M I P A V

Medical Image Processing, Analysis, & Visualization

MIPAV

User Guide

December 2, 2008

National Institutes of Health
Center for Information Technology
Rockville, Maryland

December 2, 2008 12:45 pm

Matthew McAullife, PhD mcmatt@exchange.nih.gov
301-594-2432
Building 12A, Room 2041
National Institutes of Health
Bethesda, Maryland 20892

If you find a bug, please send e-mail to bug@mipav.cit.nih.gov. Frozen menus and JAVA exceptions dialogs are common signs. Please include as much information about what happened as you can. Please understand that we might need to get more information from you about what happened so we understand the problem.

If you have a feature idea, send an email to wishlist@mipav.cit.nih.gov.

Contents

List of Figures

List of Tables

Preface

The purpose of the MIPAV software program is to allow medical researchers to extract quantitative information from image datasets of various medical image modalities.

Scope of this guide

The primary purpose of the *MIPAV User's Guide* is to provide medical researchers the information they need to use the MIPAV software program.

A secondary goal is to give researchers the information needed to extend, if desired, the software's capabilities through the development of new functions in plug-in applications via use of the software's application program interface (API).

Who should use this guide

Medical researchers, medical technicians, and other people who are involved in analyzing medical images or maintaining and supporting the equipment used to produce images form the audience for the *User's Guide*.

Skills you need to perform tasks in MIPAV

Depending on the platform—Windows, MacIntosh, or Unix—that is running your workstation, it is recommended that you are familiar with installing and using software programs for that platform.

If you plan to create plug-in applications for MIPAV to add new functionality, you must have software programming skills and be familiar with Java.

How this guide is organized

The *MIPAV User's Guide* is divided into two volumes:

- Volume 1, *Basics*, explains how to use the basic features and functions of MIPAV and how to incorporate plug-in applications.
- Volume 2, *Algorithms*, presents detailed information about the purpose, background, and function of the algorithms packaged with MIPAV and gives instructions for using them.

Volume 1, Basics

The *MIPAV User's Guide*, Volume 1, *Basics*, includes the following:

- Chapter 1, "Introducing MIPAV," presents an overview of the MIPAV software program.
- Chapter 2, "Installing MIPAV," explains how to install, remove, and upgrade the MIPAV software program. It also explains how to subscribe to the MIPAV mail list and how to search the MIPAV archive.
- Chapter 3, "Getting Started Quickly with MIPAV," explains how to use MIPAV to perform common functions, such as opening an image file, saving the file, and printing a log file.
- Chapter 4, "Understanding Image Basics," provides background information on image file formats. It also provides information on how to view and adjust image file attributes.

- Chapter 5, "Working with DICOM Images," explains how to access DICOM databases, perform queries, and retrieve image files. In addition, it explains how to send files to a database.
- Chapter 6 "Connecting to SRB BIRN" explains how to connect to SRB BIRN.
- Chapter 7 "NDAR Imaging Import," provides the help for the NDAR Imaging Import tool, which is designed to assist uses in moving data from the MIPAV XML output to the NDAR shared data repository for the benefit of the greater autism community.
- Chapter 8, "Visualizing Images," provides instruction on how to customize the way image files are displayed, how to magnify and minify images, how to view images together, how to view a portion of the image, and how to change image brightness and contrast by generating histograms and adjusting color look-up tables (LUTs).
- Chapter 9, "Segmenting Images Using Contours and Masks," explains how to create, group, rearrange, and modify volumes of interest (VOIs); how to create masks; and how to use paint to further identify VOIs.
- Chapter 10, "Analyzing Images," discusses how to calculate statistics for VOIs and masks and how to generate intensity profiles, or graphs, for images.
- Chapter 11, "Changing Image Datasets Using MIPAV Utilities . . .," explains how to use the utilities included in the software.
- Chapter 12, "Using Scripts (Macros) in MIPAV," describes how to develop scripts, which you can use to customize the program.
- Chapter 13, "Developing Plug-in Programs," and Appendix D explain how to incorporate plug-in programs into MIPAV.
- Chapter 14, "Technical support" on page 594 provides information about MIPAV technical support and also explains how to use the debug mode.
- Appendix A, "References," provides a list of references that can be used to learn more about MIPAV functions.
- Appendix B, "DICOM Conformance Statement," provides a copy of the formal DICOM Conformance Statement, which specifies MIPAV's

service classes, information objects, communications protocols, and media storage application profiles.

- Appendix C, "Supported formats" on page 627 lists graphical and file formats supported by MIPAV. It also provides examples of MIPAV system files (such as the preference file) and explains how the user can interpret them; provides limited instruction on how to modify specific files.
- Appendix D, "PluginAlgorithmMedian" gives an example of MIPAV plug in.

The guide also includes a glossary of terms and acronyms.

Volume 2, Algorithms

Volume 2, *Algorithms*, includes two chapters:

- "Understanding MIPAV capabilities," which discusses the tools and application programming interface provided with MIPAV
- "Using MIPAV Algorithms," provides detailed information about the algorithms packaged in MIPAV

In addition, the book includes the glossary of terms and acronyms.

Where to find more information

Both volumes 1 and 2 of the *MIPAV User's Guide* are available as Acrobat PDF files, which you can view, download, and print. You can either print each volume, or you can print individual chapters separately. For PDFs of this guide, go to the MIPAV web site:

http://mipav.cit.nih.gov

Conventions used in this guide

This guide uses the following conventions:

This convention . . .	Stands for . . .
Italics	Names of books, guides, or manuals as references New terms or emphasis Names of executable files
Bold	User input Names of programming commands
All caps	File types, such as TIFF, GIF, or JPG
Upper- and lowercase	Names of keys
name@address.com	E-mail address format
<u>Hyperlink</u>	An internet link (position the cursor on this word and click the left mouse button)*
`Monospace`	Code sample, including keywords and variables within text and as separate paragraphs, and user-defined program elements within text

*All figure and table citations, such as Figure 1 or Table 1, are hyperlinks although they are not underscored. Clicking the citation displays the appropriate figure or table.

Both volumes of the *MIPAV User's Guide* include special information that briefly highlights particular features or functions or that provide clarification. Based on the type of information they convey, these notes are labeled "note," "tip," "example," "recommendation," "remember," "reference," "caution," and "disclaimer." The following examples indicate how these notes appear and the type of information they include.

Note: Notes provide additional information that is related to the subject at hand. They tend to be "by the way" types of information or asides.

Tip: Tip paragraphs point out application shortcuts or briefly describe special features of the software.

Example: An example paragraph provides an example of a task or incident in which something of note could occur.

Recommendation: Paragraphs that are labeled "Recommendation" suggest methods of good practice or advice.

Definition: The definitions of specific words or phrases of note appear in "definition" paragraphs.

Remember: Notes labeled "Remember" present information that was previously discussed and that is pertinent in the current topic.

Reference: A reference note highlights one or more references that contain information on the current topic.

Caution: A paragraph labeled "Caution," alerts you to be very careful about avoiding some action that could harm your equipment or data.

Disclaimer: A disclaimer indicates the possible limitations or ramifications of a topic.

1 Introducing MIPAV

In this chapter . . .

Imaging is essential to medical research and clinical practice. Biologists study cells and generate three-dimensional (3D) confocal microscopy datasets; virologists generate 3D reconstructions of viruses from micrographs. Radiologists identify and quantify tumors from Magnetic Resonance Imaging (MRI) and Computed Tomography (CT) scans.

Neuroscientists detect regional metabolic brain activity from Positron Emission Tomography (PET) and functional MRI (fMRI) scans. Analysis of these diverse image datasets require sophisticated quantification and visualization tools. Until recently, 3D visualization and quantitative analysis of an image dataset could only be performed on expensive UNIX workstations with customized software.

Because of technological advancements, medical image visualization and analysis can now be performed on an inexpensive desktop computer that is equipped with the appropriate software applications.

This *User's Guide* explains how to use one of these software applications: Medical Image Processing, Analysis, and Visualization (MIPAV). Researchers use MIPAV to extract quantitative information from image datasets of various medical image modalities. The MIPAV application can run on virtually any platform, including Microsoft Windows, Solaris, and the Macintosh Operating System (Mac OS).

This chapter provides information on the characteristics of the product:

- Platform independence
- Supported image types
- Ability to visualize image datasets of two or more dimensions
- Volume of interest (VOI) segmentation and analysis methods
- Ability to accommodate customized user-created plug-ins

Platform independence

Much research at NIH requires the segmentation, quantification, and visualization of 2D, 3D, and 4D image datasets. Researchers analyze images of varied imaging modalities, such as microscopy, microarray data, X-ray, CT, MRI, fMRI, and PET. Factors such as personal preference, data requirements, software limitations, and precedent have led to a heterogeneous distribution of computer platforms, among which are personal computers executing Windows or Linux, Macintoshes, and workstations by SGI, Sun Microsystems, or Hewlett-Packard. To analyze an image dataset, researchers may use several software applications. If each software application is platform specific, researchers may need access to several platforms to analyze a single image dataset. This often reduces efficiency while simultaneously increasing lab costs. MIPAV has been designed to help researchers increase efficiency and reduce costs by providing them with a flexible tool that can operate on virtually any platform. Researchers can use MIPAV by itself or in concert with other image processing and visualization tools.

The MIPAV application is platform independent because it is written in Java, which is an object-oriented, interpreted, programming language that was developed by Sun Microsystems. Java source code is compiled into the bytecode, which is machine-level code that is compiled specifically for the Java Virtual Machine (VM). There are versions of the Java VM for different platforms. The same program (bytecode) can run on any of those versions. If researchers run a Java program on a Windows 2000 platform, the bytecode is interpreted by the Java VM that has been specifically designed for the Windows 2000 platform. If the same program is run on a Solaris platform; the bytecode is then interpreted by the Java VM that was specifically designed for the Solaris platform.

Note: The correct version of the Java VM can be downloaded from the MIPAV web site **http://mipav.cit.nih.gov** along with the MIPAV installation program.

Supported image types

Before image dataset analysis and quantification can be performed, an application must be able to read and write image datasets in industry-standard formats. Conformance to accepted standards, such as DICOM, ensures compatibility with present and future applications and medical equipment. This protects researchers' investment in hardware and provides flexibility in reaching their goals.

MIPAV supports over 20 different industry-standard image formats including: DICOM, TIFF, Analyze, and RAW (a complete list appears in Appendix C: "Supported formats" on page 627). MIPAV reads and writes images in both big and little endian formats.

Visualization of images

The visualization of datasets with two or more dimensions is an important aspect of image dataset analysis and research. The ability to visualize the orientation, locality, or progression (time) of structures in clinical and nonclinical datasets can be vital to researchers. Confocal microscopy, CT, and MRI are examples of imaging modalities that are comprised of multiple adjacent cross-sectional image datasets that can be combined to form a 3D volume dataset. MIPAV allows researchers to visualize datasets using a variety of presentation formats, including lightbox, triplanar, cine, and animate. Once researchers display the image dataset, they can adjust the lookup table (LUT), apply prepackaged pseudo-color LUTs to highlight structures of interest, control the magnification level, adjust the transfer function, and more.

Volume of interest (VOI) segmentation and analysis

Another significant research activity is the quantification of data from image datasets. Although the visualization of image data is important, the actual quantification of the data is typically required to evaluate the researchers' hypothesis. Researchers must be able to identify regions-of-interest (ROIs) and/or volumes-of-interest (VOIs).

Note: An ROI is used in the context of 2D image datasets. VOI usually describes the analysis of volume data for datasets with more than two dimensions. This document uses the term VOI to represent both ROI and VOI.

Image segmentation is the process of identifying connected regions of images as members of a common group. In the medical field, physicians must routinely identify (i.e., segment) structures in medical image datasets to facilitate the treatment of patients. For example, many researchers who study the brain are interested in the segmentation of gray matter, white matter, and cerebrospinal fluid in MR images. The quantification of important attributes, such as volume, of various tissue types enables researchers to better understand, diagnose, monitor, and treat neurobehavioral disorders.

There are a multitude of image dataset segmentation methods; the choice of segmentation algorithm depends on the image data type and task. Automatic segmentation methods are desirable because they require little user interaction, which is subject to operator error and subjectivity. However, in practice automatic methods sometimes fail and require manual VOI correction (adjustment of the boundary that identifies the region). Thus, in MIPAV, researchers have the choice to automatically, semiautomatically, and manually segment VOIs. Contours can be manually edited, grouped, and copied to other slices in the dataset. MIPAV also offers a variety of mask-generation methods. Researchers can manually paint a mask or use one of several algorithms.

MIPAV also allows researchers to perform statistical calculations on masked and contoured VOIs. Statistical results can be saved to an ASCII text file and imported to another program, as needed.

Extensibility with Java plug-ins

A typical analysis and visualization application can be designed to meet a broad range of researcher requirements. Many components of image dataset processing, analysis, and visualization techniques are general and can be applied to many types of data. However, many datasets also require unique functionality to meet special requirements. MIPAV allows researchers, who have the programming resources, to add a customized Java plug-in to the application. To program a plug-in, researchers must have a strong understanding of the underlying structure of the application's software design.

Note: This *User's Guide* presents information on how to add and remove plug-ins from the MIPAV application. It also indicates the statements that must be included in the source code to allow the plug-in to interface properly with MIPAV. However, in-depth information is not included in this guide. If you need more information, check the MIPAV web site **http://mipav.cit.nih.gov** for the e-mail address for technical support (Figure 1).

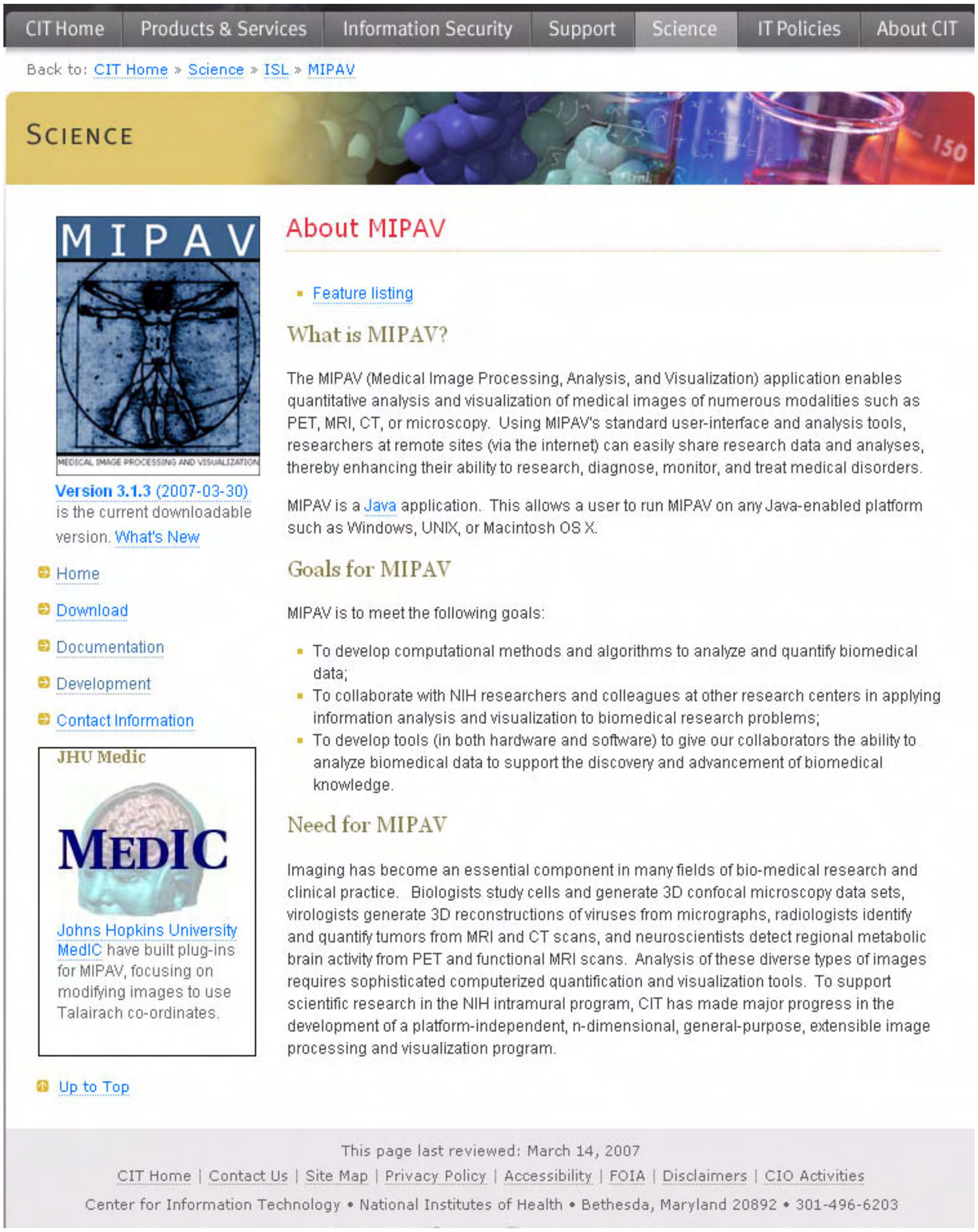


Figure 1. MIPAV home page

Sampling of MIPAV's features

MIPAV provides ready-made, general-purpose tools that meet the majority of requirements of many researchers. Researchers can use MIPAV to perform a variety of tasks. The following list shows a sampling of the tasks that researchers can performed with the program. These tasks and others are addressed in volumes 1 and 2 of this *User's Guide*.

- Visualize files and create new image dataset files
- View and modify the attributes of an image dataset, including DICOM and VOI information
- Adjust the display of an image dataset file and adjust magnification settings
- View DICOM overlays and protect patient privacy using the anonymize feature
- Send and receive image dataset files to and from databases via DICOM-compliant servers
- Contour VOIs using manual, semi-automatic, and automatic methods
- Generate graphs and calculate statistics on VOIs
- Generate and adjust histograms and LUTs using customized or preset options
- Run sophisticated, predefined algorithms, and generate logs
- Blend two image datasets and adjust opacity levels of the alpha channels so overlapping areas can be studied
- Create new plug-ins to further customize the analysis of data
- Save transformation, LUT, and VOI data, and apply them to other image datasets
- Print image dataset files, intensity profiles, statistical data, algorithmic logs, and debugging log data
- Create black and white, grayscale, and pseudo-color masks

2 Installing MIPAV

In this chapter . . .

Because MIPAV is Java-based, it can run on many different platforms, such as Windows (XP, NT, 95, 98, and 2000), Solaris, Linux, UNIX, Macintosh, and SGI workstation. This chapter explains:

- System requirements
- How to download the MIPAV installation file and install the MIPAV software
- How to subscribe to the MIPAV mailing list and search the archives
- How to view the list of enhancements made to the current version of MIPAV
- How to upgrade MIPAV
- How to remove MIPAV from your workstation

Understanding the system requirements

MIPAV runs optimally on a system with:

- **Pentium III 400-megahertz (MHz) processor**—Although MIPAV can run on a computer with a slower processor speed, for peak performance a processor that is equivalent to or faster than Pentium III 400 MHz is recommended.
- **At least 100 Megabytes (MB) of a combination of random access memory (RAM) and virtual memory (recommended)—**Minimally, MIPAV requires a base of 35 MB of RAM. However, when you open an image file, MIPAV requires additional memory to correctly display the file and quantify data. By default, MIPAV allocates 100 MB of a combination of RAM and virtual memory. However, your own memory requirements may differ.

  To more precisely determine the amount of memory that you need, estimate the number of files you generally have open at one time. Next, add the sizes of the files together and multiply the result by 10.

Example: During a typical MIPAV session, you might generally compare and contrast three image files. The estimated file sizes may be 2 MB, 4.5 MB, and 6 MB. The calculation would be: 10(2 + 8.5 + 10) + 35 = 240. Thus, in this scenario, you need 240 MB of a combination of RAM and virtual memory to run MIPAV and visualize the three image files. Note that for DICOM datasets you must add all of the files in the dataset, even if you only work with one or two images.

Note: If you need to allocate more than 100MB of memory, after you install MIPAV you need to adjust the memory allocation settings. Instructions are found in Chapter 3.

- **A minimum of 25 MB free hard disk space *plus a*dditional image file storage space**—MIPAV software components require 25 MB of hard disk space for storage. If you plan to store image files on your hard disk, you need to allocate more space.

Installing MIPAV

Installing MIPAV consists of downloading the MIPAV installation file from the MIPAV web site and saving it to your hard drive. Next, you run the installation program to install MIPAV on your computer's hard drive.

To begin the installation process, complete the following steps:

1 Enter the following address in your web browser:

http://mipav.cit.nih.gov/

The Medical Image Processing, Analysis, & Visualization (MIPAV) web page opens (Figure 2).

2 Click either the Version x.xx link or the Download link. The Downloading MIPAV page opens (Figure 2).

3 Fill out the required form – type your name and e-mail address.

4 Scroll down the page to locate and read the installation instructions provided for installing MIPAV on your workstation's platform.

5 You might choose to download the tested release version or the nightly release, which is the most recent un-tested version.

6 Click the appropriate platform link—Windows, Linux, Solaris, Macintosh, or Other. The File Download dialog box opens.

Figure 2. The Medical Image Processing, Analysis, and Visualization (MIPAV) home page

7 Click Save to the save the program to your workstation's hard disk.

8 Follow the installation instructions for your workstation's platform to install MIPAV.

Using the MIPAV mailing list

You can subscribe to the MIPAV mailing list to receive information about MIPAV software updates and other MIPAV-related announcements. The MIPAV mailing list is housed on LISTSERV at NIH. LISTSERV is an e-mail-based server that manages the subscriptions and archives for mailing lists generated by NIH organizations.

To get help on using NIH LISTSERV

1 Enter the following address in your internet browser:

http://list.nih.gov

The NIH LISTSERV web page (Figure 3) opens.

2 Select the For General Users link near the bottom of the screen. Five new links appear.

3 Select any of the links for information on that subject.

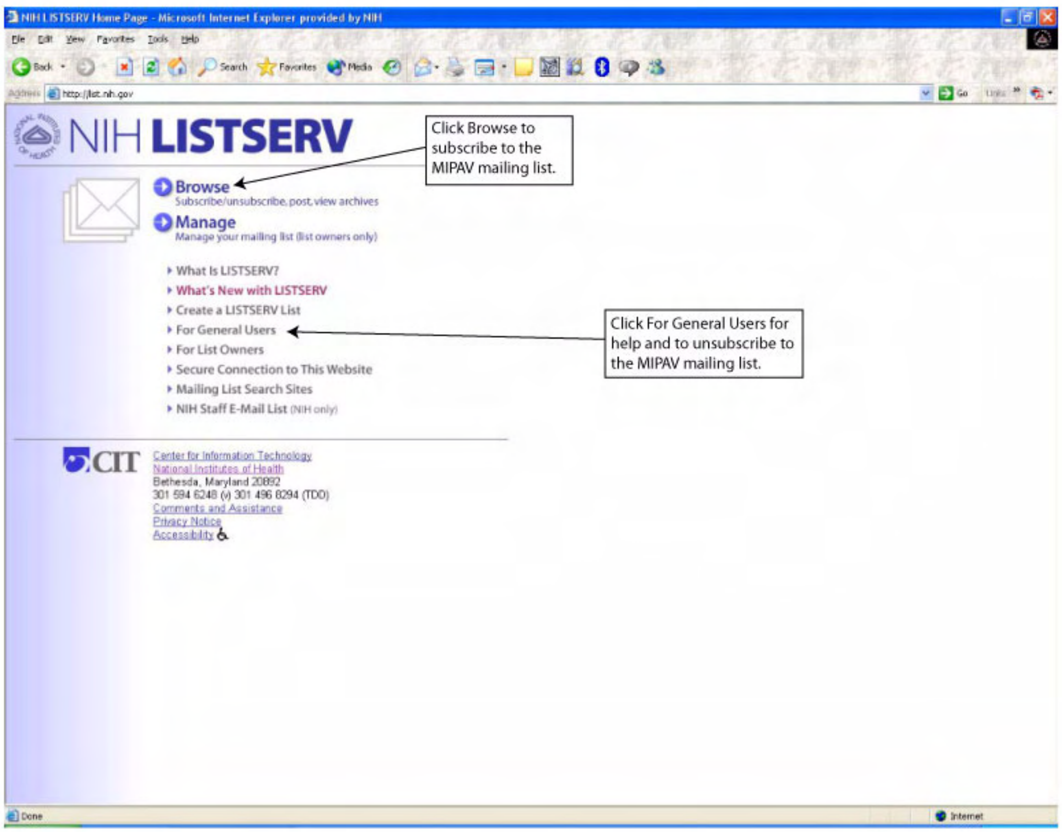


Figure 3. NIH LISTSERV page

To join the MIPAV mailing list

1 Go to the NIH LISTSERV web page (refer to Figure 3).

2 Select Browse. The List of Lists page (Figure 4) opens.

3 Select M. The M section of the list opens.

4 Scroll down to find the MIPAV mailing list.

5 Select the MIPAV link. The Archives of MIPAV page opens.

6 Select the Join or leave the list, or update options link. The Join or Leave the List, or Update Options page opens.

7 Complete the information in the page.

8 Click Join the List.

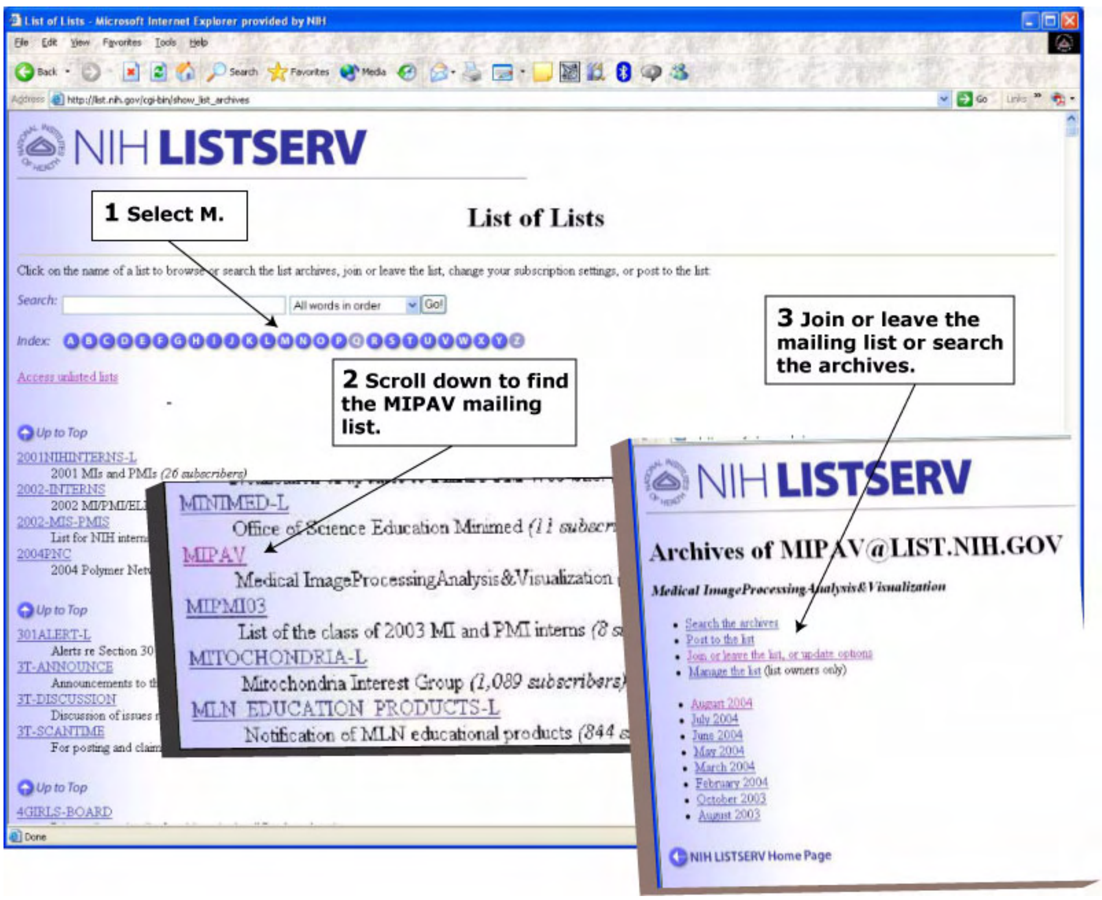


Figure 4. How to subscribe to the MIPAV mailing list

To unsubscribe to the MIPAV mailing list

1 Go to the Archives of MIPAV page.

2 Select Join or leave the list, or update options. Click Leave the List.

Viewing MIPAV news and changes

The version number of the most current version of MIPAV, along with a list of enhancements and corrections made to that version, appears on the MIPAV web site.

To view MIPAV news

1 Go to the MIPAV web site on your web browser:

http://mipav.cit.nih.gov/

The Medical Image Processing, Analysis, & Visualization (MIPAV) web page opens (refer to Figure 2 on page 53).

2 Select the What's New link. The MIPAV News page opens.

This page lists the enhancements and corrections made to the most recent version of MIPAV.

To view a list of cumulative changes to MIPAV

1 Select the cumulative list of changes link at the foot of the MIPAV News page. The MIPAV Version History page opens.

2 Read the lists of enhancements and corrections made to not only the most recent version but to all previous versions of MIPAV.

Upgrading MIPAV

Generally, when a new version of MIPAV is available, a message is sent to the MIPAV mailing list. You can also find the latest version number if you go to the MIPAV web site and select the Download link.

To upgrade MIPAV

1 Save the following information:

- Hosts table information. You must reenter this information when you reinstall MIPAV (refer to Chapter 5).
- List of plug-in programs. You must reinstall all plug-in programs when you reinstall MIPAV (refer to Chapter 10).

2. Remove the existing copy of the software.
3. Download the installation file for the new version.
4. Install the new version of the software.

Removing MIPAV

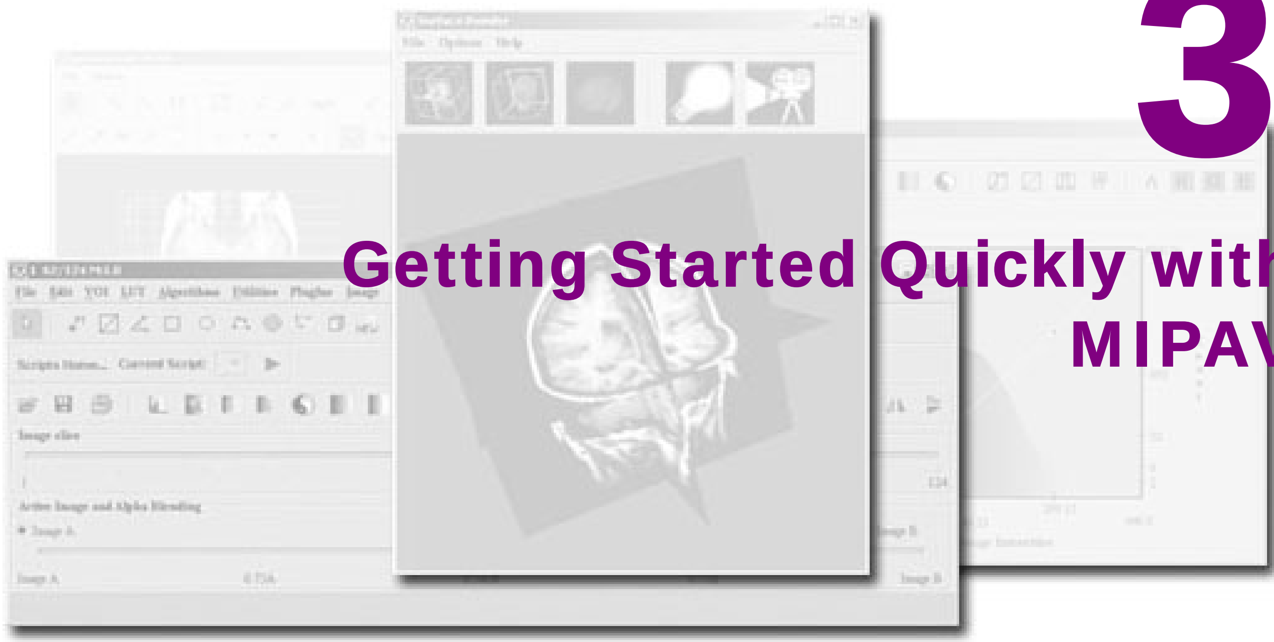
When MIPAV is removed, all of the originally installed software components are removed from your hard drive. Note that files and folders that were created after installation, such as image dataset files and plug-ins, are not removed.

To remove MIPAV

1. Navigate to the directory where the MIPAV software is installed. If you are not sure of the location:
    - a. Start MIPAV (refer to Chapter 2, "Quick Start").
    - b. Select Help > About Java in the MIPAV window. The About Java window opens.
    - c. Note the User Directory. The installation directory is the same as User Directory.
2. In the User Directory, navigate to the UninstallerData subdirectory.
3. Run the MIPAV uninstallation file. File names appear in Table 2-1. The About to Uninstall window opens.
4. Click Uninstall to begin the process. The Component Removal Status window opens.

    Components are removed from the system. When finished, the Uninstall Complete window opens.
5. Click Exit to close the window.

Note: If a component has not been removed, you can use the standard file deletion procedures for your platform to manually remove it. By default, all of MIPAV's components are stored in the directory where MIPAV was installed.



3 Getting Started Quickly with MIPAV

In this section. . .

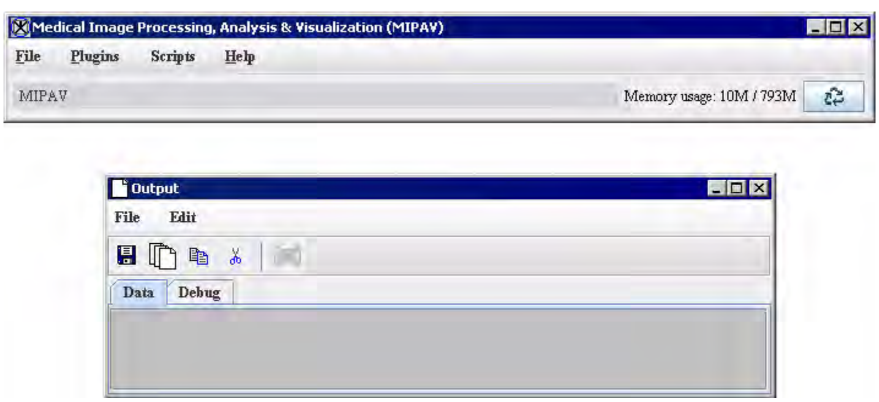
This chapter provides information to help you get started quickly using MIPAV. It explains basic tasks, such as starting and ending the program, opening and loading images, creating new images, and saving and printing images. In addition, it discusses how to create volumes of interest (VOIs), how to compare images using alphablending, and how to customize the program.

Starting MIPAV

Starting MIPAV differs depending on your platform. To begin a MIPAV session, do the following:

If your platform is . . .	Then . . .
Windows 95, 98, 2000, NT, XP	Select Start > Programs > mipav > mipav.
Linux	Open a shell. Go to the directory where you downloaded the installer. At the prompt, type **sh ./installMIPAV.bin**.
Mac Operating System (OSX)	Unzip the installer (StuffIt expander may open automatically). Double-click the installer icon on your desktop.
Unix OS (Sun Solaris, OS/2 Warp, and others	In a shell tool, command tool, or terminal window, go to the directory where you installed MIPAV. At the prompt, type **./mipav**.

Both the main Medical Image Processing, Analysis, & Visualization (MIPAV) window and the Output window (Figure 5) appear on your desktop. You can access most of MIPAV's features from the main menu bar.



The Output window menu options

Figure 5. MIPAV–Main and Output windows

File	**Save Messages**—Saves all of the messages displayed on the currently open page (e.g., Global Data page, Data page, or Debug page). When you select this command, the Save dialog box opens. **Add tab**— Adds a new tab to the Output window. When you select this option, the Add Tab dialog box opens. In this dialog box, enter the name for the new tab and press OK.
Edit	**Clear messages**—Clears all of the messages that are displayed on the currently open page. Be sure that you want to use this command. There is no undo! **Copy**—Copies the selected text on the currently open page. **Cut**—Removes the selected text from the currently open page. **Select All**—Selects all of the text on the currently open page.
The Output window toolbar	
Save Results	Saves all of the messages displayed on the currently open page. When you select this command, the Save dialog box opens.
Copy	Copies the selected text on the currently open page.
Paste	Pastes copied text into the currently open page at the cursor point.
Cut	Removes the selected text from the currently open page.
Remove selected tab	Removes selected tab from the output window.
The Output window default tabs	
Data tab	Displays messages that apply to the currently active image.
Debug tab	Displays debugging information for the currently selected image.

Figure 5. MIPAV–Main and Output windows (continued)

After starting MIPAV, you can open and load image files, create new images, access DICOM images, customize MIPAV, and end the session.

Managing memory resources

MIPAV requires at least 35 Mb of random access memory (RAM). Additional memory is needed to correctly display image files and to quantify the data. To determine the amount of additional memory needed, multiply the size of the image file that you want to display by 10. Thus, if an image file is 2 Mb, you must allocate an additional 20 Mb of RAM (in addition to the base of 35 Mb) for it to display correctly. Because memory requirements fluctuate depending on the size and number of image files open, it may be necessary to allocate additional memory during a session.

Allocating memory in MIPAV

When you run MIPAV for the first time you must specify the amount of memory that you would like to allocate, so that MIPAV can take the full advantage of the memory resources available in your machine.

1 Select Help > Memory allocation in the MIPAV window. The Change JavaRuntime Memory Allocation dialog box opens. See Figure 6.

2 Change the maximum heap size number so that it reflects the maximum amount of memory that should be allocated.

In a Microsoft Windows system, you can specify a maximum to 1.2 or 1.3 Gb. If additional memory is needed, MIPAV allocates it until the memory reaches the maximum heap size that you specify.

3 Click OK to close the dialog box.

Figure 6. Change Java-Runtime Memory Allocation dialog box

You must restart of MIPAV for the changes to take effect. To exit MIPAV, select File > Exit. Then, restart the software.

TO ALLOCATE ADDITIONAL MEMORY

The first step is to view how much memory is currently being used and to free needlessly reserved memory. Then, if necessary, you may need to allocate additional memory.

To determine how much memory is currently being used by MIPAV

1 Select Help > Memory Usage in the MIPAV window. The Memory Monitor dialog box appears (Figure 7).

When MIPAV performs a function, such as visualizing an image file, it uses memory. When the function completes (i.e., the visualized image closed), you can manually free the memory so it can be used for another function.

The Memory Monitor dialog box shows how much memory is allocated and how much memory has been used. The vertical bar on the right of the window displays a pictorial representation of the ratio of allocated memory or amount of memory used. The chart shows the memory usage for the past 3 minutes and 45 seconds. The chart is updated every second.

If you want to constantly monitor your memory resources, you can leave the Memory Monitor dialog box open on your desktop. Otherwise, close it.

2 Click Free memory to free memory.

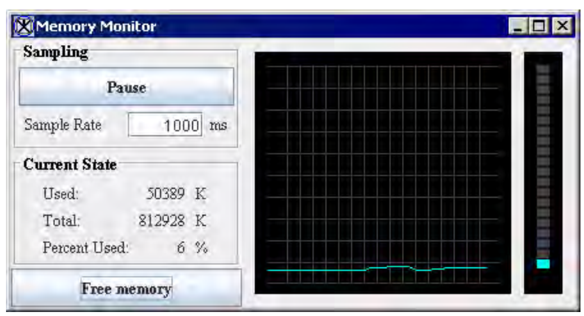


Figure 7. Memory Monitor dialog box

If you need more memory, you can allocate virtual memory or disk swap space if there is free space on your hard drive. For instructions on how to do this, refer to the documentation for your system. If you cannot assign more virtual memory or disk swap space, you may need to install additional memory.

To manually free the memory

As Java-based programs run, they often leave old variables, objects, or constructors in memory. For instance, if you delineate a VOI, the coordinates of the contours may remain in memory even after you close the image. Java provides a method called the *garbage collector* (or *memory recycling*) that automatically freeing objects that are no longer referenced by the program and clears all unnecessarily reserved memory. Generally, the software does this automatically when free memory becomes very limited. However, you can run the garbage collector at any time to free memory by clicking the Memory Usage button. Refer to Figure 8.

To allocate additional memory,

Repeat the steps described in "Allocating memory in MIPAV" on page 61.

Figure 8. Running the garbage collector to free the memory in MIPAV

Opening and loading image files

MIPAV displays images in windows, or frames, that you can move or arrange around the desktop (Figure 9). You can display as many image datasets as needed as long as your computer has enough allocated memory.

Note: The amount of memory required depends on the number of image datasets that are open and the size of each image file. For more information, refer to "Managing memory resources" on page 60.

You can either open or load an image file. When you *open* an image file, MIPAV displays the image in a new image window. *Loading* an image file imports the file into an image window in which another image file is open. In other words, two image files share a common image window.

UNDERSTANDING IMAGE WINDOWS

An *image window* (Figure 9) consists of a title bar and an image itself.

The title bar displays:

- the image file name, e.g. "3DHead";
- the number of the current slice and the total number of slices in the dataset, e.g. 62/124;
- the magnification level, e.g. M:1.0.

The number of the current slice is also shown in the lower left corner of the image.

Figure 9. An image window

You can move an image window around your desktop by clicking the title bar and dragging the window to a new location.

Sometimes, depending on the view, a toolbar or menu bar also appears in the image window. For example, the image window using the *light box view* contains a toolbar and menu bar. See also "Displaying images using the lightbox view" on page 76.

IMAGE A AND IMAGE B

In MIPAV, the first image opened in an image window (see "Understanding image windows") is referred to as *Image A*. When you load, or import, a second image in the same window, MIPAV refers to the second image as *Image B*.

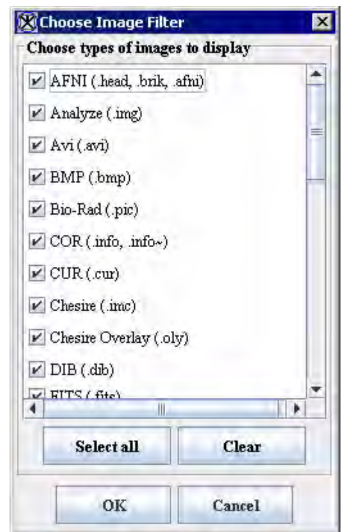
Supported formats

You can use MIPAV to open or load files of a variety of medical and generic graphics and multimedia file formats. Table 1 lists the format, extension, and whether MIPAV can read or write files of a particular format. The *extension*, which is the last few characters of a file name including the period, indicates the file format. For example, in the file name *DOE255.ima*, the extension *.ima* indicates that the file is stored in DICOM format. In the table, the Read column indicates that MIPAV can display image files of a particular format. When a file is read, it is stored in main memory so that MIPAV can access it. The Write column means that MIPAV can copy the data from the memory to a storage destination, such as your hard disk. In this context, *write* is synonymous with *save*.

For more information on file formats, the list of supported formats and other information, refer to Appendix C: *"Supported formats" Table 1 on page 628.*

IMAGE BROWSER

You can select the image formats to display using the File > Open Image(A) > Image Browser menu. This opens the **Choose Image Filter dialog box**. In this dialog box, use the check boxes to select the image types which you would like to use in MIPAV.



Select All	Selects all image types.
Clear	Removes selection.
OK	Applies the image type selection to MIPAV.
Cancel	Disregards any changes you made in this dialog box and closes the dialog box.

Figure 10. Choose Image Filter dialog box

Opening image files

The **Open Image A from disk** menu allows you to open an image which is stored on your hard drive (or network drive).

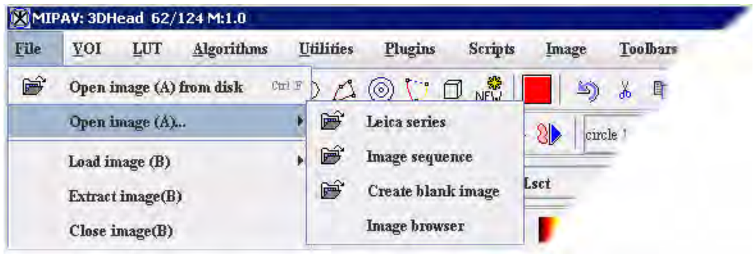


Figure 11. FIle > Open menu

The **Open Image A** menu (Figure 11) provides you with additional options such as to open a single image file, or image sequence, or Leica series images – the images that were taken on a Leica microscope. You can also call Create Blank Image to create a blank image (refer to "Creating new images" on page 91).

To open an image file or multifiles

An option on the Open Image dialog box (Figure 11), which appears after you select File > Open Image(A), is **Open as multifile**. This option lets you open the image as a *multifile*.

Multifiles are image datasets that are composed of multiple files.

1. Select File > Open Image(A) from disk. The Open Image dialog box (Figure 12) opens.
2. Navigate to the directory where the file is stored.
3. Select an image file.
4. To open the image file as a multifile, select the **Open as multifile** box.

   The name of the image appears in File Name.

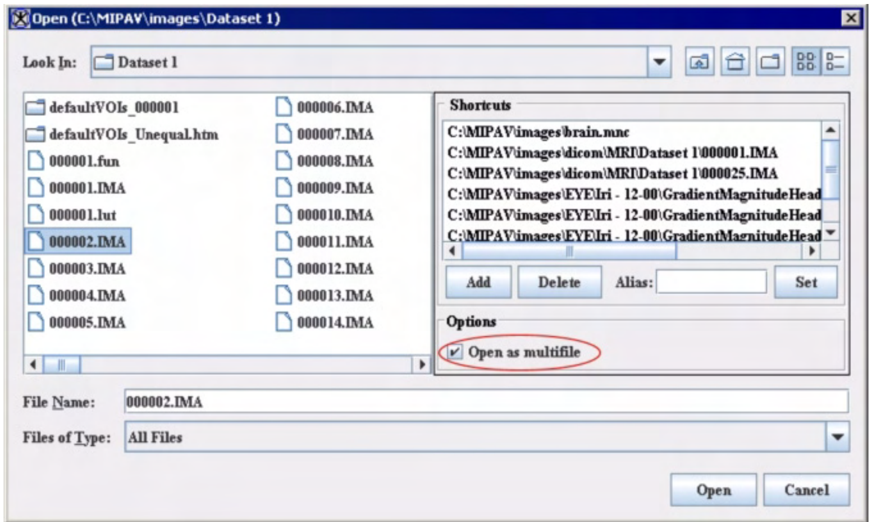


Figure 12. Open as Multifile option

If you can't find the image,

check the files types listed in Files of Type (showing in Figure 12) near the bottom of the dialog box. If the file type you are looking for does not appear in Files of Type, select All Files or All.

To form a 4D dataset in Analyze format,

use **Open as multifile** to open a series of individual 3D Analyze, formatted, consecutively ordered images.For example, you can use Open as multifile to open an array of TIFF, JPEG, BMP, etc., files if their file names meet the following format: *foo*_001.*tiff*, *foo*_002.*tiff*, *foo*_003, etc. where *foo* is the name of the file.

MIPAV window

After the first image is opened, the MIPAV window changes. It includes more menus, the title bar displays more information, and tool bars appear. Because the MIPAV window is context sensitive, it displays or hides commands on the menus depending on the characteristics of the image file that is opened. If you opened a dataset file that contains more than one image, an *image slice slider* also appears. Refer to Figure 13.

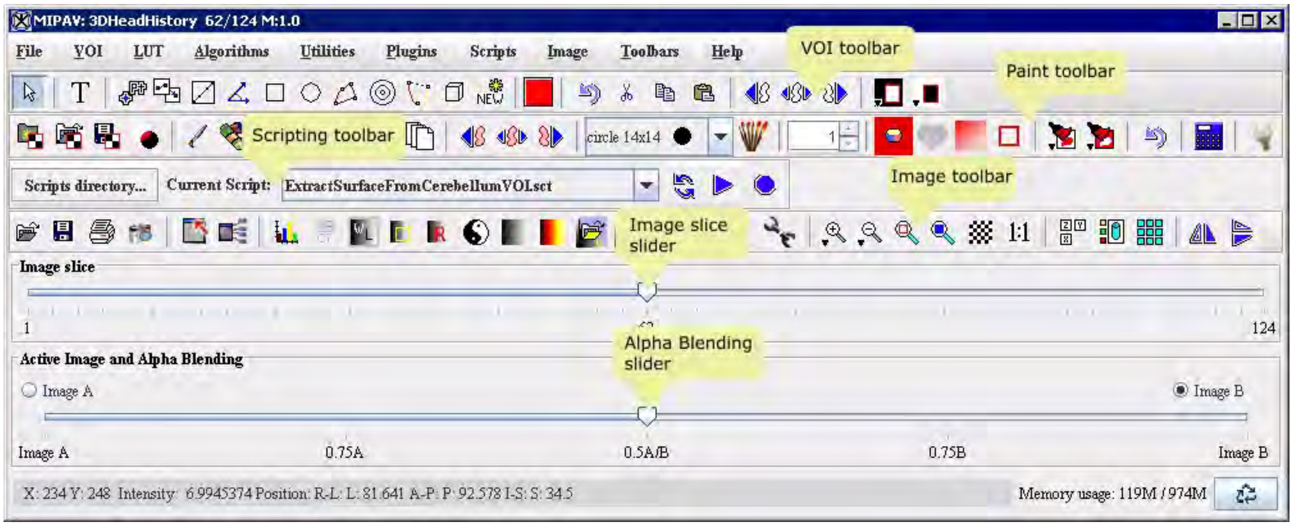


Figure 13. Expanded MIPAV window showing toolbars, image slider, and more menus after the first image is opened. See also "MIPAV toolbars" on page 71

If you prefer for MIPAV to use the style of Open and Save dialog boxes that are used by the operating system (e.g., Microsoft Windows, Unix, or Apple) on your computer, read the section on "Using platform-specific Open and Save dialog boxes" on page 124.

Loading image files

When you *load* an image file, MIPAV imports it into an existing image window. The loaded image shares the same window with another image file. Loading a file allows you to compare two datasets.

To load an image file

1. Open the first image by selecting File > Open Image(A) from disk. The image appears in an image window.
2. Select the title bar of the image window (in this case, the image that you just opened) in which you want to load another image.
3. Select one of the following:
   - **File > Load Image (B) > From Frame** to load another image dataset from an already opened image dataset;
   - **File > Load Image (B) > From File** to load another image dataset;

- **File > Load Image (B) > Create a Blank Image** to create a blank image.

4 If you chose Image (B) from File, the Open dialog box appears, see Figure 12. Select the image file you want to load, and click OK.

If MIPAV does not recognize the type of file based on its extension, the Choose Image Filter dialog box opens (see also "Image Browser" on page 66). Here, select the file type and click OK.

As the image file is loading, a pop-up window appears with the status. When the image file finishes loading, it appears in the image window. At this point, take note of several changes:

- MIPAV pseudo-color was applied to the images in the image frame. The pseudo-color indicates that two images are loaded in the same image frame.
- In the MIPAV window, the Active Image and Alphablending slider appears. The slider allows you to control which image is displayed in the foreground of the window. The image in the foreground is the *active image*. You can adjust the translucency of the alpha channels in each image using the technique.

5 If you chose Image (B) from Frame, the Load Image onto <file name> (Figure 14) opens. Select one of the images in Set as Image (B), and then click OK. The image is loaded onto the first image file.

Note: The abbreviations *(A)* or *(B)* appear after some commands. *(A)*, which is the abbreviation for "Image A," indicates that the option is applied to the first image opened in an image window. *(B)*, which is the abbreviation for "Image B," indicates that the option is applied to second image loaded in the image window. For example, if you select Close Image (B), MIPAV closes the second image (Image B) that was loaded in the image window.

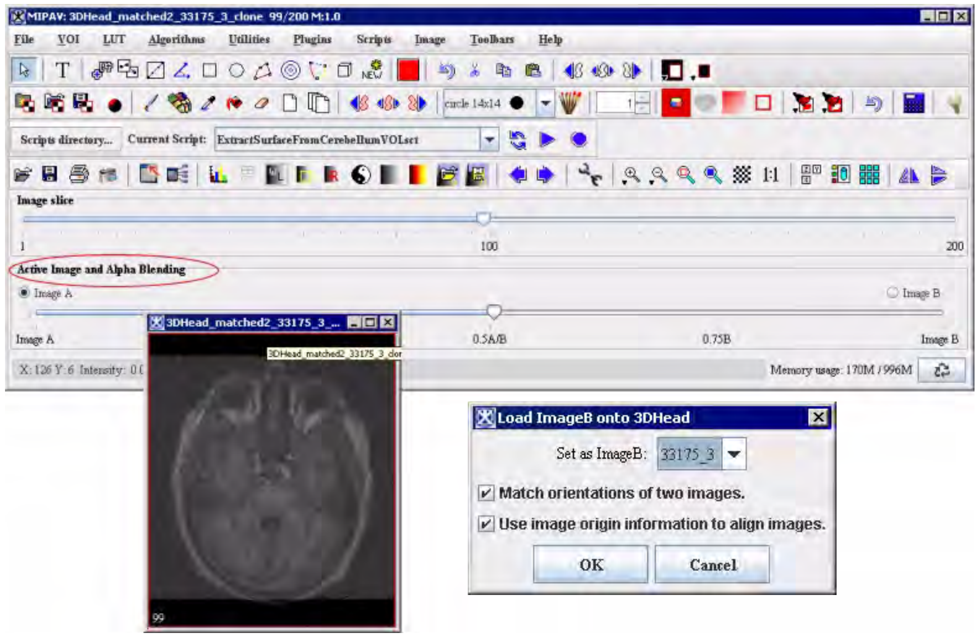


Figure 14. After loading an Image B, the Active Image and Alphablending slider appears in the main MIPAV window. The slider allows you to control which image is displayed in the foreground of the window

Note: MIPAV recognizes files types by the file extension. For example, if you select an image named “Smith.tiff”, MIPAV tries to open the file as a TIFF file. If that image is actually a *JPEG* file and mistakenly has the *TIFF* extension, MIPAV fails to open the file.

MIPAV toolbars

After you open an image in MIPAV and the MIPAV window expands to display all of its menus and toolbars, such as

- **VOI toolbar.** Volume of interest toolbar (Figure 15). The VOI toolbar contains tools that help you in selecting the specific area of interest on the image.

- **Paint toolbar.** The Paint toolbar (Figure 16) includes tools that allow you to add, adjust, or remove colors and color intensity, erase paint, and adjust the opacity level of the paint.
- **Scripting toolbar.** The Scripting toolbar allows you to locate and run previously recorded scripts, or macros, that contain two or more algorithms on images. See Figure 17.
- **Image toolbar.** The Image toolbar includes tools for opening, printing, saving, maximizing, and minimizing an image; converting an image from gray scale to color or from color to gray scale; and adding to and removing slices from an image or changing their order; and rotating, cropping, and flipping an image. See Figure 18.

By default, the MIPAV window displays only the VOI and Image toolbars.

Icon	Name	Icon	Name	Icon	Name
	Default Mode		Protractor tool		Change VOI color
	Annotation Tool		Draw rectangle VOI		Undo
	Draw a point VOI		Draw polyline VOI		Cut VOI
	Draw inter-slice polyline		Levelset VOI		Copy VOI
	Draw line VOI		Draw 3D rectangular VOI		Paste contour
	Propagate VOI down		Propagate both sides		Propagate VOI up
	Split VOI contour		Live wire VOI		View VOI properties
	Quick AND VOI mask operation		Quick NOT VOI operation		

Figure 15. VOI Toolbar

Icon	Name	Icon	Name	Icon	Name
	Add a blank mask		Fill an area with color		Paint brush editor
	Open a mask from a file		Erase a paint		Change paint color

Figure 16. Paint Toolbar

	Save the current mask		Erase a paint from a current frame		Change opacity of the paint
	AND mask operation		Erase all paints		Display a border around painted areas
	Draw using a brush		Propagate the paint to the previous slice		Masks the inside of the painted area
	Load advanced brush tools		Propagate the paint both sides		Masks the outside of the painted area
	Pick up a color from an image		Propagate the paint to the next slice		Undo
			Calculate volume		Load power paint tools

Figure 16. Paint Toolbar

Icon	Name	Icon	Name
Scripts directory...	Open the Scripts Home catalogue		Refresh the Scripts Home catalogue
Current Script: ExtractSurfaceFromCerebellumVOI.sct	Show the current script to run		Run the script from Current Scripts
			Start recording the script

Figure 17. Scripting Toolbar

Icon	Name	Icon	Name	Icon	Name
	Open image, CTRL+F		Adjust window and level		Decrement image slice
	Save image, CTRL+S		Quick LUT		Increment image slice
	Print image		Reset LUT		Link images
	Capture image to TIFF file		Invert LUT		Zoom in
	View header, CTRL+H		Gray LUT		Zoom out
	Edit attributes, CTRL+E		Hot metal LUT		Magnify region

Figure 18. Image Toolbar

Display Lookup table	Open user-defined LUT	Window region of Image B
CT preset function	Save LUT	Checker board
Magnify 1:1	Tri-planar view	Volume Tri-planar view
Volume Renderer	GPU based volume renderer	GPU rendering
View light box	Flip horizontally	Flip vertically
Multi histogram rendering		

Figure 18. Image Toolbar (continued)

Displaying MIPAV Toolbars

The Toolbars menu contains check boxes for each of the four toolbars: Image, Paint, Scripting, and VOI. Use these check boxes to select which toolbars should appear. See Figure 19.

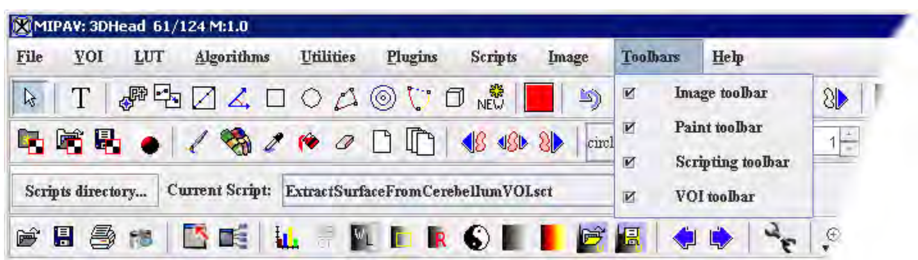


Figure 19. Toolbars menu

By marking or clearing these check boxes, you can choose which toolbars to display and which to hide. For example, suppose you just started MIPAV a moment ago. You then open an image. The MIPAV window expands in size and displays its full complement of menus. Although it displays the VOI, Scripting, and Image toolbars, you want to work with the Paint toolbar as well as the Image toolbars. However, you don't need to use the VOI and Scripting toolbars. The next section explains how to hide and show toolbars.

To hide the VOI and Scripting toolbars and display the Paint toolbar

1. Select Toolbars > VOI to hide the VOI toolbar. MIPAV removes the check mark from the check box and removes the VOI toolbar from the MIPAV window.
2. Select Toolbars > Scripting to hide the Scripting toolbar. MIPAV removes the check mark from the check box and removes the Scripting toolbar from the MIPAV window.
3. Select Toolbars > Paint to display the Paint toolbar. The program marks the check box and displays the Paint toolbar in the MIPAV window.

Because the Image toolbar is already displayed, you do not need to do anything.

Selecting views

A *view* refers to the way an image file is displayed. A view indicates how many images are shown at one time and whether images are advanced manually or automatically. Depending on the view, the size and shape of the image window in which an image is displayed can vary. MIPAV allows you to display images using the following views:

Icon	View	Icon	View
1:1	Default		Surface plotter
	Animate		Volume renderer
	Cine (movie)		Tri-planar
	Lightbox		Tri-planar dual
	Link to another image		Volume Tri-planar

This section explains how to display image files in the *default*, *lightbox*, and *cine* view. To learn how to display image files in other views, refer to Chapter 6 in the *MIPAV User's Guide*.

Displaying images using the default view

In the *default view*, MIPAV displays the images in a dataset one at a time in an image window (Figure 20). This window can display datasets of any dimension.

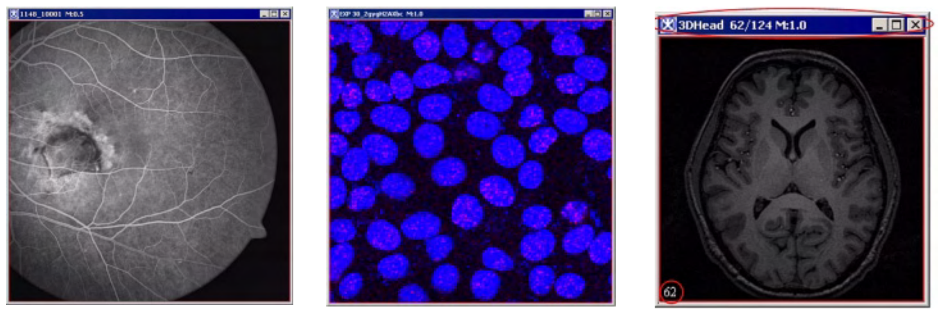


Figure 20. Images open in the default image windows

Displaying images using the lightbox view

The *lightbox view* is similar to the default view, except that all images in the dataset appear in one window at the same time (Figure 21).

To display images in the lightbox view

1 Open an image file. The image appears in an image window.

2 Select the image window. Do one of the following in the MIPAV window:

- Click the View Light Box icon;
- Select Image > Views > Light box.

The image now appears in a lightbox view.

3 Select Options > Settings in the lightbox view window to adjust the size and shape of the window. The Lightbox Settings dialog box (Figure 21) opens.

4 In the dialog box, indicate the number of rows and columns, the grid size, frame border size, color settings, and the magnification. For more information on these attributes, see the *MIPAV User's Guide*, "Visualizing Images" , "Adjusting the lightbox view" .

5 Click Close when complete. The image appears in the lightbox view, see Figure 21.

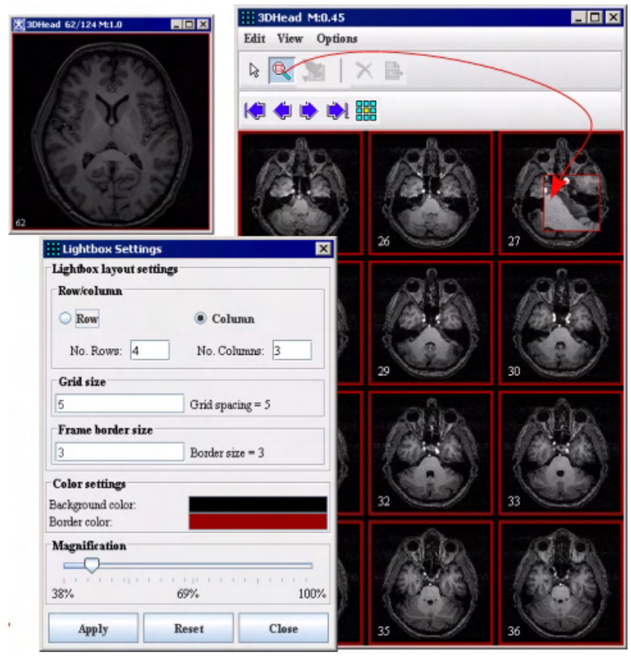


Figure 21. An image shown in the lightbox view and the Lightbox Settings dialog box

To magnify a portion of the image in the lightbox,

click Magnify Region and move the pointer to the image section to be magnified. Click Default Mode when finished.

Double-clicking an image in the lightbox view updates the 2D image frame for that image.

Displaying images using the cine view

When you view an image file in cine view, MIPAV automatically advances images one frame at a time. The effect is much like a film loop.

To display images in cine view

1 Open an image file. The image appears in the default image window.

2 Select Image > Views > Cine (Movie). The images in the image window are advanced automatically.

Adjusting magnification

MIPAV allows you to magnify images from ¼ to 32 times the size of the original image. Using the magnification tools, you can magnify or minify the entire image or just a portion of the image. Much like a traditional magnifying glass, the magnification box can also *minify*—or reduce the level of magnification—a portion of the image.

Changing the magnification level

This section explains how to change the magnification level of the image using the icons on the Image toolbar in the MIPAV window.

To learn how to adjust the magnification level using the Image toolbar, refer to

- "Magnifying images" on page 78
- "Reducing the magnification level" on page 78
- "Restoring the original level of magnification" on page 79
- "Magnifying regions within images" on page 79

To learn how to set the magnification level using the other methods, see volume 1 of the *MIPAV User's Guide*.

MAGNIFYING IMAGES

To magnify an image, click the Magnify Image icon, and then click the image. Each time you click the Magnify Image icon and click the image, the image doubles in size. If an image is too large for the current window size, scroll bars appear, and you may need to manually adjust the size of the window.

REDUCING THE MAGNIFICATION LEVEL

To reduce the magnification level of an image, click Minify Image. Each time you click the Minify Image icon, MIPAV reduces the magnification level of the image by half.

RESTORING THE ORIGINAL LEVEL OF MAGNIFICATION

To return the image to its original size or original level of magnification, click Original Magnification.

MAGNIFYING REGIONS WITHIN IMAGES

 The Magnify Region icon allows you to view a square portion, or region, of the image at a specific magnification level. If you have loaded **two images** into the same image window, the Window Region of Image B icon appears on the toolbar. Use this icon to view a region on the second image, or Image B.

To use the Magnify Region icon

1. Open an image.
2. Click Magnify Region and move it over the image.

   As you move the mouse over the image, the Magnify Region icon displays a magnified square region in the image in a red box. The number at the left corner of the square is the magnification level. See also Figure 23.

To change the size of the magnified region

You can change the size of the magnifying region to allow you to view larger or smaller regions of the image.

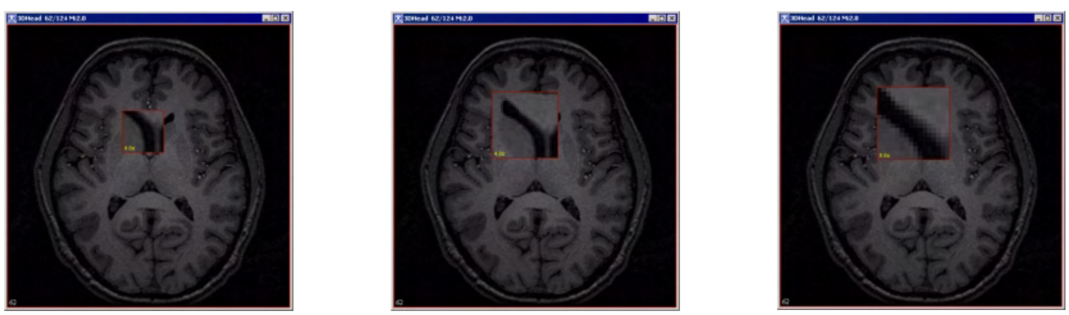
1. Right-click on the image while displaying a magnified region. The Magnification dialog box (Figure 22) opens.
2. Type either a higher number for a larger region or a lower number for a smaller region in the Width box. Valid values range from 64.0 to 198.0.
3. Click Apply. The size of the magnified region, or square, either increases or decreases in size.

Changing the magnification level of the magnified region works similarly to changing the size of the magnified region.

Magnification	Move the Magnification slide to the right to increase the magnification level or to the left to decrease the magnification level
Display intensity values	When you slide the Magnification slider right past a certain point, this check box becomes active.
Size	Type either a higher number for a larger region or a lower number for a smaller region in the Width box. Valid values range from 64.0 to 198.0.
Apply	Applies the changes.
Close	Closes the dialog box.

Figure 22. Magnification dialog box options

Tip: To review or compare a magnified region of the image with the same region at its original magnification, press Shift. To return to the Magnified view, release the key.



A–the magnified region size is set to 75; the magnification level 4 is shown in the magnified region.

B –the magnified region size is set to 130; the magnification level 4 is shown in the magnified region.

C– the magnified region size is set to 130; the magnification level 8 is shown in the magnified region.

Figure 23. The larger size magnifying glass such as shown in B and C allows to inspect a larger region of the image

To change the magnification level

1. Right-click on the image while displaying a magnified region. The Magnification dialog box (Figure 22) opens.
2. Complete the dialog box, and then press Apply.
3. The magnification level shown in the magnified region changes to the level you specified.

SHOWING INTENSITY VALUES THROUGH THE MAGNIFICATION GLASS

The Display intensity values check box on the Magnification dialog box allows you to display the intensity values within a magnified region of the image. See Figure 22.

To show the intensity values

1. Right-click on the image while displaying a magnified region. The Magnification dialog box (Figure 22) appears.
2. Select the Display intensity values check box.

   If the check box is not available, or dimmed (as it is in Figure 22), increase the magnification by sliding the magnification slider to the right until the check box is active. See Figure 24.
3. Move the mouse over the image. The intensity values appear within the magnified region (Figure 24).

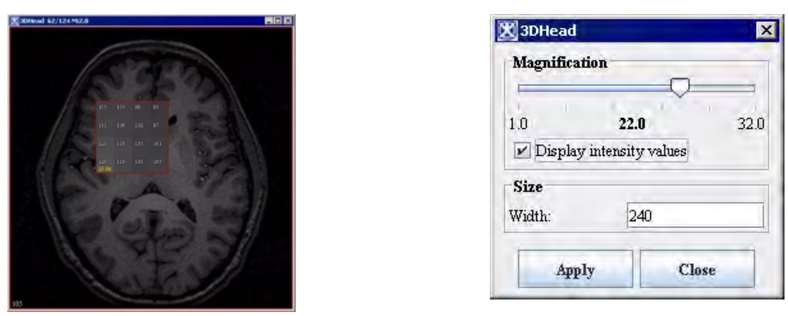


Figure 24. An image window displaying intensity values for a magnified region in the image

Tip: If you hold down the middle mouse button, MIPAV writes the intensities by position within the image to the Data page of the Output window. You can then save these messages and print them.

Improving contrast, adding color, etc.

This section explains how to improve image contrast, add color to images, and create negative images by applying a *quick lookup table* (LUT), generating and modifying a histogram, applying pseudo-color LUTs, and creating negative images by inverting their color.

histogram— is a representation of a frequency distribution by means of rectangles whose widths represent class intervals and whose areas are proportional to the corresponding frequencies.

lookup table (LUT)— maps the frequency distribution in a histogram to pseudo-color values.

Improving contrast on images quickly

MIPAV provides the following quick ways to improve the contrast on images:

- Using the Quick LUT icon
- Using the right mouse button
- Using the Adjust Window and Level icon

Using the Quick LUT icon

 An easy way to improve the contrast in an image is by using the Quick LUT icon, which is located on the image toolbar in the expanded MIPAV window.

To use the Quick LUT icon

1. Open an image file. The image appears in an image window. If an image was not previously open, the initial MIPAV window expands to include all of the menus.
2. Select the image window.
3. Click Quick LUT.
4. Hold down the left mouse key and draw a rectangle on a portion of the image.
5. Release the left mouse key. Based on the amount of dark and light in the portion of the image that you selected, MIPAV changes the contrast in the image.

   Repeat steps 3 through 5 as often as you wish to further improve image contrast. When you are satisfied with the contrast, save the image.

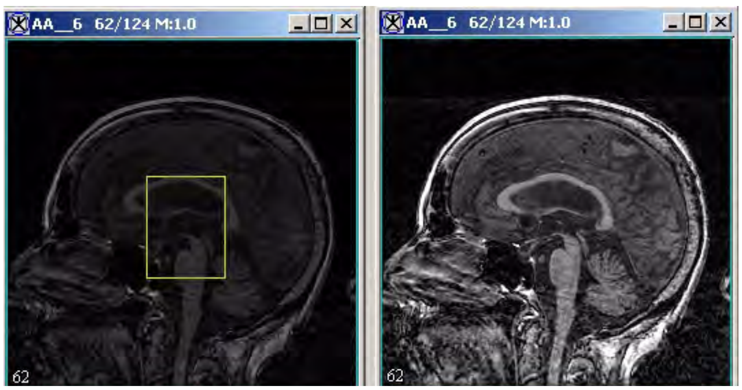


Figure 25. Images before and after applying the Quick LUT icon

USING THE RIGHT MOUSE BUTTON

The right mouse button provides you with a very simple way of changing image contrast. To do so, open an image and then hold down the right mouse button and drag it around the screen. The cursor changes from a red cross to one that is shown in Figure 26. To restore an image to its original appearance, refer to "Restoring images to their original appearance" on page 85.

Figure 26. Changing image contrast using the right mouse button

When you drag the cursor up and down or across the image, the image may become darker and gradually disappear or become lighter in appearance.

At some points you may be able to create a negative of the image.

Adjust Window and Level

The Adjust Window and Level icon on the Image toolbar provides another way to change the contrast of images.

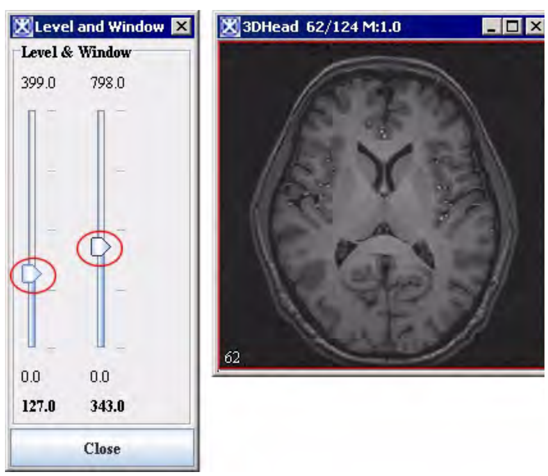


Figure 27. The Level and Window dialog box

Move each slider up or down to change the contrast of the image. The changes are immediately effective in the image.

To adjust image contrast

1 Open an image file. The image appears in an image window.

2 Click Adjust Window and Level. The Level & Window dialog box appears.

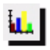
3 In the dialog box, move each slider up or down to change the contrast of the image. The changes are immediately effective in the image. Click Close when done.

RESTORING IMAGES TO THEIR ORIGINAL APPEARANCE

 To return to the original appearance of the image, click Reset LUT.

 In addition, you can click the Gray icon to restore the image to grayscale if you have changed it or added colors.

Improving contrast by generating and modifying histograms

 To generate a histogram of an image, you can use the Lookup Table icon or call LUT > Histogram LUT. To obtain a histogram summary, which is in the tabular form, use the Algorithms > Histogram Tools > Histogram Summary command.

To generate a histogram for an image

1 Open an image. The image appears in an image window.

2 Do either of the following: click Lookup Table icon or select LUT > Histogram–LUT. For images that do not contain VOIs, the Lookup Table window opens.

3 For images that contain VOIs, the Histogram dialog box appears. In the dialog box, select either Whole image or VOI region(s).

4 Click OK. A progress message appears. After a few moments, the Lookup Table window (Figure 29) opens.

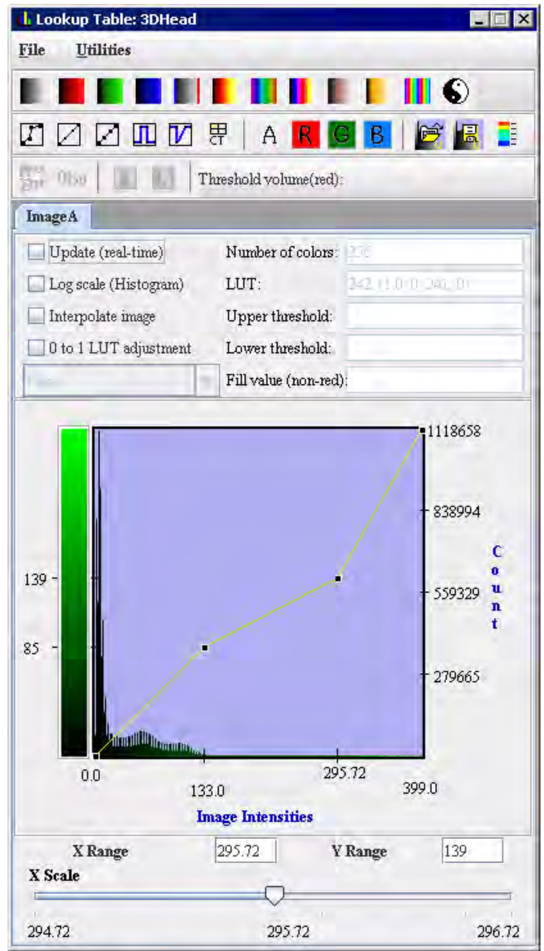
A B

Figure 28. The Lookup Table window (A) and Histogram dialog box (B)

Definition: Lookup table (LUT) indicates the intensity of each voxel in the image and, in MIPAV, allows you to remap the original intensities to other intensities.

transfer function reflects the relationship between the original image intensity values and how they are mapped into the LUT. The line in the LUT represents the transfer function.

 To change back to the original grayscale intensities, click Gray LUT in the MIPAV window.



File	*Open LUT*—Opens a previously saved LUT file. LUT files have an*.LUT* extension. *Save LUT*—Saves the LUT displayed in this window in a LUT file. *Open user defined LUT*— opens a file with the user defined LUT. *Save user defined LUT*— saves a user defined LUT. *Open Transfer Functions*—Opens a previously saved transfer function. Transfer function files have a *.FUN* extension. *Save Transfer Functions*—Saves the transfer function displayed in this window to a file. *Close LUT*—Closes the LUT window.
Utilities	*Change number of colors*—Allows you to change the number of colors displayed in the image. Valid values are 2 to 256. *CT function*—Allows you to select a preset LUT that is appropriate for the image content. Values are abdomen, head, lung, mediastinum, spine, and vertebrae. *Invert LUT*—Creates a negative of the image. *Reset transfer function*—resets the chosen LUT back to Gray LUT. *Reset histogram and LUT A*—Returns image A to its original values. *Reset histogram and LUT B*—Returns image B to its original values. This command is only available if two images are open.
LUT toolbar	– Gray LUT; – Red LUT; – Green LUT; – Blue LUT; – Cool-Hot LUT; – Gray/Blue/Red LUT; – Hot Metal LUT; – Spectrum LUT; – Skin LUT; – Bone LUT; – Stripped LUT; – Invert LUT.

Figure 29. Lookup Table dialog box

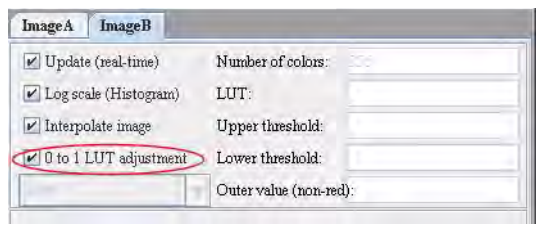
Functions toolbar	– Transfer function; – Reset Transfer function; – Even Distributed Transfer function; – Dual Trashed function; – Dual Inverse Trashed function; – CT preset function; – Edit Alpha function; – Edit Red LUT; – Edit Green function; – Edit Blue function; – Open user Defined LUT; – Save User Defined LUT; – Generate LUT.
Update (real-time)	Changes the image as you make changes to the LUT, which allows you to see the effect of your changes immediately on the image.
Log scale (histogram)	Displays the image's histogram count in log scale along the *Y* axis.
Interpolate image	Displays image using interpolation, which reduces pixilated image to appear more smooth. **Caution:** Depending on the memory resources of your workstation, interpolation can be very lengthy.
0 to 1 LUT adjustment	Appears only for in the ImageB tab when two images are loaded in the same image frame.It processes the image for contrast as follows: for contrast <0, it uses a linear function to calculate the entire contrast look-up-table; for contrast 0, the contrast look-up-table is an identity table, and for contrast greater than 0, a linear function is used to calculate the linear region and for regions near **min** and **max**, the nonlinear function $y=x^n$ is used. . 
Number of colors	Allows you to change the number of colors displayed in the image.
LUT	Displays the image intensities.
Upper threshold **Lower threshold** **Fill value**	The threshold options is a tool for setting thresholds for a gray level of the image in order to segment the image into an object and a background. This tool does applys a pceudo color (spesified in the Fill Value parameter) to the image, to show the pixels selected by the threshold setting, that is, those pixels with values equal to and between the thresholds.

Figure 29. Lookup Table dialog box (continued)

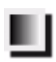
Applying color to images using predefined LUTs

MIPAV provides a variety of pseudo-color LUTs. When a pseudo-color LUT is applied to an image, the grayscale intensities are remapped to the pseudo-color intensity values.

> **Using interpolation to smooth images**
>
> If you are zooming in on a portion of an image and want to reduce the appearance of pixilations in the image, make sure to select the Interpolate image check box. Interpolation smooths the pixilation.

To apply a pseudo-color LUT

1. Open an image.
2. Select the image window.
3. Click Displays Lookup Table icon to call the Lookup Table dialog box.
4. In the dialog box that appears, select one of the LUTs on the LUT toolbar.
5. Check the Update (real-time) box.
6. MIPAV immediately applies the LUT to the image.

 To change back to the original grayscale intensities, click Gray LUT in the MIPAV window.

Creating negatives of images

 The invert LUT icon creates a negative of an image.

To create a negative image

1. Select the image window of the image you want to invert.
2. Click Invert LUT.

Based on the 256-step color values scale, MIPAV assigns the inverse value to each pixel of the image (refer to Figure 30).

Tips: You may wish to apply a LUT to the image or adjust the histogram of the image first before applying creating the negative.The Invert LUT icon is a toggle. To change the image back to its previous appearance, simply select Invert LUT again. To change the image back to its *original* appearance, select Reset LUT.

Note: Invert LUT appears on two different windows: in the Image toolbar in the MIPAV window (select Toolbars > Image toolbar to display the toolbar) and in the LUT toolbar in the Lookup Table window.

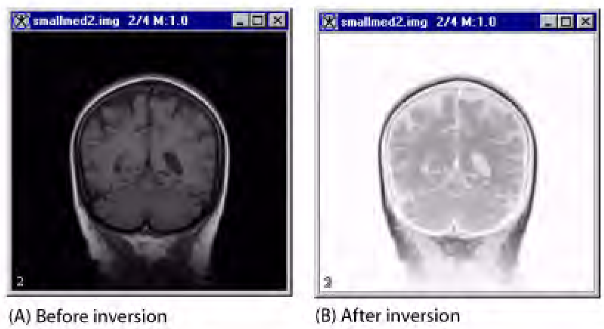


Figure 30. Image before and after inversion

Comparing images using alphablending

Alphablending is a technique that adds transparency information to translucent objects. When two images share the same window such as when you *loading the image*, you can adjust the alphablending settings so that you can see a blend of both images and can compare overlapping regions in two datasets. See also "Loading image files" on page 69.

To use the alphablending function

1. Open the first image, then load the second image in the same window.
2. Adjust the alphablending slider at the bottom of the MIPAV window.
3. Move the slider to select the best ratio for the datasets of interest.

The level of translucency for one image is inversely proportional to the other. Thus, if image A is 75 percent transparent (25 percent opaque), then image B is 75 percent opaque (25 percent transparent).

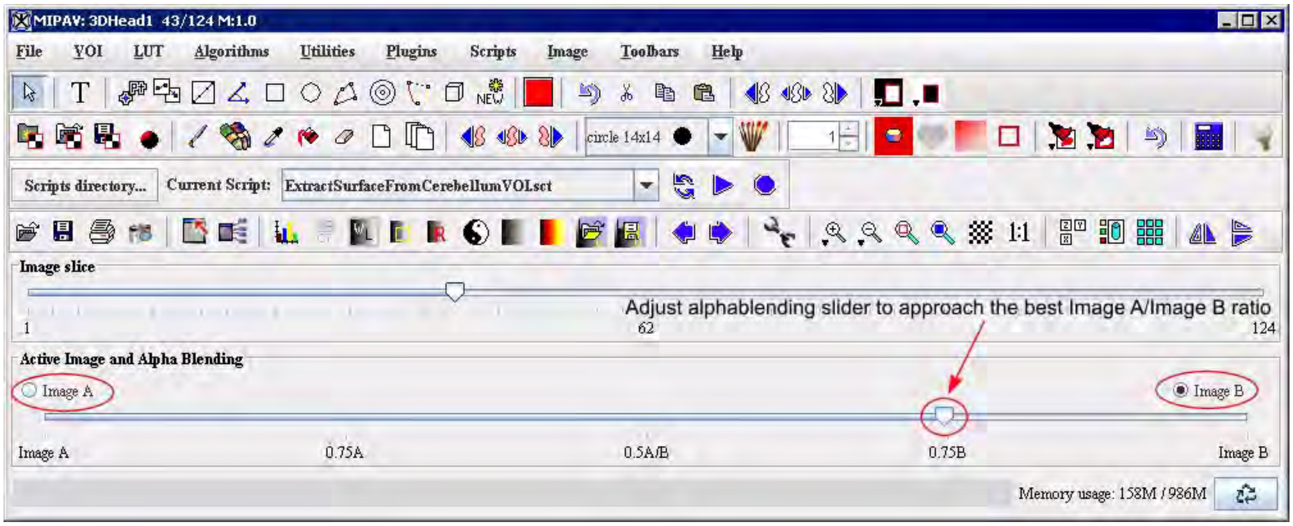


Figure 31. MIPAV window showing the alphablending slider at the bottom of the window

Creating new images

To create a new image file

1 Do one of the following:

- If an image file is not already open, select File > Open Image(A) > Create Blank Image.
- If an image file is already open, select File > Load Image (B)> Create Blank Image. The Raw dialog box (Figure 32) appears.

2 In the dialog box, select the image type.

3 Select the units of measure for each dimension.

4 Enter the header offset and byte ordering information.

5 Click OK. A blank image (Figure 32) appears in an image window.

6 Use the paint and VOI tools to create an image.

7 Click File > Save image as. The Save dialog box opens.

8 Type the name of the file in File Name, and select the file type in Files of type.

9 Click OK. MIPAV saves the image under the file type you selected.

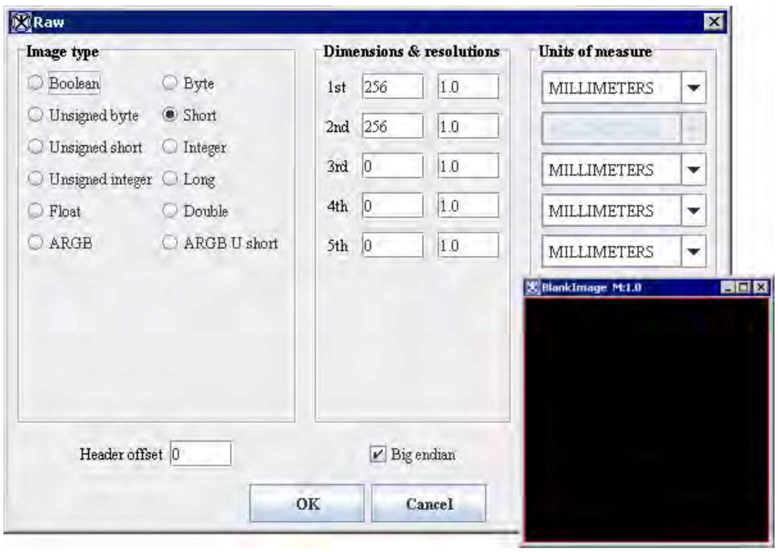


Image type	Synonymous with data type. The image type determines the number of intensities that can be represented in an image. For example, a Boolean image can display two intensities: 1 and 0. • *Boolean*—1 bit per pixel (1 on, 0 off) • *Unsigned byte*—1 byte per pixel (0, 255) • *Unsigned short*—2 bytes per pixel (0, 65535) • *Unsigned integer*—4 bytes per pixel (0, 2^{32} -1) • *Float*—4 bytes per pixel (-3.4E38, 3.4E38) • *ARGB*—3 bytes per pixel, plus 1 byte; 8 bits per color channel (alpha, red, green, and blue) • *Byte*—1 byte per pixel (-128, 127) • *Short*—2 bytes per pixel (-32768, 32767) • *Integer*—4 bytes per pixel (-2^{31}, 2^{31}-1) • *Long*—8 bytes per pixel (-9.22E18, 9.22E18) • *Double*—8 bytes per pixel (-1.8E308, 1.8E308) • *ARGB U short*—2 bytes per color channel and 2 bytes for alpha channel

Figure 32. Raw dialog box

Dimensions and resolutions	Degree of manifolding of a quantity such as space or time. Two-dimensional datasets are composed of one image (the two dimensions are length and width). Generally, three-dimensional datasets are composed of more than one image. The third dimension is generally space. The fourth dimension is generally time. (Either time or space can be the third or fourth dimensions.) Zeros in the text boxes indicate that the dimension is not represented in the image. For example, if the text boxes for the first and second dimensions are filled, and the rest of the text boxes are filled with a zero, the image only has two dimensions. *Dimensions* • *1st*—Width (along *x* axis) • *2nd*—Length (along *y* axis) • *3rd*—Depth (along *z* axis) • *4th*—Time (along *t* axis) • *5th*—Fifth dimension *Resolutions* Size of pixel or voxel per dimensions 1 through 5.
Units of measure	Indicates the unit of measurement for each of the applicable dimensions.
Header offset	Indicates the size of the space reserved at the beginning of the file where specific types of information is kept. This space, which is called the *header,* precedes the image data. If you know the length of the header, type it in this box. When MIPAV accesses the file, it skips the header offset and begins to read the image data. Note that not all image file formats have a header.
Big endian	Indicates whether image data is stored in the big endian format. If not, the image data is stored in the little endian format. *Endianess* refers to the byte ordering of the data. Some computers order the data with the least significant byte (LSB) first followed by the most significant byte (MSB). This byte order is referred as *little endian* or Intel byte ordering. Machines that use little-endian byte ordering are VAXes, Intel x86, and Pentium. The reverse is MSB and then LSB, which is referred as *big endian* or Motorola byte ordering. Machines that use big-endian byte ordering are IBM System 3D, RISC, and a Motorola 680x0. MIPAV is biendian; it supports both big- and little-endian byte-ordering formats.
OK	Applies the parameters that you specified and creates a blank image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not create a blank image.
Help	Displays online help for this dialog box.TBD.

Figure 32. Raw dialog box

Delineating volumes of interest (VOIs)

MIPAV provides tools that allow you to automatically, semiautomatically, and manually identify and modify volumes of interest (VOIs).

volume of interest—the portion of the image in the dataset on which you want to focus. It may be either one slice or multiple slices throughout the dataset.

Each VOI can be formed from multiple contours in a single slice or multiple slices. Once an object is segmented and defined by a VOI, statistics of the volume can be calculated.

MIPAV supports over 32,000 unique VOIs on a single dataset. Additionally, you can move or delete nodes on the VOI and add new points. VOI types include:

- Point, which is created by using
- 2D line
- Rectangular
- 3D rectangular
- 2D elliptical
- Polygonal
- 3D polygonal
- Interactive level-set

Generating contour VOIs using predefined shapes

MIPAV includes icons of predefined shapes, such as points, lines, ellipses, or rectangles, that you can use to create VOIs. These icons are on the VOI toolbar.

To generate contour VOIs using the predefined shapes

1 Select one of the contour icons from the VOI toolbar in the expanded MIPAV window. See also Figure 15.

2 Move the pointer to the image window. The pointer changes to a cross-hair shape. Do one of the following:

- **Points, levelset:** Position the cursor on the area where the point or levelset should be drawn. Click the mouse button.
- **Straight lines, rectangles (2D and 3D), ellipsoids:** Position the cursor on the area where the contour should begin.

  Click the mouse button.

  While holding down the mouse button, drag the cursor until the contour is the desired size.
- **Polylines, polygons:** Position the cursor over the area where the contour should begin.

  Click the mouse button. A point appears. Alternate between moving the mouse and selecting the mouse button to outline the VOI.

  To complete a polyline, double-click the mouse button. To complete a polygon, connect the first and last nodes.

Tip: To draw the same shape several times in succession, hold down the Shift key while you select the applicable icon from the VOI toolbar and draw.

Adding and moving boundary points on VOIs

No matter which method chosen to delineate a VOI, you can add points and change the boundaries of the VOI. For example, suppose you created a VOI on an image with the Rectangle VOI icon. If you notice that a part of the image that should be included in the VOI is not included, the boundary of the rectangle needs to be adjusted to include the missing portion of the image.

To add a point and change the boundary of the VOI

1. Select the VOI. Notice that white points appear at the corners of the VOI and a small cross appears in the middle of the VOI.
2. Place the cursor on the portion of the VOI you want to adjust. The cursor changes from a cross to.
3. Click once. A white point appears on the line.
4. Select the point and drag it to include the missing part of the image.

Tip: Circular VOIs are composed of a continuous series of points around the diameter of the circle. You only need to select one of those points and drag it to enlarge the circle.

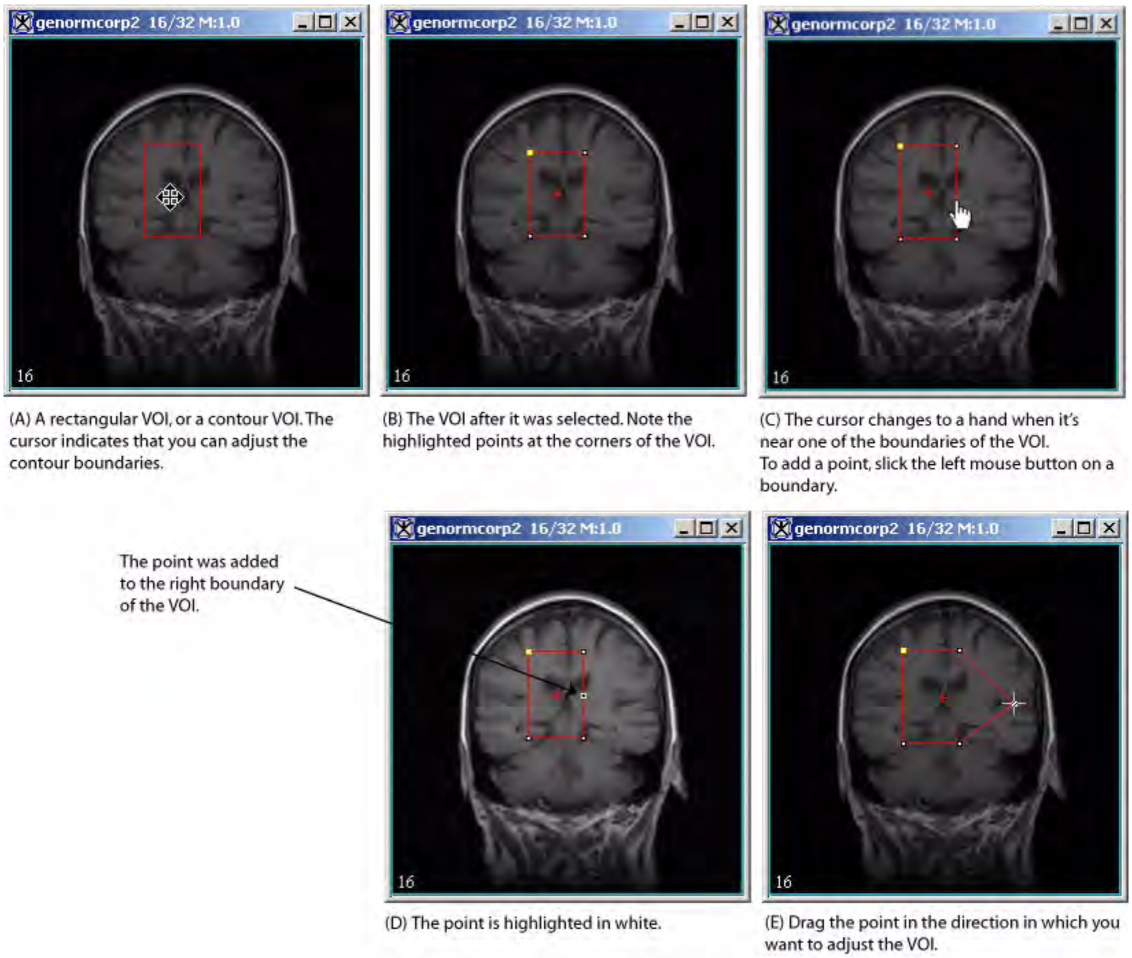


Figure 33. Adding a point and adjusting the boundary of a rectangular VOI

Automatically adjusting contour boundaries

After a contour is drawn, it might be necessary to adjust the boundaries so it more closely matches the VOI region.

To adjust contours

1. Click a contour. The nodes become visible.
2. Select VOI > Evolve boundary 2D >Active Contour. The Evolve Boundary dialog box opens.
3. Modify the information in the window if necessary.
4. Click OK.

   A new contour, that more closely outlines the VOI, appears on the image. The old contour also remains.

 To delete an old contour, select the contour, and then select Cut Selected Contour, or press the Del key on the keyboard.

Scale of the Gaussian	Enter values for X,Y, and Z directions which will be used to correct blurring. The default value is 2.0
Resolution options	If this box is checked, the algorithm uses the image resolution to normalize the Z scale.
Evolve Boundary	Single slice – the VOI boundary will be evolved only for the current slice; Propagate to Adjacent Slices – the VOI will be propagated to adjacent slices; Replace Original Contour – the original VOI will be replaced with the new one.

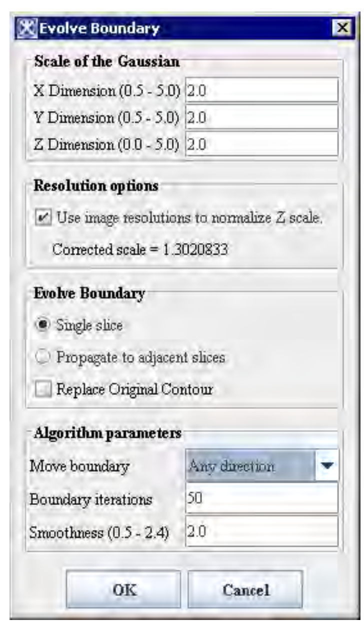


Figure 34. Evolve Boundary dialog box

Algorithm parameters	Move Boundary – depending on the selected option, the VOI will be moved in 1) any direction, 2) only inward, 3) only outward. Boundary Iterations– specify the number of iterations needed to calculate the new boundary. Smoothness– enter the number from 0.5 to 2.4 to specify smoothness.
OK	Applies the algorithm according to the specifications in this dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes it.
Help	TBD.

Figure 34. Evolve Boundary dialog box (continued)

Annotating images with text

 Another icon on the VOI toolbar is the Annotation tool icon, which you can use to place text directly on an image at any position on the image. You can use any font family, or typeface, that is installed on your computer to display the text.

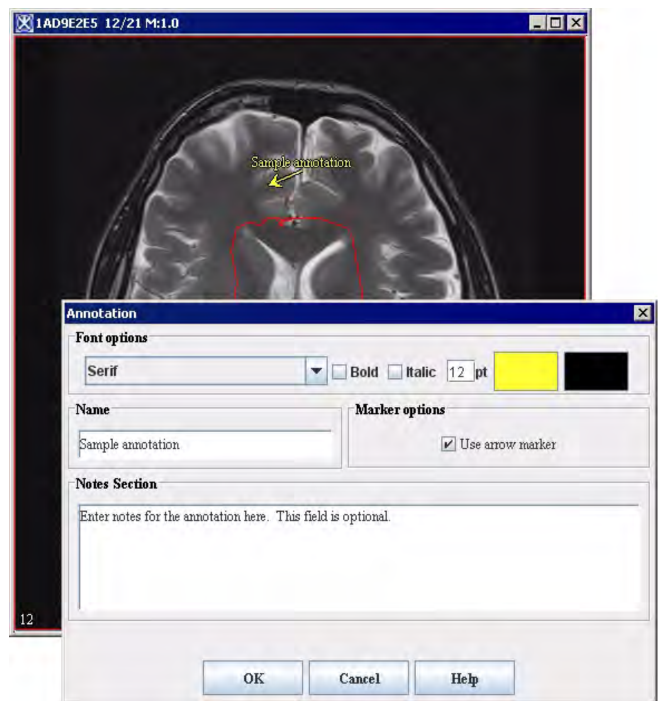
To annotate images with text

1. Open an image.
2. Click the Annotation tool icon. The Annotation dialog box (Figure 35 on page 99) opens.
3. Click the image on which you want the text to appear. The image becomes the active image.
4. Select the font family and style (bold or italic). To select a regular font style, simply specify the font family.
5. Type the font size of type in the **pt.** box.
6. Press Enter. The Annotation dialog box either enlarges or reduces in size depending on the type size you chose (see "Previewing Font Selection" on page 100 below).
7. Click Color to select the color of the text. The color of the type in the Text box changes to the color you have chosen.

8 Select "Enter text here" and type the text that you want to display on the image in the Text box.

9 Click OK. The text appears near the top of the image.

10 Click the text and drag it to any position on the image.

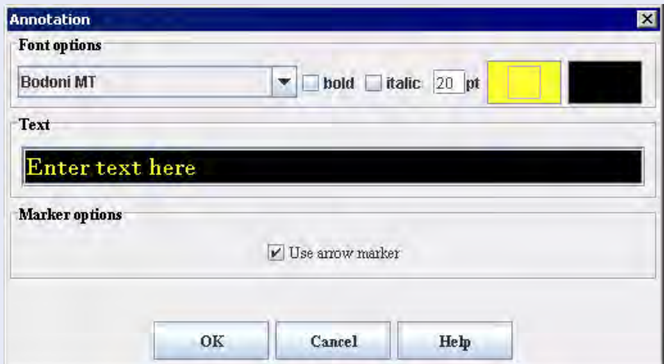


Font options	**Font**—Specifies the font family, or typeface, for the text. When you select another typeface, "Enter text here" appears in that typeface.
	Bold—Specifies that the type style should be bold.
	Italic—Specifies that the type style should be italic.
	Pt. (point) size—Specifies the point size of the typeface.
	Color—Specifies the color of the text. When you click this box, the Pick VOI Color dialog box opens.
Text	Displays the text that should appear on the image.
Marker options	The arrow marker appears if you check the Use Arrow Marker box.
OK	Applies the parameters that you specified and places the text on the image, which you can move to any position on the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not place text on the image.
Help	Displays online help for this dialog box.

Figure 35. Annotation dialog box

Previewing Font Selection

The Annotation dialog box allows you to preview the font family, style, size, and color of the text before you click OK and add the text to the image.



To edit the text

1. Double-click the added text on an image. The Annotation dialog box opens.
2. Make any changes you want to the text, font family, font style, and color.
3. Click OK. The changes you made to the text should appear on the image.

To delete the text

1. Select the added text on an image.
2. Press Del. MIPAV removes the text from the image.

To restore the text to the image, select Edit > Undo VOI. The text reappears on the image.

Calculating VOI statistics

Once a VOI is drawn, you can calculate the number of voxels in the VOI, the volume, and area. You can also calculate the average and standard deviation of the voxel intensity and the center of mass. For 2D images, you can calculate the principal axis and the eccentricity. MIPAV provides two methods for you to obtain VOI statistics:

- *By using VOI properties*—This method is fast and simple and provides statistics for the entire VOI. Although it allows you to save the results in a text file, that is an additional step. However, using this method, you can type additional information directly onto the Data page with the statistics.
- *By using the Statistics Generator*—Using this method, you can obtain statistics on the entire VOI, on a single slice of the VOI, or by contour and slice. The Statistics Generator also automatically saves the results in either a tab-delimited file or an XML file of your choosing. In addition, it displays statistics in a tabular format.

Note: MIPAV can calculate statistics for only one VOI at a time. In addition, MIPAV calculates the volume and area in a VOI using image pixel (voxel) resolutions.

Using VOI properties

To calculate VOI statistics

1. Select the desired VOI in the image window.
2. Select VOI > Properties in the MIPAV window. The VOI Statistics dialog box appears.
3. Select the statistics to be calculated in Statistics to Calculate. Click Calculate. The VOI Statistics dialog box remains on the desktop. In a few moments, statistical data appears on the Data page in the Output window.
4. To close the dialog box, click Cancel in the VOI Statistics dialog box when complete.
5. To add information to the statistics, just type it in into the statistics on the Data page.
6. To remove the data, select them first, and then click the Cut icon or select Edit > Cut to cut the selected data. MIPAV removes the selected text from the Output window and copies it to the clipboard for use in other applications.

7 Select the data that you want to copy, and then click Copy or select Edit > Copy to copy the data to another location in the window or to another application (such as a word processor).

8 Click Save or select File > Save messages to save the data and any comments you've added to a text file.

9 Click Clear Messages or select Edit > Clear Messages to clear the window.

VOI Properties	
Name of VOI	Shows the name of VOI.
Thickness of VOI	Shows the number which represents how many slices has a selected VOI.
Color of VOI	Shows the color that was used to outline the VOI.
Show contour bounding box	Highlights the VOI and, when you select the VOI, displays a box that encompasses all of the VOI's borders and lists the measurements and position of each boundary.
Use additive polarity for VOI	If selected, put VOI names, such as "polygon1" to indicate VOI location.
Include for processing	This option is very helpful if you have a set of VOIs delineated on your image and want to use some of them for masking. See "Using the Include for Processing option" .
Show VOI name	If selected, shows the name of the chosen VOI. E.g. "polygon1."

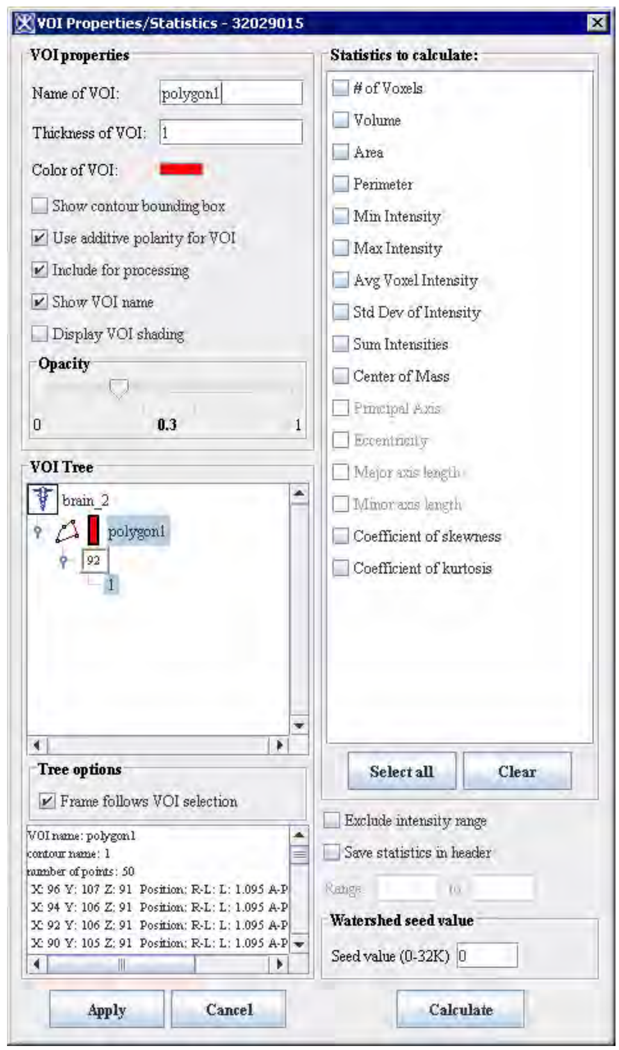


Figure 36. VOI Statistics dialog box

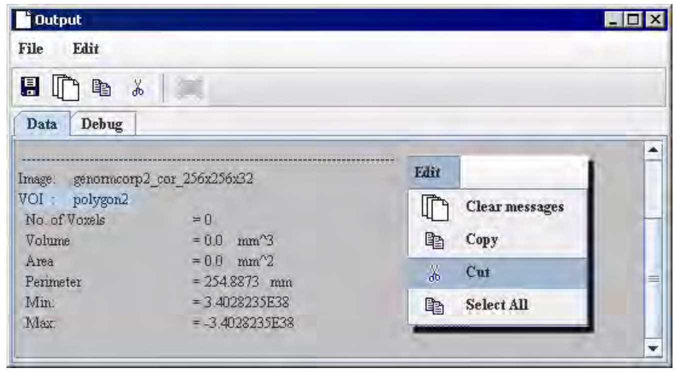
Display VOI shading	Shades the area inside a chosen VOI. When this option is selected, the **Opacity slider** becomes also available, so you can regulate the opacity of the shaded area.	
Statistics to calculate		
	Provides a list of statistics. Select the statistics that you want to include in the report.	
Select all	Selects all of the statistics listed in the Statistics to calculate list.	
Clear	Clears all of the check boxes that you selected in the Statistics to calculate list.	
VOI Tree	Displays all VOIs delineated on the image in an hierarchical view.	
Tree options	Frame follows VOI selection - if this option is selected, the current selected VOI appears in a frame in the VOI tree.	
Exclude intensity range		Allows you to select specific intensity ranges that you want to exclude from the calculation.
Watershed seed value (0-32K)		Indicates the basin value used when running the Watershed algorithm on images.
Apply	Applies the changes you made in this dialog box and leaves the dialog box open for you to make further changes.	
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not run a statistics report.	
Calculate	Calculates the statistics requested in this dialog box and displays them in the Data page of the Output window.	

Figure 36. VOI Statistics dialog box (continued)

Using the Include for Processing option

The Include for Processing option is very helpful if you have a set of VOIs delineated on your image and want to use some of them for masking. In that case, select the VOI(s) of interest, then activate the Include for Processing option, and after that, call one of the mask tools, e.g. the Quick AND VOI mask operation. The tool will apply on the selected VOI(s). In case when the Include for Processing option has not been activated, then the Quick AND VOI mask operation will apply on the whole image and darken it. See Figure 37 (a) and (b).

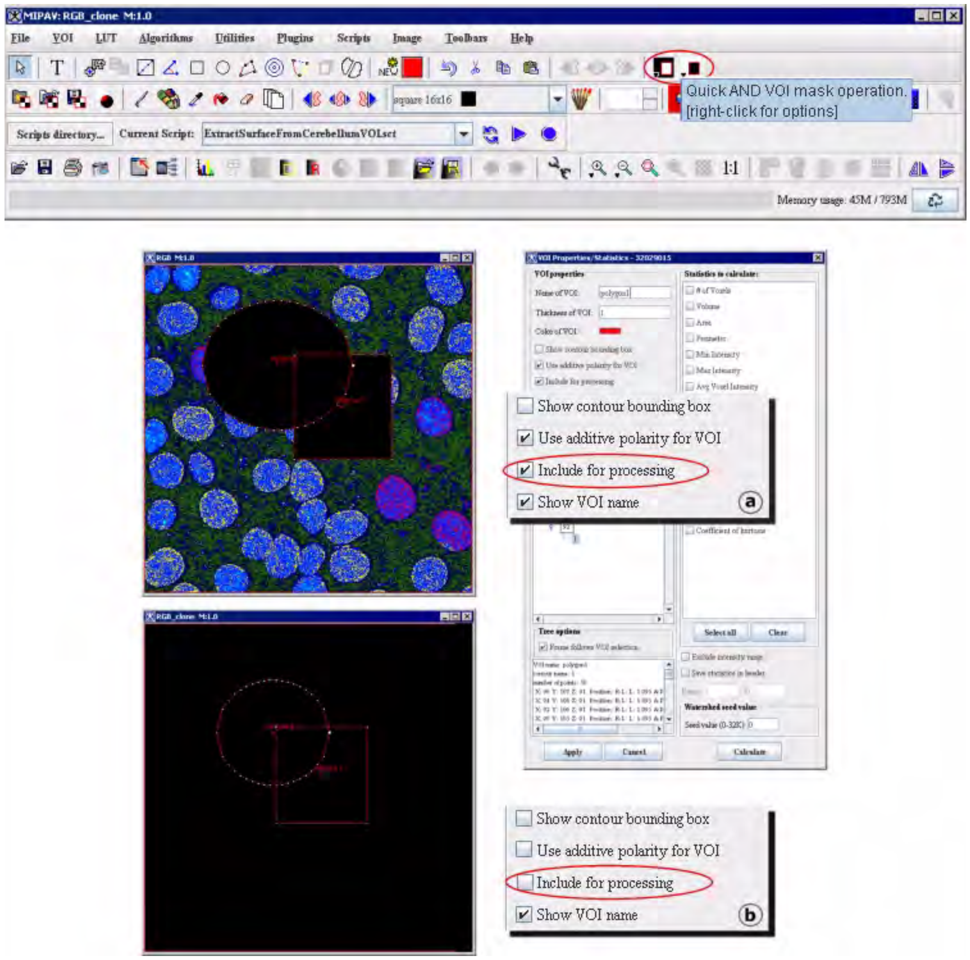


Figure 37. Using the Include for processing option for the Quick AND VOI mask operation. (a) The Include for Processing option is activated and (b) the Include for Processing option is disabled.

Using the Statistics Generator

If you plan either to use a database or spreadsheet to keep track of VOI statistics or to obtain them in an XML format, use the Statistics Generator. As mentioned earlier, the Statistics Generator can provide statistics on an entire VOI(s) or a particular slice or by contour and slice.

Using the Statistics Generator involves three tasks:

- Selecting VOI and save options, refer to page 105
- Selecting statistics options, see page 106

- Reviewing the statistics, see page 107

SELECTING VOI AND SAVE OPTIONS

The first task is to select the VOIs on which you want to obtain statistics and select the file in which the resulting statistics should be saved.

To Calculate Statistics on VOI Groups

1 Delineate or select the VOI(s) on the image.

2 Select VOI > Statistics Generator in the MIPAV window. The Calculate Statistics on VOI Groups window appears (Figure 38). This window displays all of the VOIs on the image in the VOI group list on the left.

Figure 38. Calculate Statistics on VOI Groups window

3 Select the VOI on which you want to obtain statistics.

4 Press Send Selection Right. The name of the VOI appears in the VOI group list on the right.

5 Use Browse to choose a file name in the VOI Statistic File Destination box.

6 Select either one of the two formats in the Output Format group: Tab delimited or XML.

7 Click Statistics Options. The Statistics Options page opens.

8 Proceed to the next task: Selecting statistics options.

SELECTING STATISTICS OPTIONS

The Statistics Options page (Figure 39) provides a list of statistics from which you can select and the options to obtain statistics by slice, by contour and slice, or by the total VOI.

To select statistics to perform on VOIs

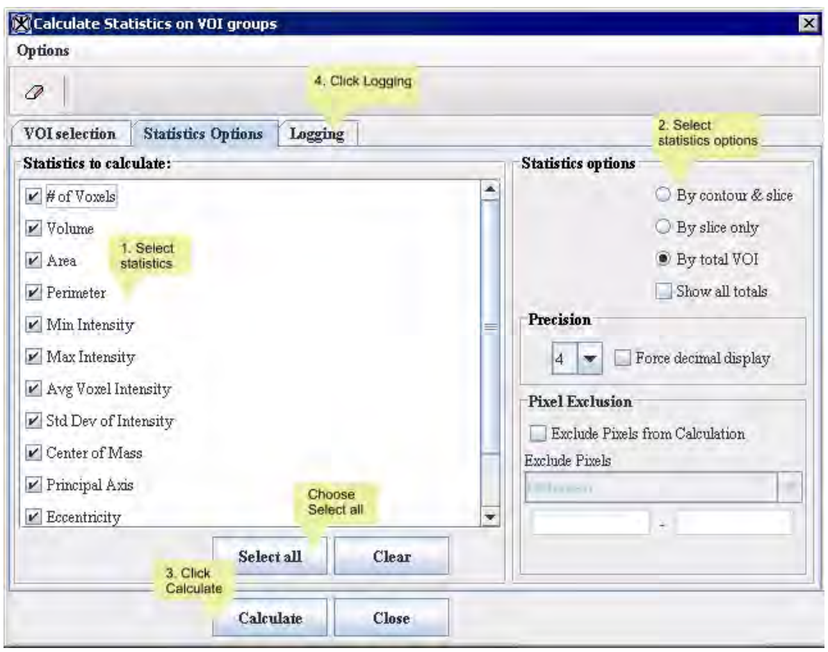


Figure 39. Statistics Options page

1 Do either of the following in the Statistics to calculate group:

- Select one or more of the listed types of statistics.
- Click Select all to obtain all of the statistic types.

2 Select one of the following options in the Statistics options group:

- By contour & slice

- By slice only
- By total VOI (the default selection)

3 Select Show all totals if you want to record the totals for each type of statistic.

4 Click Calculate.

5 Select Logging. The Logging page (Figure 40) appears.

6 Proceed to the next task: "Reviewing the statistics".

REVIEWING THE STATISTICS

The Logging page (Figure 40) displays the statistics in tabular form. Whether or not you chose a type of statistic on the Statistics Options page, the table includes a heading for each type. Blanks cells in the table indicate that you did not choose to obtain that particular type of statistics.

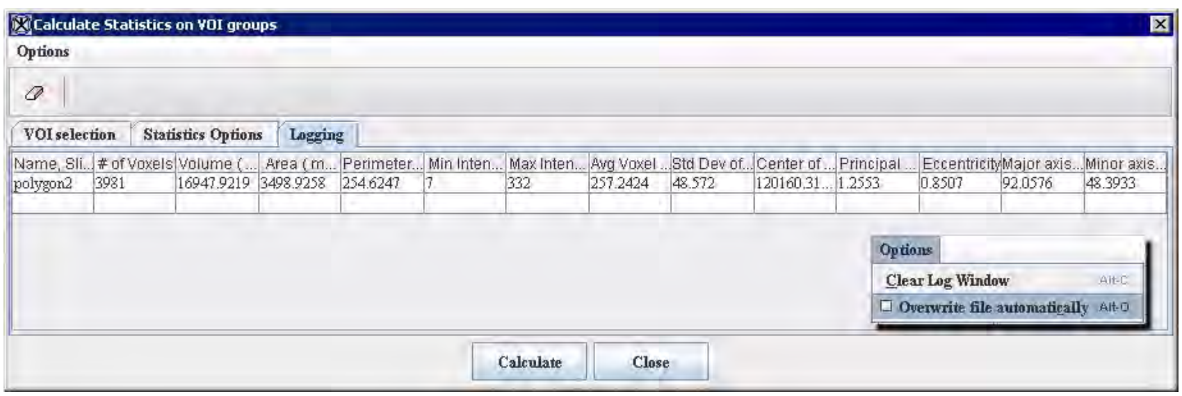


Figure 40. The Logging page in the Calculate Statistics on VOI Groups window

Each time you calculate the statistics for a VOI the Statistics Generator adds another row of statistics to the table. Note that the first column in the table lists the name of the VOI and, if appropriate, the slice and contour numbers. Also, you can change the width of each of the columns in the table by dragging the line between the columns in the heading.

You can include and review the statistics file in a database or in a spreadsheet program by double-clicking on the file name in Windows Explorer window and, in the Open with dialog box, selecting the application in which you want to open the file.

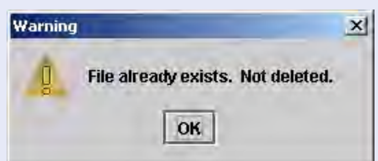
When the number of rows in the table exceeds the length of the Logging page, scroll bars appear on the right side of the table to allow you to scroll from the beginning or to the end of the table.

If at any time you want to clear, or erase, all of the rows of the table, select Options > Clear log window. The complete table disappears from the Logging page, which is now totally gray.

Overwriting statistics files

If you previously ran the Statistics Generator and obtained statistics, after you click Calculate a message appears stating that a statistics file already exists. It asks whether to overwrite the file or to cancel the action.

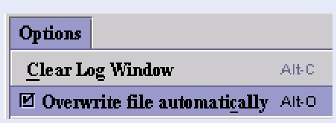
If you not want to overwrite the file, click Cancel. The following warning message appears.



Return to the VOI selection page and choose another file name in the VOI statistic file destination box. Then click Calculate to obtain the new set of statistics. The Statistics Generator calculates the statistics and saves them in the file that you indicated. The statistics appear on the Logging page.

If it's all right to overwrite the file, click Overwrite. The Statistics Generator calculates the statistics and overwrites the previously recorded statistics file.

Tip: If you always want the Statistics Generator to overwrite the file, either select Options > Overwrite file automatically or press Alt+O.



Go to the next task: "Reviewing the statistics".

Modifying image resolutions

To modify the image resolution

1 Open an image.

2 Select Image > Attributes > Edit attributes in the MIPAV window. The Image Attributes dialog box opens. SeeFigure 41.

3 Click Resolutions. The Resolution page appears.

4 Modify the resolutions.

5 Click Apply.

6 Click OK or Close when complete. The window closes.

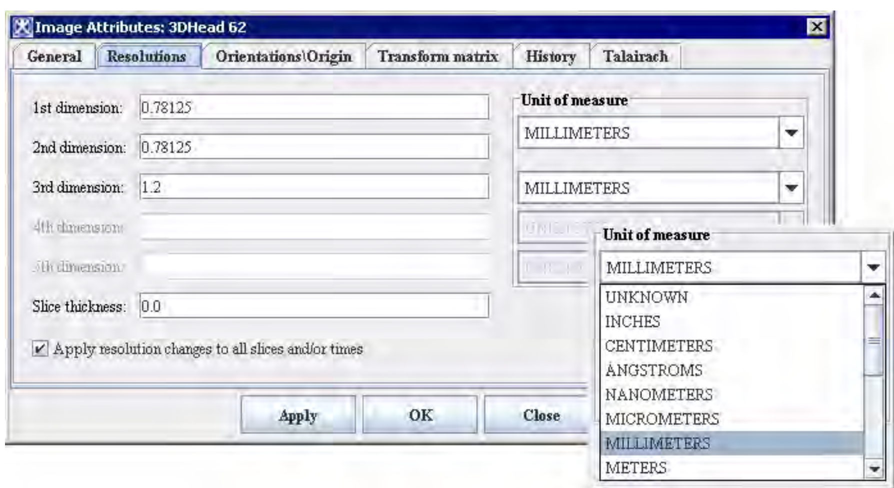


Figure 41. The Resolutions tab of the Image Attributes dialog box; the Unit of Resolutions list box shows different units available in MIPAV

Generating graphs (intensity profiles)

MIPAV can generate a graph of the intensity values of a region bound by a VOI. You can then save this graph or *intensity profile*, to a file for future reference.

Generating new graphs

You can generate an intensity profile for any VOI.

To generate a graph of a VOI

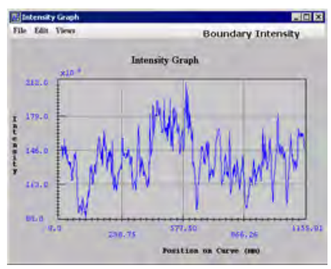
1 Select a VOI in an image window.

2 Use the right mouse button to call the context menu, then select one of the following:

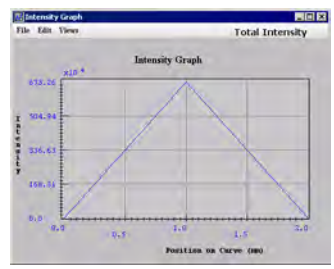
- Graph > Boundary Intensity
- Graph > Total Intensity
- Graph > Average Intensity
- Graph > Total Intensity with Threshold
- Graph > Average Intensity with Threshold

The chosen Intensity Graph window appears. Each function on the graph represents the intensity levels within each channel. Refer to Figure 42.

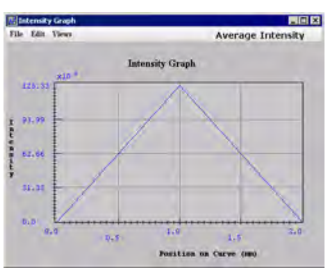
3 Close the graph by either selecting File > Close graph or pressing CTRL+X.



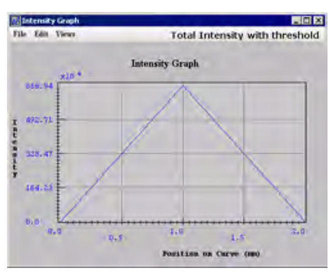
Boundary Intensity



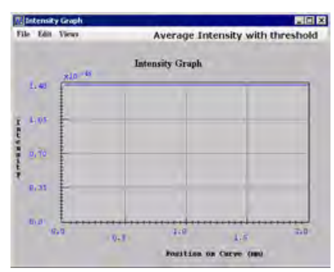
Total Intensity



Average Intensity



Total Intensity with Threshold



Average Intensity with Threshold

Figure 42. The Intensity graphs

SAVING GRAPHS TO A FILE

To save a graph

1 Select File > Save graph in the Intensity Graph window or press Ctrl S.

The Save dialog box appears. See Figure 43.

2 Type a name for the graph in File name.

3 Make sure that Graphs (*.plt*) appears in Files of type. Refer to Figure 43.

4 Click Save. MIPAV saves the file under a specified name.

PRINTING GRAPHS

To print graphs

1 Select File > Print graph in the Intensity Graph window. The Print dialog box appears.

2 In the dialog box, select the printer and adjust the print options if necessary.

3 Click OK to print the graph or image on your default printer.

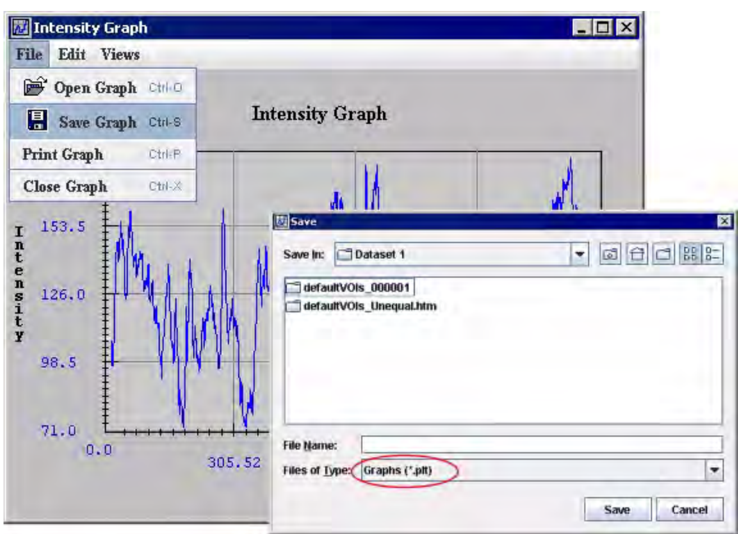


Figure 43. Saving and printing intensity graphs

Saving and printing images

This section explains how to save images to the same format (as when opened) or in a different format. It also shows how to print images. In addition, it explains how to save an image as a TIFF file.

Saving images to the same format

To save an image file in the same format (as when opened)

1 Do one of the following in the MIPAV window:

- Click the Save Image icon or use the Ctrl+S combination of keys.
- Select File > Save Image.

The Save dialog box appears (Figure 44).

2 In the dialog box, type the name of the file in File name and click Save.

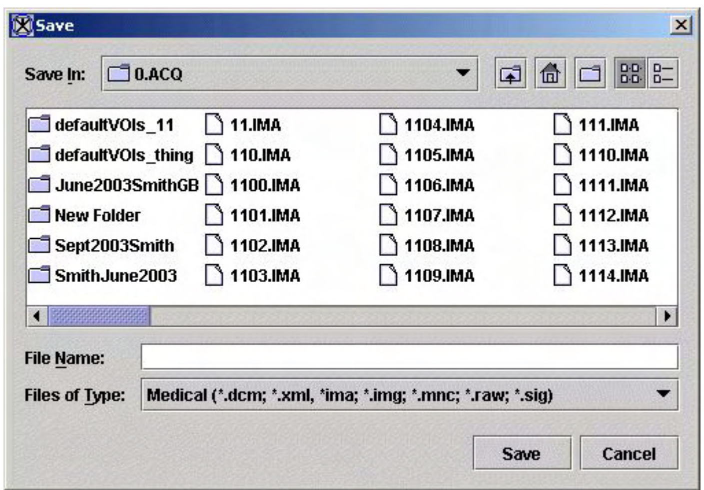


Figure 44. Save dialog box

Note: If you prefer for MIPAV to use the style of Open and Save dialog boxes that are used by the operating system (e.g., Microsoft Windows, Unix, or Apple) on your computer, read the section on "Using platform-specific Open and Save dialog boxes" on page 124.

Capturing images as TIFF(RGB) files

MIPAV allows you to capture entire images or portions of images as RGB TIFF files.

1 Select File > Capture image to TIFF(RGB) in the MIPAV window. The Capture Screen dialog box opens. See Figure 45.

2 To capture only a portion of the image:

- Select **Region**, then draw a rectangle with the mouse around the region in the image you want to save. Click OK. The Save dialog box opens. Type the name of the file in File Name, and select the TIFF type of file. Click Save. The region is saved under the specified file name.

3 To capture the entire image:

- Select **Window,** then click OK. The Save dialog box opens. Follow the instructions provided by the dialog box to save the image in TIFF format.

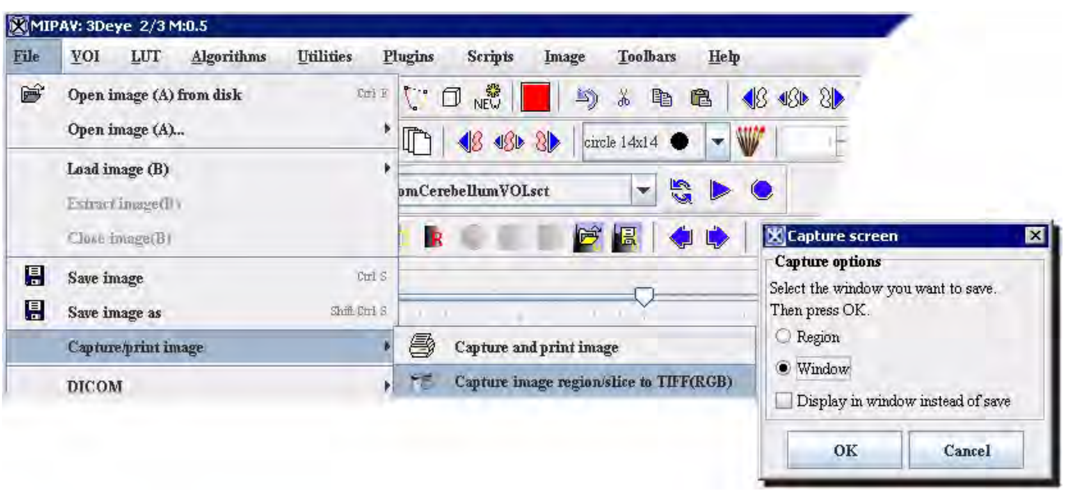


Figure 45. Capturing images or image regions

To display the region in a separate window rather than save it, select **Display in Window Instead of Save**. The selected region appears in a separate window.

To display the image in a separate window rather than save it, select **Display in Window Instead of Save**. The entire image appears in a separate window.

Saving images as

MIPAV uses the file extension to save image into various formats. Thus, saving an image with the extension of *TIFF* causes the image to be saved as a TIFF image. Saving an image with the extension of *IMG* causes the image to be saved as an Analyze image. See Table 1 for file extensions supported by MIPAV.

To save images as RGB TIFF files

1 Select File > Save as in the MIPAV window.

The Save Image as dialog box appears.

2 Type the name of the image in File name. Make sure you add *.tiff* as the extension.

3 Click Save. MIPAV saves the file as a TIFF file under the name you specified.

To save images to MINC format

1 Select File > Save As.The Save Image as dialog box appears. See Figure 44.

2 Type the name of the file in File name. Make sure you add *.mnc* as the extension.

3 The Attributes to Save dialog box appears. See Figure 47.

4 Complete the text boxes with the appropriate information. Note that the *x*, *y*, and *z* values differ from DICOM.

5 Click Save. MIPAV saves the image to MINC format.

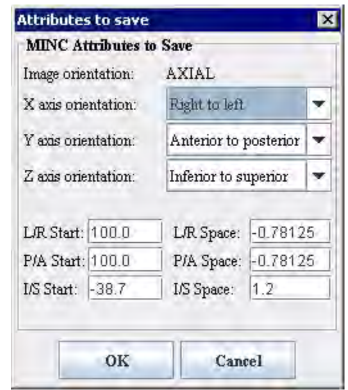
Image orientation	Specify the image orientation here if need. Note that, often the proper value is already entered in the dialog box. Then use the appropriate list boxes to specify the X, Y and Z axis orientation. L/R Start – TBD L/R Space – TBD P/A Start – TBD P/A Space – TBD I/S Start – TBD I/S Space – TBD 
OK	Saves the selected image based on your choices in this dialog box.
Cancel	Disregards changes you made in this dialog box and closes the dialog box.

Figure 46. MINC Attributes to Save dialog box

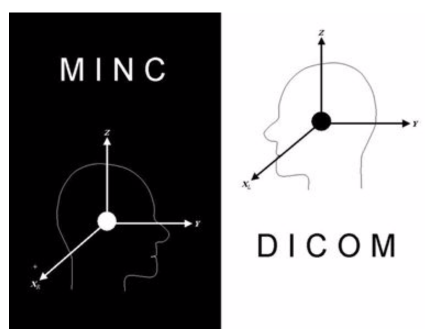


Figure 47. The MINC Attributes to Save dialog box and comparison of MINC and DICOM image orientation

SAVING IMAGES TO XML FORMAT

To record a history of actions that were performed on images, you should save the images to XML format. When you save images to XML format, MIPAV creates two files for each image: a RAW file, which contains image data, and an XML file. The XML file includes image attributes, such as action history. You can view the attributes of an image by selecting Image > Attributes > Edit Attributes or by viewing the *.xml* file using a text editor or an internet browser. For more information on recording a history of actions, refer to "Saving a history of actions on images (TBD)" on page 126.

To save images to XML format

1. Select File > Save as.

   The Save Image as dialog box opens.

2. Accept the name currently in File name or type a new name. Make sure you add *.xml* as the extension.

3. Click Save.

   MIPAV saves the file under the name and extension you specified and begins, from this point on, recording a history of actions performed on the image.

SAVING IMAGES TO ANOTHER FORMAT OR RENAMING IMAGES

To save images to format different from the original file

1. Select File > Save as.The Save Image as dialog box appears (Figure 48).

2. Type the new name in File name. To save the file in a different format, change the file extension. A list of extensions appears in Table 1 on page 628.

3. Click Save. MIPAV saves the file under the name and extension you specified.

To rename images

1. Select File > Save as. The Save Image as dialog appears (Figure 48).

2. Type the new name in File name. Either keep the extension already specified or change it to another file type. A list of extensions appears in Table 1 on page 628.

3 Click Save. MIPAV saves the file under the name and extension you specified.

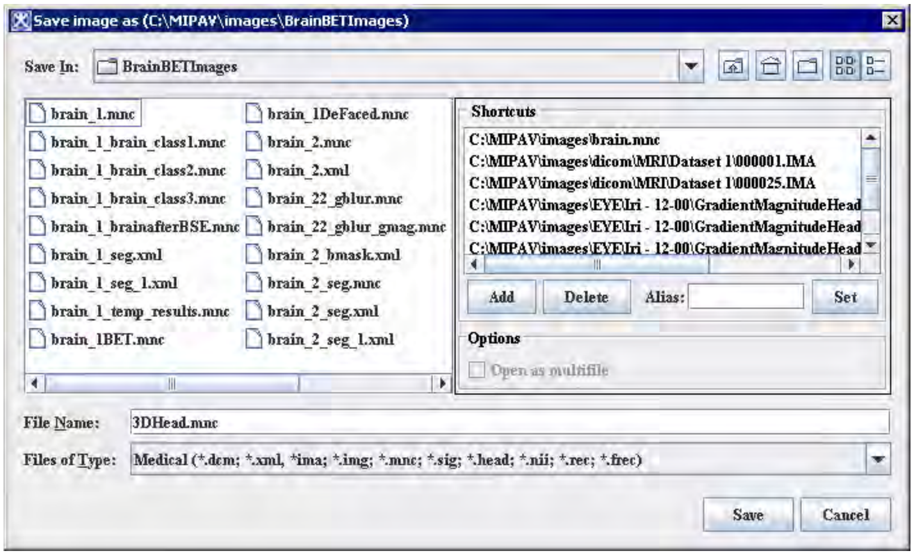


Figure 48. Save Image as dialog box

Printing Images

MIPAV offers the following printing command **File > Capture and Print Image**. This command calls the Print dialog box. Using this dialog you can set up a printer (to do it press the Properties button) or a file for printing (using the Print to File option), select the print copies and print range (all pages, current page, or selected pages).

To print an image with its current options:

Choose File > Capture and Print Image, and click OK.

To set printer and page setup options:

1 Choose File > Capture and Print Image.

2 Select an installed printer from the pop-up list at the top of the dialog box.

3 Set additional options, such as paper size and layout, as desired. The available options depend on your printer, print drivers, and operating system.

Customizing MIPAV

You can set the following MIPAV configuration options:

Showing or hiding the splash screen on start-up

The *splash screen* is the window that first appears briefly when you start MIPAV. It appears immediately before the MIPAV window and the Output window open. The splash screen displays the name of the program and the MIPAV logo.

By default, MIPAV always displays the splash screen on start-up unless you decide to hide it. To do so, you need to change the option in the MIPAV Options dialog box.

To hide the splash screen or prevent it from appearing on start-up

1 Select Help > Program Options. The MIPAV Options dialog box (Figure 50) opens.

2 Clear the Display Splash Screen box.

3 Click Close.

After you quit MIPAV and then start it again, the splash screen doesn't appear.

TO SHOW THE SPLASH SCREEN

After hiding the splash screen, you may later decide to display it.

1 Select Help > Program Options. The MIPAV Options dialog box (Figure 50) opens.

2 Mark Display splash screen. A check mark appears in the check box.

3 Click Apply, and then click Close.

After you quit MIPAV and then start it again, the splash screen appears.

Display Splash screen	Displays the MIPAV opening splash screen, or title screen, when MIPAV is started. By default, this check box is clear.
Use platform-style File dialog boxes	Uses the style of Open and Save dialog boxes that are provided with the operating system of your computer, rather than the MIPAV style. For more information, refer to "Using platform-specific Open and Save dialog boxes" on page 124. By default, this check box is clear.
Show Scripting toolbar	Shows the Scripting toolbar, which you can use to create and run scripts, in the MIPAV window. By default, this check box is clear.
Show Paint toolbar	Shows the Paint toolbar, which allows you to paint directly on images. By default, this check box is clear.
Show Paint border	Shows a border around the painted area. By default, this check box is clear.
Snap paint cursor to pixels	Causes the paint cursor to snap to the nearest pixel.

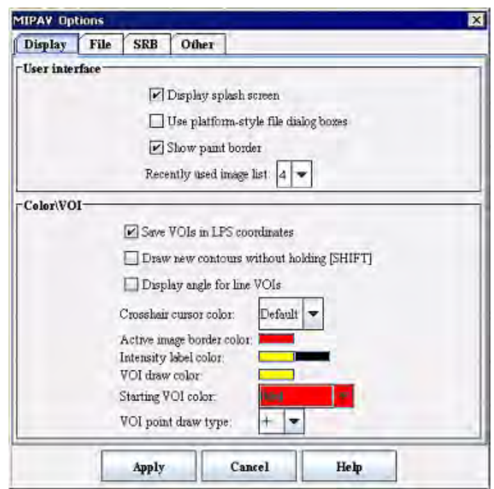


Figure 49. The Display page in the MIPAV Options dialog

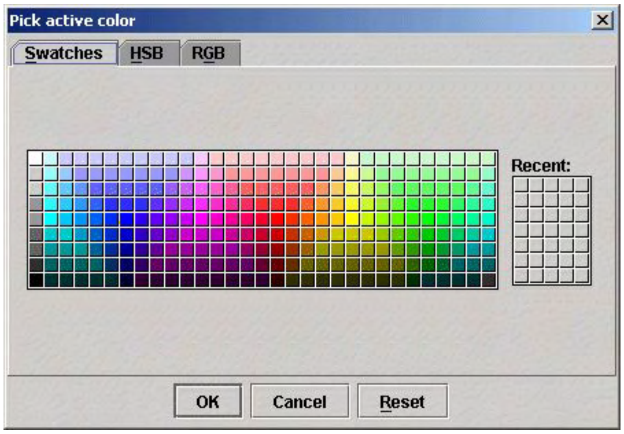
Recently used image list	Specifies the number of recently displayed images that appear near the bottom of the File menu on the MIPAV window. To open a recently displayed image, simply select the image on the File menu. The number of images that can be displayed can be from 2 to 9. By default, the number of images displayed on the File menu is 4.
Crosshair cursor color	Specifies the color of the crosshair cursor. You can choose from several colors and styles. By default, the color is set to "Default."
Active image border color	Specifies the color of the border around images. When you click on this box, the Pick Active Color dialog box, which allows you to select a different color, opens.  By default, the color is red.
VOI draw color	Allows you to choose the color that is used to draw VOIs. After you finish drawing VOIs, the completed VOIs appear in the Starting VOI color.
Starting VOI color	Allows you to select the initial color for completed VOIs. That is, when you draw VOIs, they appear in the color chosen for VOI draw color; the completely drawn VOIs appear in the color you selected for this field.
Apply	Saves and immediately applies all of the selected parameters in this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not save the specified options.
Help	Displays online help for this dialog box

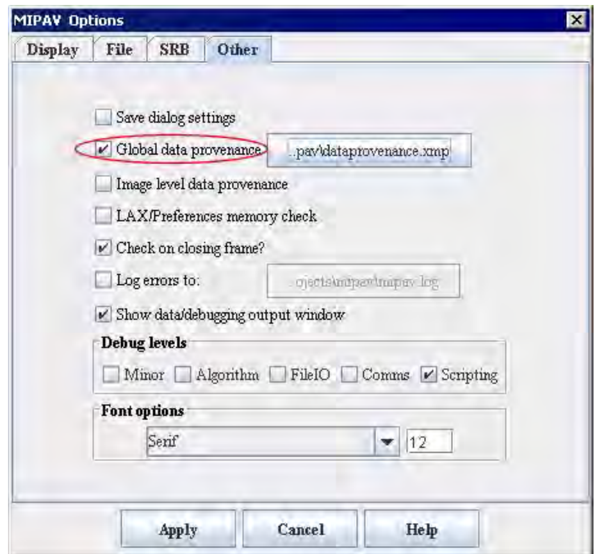
Figure 49. The Display page in the MIPAV Options dialog

Prompt overwrite on save	Displays a prompt, or message, when you save a file under the same name as an already existing file. The message indicates that saving the file under the same name as an existing file overwrites the file.
Save all on save	Saves the active image and all VOIs presently displayed on the image so that the next time you open the image MIPAV opens the image and VOIs. The VOIs are saved in a subdirectory of the directory in which the image is stored. By default, this check box is clear.
Save dialog settings	Saves the settings you specified on algorithm dialog boxes so that the next time you use the algorithms, your specific settings override the default settings. By default, this check box is clear.
Save XML header with Analyze images	Saves an XML header for an Analyze image when the image is saved. If you choose this option, the image has two headers: the normal header and an XML header. Both point to the image file. By default, this check box is clear.
Save thumbnails for XML files	Saves a thumbnail image in the XML header, allowing you to view the thumbnails in the image browser
Compress image in zip format when saved as XML	Compresses images in a zip format when they are saved in XML format. When an image is saved in XML format, MIPAV produces a RAW file that describes the image and an XML file that includes image attributes. When you select this check box, MIPAV compresses only the RAW file, which compensates for the size of the RAW file. By default, this check box is clear.
Default frame rate for Save-image-as AVI	Specifies the default frame rate in frames per second for images that are saved as AVI. By default, this check box is clear. If you select this option, the default frame rate is 10.0.
File filter default	Specifies the types of files that should be shown in the Files of type box in the Open Image and Save Image as dialog boxes. Refer to "Choosing the default file types to display or save" on page 129 for more information.
Apply	Saves and immediately applies all of the selected parameters in this dialog box.

Figure 50. The File page in the MIPAV Options dialog box

Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not save the specified options.
Help	Displays online help for this dialog box

Figure 50. The File page in the MIPAV Options dialog box (continued)



Save dialog settings	Saves the active dialog box settings in a log file.
Global data provenance	Keeps a global record of all of the actions—algorithms and utilities—performed on images. The history may be viewed in the **MIPAV system data provenance** dialog box. By default, this check box is clear.
Image level data provenance	Keeps a record of all of the actions—algorithms and utilities—performed on images. The history may be viewed in the **Image data provenance** dialog box while the actions are performed, or in the XML file when an image is saved to an XML file. By default, this check box is clear.
LAX/ Preferences memory check	Enables MIPAV developers to debug the program. If you experience problems with the program, you may be asked to select this check box. **Recommendation:** Do not select this check box unless otherwise instructed by MIPAV development. By default, this check box is clear.
Check on closing frame?	Adds messages to confirm deletions of images. By default, this check box is clear.
Log errors to:	Allows you to select the file in which MIPAV records any errors that occur. By default MIPAV logs errors to C:\Program Files\mipav\mipav.log unless the name and path of the log file was changed during installation.
Debug levels: Minor	Records only minor error messages in mipav.log.

Figure 51. The Other page in the MIPAV Options dialog box

Debug levels: Algorithm	Records only error messages with algorithms in mipav.log.
Debug levels: FileIO	Records only FileIO (file input and output) error messages in mipav.log.
Debug levels: Comms	Records only error messages involving communications in mipav.log.
Debug levels: Scripts	Records all error messages in mipav.log.
Apply	Saves and immediately applies all of the selected parameters in this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not save the specified options.
Help	Displays online help for this dialog box.

Figure 51. The Other page in the MIPAV Options dialog box (continued)

Jargon version:	A version of Jargon API library, which should be used to communicate to the SRB server. Select one from the list;
Transfer Mode	Parallel or sequential specifies the file transfer protocol. There are SRB parallel transfer protocol and SRB sequential transfer protocol available;
Temporary Directory	Is your local directory, where you keep your files which need to be transferred FROM or TO the SRB server. See also "Connecting to SRB BIRN" .
Apply	Saves and immediately applies all of the selected parameters in this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not save the specified options.
Help	Displays online help for this dialog box.

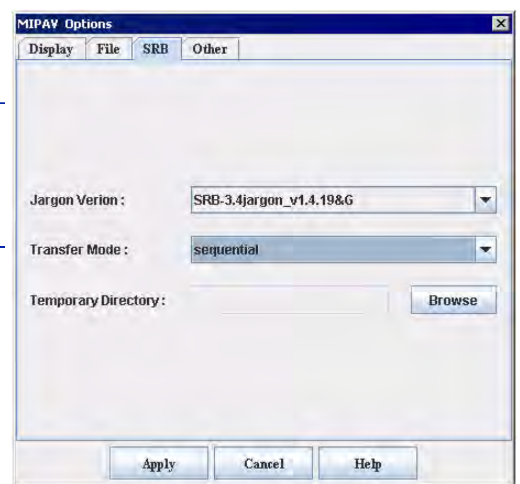


Figure 52. The SRB tab

Using platform-specific Open and Save dialog boxes

By default, MIPAV uses its own version of Open and Save dialog boxes, which provide you with the ability to assign aliases, or shortcuts, to frequently used images. Aliases make it easy to locate images, a feature that may be attractive if you work with the same images for a period of time. However, you may prefer instead to use the style of Open and Save dialog boxes that are provided with the operating system of your computer. To do so, you need to select the Use platform-style File dialog boxes check box in the MIPAV Options dialog box. See Figure 53 on page 125.

If this check box is selected and your computer is running Microsoft Windows, MIPAV displays the Windows style of Open and Save dialog boxes. If you use a Sun terminal, when this check box is selected, MIPAV displays the standard Unix-style Open and Save dialog boxes. On Apple MacIntosh computers, MIPAV displays dialog boxes that are standard with that operating system.

To use platform-specific dialog boxes

1. Select Help > Program Options. The MIPAV Options dialog box opens. See Figure 53 on page 125.
2. Mark **Use Platform-Style File Dialog Boxes**. A check mark appears in the check box.
3. Click Apply, then click Close. The dialog box closes.

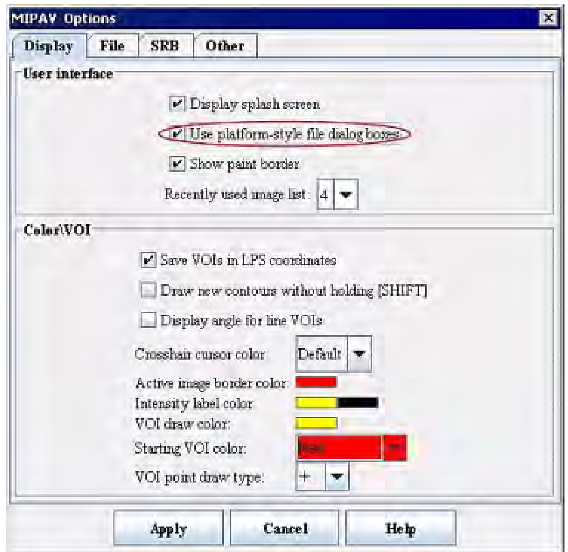


Figure 53. MIPAV Options dialog box showing Use platform-style File dialog boxes check box

Debugging MIPAV

You can track debugging information and error messages generated by MIPAV during a session by placing it in debug mode. If errors occur during the session, the program displays any error messages in the Output window on the Debug page. By default, the debug mode is off when you start the program.

Recommendation: The debugging information is primarily intended for MIPAV developers and not for users. The best course for a user is to leave the debug mode *Off* unless a MIPAV developer asks for he/she to turn it on.

For more information about MIPAV debug mode, please refer to "Troubleshooting" , "Placing MIPAV in debug mode" on page 595.

Saving a history of actions on images (TBD)

MIPAV allows you to keep a history of all of the actions—the algorithms and utilities—that you perform on images. The history includes the specific parameters that were set for the action. In addition, if you save the images as XML, the history of actions and their parameters appears in the dataset attributes section near the beginning of the MIPAV header file (the XML file). Each action appears between the <HISTORY> and </HISTORY> tags.

By default, MIPAV does *not* record a history of actions unless you specifically select the Record history check box in the MIPAV Options > Other dialog box. TBD.

To record a history of actions taken on images TBD

1 Select Help > Program Options in the MIPAV window. The MIPAV Options dialog box appears.

2 Mark Record history. A check mark appears in the check box.

3 Click Apply. From this point on, MIPAV records the actions performed on all images until you clear the Record history check box.

Refer to Figure 54 and "To view the history of actions recorded in the XML file TBD" on page 128.

To view the history of actions as they are performed

The History page in the Image Attributes dialog box allows you to view the list of actions *immediately* after they are performed.

The following example illustrates how to do this:

1. Make sure that the Record History check box was selected in the MIPAV Options dialog box.
2. Open a sample image.
3. Select Image > Attributes > Edit attributes. The Image Attributes dialog box opens.
4. Select History. The History page appears.
5. Click the image to make it the active window.
6. Select Utilities > Flip > Horizontal. The image is flipped horizontally, and the words "Flip(Y_AXIS)" appear on the History page in the Image Attributes dialog box.
7. Perform other actions on the image, such as algorithms or other utilities. Note that these actions appear on the History page.

You can type directly onto the History page of the Image Attributes dialog box, letting you add comments to the history, which you can save and print.

Under What Circumstances Do Actions Fail to Appear on the History Page?

If MIPAV is fully operational, actions performed on an image always appear on the History page of the Image Attributes dialog box and in the XML file for that image. However, it's important to know whether the actions were performed on the *original image* or on the *replacement image*.

Several algorithms and utilities offer a choice of either generating a new image (New image check box) or replacing the original image (Replace image check box). If you choose New image and want to see the actions performed on that image, you must select Image > Attributes > Edit Attributes to open an Image Attributes dialog box for the new image. If, however, you select Replace image, the History page on the Image Attributes dialog box for the original image displays the actions.

To illustrate this point, open a sample image; select Image > Attributes > Edit Attributes to display the Image Attributes dialog box; and select History to display the History page. Perform an action such as Utilities > Image Margins, mark New image in the dialog box, and then click OK. Notice that the action does not appear on the History page. Click on the new image; select Image > Attributes > Edit Attributes; and select History. The History page shows the AddImageMargins action for the new image.

To view the history of actions recorded in the XML file TBD

1. Save the sample image as an XML file by selecting File > Save Image as. The Save Image as dialog box opens.
2. Select the directory where you want to save the file in the Save in box.
3. Type the name of the file—use "SampleImage" as the file name—and type an ".xml" file extension in the File Name box so that the complete file name is "SampleImage.xml."
4. Click Save.
5. Open Windows Explorer by right-clicking on Start and then selecting Explore. The Windows Explorer window opens.
6. Navigate to the directory where you stored the XML file. This directory should have two files named "SampleImage": "SampleImage.raw" and "SampleImage.xml."
7. Double-click "SampleImage.xml." An internet browser page, such as Internet Explorer or Mozilla, appears displaying an XML page.

Notice that, immediately after the <Dataset-attributes> tag, the following line appears:

<History>Flip(Y_AXIS)</History>

Each action that you perform on the image appears between its own <History> and </History> tags. See also Figure 54.

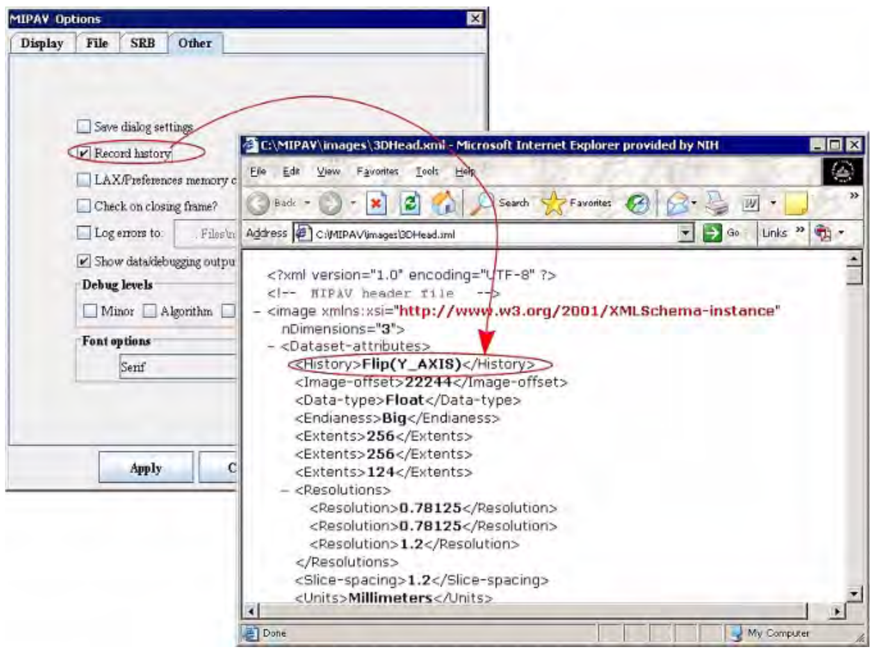


Figure 54. The Record History check box is located in the MIPAV Options dialog box > Other tab. Actions recorded are shown in the header of open XML file. TBD

Choosing the default file types to display or save

When you using MIPAV, chances are that you are working with the same type of files every day or perhaps for a certain period of time. For example, suppose you normally work with medical files (files whose extensions are .dcm, .xml, .ima, .img, .mnc, .sig, .head, etc.). In this situation, when opening or saving images, you want to see only the medical files, not other file types. One of the ways of customizing MIPAV is the ability to show only specific types of files by default.

The File Filter Default box on the MIPAV Options dialog box allows you to choose which types of files should be shown in the Files of type box in the Open Image and Save Image as dialog boxes.

To choose the default file types to display or save

1. Select Help > MIPAV Options. The MIPAV Options dialog box opens.
2. Click File. The File page appears.
3. Click File filter default. The Choose File Filter dialog box appears.
4. Select one of the file filters. You can choose one of the following collections of file types:
   - All (*.*)
   - AVI (*.avi)
   - Class files (*.class)
   - FreeSurfer (*.asc)
   - General (*.gif, *.jpeg, *.jpg, *.pict, *.psd, *.tif, *.tiff)
   - Graphs (*.plt)
   - Look-up tables files (*.lut)
   - Matrix files (*.mtx, *.mat)
   - Medical (*.dcm, *.xml, *.ima, *,img, *.mnc, *.sig, *.head)
   - Microscopy (*.avi, *.ics, *.lsm, *.pic, *.stk, *.tif, *.tiff, *.xml)
   - Misc. (*.avi, *.bmp, *.pcx, *.png, *.tga, *.xbm, *. xpm)
   - Optical (*.avi, *.bmp, *.img, *.jpg, *.pict, *.psd, *.tif, *.tiff, *.xml)
   - Project (*.xml)
   - Script files (*.sct)
   - Surface files (*.sur, *.wrl)
   - Transfer function files (*.fun)
   - VOI files (*.voi)
   - Nonlinear transformation files (*.nlt)
   - User Defined
5. Click OK. The Choose File Type dialog box closes and the file type you chose appears on the File filter default button.
6. Click Apply in the MIPAV Options dialog box. This change takes effect immediately.

See Figure 54 on page 129.

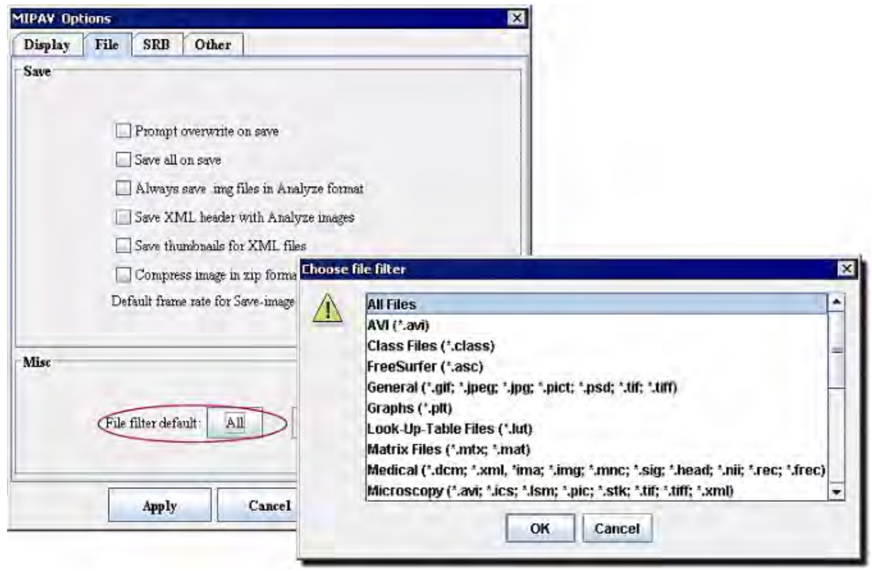


Figure 55. The File Filter Default button is located on File toolbar of the MIPAV Options dialog box. The Choose File Filter dialog box appears when you press that button

To verify that the filters you chose are used as the default filters

1. Select File > Open > Image(A). The Open Image dialog box appears. The Files of type box should show the filters you chose.
2. Select File > Save Image as. The Save Image as dialog box opens with the Files of type box listing the file filters you chose.

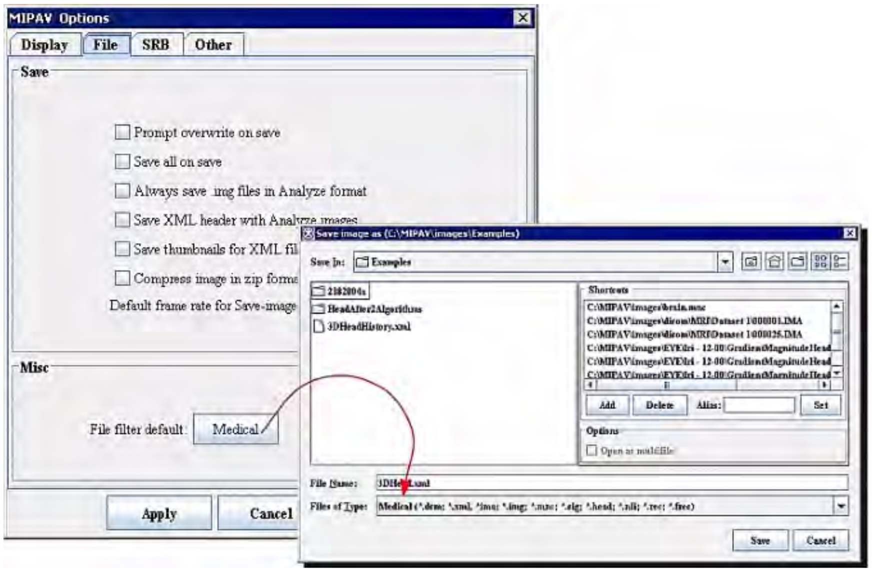


Figure 56. Setting up the Medical file filter as a default filter. The Save As dialog box offers only Medical file types as types to save

Adding shortcuts

MIPAV allows you to create shortcuts using the Ctrl, Alt, and Shift keys plus the alphanumeric key of your choice. Shortcuts provide a quick way for you to perform a command on a menu or a command on one of the toolbars.

Example: Suppose you frequently clone images. Rather than taking the time to select Utilities > Clone, you may find it easier and faster to simply press Ctrl +C, the keys you assigned to the shortcut.

To add shortcut keys

1 Select Help > Shortcut editor. The Shortcuts dialog box opens.

2 Click Add. The New Shortcut dialog box opens. See Figure 57–1.

3 Select Ctrl, Alt, or Shift or any combination of these check boxes. You can select only one of these check boxes, or you can select any combination of these check boxes. Valid selections can include only Ctrl, only Alt, or only Shift, or you can select Ctrl and Alt, Ctrl and Shift, Alt and Shift, or all three of the check boxes. See Figure 57–2.

4 Type an alphanumeric character in Character, and then press OK.

5 A message appears directing you to select a command on a menu or in a toolbar. See Figure 57–3. Press OK again. The message disappears.

6 Select a toolbar icon or select a command on a menu in the MIPAV window to which you want to assign this shortcut. A message appears indicating that the shortcut was successfully created. See Figure 57–4.

7 Click OK. The message disappears, and the shortcut appears on the Shortcuts dialog box.

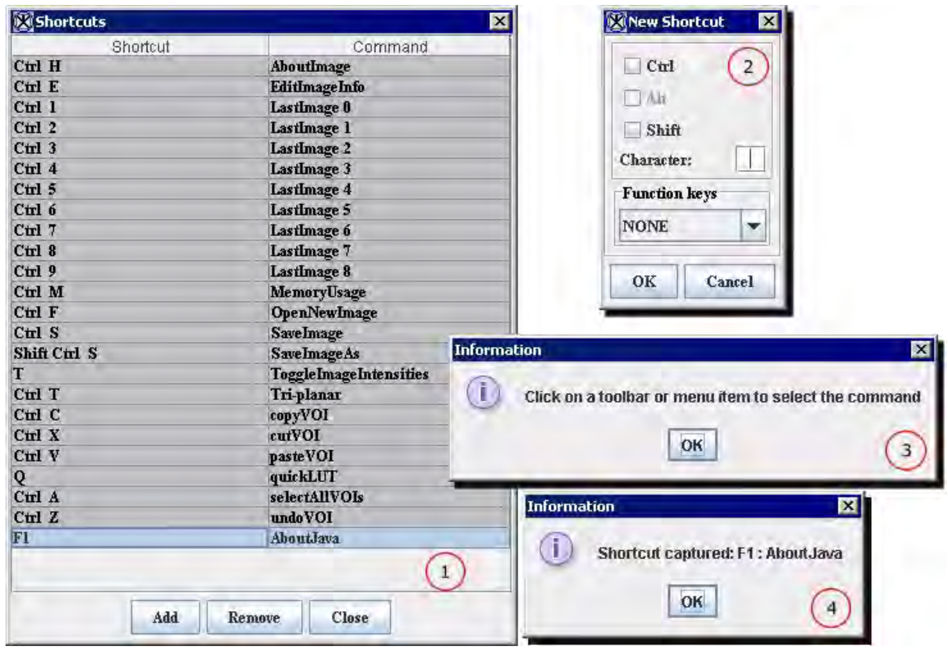


Figure 57. Creating a new shortcut. Steps 1–4 are explained in the text above

To remove shortcut keys

1 Select Help > Shortcut editor. The Shortcuts dialog box opens.

2 Select the shortcut you want to delete.

3 Click Remove. MIPAV removes the shortcut, and the shortcut disappears from the list of shortcuts in the Shortcuts dialog box.

Note: Make sure you want to delete the shortcut. You cannot undo the deletion.

Developing and using plug-in programs

A *plug-in* program allows you to add customized functionality to MIPAV. Before you create a plug-in, you should have a strong understanding of the underlying structure of MIPAV's software design and data structure. Because plug-ins are written in Java, you should also have a basic understanding of that programming language. After you create the plug-in, you must then install it. Once it is installed, you can access it from the Plug-Ins menu in the main MIPAV window. Plug-ins are addressed in Chapter 12, "Developing Plug-in Programs", in the *MIPAV User's Guide*.

Quitting MIPAV

To end a MIPAV session

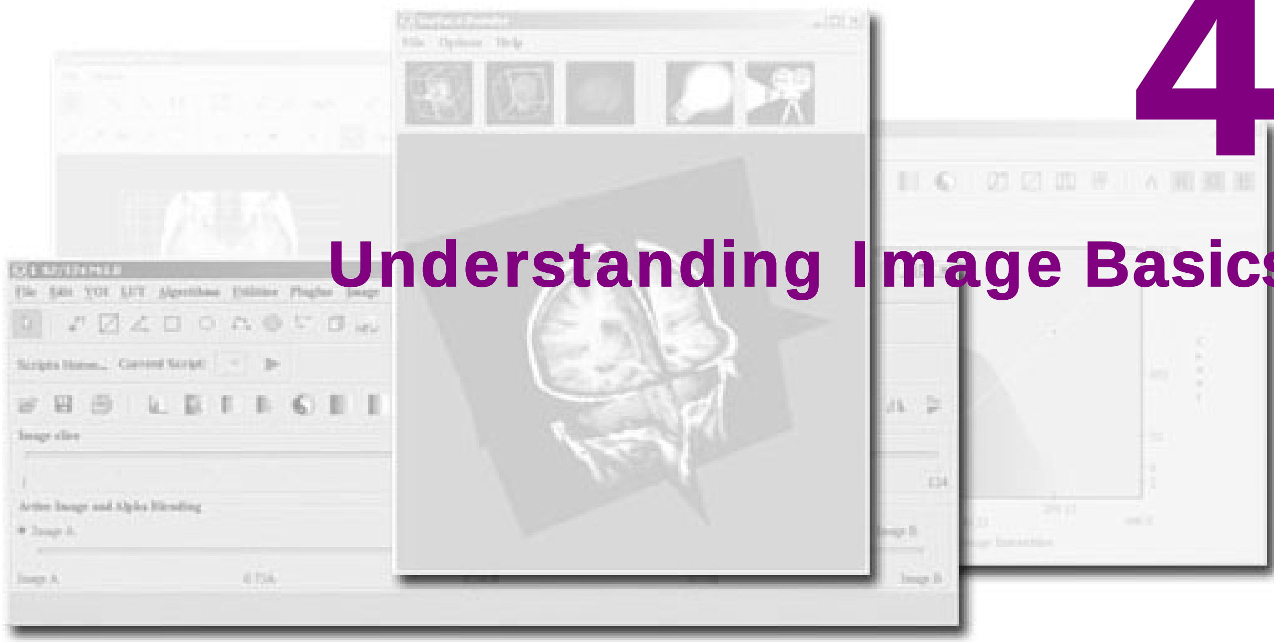
1. Select File > Exit-MIPAV. The MIPAV-Exit dialog box appears asking if you really want to exit the program. See Figure 58.
2. Click Yes to end the session.

 You can also exit MIPAV by clicking on Close at the top right of the MIPAV window.

Figure 58. MIPAV Exit Confirmation dialog box

4

Understanding Image Basics



In this chapter . . .

This chapter also explains the dimensionality and coordinate system used by MIPAV, the data types MIPAV supports, the header offset information, endianness, and how to view and edit image attributes.It also provides a basic information about image file formats supported by MIPAV. More information about graphic file formats and the formats that MIPAV supports can be found in Appendix C: "Supported formats" .

Supported graphic formats overview

MIPAV supports three categories of graphic file formats: vector (i.e., volume of interest, or VOI), bitmap (i.e., 2D to 5D grayscale or RGB images), and 3D images.

Vector file formats are usually used to store line-based elements. These elements can be geometric shapes, such as polygons, curves, or splines. MIPAV can read and generate several vector-formatted file types including the MIPAV lookup table (LUT), MIPAV MTX, MIPAV Graphics Plot (PLT), and MIPAV VOI. See also Appendix C: "Supported formats" on page 627.

Bitmap file formats are usually used to store real-world graphics data e.g., photographs or medical images such as CT scans. The data portion of bitmap files contains numerical data that indicates the color of each pixel or voxel in the image. MIPAV can read and generate a number of bitmap formatted file types, such as Adobe Photoshop (PSD), Graphics Interchange File (GIF), Sun Raster (RS), Tagged Image File Format (TIFF), and Truevision Graphics Adapter (TGA).

MIPAV uses 3D graphic formatted files to store descriptions of the color and shape of 3D models of real-world and imaginary objects.

To support the wide range of image-processing needs of the NIH intramural research community, MIPAV reads and writes the image file types indicated in Table 1, refer to Appendix C: "Supported formats" on page 627.

Understanding the MIPAV dimensionality and coordinate system

Internally, MIPAV is designed to store images of any dimensionality and thus is *n* dimensional. Figure 59 is a model of an example 3D-image where each position within the grid represents a *voxel*.

voxel—The smallest distinguishable cube-shaped part of a 3D-image.

The voxel in the upper left-most position is considered the origin of the dataset. (A volumetric dataset is a set of cross-sectional images). Ideally, each voxel would be isotropic (i.e., cubic), but clinical datasets are often acquired in such a manner that the resolution in the *x* and *y* dimensions is greater than the resolution in the *z* dimension, thus producing rectangular voxels (elongated in the *z* direction). This voxel anisotropy is taken into account in the execution of the algorithms found in MIPAV.

Because MIPAV's image storage buffer is *n*-dimensional, it can store 2D, 3D, and 4D+ medical images. CT and MRI are examples of 3D medical images where the three dimensions (*x*, *y*, and *z*) are spatial dimensions. Other 3D datasets include fluoroscopy volumes, where the first two dimensions are spatial and the third dimension is time.

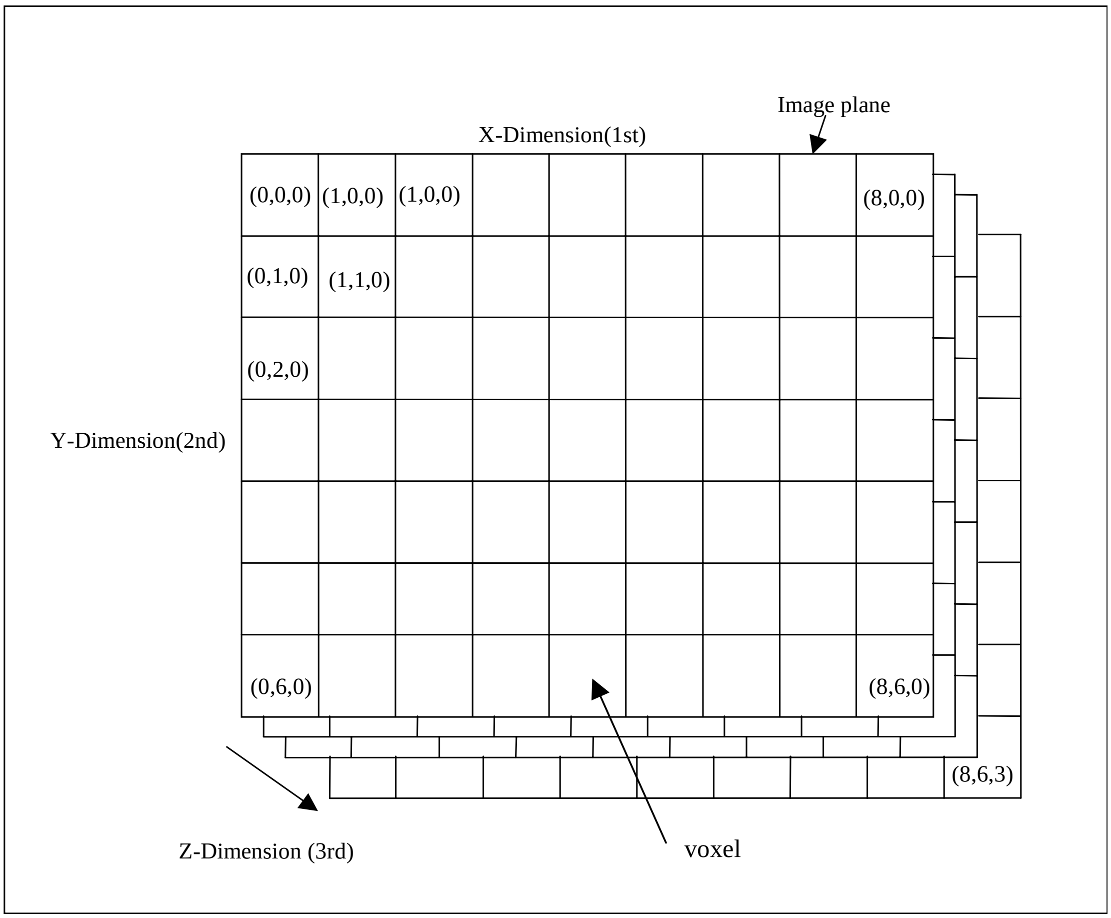


Figure 59. Example of 3D image (origin at top left)

Data types

A *data type* is the classification of a particular type of information. For example, a floating point data type indicates a number with a decimal point. Data read from image files are stored in MIPAV's data structures. MIPAV supports all the basic data types shown in Table 1 and may in the future support more composite data types including color and complex numbers. The image data read from the file is stored in a data buffer of the same basic data type, preventing the reduction of image information that results from conversion into a fixed data type.

Table 1. Data types supported by MIPAV

Data type	Description
Boolean	1 bit per pixel/voxel (1 on, 0 off)
Signed byte	1 byte per pixel/voxel (-128, 127)
Unsigned byte	1 byte per pixel/voxel (0, 255)
Signed short	2 bytes per pixel/voxel (-32768, 32767)
Unsigned short	2 bytes per pixel/voxel (0, 65535)
Integer	4 bytes per pixel/voxel (-2^{31}, $2^{31}-1$)
Long	8 bytes per pixel/voxel (-9.22E18, 9.22E18)
Float	4 bytes per pixel/voxel (-3.4E38, 3.4E38)
Double	8 bytes per pixel/voxel (-1.8E308, 1.8E308)
Color 32	3 bytes per pixel/voxel, plus 1 byte; 8 bits per color channel (alpha, red, green, and blue)

Headers

If you attempt to open a raw image, you may need to supply MIPAV with the *header offset*. The header offset indicates the size of the space reserved at the beginning of the file where specific types of information is kept. This space, which is called the header, precedes the image data. If you know the length of the header, enter it in the header offset text box. When MIPAV accesses the file, it skips the header offset and begins to read the image data. **Note that** not all image file formats have a header.

Endianness

Endianness refers to the byte ordering of the data. Some computers order the data with the least significant byte (LSB) first followed by the most significant byte (MSB). This byte order is referred as "little endian" or Intel byte ordering. machines that use little-endian byte ordering are VAXes, Intel x86, and Pentium. The reverse is MSB then LSB, which is referred as "big endian" or Motorola byte ordering. Machines that use big-endian byte ordering are IBM System 3D, RISC, and a Motorola 680x0. MIPAV is biendian; that is, it supports both big- and little-endian byte-ordering formats.

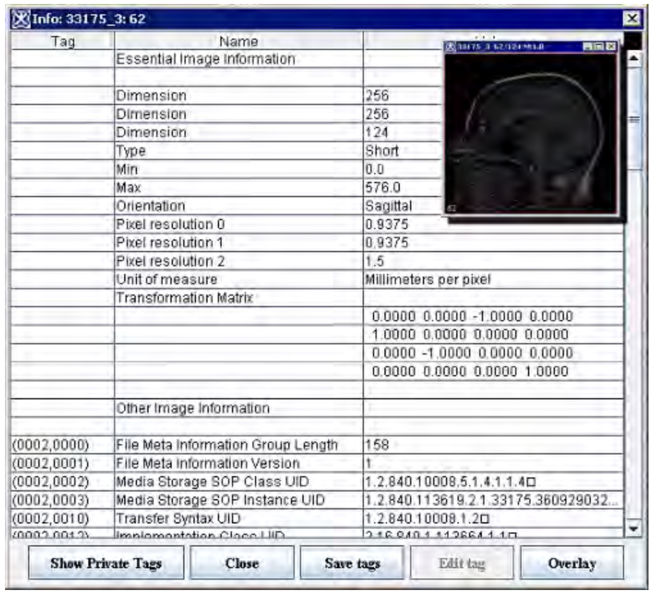
Viewing and editing image attributes

Information about the image appears in the Image Information dialog box and in the Image Attributes dialog box.

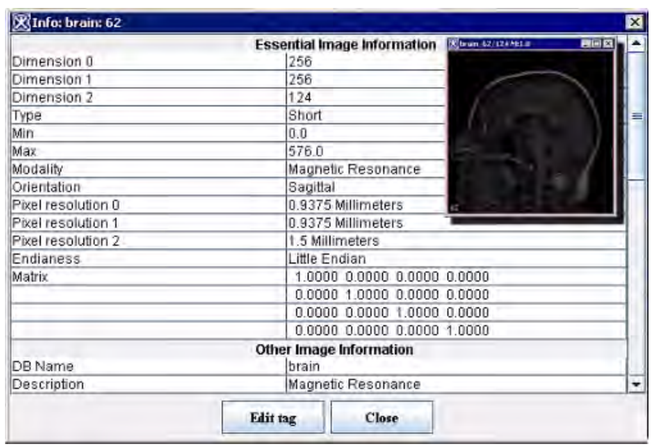
The Image Attributes window displays the name of the image file and information about the resolution of the image. The About Image window displays more detailed information about the image, such as the orientation of the image or the number of bits per pixel or voxel.

Viewing image attributes

To view the attributes of an image, select Image > Attributes > View Header in the MIPAV window. For DICOM datasets, the Image Information dialog box similar to the one shown in Figure 60-a appears. For non-DICOM datasets, an Image Information dialog box similar to the one in Figure 60-b appears.



(a) the Info dialog box for DICOM images



(b) the Info dialog box for non-DICOM images

Figure 60. The Info dialog box for DICOM (a) and non-DICOM (b) images

Editing image attributes

You can edit some of the image attributes for both DICOM datasets and non-DICOM datasets in the Image Attributes dialog box (Figure 61). The Image Attributes dialog box includes five tabbed pages, which are the following:

- **General page** displays the name of the image (without its extension, or suffix) and lists the image modality
- **Resolutions page** indicates the unit of measure for each dimension
- **Orientations/Origin page** lists the image orientations
- **Transform Matrix** displays and lists the values for the transform matrix
- **Talairach** displays Talairach transform

By default, the General page always appears first when the Image Attributes dialog box opens.

IMAGE ATTRIBUTES DIALOG BOX

Apply	applies the changes to the current image, but doesn't close the dialog box.
OK	applies the changes to the current image and closes the dialog box.
Close	closes the dialog box without applying the changes to the image.
Help	opens the on-line help.

General tab

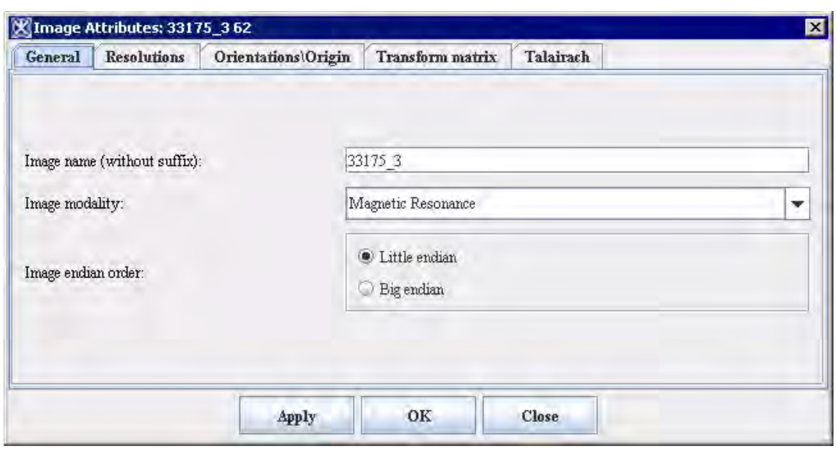
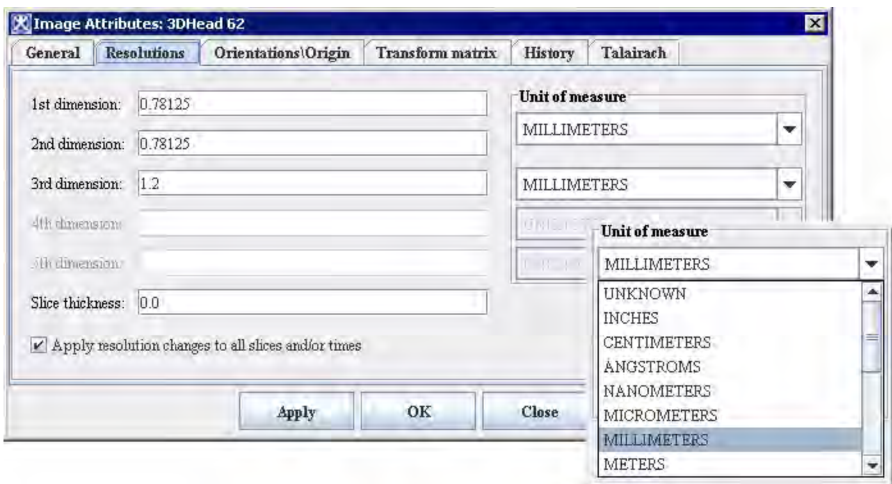


Image name (without suffix)	displays the name of the current image. You can change the image's name here.

Figure 61. The Image Attributes dialog box options

Image modality	shows image modality. You can change the modality by selecting it from the list.
The list includes:	Unknown Modality, Biomagnetic Imaging, Color Flow Doppler, Computed Radiography, Computed Tomography, Diaphanography, Digital Radiography, Duplex Doppler, Endoscopy, External Camera Photography, FA, General Microscopy, Hardcody, ICG, Intraoral Radiography, Laser Surface Scan, Magnetic Resonance Angiography, Mammography, Magnetic Resonance, Nuclear Medicine, Other, Positron Emission Tomography, Panoramic X-ray, Radio Fluoroscopy, Radiographic Imaging, Radiotherapy Dose, Radiotherapy Image, Radiotherapy Plane, Radiotherapy Record, Radiotherapy Structure Set, Red Free, Slide Microscopy, Single Photon Emission Computer Tomography, Thermography, Ultrasound, and X-ray Angiography.
Image endian order	Here, use the appropriate radio button to set the image endiannes.

Resolutions tab



1–5 dimension	shows resolution for each dimension.
Slice thickness	shows the slice thickness.
Units of measure	displays the units of measure used in the image.

Orientation/Origin tab

Figure 61. The Image Attributes dialog box options (continued)

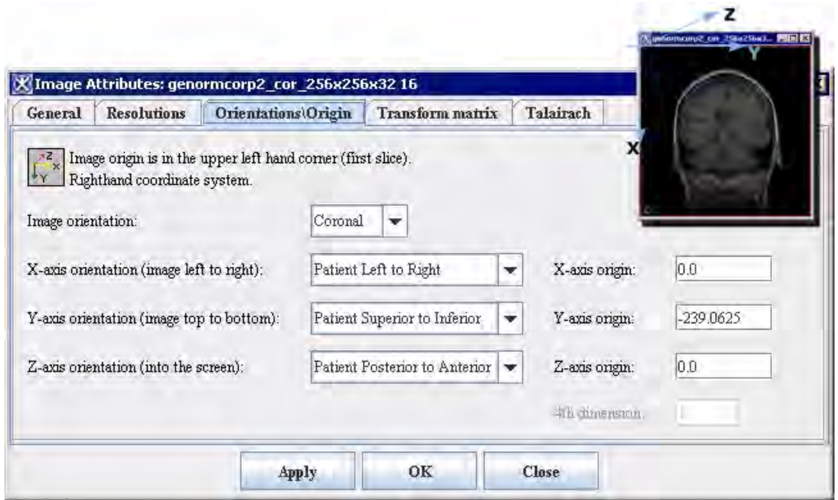
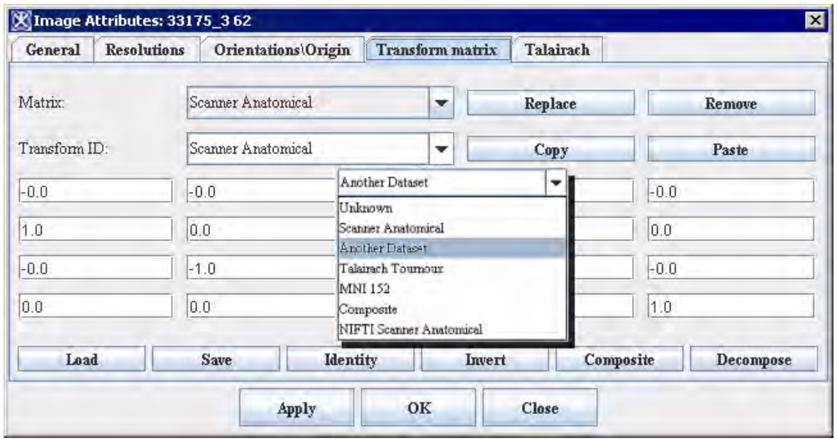


Image Orientation	Medical image data are stored in a variety of ways. In order to orient a two-dimensional image slice or three-dimensional volume in space, one needs to know how the order of the image data relates to a standard set of axes. For medical imaging, these axes are generally chosen to correspond to the directions **posterior/anterior**, **superior/inferior** and **left/right**. This information could be found from the image header. To change image orientation, select one from the list: Axial, Coronal, Sagittal, or Unknown. See also "About medical image coordinate systems."
X-axis orientation (image left to right) **Y-axis orientation (image top to bottom)** **Z-axis orientation (into the screen)**	To specify X, Y, or Z -axis orientation, select one from the list: Unknown, Patient Right to Left, Patient Left to Right, Patient Anterior to Posterior, Patient Posterior to Anterior, Patient Inferior to Superior, Patient Superior to Inferior. Here, • The head end is referred to as the **superior end**, while the feet are referred to as the **inferior end.** • **Anterior** refers to the "front" of the individual, similarly, **posterior** refers to the "back" of the individual. • Left and right are self explanatory.
X-axis origin **Y-axis origin** **Z-axis origin**	The image origin is generally at the corner of the image, often the upper left, the axis directions are: the X- axis is across, Y -axis is down, and Z-axis goes into the screen. Usually, it differs from matrix coordinates and Cartesian coordinates. E.g. for the above image, the origin coordinates shown in the dialog box are as follows: x=0,z=0, and y=-239.0625, which corresponds to the scanner frame of reference and scanner origin coordinates. If you mouse over the image, MIPAV will show you the coordinates from the image frame of reference.
4-th dimension	For 4D images, it specifies the units and values used for the 4-th dimension.
Transform matrix	

Figure 61. The Image Attributes dialog box options (continued)



Matrix	displays a list of transformation matrix for the chosen image
Transform. ID	displays transformation ID type for the chosen image. To change the current transformation matrix, select a new transform ID first, and then click Replace. This will replace the transformation matrix.
Replace	replaces a current transformation matrix to a new one that you choose.
Add a New	adds a transformation matrix that has been selected from the Transform ID list to the image.
Rename	renames a transformation matrix.
Copy	copies the current transformation matrix to the Clipboard.
Paste	pastes a copied transformation matrix.
Load	loads a transformation matrix (from a file with the **.mtx* extension)
Save	saves a transformation matrix to a file.
Identity	transforms the current transformation matrix to an identity matrix.
Invert	inverts the current transformation matrix.
Composite	TBD.
Decompose	TBD.
Talairach	

Figure 61. The Image Attributes dialog box options (continued)

Shows ACPC and Talairach transformation matrices	
Include Talairach – displays a Talairach transformation matrix for the image.	
Load – loads a new matrix.	
Save – saves a matrix to a file.	

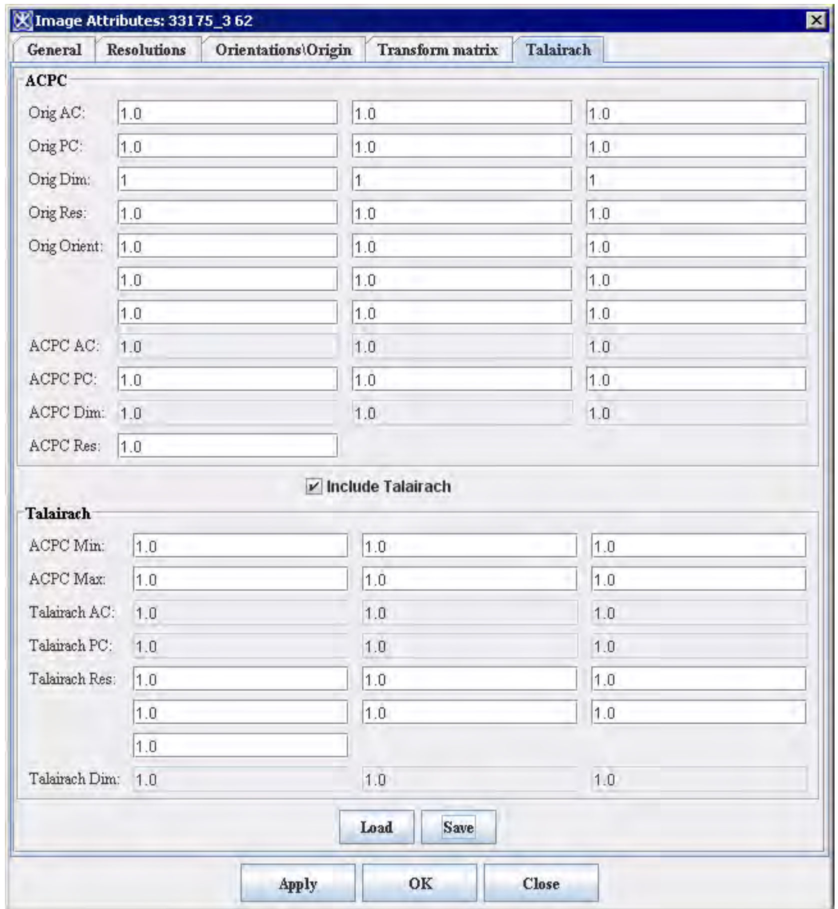


Figure 61. The Image Attributes dialog box options (continued)

To modify the name of the image dataset

1. Open either a DICOM or non-DICOM image dataset.
2. Select Image > Attributes > Edit Attributes in the MIPAV window. The Image Attributes dialog box opens.
3. Select the name of the image in the Image name (without suffix) box.
4. Type a new name of the image.
5. Click either Apply or OK. The new image name appears in the header on the image window.

What's the Difference Between Clicking Apply or Clicking OK?

Click Apply when you want the dialog box or window to remain open after it makes the change so that you can make further modifications. If instead you click OK, the dialog box or window closes immediately after making the current change.

To edit the dimensions and units of measure

1. Open either a DICOM or non-DICOM image dataset.
2. Select Image > Attributes > Edit Attributes in the MIPAV window. The Image Attributes dialog box opens.
3. Click Resolutions. The Resolutions page in the Image Attributes dialog box appears.
4. Type the changed values in the dimension text boxes on the left and then select the appropriate value in the Unit of measure boxes on the right.
5. Click either Apply or OK.

To edit the *x*, *y*, and *z* orientations of image datasets

1. Open either a DICOM or non-DICOM image dataset.
2. Select Image > Attributes > Edit Attributes in the MIPAV window. The Image Attributes dialog box opens.
3. Click Orientations. The Orientations page (Figure 61) in the Image Attributes dialog box opens.
4. Select the appropriate values in the boxes on the right.
5. Click Apply or OK.

To change the origin for the first image slice

1. Open either a DICOM or non-DICOM image dataset.
2. Select Image > Attributes > Edit Attributes in the MIPAV window. The Image Attributes dialog box opens.
3. Click Orientation. The Orientation page (Figure 61) in the Image Attributes dialog box appears.

4 Type the appropriate values in the dimension boxes.

5 Click Apply or OK.

To change the transform matrix

1 Open either a DICOM or non-DICOM image dataset.

2 Select Image > Attributes > Edit Attributes in the MIPAV window. The Image Attributes dialog box opens.

3 Click Transform Matrix. The Transform Matrix page (Figure 61) in the Image Attributes dialog box appears.

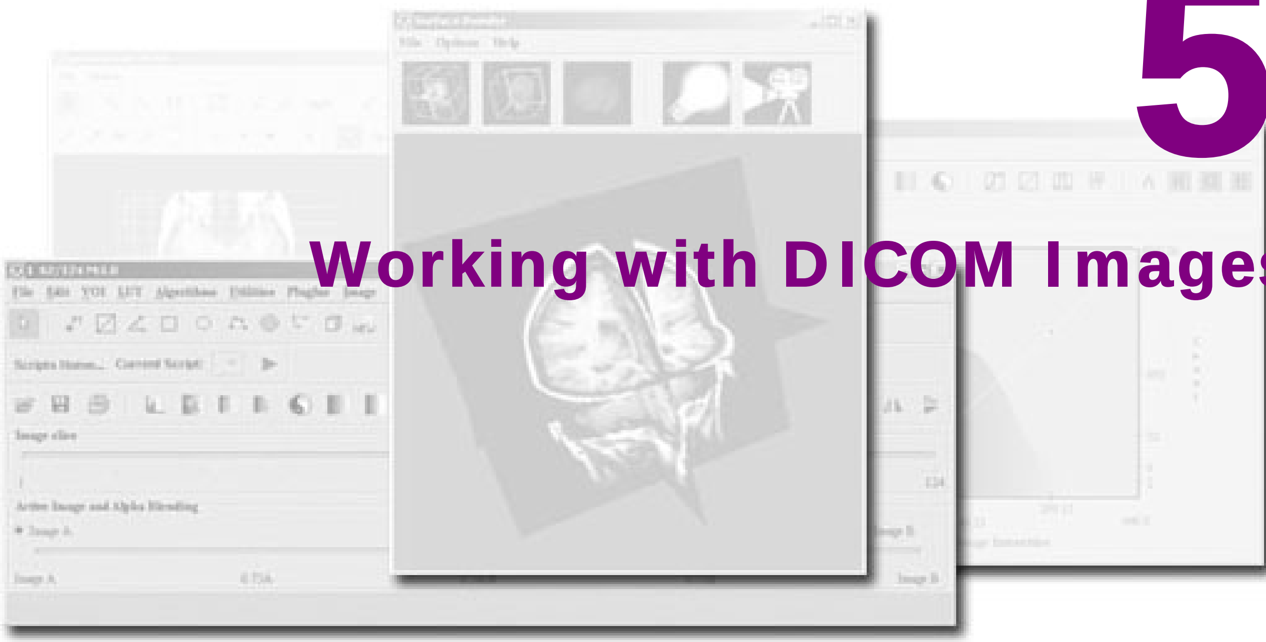
4 Select the appropriate transform ID in the Transform ID box.

5 Load and save transform matrices.

6 Click Apply or OK.

5

Working with DICOM Images



In this chapter . . .

Digital Image Communication in Medicine (DICOM) defines a standard method of transmitting digital medical-image information between devices, such as computers, servers, or imaging equipment. The standard not only prescribes how to communicate with other imaging equipment and databases, but also how to store images.

This chapter discusses how MIPAV works with DICOM images. It also explains how to browse DICOM images and send and retrieve DICOM images, which includes the following tasks:

- Setting up a hosts table to indicate where you want to send images ("internet protocol (IP) address: A numeric ID that is used to identify your computer on a Transmission Control Protocol/Internet Protocol

(TCP/IP) network. An example of a TCP/IP network is the Internet." on page 159)

- Testing server connections ("Testing the connection" on page 168)
- Posing queries and retrieving DICOM images ("Posing queries and retrieving images" on page 169 and "Receiving and sending image files" on page 173)
- Sending DICOM images to another device ("To send image files" on page 174)

It also explains how to edit DICOM tags, protect patient privacy by removing DICOM tag information from images, and convert image formats, such as Analyze or TIFF, to DICOM format.

Understanding how MIPAV works with DICOM images

You can use MIPAV to access images and information that were generated by DICOM-compatible imaging equipment. Some researchers at NIH use MIPAV to access DICOM-compatible archive systems, such as the Kodak Cemax-Icon Picture Archiving and Communication System (PACS), which includes the AutoRad devices located in the NIH clinical center. The PACS system consists of software, hardware, and networks. PACS provides the means for medical specialists to digitally acquire images, transmit the images and image-related data using a communications network, display and interpret images, and store and retrieve images. PACS also provides an interface to other systems that contain patient data.

Note: The DICOM standard is detailed in the 16-part documentation set *Digital Imaging and Communications in Medicine (DICOM) Standard* (refer to Appendix B, DICOM Conformance Statement.) MIPAV is DICOM version 3.0 compliant. Compliance information appears in Appendix B of this document.

Figure 62 shows the communication pathways between MIPAV, the Internet, a DICOM-compatible server, a patient database, an imaging device, and a remote MIPAV workstation.

Digital images are acquired on a medical imaging device, such as a Computed Tomography (CT) scanner. The image can remain on the device or be transferred via a DICOM-compatible server to a patient database or other archive media. Once the image is acquired and stored, you can then use MIPAV to query and retrieve, receive, or send DICOM image files.

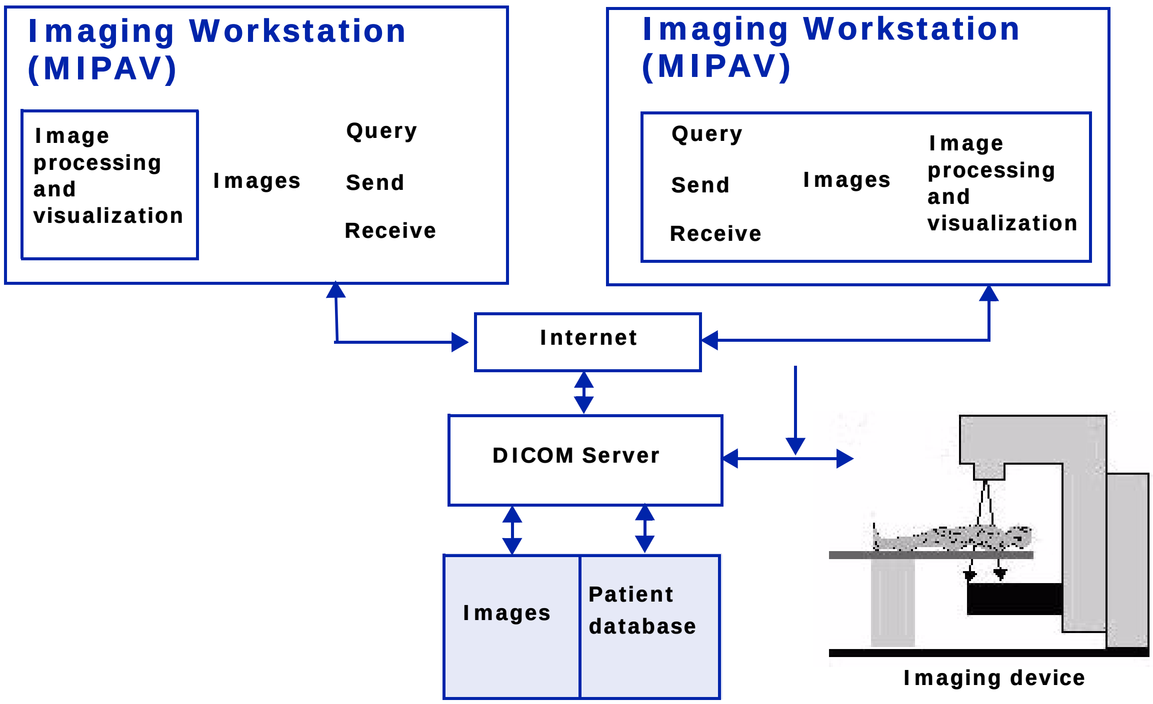


Figure 62. Communication pathways between MIPAV and DICOM

Browsing DICOM images

MIPAV includes a browser that is specifically designed for examining DICOM images. The DICOM browser allows you to view a list of all of the images in a dataset and quickly display each one individually. The browser provides a fast way of locating and selecting specific image slices.

Tip: You can open a DICOM browser at any time after starting MIPAV; and, depending on your computer's memory resources, you can open more than one DICOM browser.

Displaying an image dataset in the DICOM browser

1. Select File > DICOM browser (see Figure 63-1, and Figure 63-2). The Choose Directory dialog box opens, see Figure 63-3.
2. Select the directory in which the DICOM image datasets are stored.
3. Click Open. The DICOM browser (Figure 63-4) appears showing the directory tree in the panel on the upper left of the window.
4. Navigate, if necessary, to the correct subdirectory and select it.
5. Click Parse directory (Figure 63-4). Information appears in the three panels on the right side of the window: patient and study information at the top, dataset information in the middle, and image information at the bottom.
6. Select one of the images listed in the image information panel (Figure 63-5). The image appears in the panel on the left side of the window below the tree structure (Figure 63-6).

   You can now continue to browse the images, or you can perform one or more of the following tasks:
   - Adjust the image contrast and brightness, refer to page 153;
   - Open selected images, refer to page 153;
   - View another image dataset, refer to page 153;
   - Create a movie from the image dataset, see page 154.

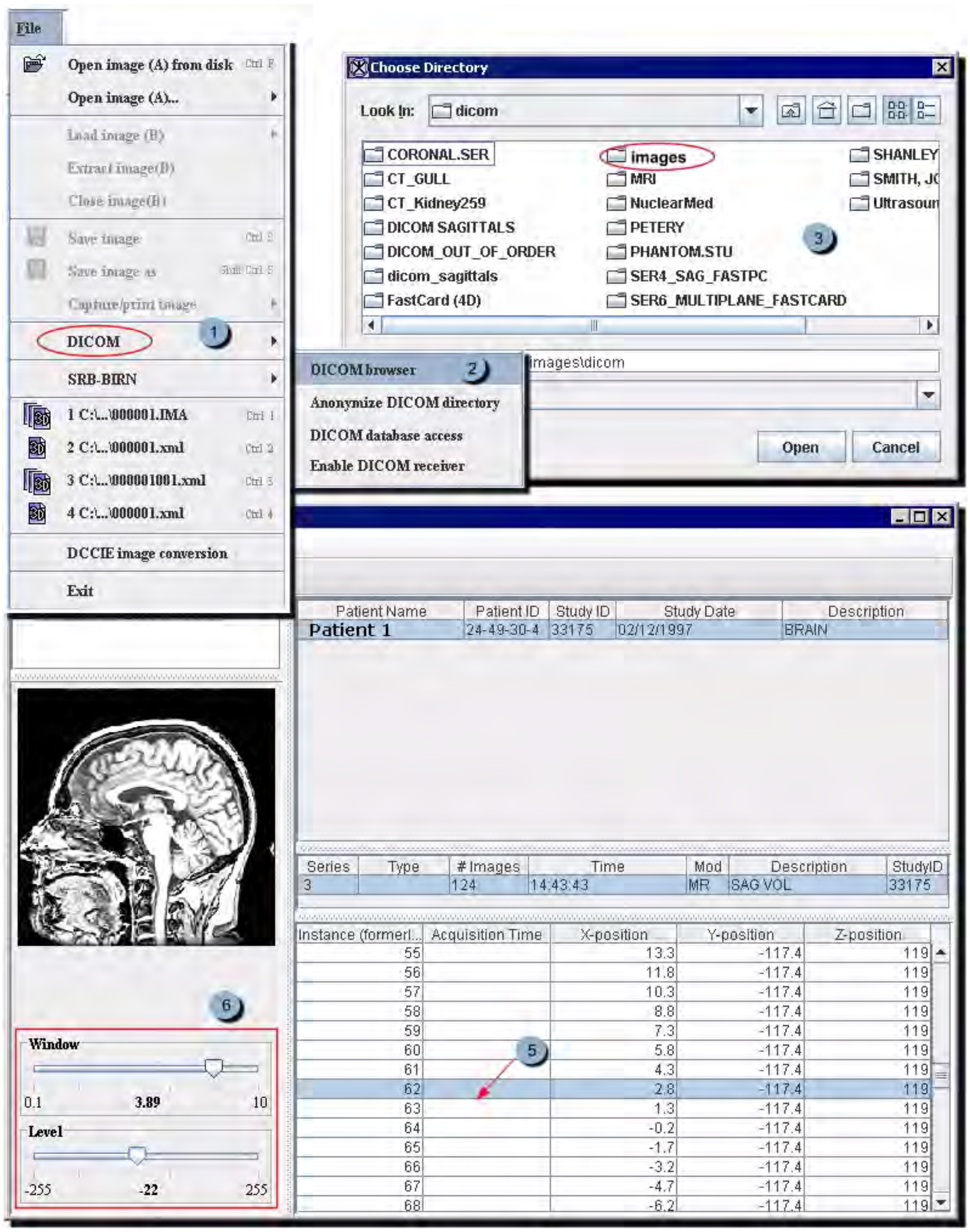


Figure 63. Opening the DICOM browser and parsing the image directory

---

Tip: You can select consecutive slices by clicking the first slice, pressing and holding down Shift, and then clicking the last slice. To select nonconsecutive slices, press and hold down Ctrl and then click each slice.

To adjust image contrast and brightness in the DICOM browser

The Window and Level sliders that appear in the left bottom panel of the DICOM browser allow you to adjust both the contrast and brightness in the image being displayed. Refer to Figure 63-6.

1. Select a slice from the slice listing in the DICOM browser.
2. Move the Window slider to the right to lessen contrast or to the left to intensify contrast.
3. Move the Level slider to the right to increase brightness or to the left to decrease brightness.

To open selected images in the image dataset

 One of the advantages of using the DICOM browser is that you can quickly identify which images in the dataset you wish to work with. Once you've determined the required images, you can open just the slices you selected.

1. Display an image dataset in the DICOM browser.
2. Browse through all of the images in the dataset to identify the slice numbers of the images.
3. Select each of the appropriate slice numbers.
4. Select the Open Selected Images icon. MIPAV opens all of the slices in an image window. You can now use the Image Slice slider in the MIPAV window to view and work with each slice.

To view another image dataset

 Once you've opened the DICOM browser, you can quickly switch between working directories. In order to do that:

1. Click the New Top Directory icon. The Choose Directory dialog box opens.
2. In the dialog box, select a directory of interest.Click Open.
3. The DICOM browser appears showing the directory tree in the panel on the upper left of the window.

4 Navigate, if necessary, to the correct subdirectory and select it.

5 Now, click Parse Directory. Information about the new image dataset replaces the information about the previous dataset in the three panels on the right of the window. See also "DICOM browser interface summary" on page 157.

To create a movie from the image dataset

 Using the DICOM browser you can create an *.AVI file or movie from the selected dataset or chosen slices.

1 Obtain and install, if you have not already done so, the following file from the Sun Java web site: `jmf-2_1_1e-windows-i586.exe`. This file installs file formats necessary for creating movies.

2 Display an image dataset of interest in the DICOM browser.

3 Browse through all of the images in the dataset to identify the slice numbers of the images you want to appear in the movie.

4 Select each of the appropriate slice numbers. Note that in order to create a movie, you must select at least two slices.

5 Select the Extract image to AVI movies icon. The Set AVI Options dialog box (Figure 64) opens.

6 Specify whether to subsample the image by 1, 2, 3, or 4.

7 Click OK. The Save dialog box opens.

8 Type a name for the movie in File Name.

9 Click Save. The Choose Type of AVI File dialog box (Figure 65) opens.

10 Select one of the methods of writing the file.

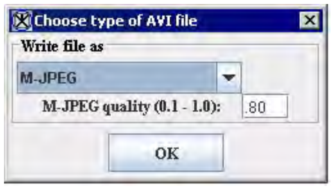
11 Type the value for the M-JPEG quality if you selected M-JPEG as the writing method.

12 Click OK. A pop-up window appears with the status.

When the algorithm finishes running, the pop-up window closes. The movie is saved in the directory you specified, and, depending on the file format, the movie may appear in a new window.

Subsample image	Subsamples each image dimension by a factor of 1, 2, 3, or 4.
Frame rate	Specifies how many frames, or slices, should appear in 1 second.
OK	Applies the parameters that you specified.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not save the file.

Figure 64. Set AVI Options dialog box



Write file as	Specifies the file format in which to save the movie. Choose one of the following: • 24-bit uncompressed RGB, 8-bit RLE with LUT, Quicktime movie, M-JPEG, IR32, IR41, Indeo Video 5, MS-MPEG4 V1
M-JPEG quality (0.1 - 1.0)	Specifies the level of quality, which may be from 0.1 to 1.0, in the M-JPEG file type. This box is only available if you selected M-JPEG in the Write file as box.
OK	Applies the parameters that you specified.

Figure 65. Choose Type of AVI File dialog box

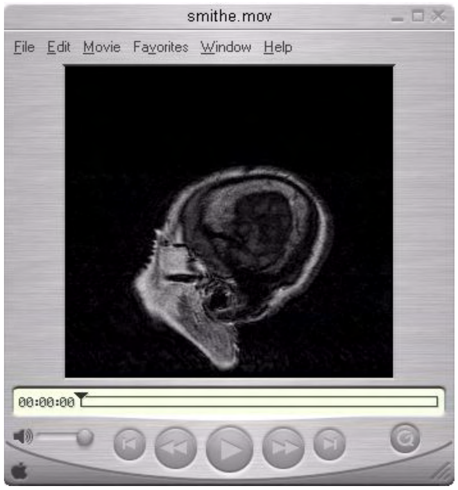


Figure 66. Quicktime movie—one of the eight supported file formats for making movies from DICOM images

To configure columns in the DICOM browser

You can customize the DICOM browser by choosing which columns to display. In order to do that:

1. Click the Configure Columns icon.
2. In the Configure DICOM Columns dialog that appears, select a field from the left dialog window and then click the Add button. The field now appears in the Selection window.
3. Repeat step 2 as many times as necessary.
4. Then, use the Move Up and Move Down buttons to set the column order. Refer to Figure 67.
5. The field at the top of the list will appear as the leftmost column in the browser. You can easily change the order by working within the Selection box. Click a column name to select it, and then click Move Up or Move Down button to move the column up or down in the list.

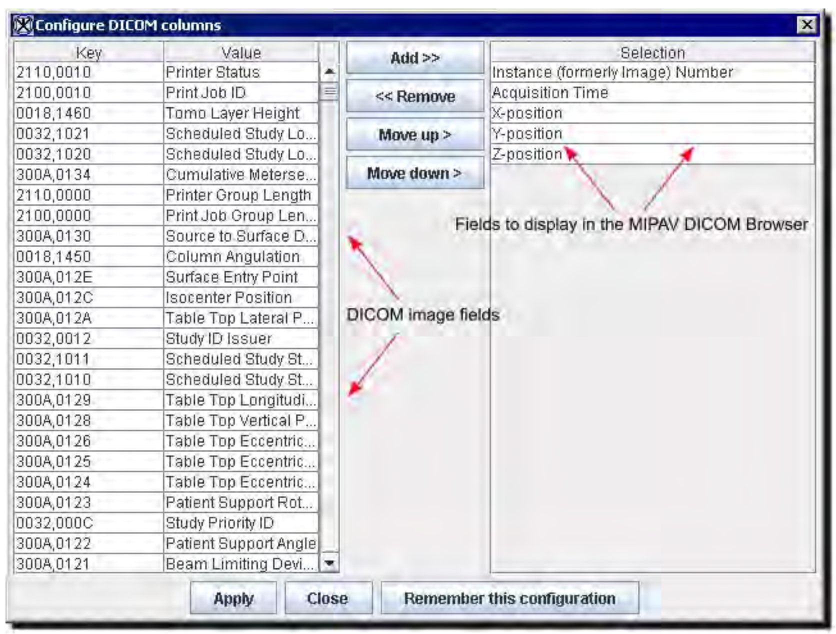


Figure 67. Configure DICOM columns dialog box

Selecting all rows (slices)

To select all rows in the DICOM browser (or all slices in the dataset), simply click the Select All Rows icon on the browser's toolbar.

DICOM BROWSER INTERFACE SUMMARY

Menu: File	**Disregard series numbers** – DICOM images use the DICOM Universal Identifier (UID) stored in the DICOM header to name the image files on the local disk. This long string of numbers and dots is designed to be unique among all DICOM images ever created. The **Disregard series numbers** option, if checked, disregards this number when shows images in MIPAV DICOM browser. Exit – exit the browser.
Toolbar	– Open Selected Image(s); – Extract Images to AVI Movie; – Parse Directory; – Configure DICOM Columns; – New Top Directory (use to open a new image dataset); – Select All Rows.

Sending and retrieving DICOM images

You can use MIPAV to query a DICOM-compatible archive or an imaging device that has, or is connected to, a DICOM server. A DICOM server is an application that processes DICOM query and retrieve requests. Because MIPAV is not equipped with a server, you cannot use this process to obtain DICOM files from another researcher who is also using MIPAV. You must use the receive process outlined in "Receiving and sending image files" on page 173 instead. You can then select images from the query results and store them on your hard drive or another destination. The tasks involved in sending and retrieving DICOM images include the following:

- Setting up the hosts table, refer to page 158
- Testing server connections, on page 168
- Posing queries and retrieving images, refer to page 169
- Receiving and sending image files, on page 173

To set up the hosts table and to query, retrieve, receive, and send DICOM files, you use the DICOM Communication Panel dialog box. To view this dialog box, select File > DICOM Database Access in the MIPAV window. The DICOM Communication Panel dialog box opens.

This dialog box includes four tabbed pages:

- **QR Client**, which appears by default when the dialog box is first opened. Use this page to search for an image for a particular patient, physician, and time period.
- **Send** allows you to select the files to be sent to the DICOM imaging device, to select the specific imaging device from a list of available devices to which you want to send the files, and to send the files. This page also allows you to test the connection to the device that you selected.
- **Hosts** allows you to create a servers to which you are connected, edit the settings for servers, delete servers, and set the default server. It also allows you to create, edit, or delete storage destinations and set the default storage destination.
- **Help** provides instructions for setting up the hosts table and for sending and receiving images.

The following sections discuss these pages and explain in detail the contents of each page and how to use them.

Setting up the hosts table

Before creating queries or receiving and sending images, you must set up the hosts table, which is on the Hosts page (Figure 68) in the DICOM Communication Panel dialog box. The hosts table stores information (application entity [AE] title, alias, IP address, and port number) about the devices to which you want to connect. Conversely, your computer's connection information must be stored in the hosts table of the devices to which you want to connect. Note that the title bar for the DICOM Communication Panel dialog box contains the internet protocol (IP) address for your workstation.

internet protocol (IP) address: A numeric ID that is used to identify your computer on a Transmission Control Protocol/Internet Protocol (TCP/IP) network. An example of a TCP/IP network is the Internet.

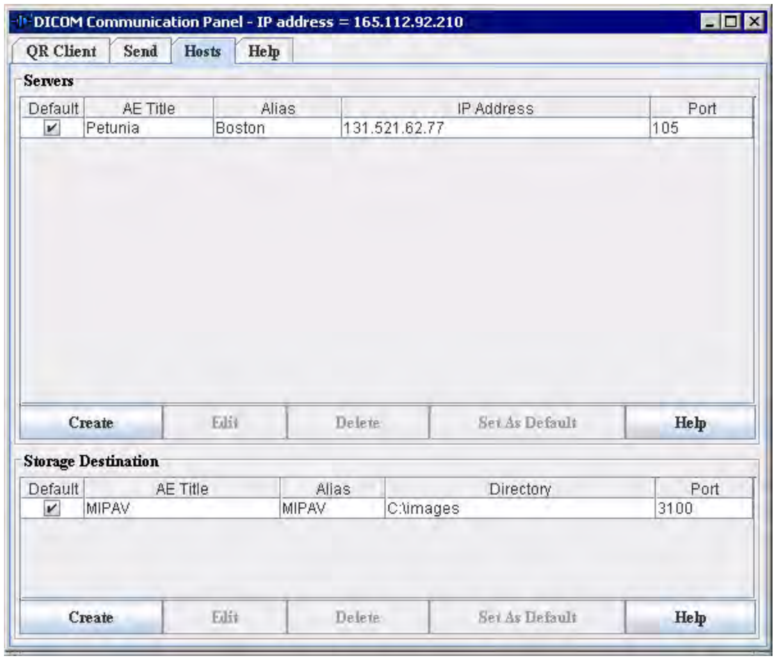
To display the Hosts page, select the Hosts tab in the dialog box. The Hosts page, shown in Figure 68, appears. The **Servers panel** on the Hosts page includes fields that contain identifying information for the devices to which you want to connect. (Although the panel is named Server, you should create an entry for any type of device to which you want to connect.) The Storage Destination panel holds entries that indicate where you want images to stored once you retrieve or receive them.

The following sections explain how to:

- Create, edit, or delete entries in the Servers panel, refer to page 159
- Create, edit, or delete entries in the Storage Destination panel, refer to page 165

CREATING, EDITING, AND DELETING SERVERS

You can create, edit, and delete entries in the Servers panel on the Hosts page (Figure 68). You must create an entry if you want to use MIPAV to query a server.



Servers	**Default**—Indicates that the device is the default. When you pose a query, data is sent to the default device.
	AE Title—Lists the title of the device's DICOM application entity. The AE Title is often set by the device's system administrator. The hosts table associates the AE Title with the IP address. The AE Title can be 16 alphanumeric characters or less. **Note that the AE Title is case sensitive.**
	Alias—Specifies the alternative name for the device. The alternative name can be up to 16 alphanumeric characters.
	IP Address—Specifies the internet protocol address for the device. An IP address identifies a device on a Transmission Control Protocol/Internet Protocol (TCP/IP) network. (An example of a TCP/IP network is the internet.) The hosts table associates the IP address with the AE Title. IP addresses are written as four numbers separated by periods.
	Port—Identifies the port for the device's application program that you query, retrieve, receive, and send image files. A port is a logical connection endpoint that is the means that a program uses to identify a specific program running on a computer on a network (such as TCP/IP).
Create	Allows you to add a new DICOM server. When you click Create, the Create Server dialog box opens.
Edit	Allows you to edit a currently listed DICOM server. When you select the server you want to modify and click this button, the Edit Server dialog box appears.

Figure 68. Hosts page in the DICOM Communication Panel dialog box

Delete	Deletes the DICOM server from the list of servers. To delete a server, select the server and then click this button. A message appears asking you to confirm the deletion.
Set as default	Allows you to select the server to be used as the default server. Select a server and then click this button. The check mark moves to the Default check box beside the server you selected.
Storage destination	**Default**—Indicates, when selected, the default storage destination. When you retrieve or receive image data, it is stored in the default storage destination.
	AE Title—Lists the title of the DICOM application entity. If the storage destination is on your machine, the AE title is typically preceded by MIPAV_. An example of a valid AE title is MIPAVSMITH23. The AE Title can be 16 alphanumeric characters or less. **Note that the AE Title is case sensitive.**
	Alias—Specifies an alternative name for the storage destination. Alphanumeric characters are accepted.
	Directory—Shows the full path name of the directory where images are stored. Alphanumeric characters are accepted.
	Port—Identifies the port that your computer uses to receive images. By default, the port number is 3100.
Create	Allows you to add a new storage location. When you click Create, the Create Storage Destination dialog box appears.
Edit	Lets you modify a listed storage location. Select the location you want to modify and click this button. The Edit Storage Destination dialog box opens.
Delete	Allow you to delete a listed storage location. To delete a storage location, select the location and then click this button.
Set as default	Allows you to select the storage location that is used by default. Select the location and click this button. The check mark moves from the current default location to the Default check box beside the location you selected.

Figure 68. Hosts page in the DICOM Communication Panel dialog box (continued)

Connecting to a DICOM device

Before you begin, obtain the AE title, alias, IP address, and port number of the device to which you want to connect. The connection information for the NIH Kodak Cemax-Icon PACS appears when you click the Help tab. The Help page appears (Figure 69). You must also contact the administrator of the device (including the NIH Kodak Cemax-Icon PACS) so that your connection information can be stored in their hosts table as well. Once you have all of the information, you can create an entry in the hosts table entry.

Contact Alberto Goldszal, CC, 6-7700 ext.226 to have your machine properly added to the server host table. You will need your IP address which can be found on the title bar of this frame.

Figure 69. The Help page in the DICOM Communications Panel dialog box

To create servers in the Server panel

1 Click Create in the Servers panel on the Hosts page in the DICOM Communications Panel dialog box. The Create Server dialog box opens (Figure 70).

AE title	Lists the title of the device's DICOM application entity. The AE Title is often set by the device's system administrator. The hosts table associates the AE Title with the IP address. The AE Title can be 16 alphanumeric characters or less. **Note that the AE Title is case sensitive.**
Alias	Specifies the alternative name for the device. The alternative name can be up to 16 alphanumeric characters.
Address	Specifies the internet protocol (IP) address for the device. An IP address identifies a device on a Transmission Control Protocol/Internet Protocol (TCP/IP) network. (An example of a TCP/IP network is the internet.) The hosts table associates the IP address with the AE Title. IP addresses are written as four numbers separated by periods.
Port	Identifies the port for the device's application program that you query, retrieve, receive, and send image files. A port is a logical connection endpoint that is the means that a program uses to identify a specific program running on a computer on a network (such as TCP/IP).
OK	Applies the parameters that you specified and creates a new server.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not create a server.
Help	Displays online help for this dialog box.

Figure 70. Create Server dialog box

2 Type the required information in AE Title, Alias, IP Address, and Port (refer to Figure 68 for information on these fields).

3 Click OK when complete. The Create Server dialog box closes and the entry appears in the Servers panel.

To edit servers in the Servers panel

1 Select the server in the Servers panel that you want to edit.

2 Click Edit. The Edit Server dialog box opens.

3 Modify the information in AE Title, Alias, IP Address, and Port as required.

4 Click OK. The Edit Server dialog box closes, and the changed entry appears in the Server panel.

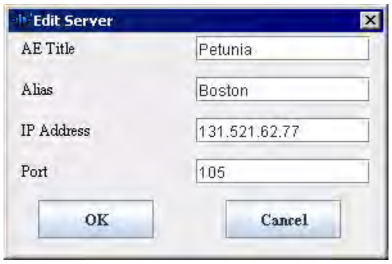


Figure 71. The Edit Server dialog box

To delete servers

1 Select the server in the Servers panel that you want to delete.

2 Click Delete. The Confirm Delete message opens.

3 Click Yes. MIPAV deletes the server from the Servers panel.

To select a default server

A default server is the server that is automatically selected when you first display the Hosts page of the DICOM Communications Panel dialog box. Unless you select another server as the default, it is the server to which MIPAV automatically sends images.

You can select the default server in either of the following ways:

- Double-click the server that you want as the default server in the Server panel.
- Select the server in the Servers panel that you want as the default server. The server is highlighted. Click Default.

A check mark moves from the check box for the previous default server to the Default check box beside the server you selected.

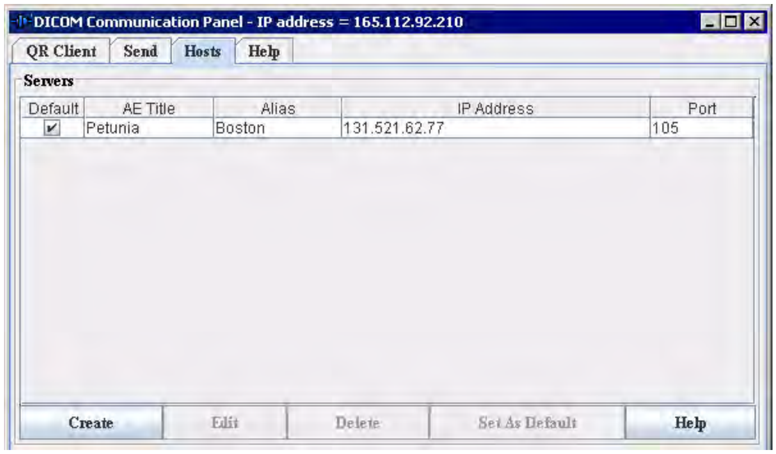


Figure 72. The default server Kodak is selected in the Servers panel on the Hosts page

CREATING, EDITING, AND DELETING STORAGE DESTINATION ENTRIES

You can create, edit, and delete storage destination entries in the Storage Destination panel of the Hosts tabbed window (Figure 73). The storage destination indicates where to store the image files you retrieve or receive. You must create at least one storage destination.

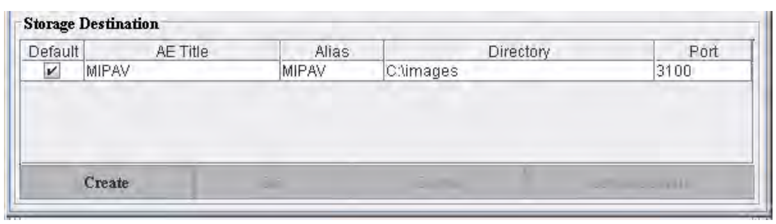


Figure 73. Storage Destination panel

To create storage destinations

1 Click Create in the Storage Destination panel on the Hosts page in the DICOM Communications Panel dialog box. The Create Storage Destination window opens (Figure 74).

2 Type the required information in AE Title, Alias, Directory, and Port (refer to Figure 74 for information about these fields).

3 Click OK when complete. The Create Storage Destination dialog box closes. The new entry appears in the Storage Destination panel.

To edit storage destinations

1 Select the storage destination in the Storage Destination panel that you want to change.

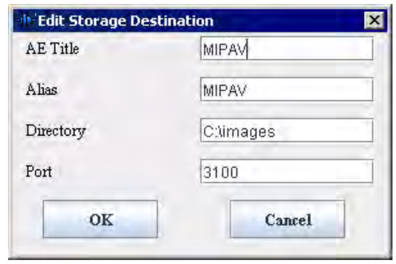
2 Click Edit. The Edit Storage Destination dialog box opens.

3 Modify the information in AE Title, Alias, Directory, and Port as required.

4 Click OK when complete. The Edit Storage Destination dialog box closes. The modified information appears in the Storage Destination panel.

AE title	Lists the title of the device's DICOM application entity. The AE Title is often set by the device's system administrator. The hosts table associates the AE Title with the IP address. The AE Title can be 16 alphanumeric characters or less. Note that the AE Title is case sensitive.
Alias	Specifies the alternative name for the device. The alternative name can be up to 16 alphanumeric characters.

Figure 74. Create Storage Destination and Edit Storage Destination dialog boxes



Directory	Shows the full path name of the directory where images are stored. Alphanumeric characters are accepted.
Port	Identifies the port for the device's application program that you query, retrieve, receive, and send image files. A port is a logical connection endpoint that is the means that a program uses to identify a specific program running on a computer on a network (such as TCP/IP).
OK	Applies the parameters that you specified and creates a new storage location.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not create a storage location.
Help	Displays online help for this dialog box.

Figure 74. Create Storage Destination and Edit Storage Destination dialog boxes (continued)

To delete storage destinations

1. Select the storage destination in the Storage Destination panel that you want to delete.
2. Click Delete. A Confirm Delete message opens.
3. Click Yes. MIPAV deletes the storage destination from the Storage Destination panel.

To select a default storage destination

A default storage destination is the one that is automatically selected when you initially display the Hosts page of the DICOM Communications Panel dialog box. Unless you select another storage destination as the default, MIPAV automatically stores images in this location. You can select the default storage destination in either of the following ways:

- In the Storage Destination panel, double-click the entry that you want as the default storage destination.
- In the Storage Destination panel, select the entry that you want as the default storage destination. The server is highlighted. Click Default.

A check mark moves from the check box for the previous default storage destination to the Default check box beside the storage destination you selected.

Testing the connection

Before querying, retrieving, or sending images, you may want to test the connection between your machine and the device that you want to receive the images.

To test the connection

1. Click Send in the DICOM Communication Panel dialog box. The Send page appears. See Figure 78.
2. Select the pathway you want to test by selecting the destination from the drop-down list near the center of the Send tabbed window. The destinations are the entries from the Servers panel in the Hosts tabbed window.
3. Click Test connection.
4. If there is a problem, connection failure messages appear in the Send Status panel as shown in Figure 75.

To resolve connection errors

Do the following:

- Make sure the AE Title, Alias, IP Address, and Port number for the device are entered correctly in your hosts table.
- Contact your system administrator for the device to which you want to send the images. Make sure your AE Title, Alias, IP Address, and Port number was correctly added to the hosts table for that device.
- If you are attempting to send images to another MIPAV machine, make sure that the receiver in the other MIPAV application is turned on. To turn on the receiver, ask them to select File > DICOM Receiver On/Off in the MIPAV window.

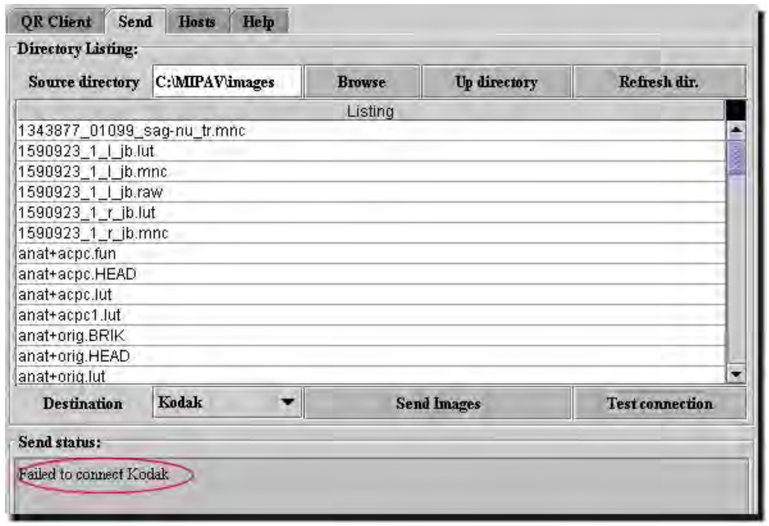


Figure 75. Error message in the Send status panel on the Send page

Posing queries and retrieving images

Use the QR Client page (Figure 76) in the DICOM Communication Panel dialog box to query a DICOM-compatible server, or device, and then retrieve files from the query results.

Note: Before you begin, make sure your hosts table has the name of the device you want to query or from which you want to receive images. Conversely, your computer's connection information should also be in the other device's hosts table. For more information on setting up a hosts table, see "internet protocol (IP) address: A numeric ID that is used to identify your computer on a Transmission Control Protocol/Internet Protocol (TCP/IP) network. An example of a TCP/IP network is the Internet." on page 159.

To pose queries

1 Click QR Client in the DICOM Communication Panel dialog box. The QR Client page appears (Figure 76).

2 Construct your query by completing the fields in the Patient Query Information and Query Duration panels. As you fill in the information, keep the following in mind:

- You must specify at least one of the following: patient name, patient ID, study number, or physician.
- If the exact name, ID, or study number is not known, you can use the asterisk as a wildcard character. For example, if you type "*****o*****ston" in the patient name field, all names that match this pattern, such as Houston or Johnston, appear in the result.

3 Click Send Query.

The query result appears in the Query Result panel. To retrieve images, go to "To retrieve images" on page 172.

To erase the query

To erase the query and start again, click Clear in the Patient Query Information panel.

Patient Query Information	
Patient Name	Indicates the full name of the patient. Alphanumeric characters are accepted. You can use an asterisk as a wild card character.
Patient ID	Indicates the primary hospital identification number or code for the patient. Numeric characters are accepted.

Figure 76. QR Client page in the DICOM Communication Panel dialog box

Study Number	Displays the user- or equipment-generated study identifier.
Physician	Indicates the physician(s) responsible for patient care at the time of the study.
Query Duration	
Displays the time period the query covers. The options are: Today, One day, One week, One month, Three months, Six months, One year, Start date and End date.	
Start Date	Indicates the date when medical specialist began acquiring study information.
End Date	Indicates the date when medical specialist completed the acquisition of study information.
Today's Date	Displays the current date.
Query Result	
Pat. Name	Patient Name. Indicates the full name of the patient. You cannot edit this field; it is read-only.
Pat. ID	Patient Identification. Displays the primary hospital identification number or code for the patient. You cannot edit this field; it is read-only.
Referring Physician	Indicates the name of the physician who referred the patient. You cannot edit this field; it is read-only.
Query Retrieval Information	
Status	Provides an update of the query retrieval process.
#	Indicates the image number that is currently being retrieved.
Source	Indicates the DICOM server from which the images are being retrieved.
Destination	Indicates where the image is stored (storage destination).
Error	Indicates whether an error has occurred.
ID	Displays the unique ID that is assigned to each query.

Figure 76. QR Client page in the DICOM Communication Panel dialog box (continued)

To view information at different levels

You can view information at the patient, study, series, or image level. To go up or down a level, click Up or Down. Note that the level that is currently displayed appears at the left of the Up and Down buttons.

To sort the query result list

Click Pat (patient), Pat ID (patient ID), or Referring Physician in the Query Result panel. The query result list is sorted in ascending order based on the column you chose. If you click the column title a second time, the list is resorted, this time in descending order, but again based on the title you chose.

To retrieve images

After you pose a query and receive the results, you can select the files that you want to retrieve from the list of query results.

1 Turn on the DICOM receiver if you have not already done so.

To turn on the receiver, click File > DICOM Receiver On/Off in the MIPAV window. A check mark appears next to the option.

2 Select the images you want to copy from the Query Result panel on the QR Client page.

3 Click Retrieve Image in the Patient Query Information panel to copy the images to the default storage destination indicated in the hosts table.

If you need to check the default storage destination:

a Click Hosts to display the Hosts page

b View the storage destinations in the Storage Destination panel

c Click QR Client to return to the QR Client page

As the images are copied, status information appears in the Query Retrieval Information panel.

Note: If you have problems connecting to the device, you may want to test the connection. See Section "Testing the connection" on page 168 for details.

To sort the query retrieval information

To sort the information in the Query Retrieval Information panel, you can click Status, #, Source, Destination, Error, or ID.

To cancel the retrieval

Click Cancel in the Query Retrieval Information panel.

Receiving and sending image files

When you receive image files, you do not pose a query or retrieve files from a list; you passively receive the images that are sent to you from another device. You can receive images from any device, including a computer running another copy of MIPAV, if your DICOM Receiver is turned on and the appropriate connection information was entered in the hosts table.

You can send DICOM images to DICOM-compatible servers, imaging equipment, or to other MIPAV applications that are up and running. Only a copy of the image files are sent, the original files remain in your source directory.

To receive image files

1 Turn on the DICOM receiver if you have not already done so. To turn on the receiver, click File > DICOM > Enable DICOM Receiver in the MIPAV window. A check mark appears next to the command.

Figure 77. The Enable DICOM Receiver command is located on the File menu in the MIPAV window

2 Click Hosts. The Hosts page appears.

3 Make sure that the device that is sending the images has an entry in the Servers panel. (Conversely, your connection information should be in their hosts table.)

4 Make sure that the default storage destination in the Storage Destination panel is the location where you want to store the images.

5 Wait until the images are sent to you. After the images are sent, you can turn off the DICOM receiver by unchecking the File > DICOM >Enable DICOM Receiver option.

To send image files

1 Click Send. The Send page (Figure 78) appears.

2 Verify that the directory listed in Source directory is the directory in which the image files you want to send are located.

If it is the correct directory, the image files that you want to send appear in the Listing panel.

If it is not the correct directory, do either of the following:

- Click Browse to navigate to the correct directory.
- Type the full path of the directory in Source directory. Click Up Directory to view the parent directory.

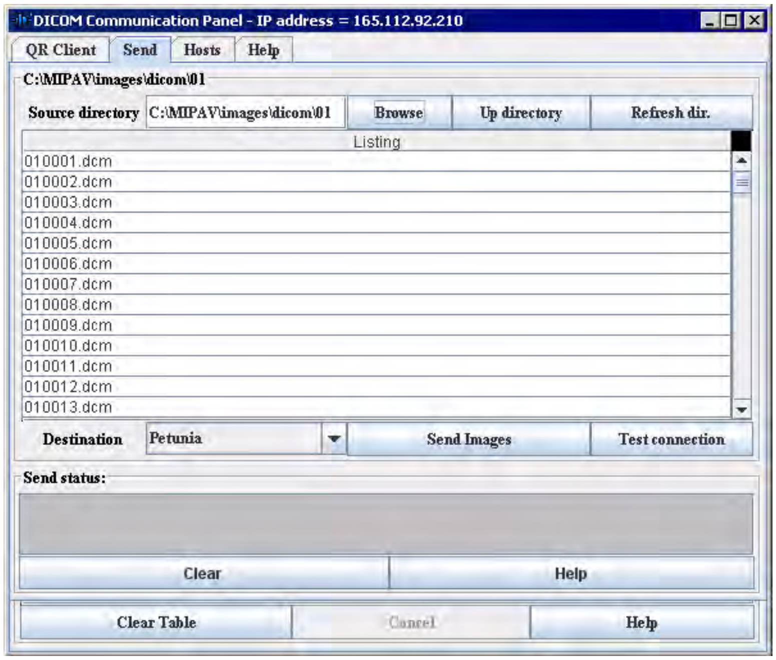
3 Select the destination to which you want to send the image files in Destination.

4 Select the images you want to send in the Listing panel.

5 Click Send Images to transmit copies of the selected images. Status messages that appraise you of the status of the send process appear in the Send status panel.

Note: The destinations reflect the entries in the Server panel on the Hosts page. If the destination does not appear in Destination, you need to create a new server in the Server panel on the Hosts page. For instructions, refer to page 159.

Tip: To select more than one image file, hold down the <CTRL> or <SHIFT> key while you click the files you want to send.



Source directory	Indicates the full path name for the directory where the image files you want to send are kept. To change the source directory, use Browse or Up directory.
Browse	Allows you to navigate to the directory for the images you want to send. When you click this button, the Open dialog box appears.
Up directory	Moves to the parent directory of the directory shown in Source Directory.
Refresh dir.	Updates the image files shown in the Listing panel.
Listing panel	Displays a read-only list of the image files in the source directory.
Destination	Indicates where the image files should be sent. Destinations in the drop down menu are the same as those listed in the hosts table (*Server* panel).
Send Images	Transmits copies of the selected image files to the destination indicated in Destination.
Test connection	Tests the connection between your server, or device, and the receiving or sending device.
Send status	Displays messages about the send process, such as whether it was completed successfully.
Clear	Erases all messages from the Send status panel.

Figure 78. The Send tab of the DICOM Communications Panel dialog box

Displaying and editing DICOM tag information

Once an image file is copied to the storage destination (usually your hard drive or a local device such as a zip drive), you can view and edit the information stored in the DICOM tags. As an option, you can display any of the DICOM tags as an overlay on the image, selecting the specific tags to display. You can also use MIPAV's **anonymize feature** to protect patient confidentiality. The anonymize feature allows you to delete information from selected DICOM tags in a single step from one image or from a whole directory of images. Refer to "Protecting patient privacy using Anonymize" on page 181.

To display the DICOM tags in an overlay over the image

1 Open a DICOM image in an image window.

2 Select Image > Show Image/DICOM Overlay (if it is not already selected) in the MIPAV window. The image window opens with the DICOM information overlaying the image (Figure 80).

Note: You can open a DICOM image file as you would any other image file. Refer to Chapter 3 for more details.

To choose which DICOM tags appear in the overlay

You may want only particular DICOM tags to appear in the overlay and at a specific spot in the overlay.

1 Open a DICOM image in an image window.

2 Select Image > Show Image/DICOM Overlay (if it is not already selected) in the MIPAV window. The image window opens with the DICOM information overlaying the image.

3 Select Image > DICOM Overlay Options. The DICOM Overlay Options dialog box opens.

4 Decide where you want to display the DICOM tags—in the upper left corner, lower left corner, upper right corner, or lower right corner of the image.

5 Select a button under the Upper left corner, Lower left corner, Upper right corner, or Lower right corner panel as appropriate. The Select DICOM Tag for Overlay dialog box opens.

6 Select the DICOM tag that you want to display in the overlay.

7 Click OK. The Select DICOM Tag for Overlay dialog box closes, and the selected DICOM tags and names appears on the buttons in the panel on the DICOM Overlay Options dialog box.

8 Click OK. The DICOM Overlay Options dialog box closes. The tags you selected appear on the overlay on the image.

See Figure 80.

If you decide that you do not want the tag to display on the overlay, simply select the button in the DICOM Overlay Options dialog box and select [Blank Overlay] at the top of the Select DICOM Tag for Overlay dialog box.

In the Select DICOM Tag for Overlay dialog box

- To list tags in the ascending numerical order, click Tag;
- To list tags in the descending numerical order, click SHIFT+Tag;
- To list names of DICOM tags in alphabetical order, click Name;
- To list names of DICOM tags in reverse alphabetical order, click SHIFT+Name.

To show grid on a DICOM image

Open the image and then call Image > Show Overlay Grid. To asses the grid options, call Image > Grid Options.

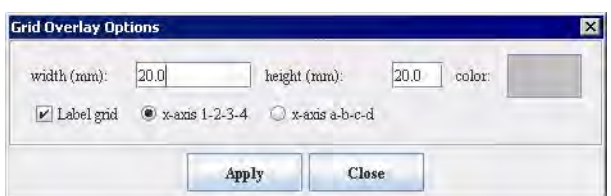
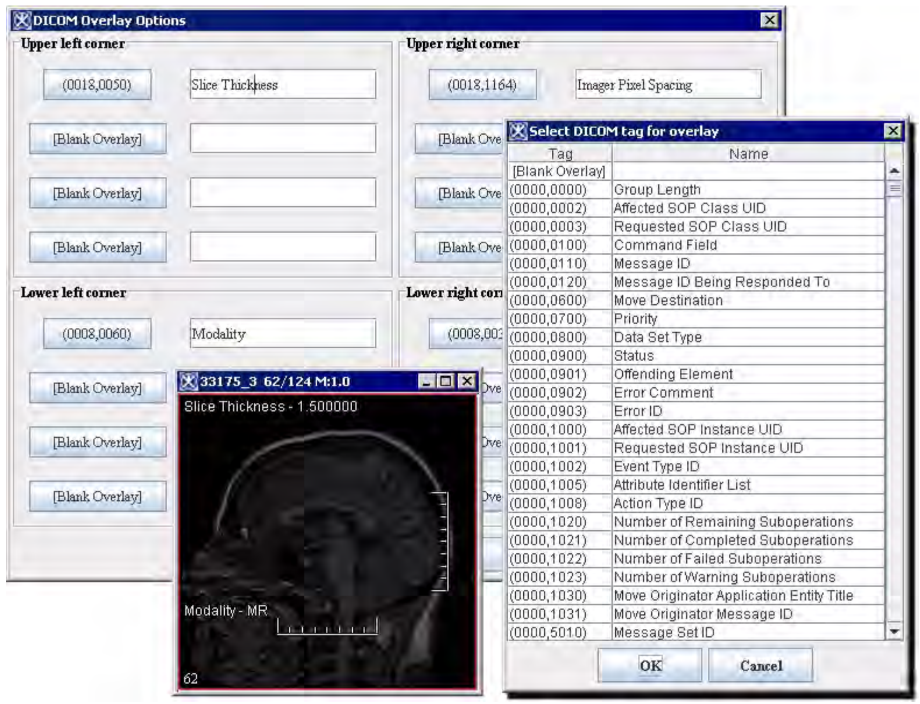
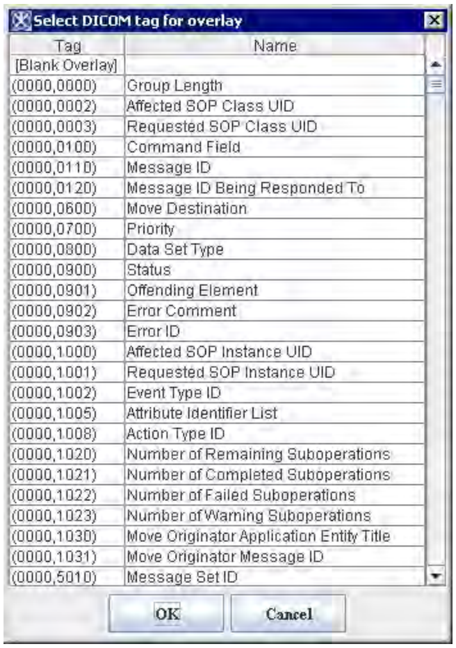


Figure 79. The Grid Overlay Options dialog box



Upper left corner	Specifies the DICOM tags you want to display in the upper left corner of the image. You can display from 1 to 4 DICOM tags.
Lower left corner	Specifies the DICOM tags you want to display in the lower left corner of the image. You can display from 1 to 4 DICOM tags.
Upper right corner	Specifies the DICOM tags you want to display in the upper right corner of the image. You can display from 1 to 4 DICOM tags.
Lower right corner	Specifies the DICOM tags you want to display in the lower right corner of the image. You can display from 1 to 4 DICOM tags.
OK	Displays the tag you selected on the button in the DICOM Overlay Options dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

Figure 80. DICOM Overlay Options dialog box



Tag	Lists each DICOM tag beside its name.
Name	Lists each DICOM name for the tag.
	• To list tags in the ascending numerical order, click Tag; • To list tags in the descending numerical order, click SHIFT+Tag; • To list names of DICOM tags in alphabetical order, click Name; • To list names of DICOM tags in reverce alphabetical order, click SHIFT+Name.
OK	Displays the tag you selected on the button in the DICOM Overlay Options dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

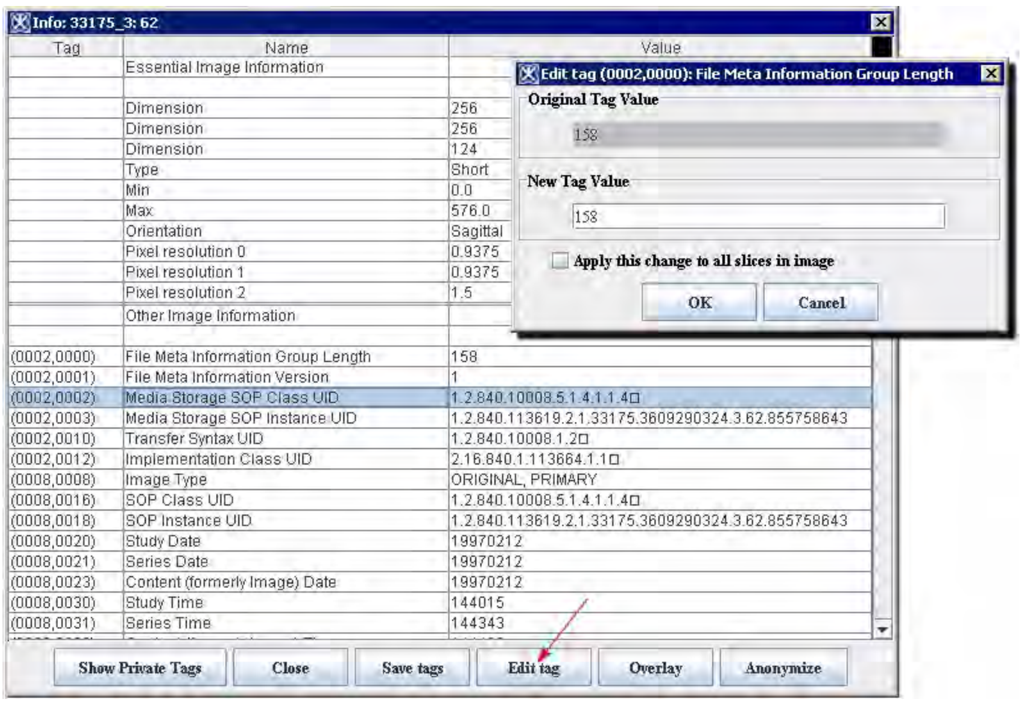
Figure 81. Select DICOM Tag for Overlay dialog box

To remove the DICOM tag overlay from the image

Show Image/DICOM Overlay is a toggle command. You click the command to turn it on or to turn if off. A check mark appears in the check box beside the command if it is on. To remove the DICOM tag overlay from the image, simply select Image > Show Image/DICOM Overlay to turn it off.

To edit DICOM tag information

1. Open a DICOM image.
2. Click Image > Attributes > View Header. The Info dialog box opens (Figure 82).
3. In the dialog box, highlight the tag which you would like to edit, and then press the Edit Tag button.



Tag column	Lists the DICOM tags.
Name column	Lists the name of the DICOM tags.
Value column	Lists the value assigned to each DICOM tag.
Show/Hide private tags	Shows or hides private tags (those that contain patient private information).
Close	Closes the dialog box.
Save tags	Saves the values of the tags that were changed.
Edit tag	Allows you to make changes to the values for the tags.
Anonymize	Anonymizes the image.

Figure 82. Info dialog box (DICOM)

4 Click Edit Tag. The Edit Tag dialog box appears (Figure 82).

5 Type the new value in the New Tag Value text box. To apply the new tag value to all of the slices in the image dataset, click Apply this change to all slices in image.

6 Click OK. The tag information is changed.

Note: You cannot edit the following tags in the Info window: Dimension, Type, Min, Max, Orientation, Pixel Resolution, Unit of Measure, and Transformation Matrix. The Dimension and Unit of Measure tags are edited in the Image Attributes window.

Protecting patient privacy using Anonymize

MIPAV's anonymize feature allows you to remove information from a selected subset of DICOM tags. The tags in this subset are those indicated in *DICOM Supplement 55: Attribute Level Confidentiality*. These tags include, but are not limited to, the patient's name, ID, birth date, sex, ethnic group, referring physician, age, size, weight, and occupation.

To anonymize the current DICOM image

1 Open a DICOM image.

2 Click Image > Attributes > View Header in the MIPAV window. The Info dialog box opens.

3 Click Anonymize. The Anonymize Sensitive Info dialog appears.

4 In the dialog box, select the tags you want to anonymize.

- To select all of the tags, click Select all.
- To remove all of the check marks from the check boxes, click Clear.

The tags that are dimmed (appear grayed) are not included in the DICOM image file.

5 Click OK. A confirmation message appears.

6 Click OK to confirm. The information in each specified tag is deleted from the image file. Although the information in these tags are cleared, the tags remain in the image file. Refer to Figure 83.

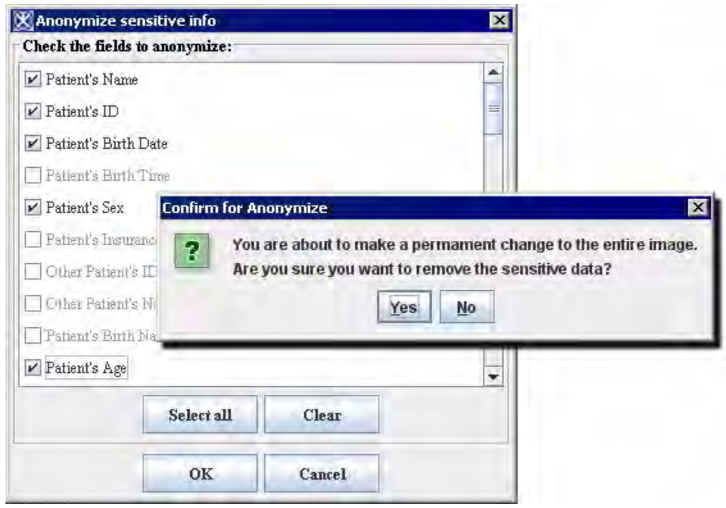


Figure 83. The Anonymize Sensitive Info dialog box and the confirmation message

Note: The lists of tags in the Info dialog box does not update right away. Also, the name of the displayed image on the title bar is not saved into the file.

To anonymize any image dataset

You can also use the Anonymize command to anonymize several image series at one time. Refer to Figure 84, Figure 85, Figure 86, and Figure 87.

Note: This function can recursively anonymize images in subdirectories of the root directory.

1. Select File > DICOM > Anonymize DICOM directory. The Choose Directory dialog box opens.
2. Select a starting or root directory, which is a starting point for the selection. You can then select the image subdirectories to anonymize.
3. Click Open. The Anonymize DICOM Directory dialog box opens.
4. In the Anonymize DICOM Directory dialog box, click File > New Directory. The Choose Directory dialog box opens. Refer to Figure 84-1.
5. Select one of the following:

- Selected directory
- Subdirectory
- No directory name change

6 Then, click Browse beside Image destination directory. The Select Destination Directory dialog box opens. See Figure 84-2.

7 Select a destination directory. See also a note on page 187.

8 Click Select. The directory you selected appears in the Image destination directory box. It also replaces the name in the Translation/Key file destination directory box.

9 Now you can decide where to store the translation/key file.

---

translation / key file records the original names in the files and correlates them to the new anonymous names. The name of the key file is patient.key.doc.

By default, the translation/key file is kept in the same directory as the images; however, if this is undesirable, you may choose a new location for the file. If you decide to change the location, go to the next step. If you decide against changing the location, go to step 12.

10 Click Browse beside Translation/Key file destination directory if you want to select another destination for the translation/key file. The Select Destination Directory dialog box opens. See Figure 84-3.

11 Use the dialog box options to browse you disk to select a designated catalog. Then click Select. The new destination appears in the Translation/Key File destination directory box.

Note: The name of the anonymized files depend on the anonymous name, which is set later in step 14.

---

12 Click Tag Options to display the Tag Options page. See Figure 84-4.

The directory selection process on the Directory page provides the images to anonymize and the location of where to put them when done. However, it does not select what to remove from the images or what information to call the images when done. This is the goal of the Tag Options page. It allows you to choose the tags to remove from the images being processed and it gives you the opportunity to give an easily remembered name to the destination images.

13 Select the DICOM tags you want to anonymize. To select all of the tags, click Select all. To remove all of the tags, click Clear. See Figure 84-5.

14 Type the anonymous name for the image files in Anonymous name.

By default, a preset random value appears in Anonymous name, but you can replace this value with a more familiar name.

15 Type a value in Sequence beginning. When more than one image source directory is chosen, the sequence beginning number affects the name of the image file. The first image set is given the sequence number; the next image is given a number one higher.

Example 1: The first image set is titled d0884_0. The second image set was chosen to be anonymized at the same time, it would be saved d0884_1.

16 Click Directory to return to the Directory page.

17 Select the directories in the Image source directory table that you want to anonymize.

18 Click Run. MIPAV displays the Logging page, where it displays messages about each of the images being processed until they are all processed. To stop the process, click Stop. The log appears in the Log tab. See See Figure 84-6.

19 To close the Anonymize DICOM Directory window after all images are processed, click Close.

Example 2: The number of image directories chosen affect precisely where the processed images are stored. When one directory is chosen as the source in step , all anonymized DICOM files are placed directly in the chosen destination directory. However, selecting more than one source directory in step 17, the anonymized DICOM files are placed into subdirectories of the Destination directory (refer to the following table).

If you select . . .		Then files are placed in . . .
Source directories	/images/a /images/b	/destination/a /destination/b
Destination directory	/destination	

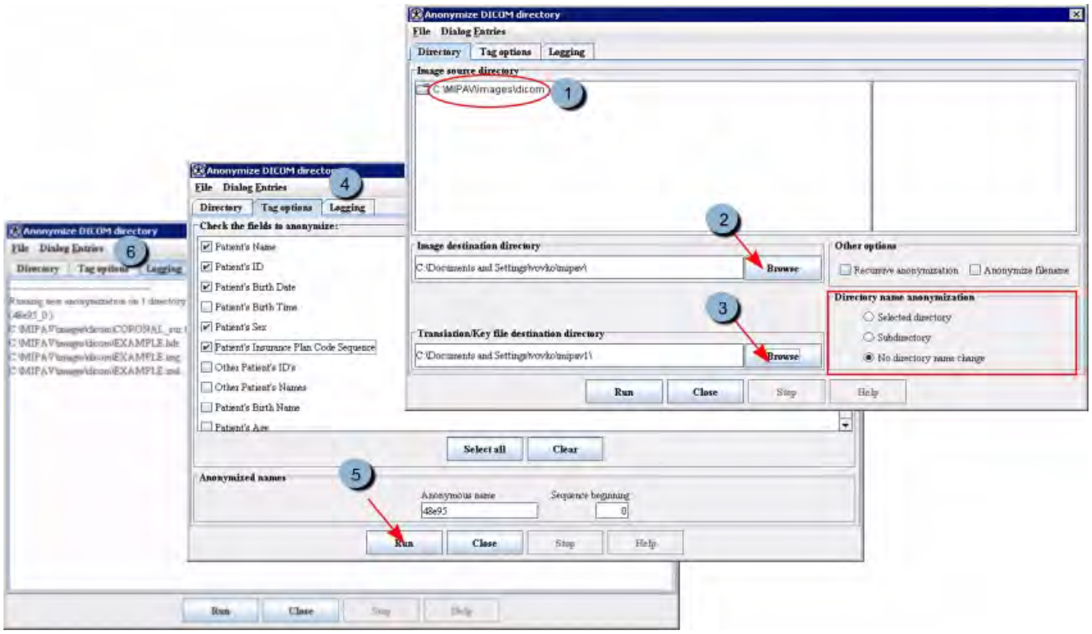
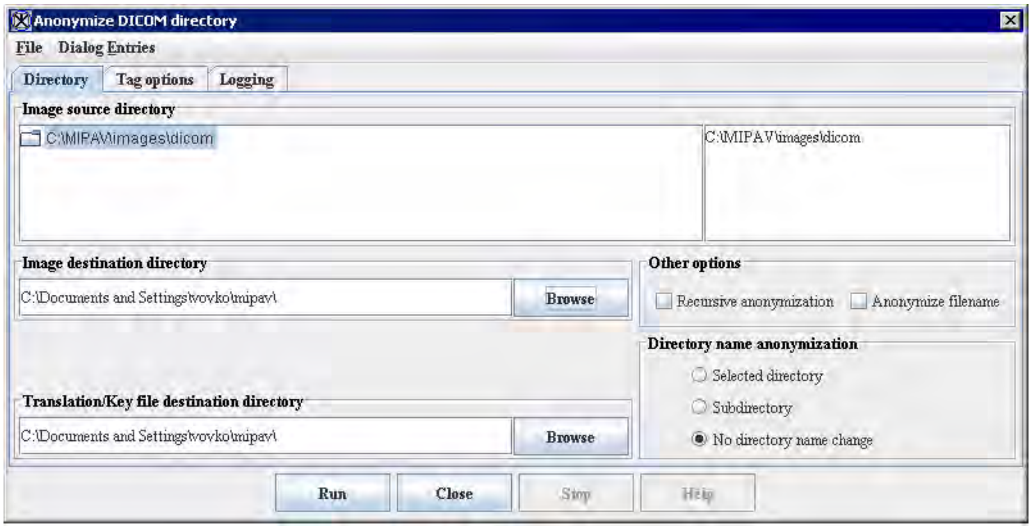


Figure 84. Anonymization of the DICOM directory

ANONYMIZE DICOM DIRECTORY DIALOG BOX OPTIONS

The dialog includes three tabbed pages:

- **Directory page**—Allows you to select which images to process and where to place the output. This page is the page that is displayed when this window opens. Refer to Figure 85.
- **Tag Options page**—Allows you to select DICOM tags to remove from images and how to name anonymized images, which becomes a name tag.
- **Logging page**—Displays the current progress of the anonymization.



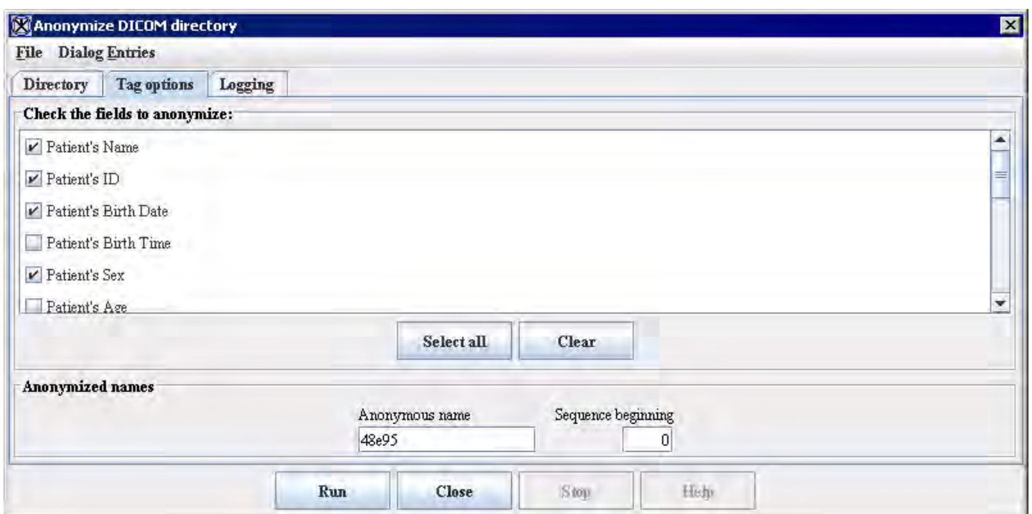
File	**New directory**—Allows you to select a new root directory to anonymize. **Renew list**—Refreshes the file listing in Image source directory.
Dialog Entries	**Clear Log Window**—Removes all of the log messages from the Logging page.
Image source directory	Specifies the root directory to be anonymized.
Image destination directory	Specifies the root directory where the anonymized images should be stored.
Browse	Allows you to select a new image destination directory.
Translation/key file destination directory	Specifies the directory where the translation file is kept. The key file "patient.key.doc" is a text-formatted file that contains the name of the old file and its new name.
Browse	Allows you to select a new translation file directory.
Other options	
Recursive anonymization	Processes images in the selected directory as well as any DICOM images in any subdirectories. Refer to the note on page 187.
Anonymize filename	If checked, anonymizes not only the directory name, but also file names inside the directory.

Figure 85. Directory page in the Anonymize DICOM Directory window

Directory name anonymization	**Selected directory**—Anonymizes the name of only the selected directory on the top level. **Subdirectory**—Anonymizes the name of only the subdirectories. **No directory name change**—Does not change the name of the directories.
Run	Starts the process of anonymizing the images. Status messages appear on the Logging page.
Close	Closes this dialog box.
Stop	Stop the process of anonymizing the images.
Help	Displays online help for this dialog box.

Figure 85. Directory page in the Anonymize DICOM Directory window (continued)

Note: When the Recursive anonymization check box is selected, MIPAV processes images in the selected directory as well any DICOM images in any subdirectories. When this check box is clear, MIPAV only processes the images in the directory that is selected, regardless of any subdirectories.



Check the fields to anonymize	Allows you to anonymize only the specific tags that you select. You can select only one tag or as many as needed.
Select all	Specifies that you want to anonymize all of the tags listed. When you select this button, MIPAV places a check mark in each of the check boxes.
Clear	Removes the check marks from all of the check boxes.

Figure 86. Tag Options page in the Anonymize DICOM Directory window

Anonymized names	**Anonymous name** specifies a first part of the name assigned to the image files. MIPAV uses a combination of this name and the sequence number specified in Sequence beginning. **Sequence beginning** indicates the starting number for the series of image files. The default number is 0.
Run	Starts the process of anonymizing the images. Status messages appear on the Logging page.
Close	Closes this dialog box.
Stop	Stop the process of anonymizing the images.
Help	Displays online help for this dialog box.

Figure 86. Tag Options page in the Anonymize DICOM Directory window

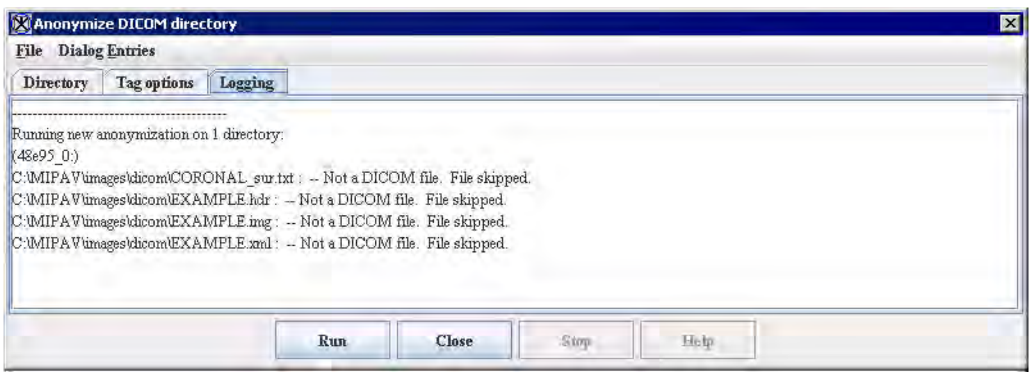


Figure 87. The Logging page in the Anonymize DICOM Directory window listing processing messages and MIPAV removes tags

Converting non-DICOM image files to DICOM format

You can convert a non-DICOM file, such as a TIFF or Analyze file, to DICOM format.

To convert a non-DICOM image file to DICOM

1. Open a non-DICOM image.
2. Click File > Save Image As in the MIPAV window. The Save window appears.
3. Select the directory where you want to store the dataset.

Tip: If you are converting an image dataset that has more than one file, you may want to save the files to a new folder. To do this, select Create New Folder at the top right of the dialog box.

4 In File Name, type the name of the file, including the file extension of `.dcm`.

5 Click Save. The Attributes to Save dialog box appears.

You need to enter DICOM tag information in this dialog box, which has the following four pages:

- Required page (Figure 88)
- Patient page (Figure 89)
- Study page (Figure 90)
- Series page (Figure 91)

6 Enter the required information on the Required page (Figure 88).

Although the fields on the Patient, Study, and Series pages are optional, you must furnish all of the information on the Required page before you can convert the file.

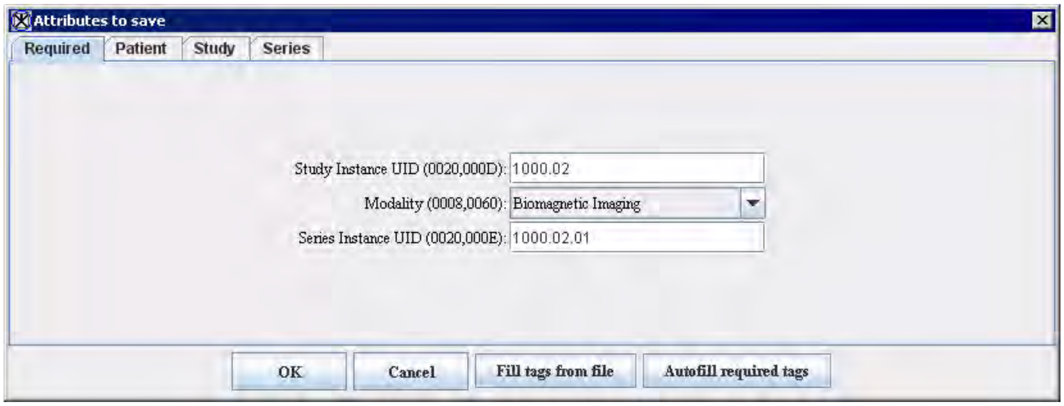
7 Enter the information, if desired, on the Patient, Study, and Series pages (Figure 89, Figure 90, and Figure 91).

8 Click OK. If you are converting a dataset that has more than one slice, the Save Range of Slices dialog box (Figure 92) appears.

9 Complete the Save Range of Slices dialog box, then click OK

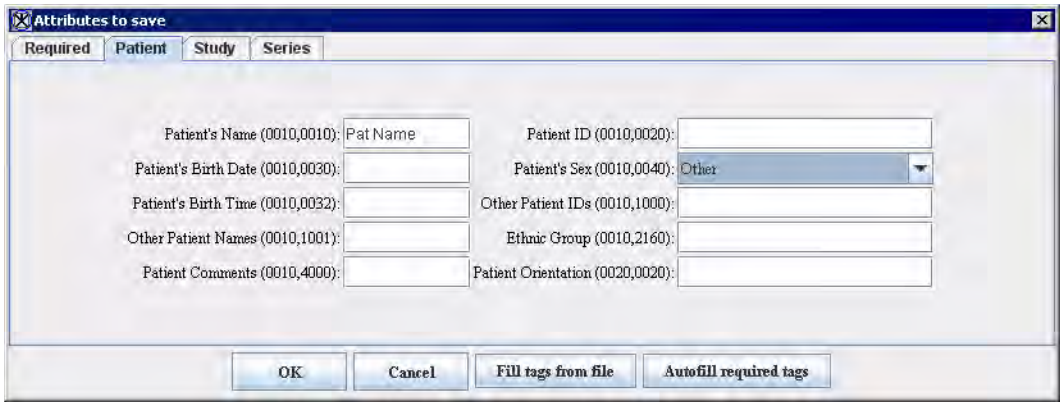
The dataset is converted to DICOM format and saved in the directory you specified. If you are converting a dataset that has several images, each file is sequentially numbered. For example, `doetumor001.dcm`, `doetumor002.dcm`, `doetumor003`, and so on.

ATTRIBUTES TO SAVE DIALOG BOX OPTIONS



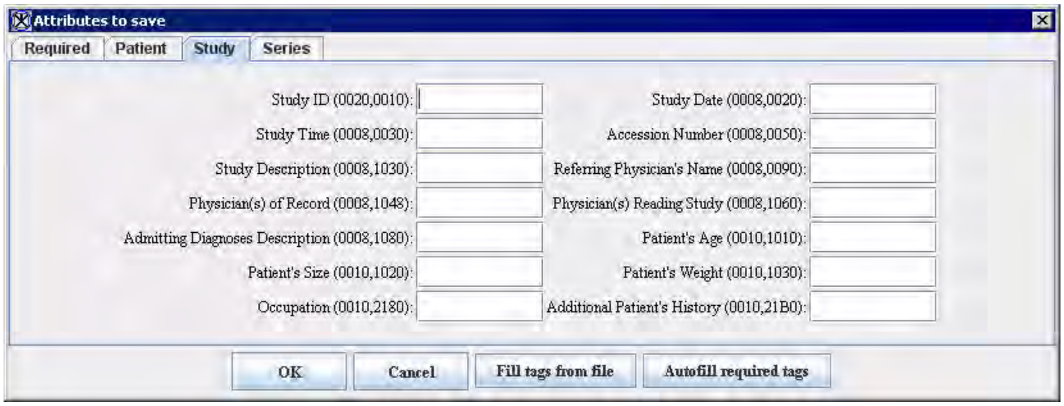
Study Instance UID (0020,000D)	Unique identifier for the study. Only numeric characters with optional periods are allowed.
Modality (0008,0060)	Type of equipment that was used to acquire the data used to create the images in the dataset. Options are: Biomagnetic Imaging, Color Flow Doppler, Computed Tomography, Duplex Doppler, Computed Radiography, Diaphanography, Digital Radiography, Endoscopy, General Microscopy, Hard Copy, Intraoral Radiography, Laser Surface Scan, MR Angiography, Mammography, Magnetic Resonance, MR Spectroscopy, Nuclear Medicine, Other, PET, Panoramic X Ray, Radio Fluoroscopy, Radiographic Imaging, Radiotherapy Dose, Radiotherapy Image, Radiotherapy Record, Radiotherapy Structure, Slide Microscopy, SPECT, Thermography, Ultrasound, X Ray Angiography, and External Photography.
Series Instance UID (0020,000E)	Unique identifier for the study. Only numeric characters (with optional periods) are allowed.
OK	Applies the tags that you specified and closes this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not apply any of the tags.
Help	Displays online help for this dialog box.
Fill tags	Allows you to specify a file that contains the DICOM tags. When you select this button, the Open Tags File dialog box appears.

Figure 88. Required page in the Attributes to Save dialog box



Patient's Name (0010,0010)	Patient's full name.
Patient's Birth Date (0010,0030)	Date of patient's birth.
Patient's Birth Time (0010,0032)	Time of patient's birth.
Other Patient Names (0010,1001)	Other names used to identify the patient.
Patient Comments (0010,4000)	User-defined comments about the patient.
Patient ID (0010,0020)	Primary hospital identification number or code used to identify the patient.
Patient's Sex (0010,0040)	Gender of the patient. Options are: Unknown, Male, Female, and Other.
Other Patient IDs (0010,1000)	Other IDs used to identify the patient.
Ethnic Group (0010,2160)	Ethnic group or race of the patient.
Patient Orientation (0020,0020)	Patient direction of the rows and columns of the image.
OK	Applies the tags that you specified and closes this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not apply any of the tags.
Help	Displays online help for this dialog box.
Fill tags	Allows you to specify a file that contains the DICOM tags. When you select this button, the Open Tags File dialog box appears.

Figure 89. Patient page in the Attributes to Save dialog box

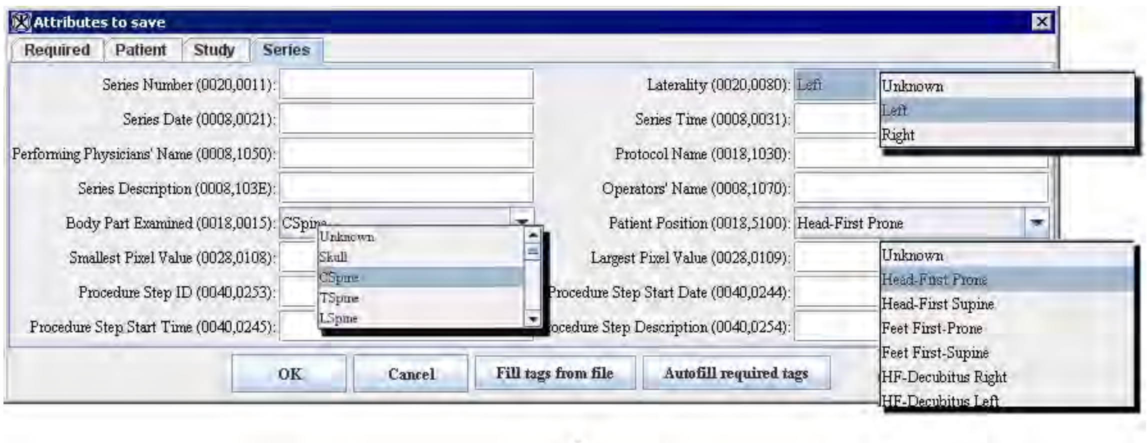


Study ID (0020,0010)	User- or equipment-generated study identifier.
Study Time (0008,0030)	Time the study started.
Study Description (0008,1030)	Institute-generated description or classification of the study (component) performed.
Physician(s) of Record (0008,1048)	Physician responsible for the overall patient care at the time of the study.
Admitting Diagnoses Description (0008,1080)	Description of the admitting diagnoses.
Patient's Size (0010,1020)	Length or size of the patient in meters.
Occupation (0010,2180)	Occupation of the patient.
Study Date (0008,0020)	Date the study started.
Accession Number (0008,0050)	An RIS-generated number which identifies the order for the study.
Referring Physician's Name (0008,0090)	Patient's referring physician.
Physician(s) Reading Study (0008,1060)	Physician(s) reading the study.
Patient's Age (0010,1010)	Age of the patient.
Patient's Weight (0010,1030)	Weight of the patient, in kilograms.
Additional Patient's History (0010,21B0)	Additional information about the patient's history.
OK	Applies the tags that you specified and closes this dialog box.

Figure 90. Study page in the Attributes to Save dialog box

Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not apply any of the tags.
Help	Displays online help for this dialog box.
Fill tags	Allows you to specify a file that contains the DICOM tags. When you select this button, the Open Tags File dialog box appears.

Figure 90. Study page in the Attributes to Save dialog box (continued)

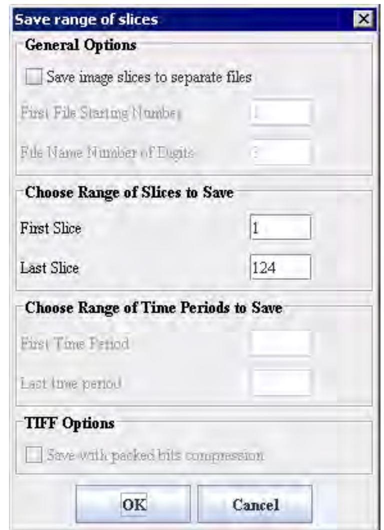


Series Number (0020,0011)	A number that identifies this series.
Performing Physicians' Name (0008,1050)	Name(s) of the physician(s) administering the series.
Series Description (0008,103E)	User-provided description for the series.
Body Part Examined (00018,0015)	A text description of the body part that was examined. Options are: Unknown, Skull, CSpine, TSpine, LSpine, SSpine, Coccyx, Chest, Clavicle, Breast, Abdomen, Pelvis, Hip, Shoulder, Elbow, Knee, Ankle, Hand, Foot, Extremity, Head, Heart, Neck, Leg, Arm, and Jaw.
Smallest Pixel Value (0028,0108)	Minimum value of all images in this series.
Procedure Step ID (0040,0253)	Identification of that part of a procedures that was performed during this step.
Procedure Step Start Time (0040,0245)	Time when the procedure step started.
Laterality (0020,0080)	Options are: Unknown, Left, and Right.
Series Time (0008,0031)	Time series started.

Figure 91. Series page in the Attributes to Save dialog box

Protocol Name (0018,1030)	User-defined description of the conditions under which the series was performed.
Operator's Name (0008,1070)	Name(s) of the technologist(s) supporting the series.
Patient Position (0018,5100)	Patient position relative to the imaging-equipment space. Options are: Unknown, Head-First Prone, Head-First Supine, Feet First-Prone, Feet First-Supine, HF-Decubitus Right, HF-Decubitus Left, FF-Decubitus Right, FF-Decubitus Left.
Largest Pixel Value (0028,0109)	Maximum value of all images in this series.
Procedure Step Start Date (0040,0244)	Date when the procedure step started.
Procedure Step Description (0040,0254)	Institute-generated description or classification of the procedure step that was performed.
OK	Applies the tags that you specified and closes this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not apply any of the tags.
Help	Displays online help for this dialog box.
Fill tags	Allows you to specify a file that contains the DICOM tags. When you select this button, the Open Tags File dialog box appears.

Figure 91. Series page in the Attributes to Save dialog box (continued)



Save image to separate files	
Saves each image slice to a separate file.	
First File Starting Number	Specifies the starting number for the first file.
File Name Number of Digits	Specifies the number of digits in the file name.
Choose Range of Slices to Save	
First slice	Specifies the number of the first slice.
Last slice	Specifies the number of the last slice.
Choose Range of Time Periods to Save	
First time period	Specifies the first time period.
Last time period	Specifies the last time period.
TIFF options	
Save with packed bits compression	Indicates to save the TIFF files with packed bits compression.
OK	Applies the parameters you specified and closes this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not apply any of the changes.
Help	Displays online help for this dialog box.

Figure 92. Save Range of Slices dialog box

6 Connecting to SRB BIRN

In this chapter . . .

Biomedical Informatics Research Network (BIRN, http://www.nbirn.net/) is a shared biomedical IT infrastructure supported by NIH, which hastens the process of new understanding and treatment of disease through use of distributed knowledge. BIRN supports collaboration between groups with different expertise and resources (e.g., technical, scientific, social and political). **The Storage Resource Broker** (SRB), developed at SDSC, is a client-server middleware designed for managing file collections in a heterogeneous, distributed environment. All files within the environment are part of a single data grid BIRN file system where a file's logical location within the file system is represented independently of its physical location. The SRB middleware is capable of managing large data sets and is currently managing the BIRN data grid.

Get Connected to SRB BIRN

To be able to connect to SRB BIRN from inside MIPAV and open the image files using SRB you need to:

1. Obtain a SRB user account and/or NDAR portal user account, see Section "Requesting the SRB account" on page 197;
2. Set up the MIPAV SRB options, see Section "Setting MIPAV SRB options and opening files" on page 198;
3. Set up the SRB connection parameters, see Section "Setting the SRB connection parameters" on page 199;
4. Upload your files to the SRB BIRN (optional), see Section "Uploading files to SRB BIRN" on page 201.

Requesting the SRB account

Before connecting to SRB BIRN from inside MIPAV, you should obtain the user account to the NDAR portal, first. In order to do that, connect to the portal using the following link:

https://nih-cit-dev-apps.nbirn.net/gridsphere/gridsphere?cid=ndar-login

Then, open the Request an Account tab and follow the instructions provided by the tab. Refer to Figure 93.

After your request has been approved, you should receive the confirmation email from the NIH help desk. This email should contain your username and instructions on how to log in to the NDAR portal. At that time, log in to the NDAR portal, enter your email address that you entered in the account request, and create your password. Once you have done creating your password, you can log in to the NDAR portal using the username provided in the helpdesk email and the password you just created. [1]

1. https://nih-cit-dev-apps.nbirn.net/gridsphere/gridsphere?cid=NDAR

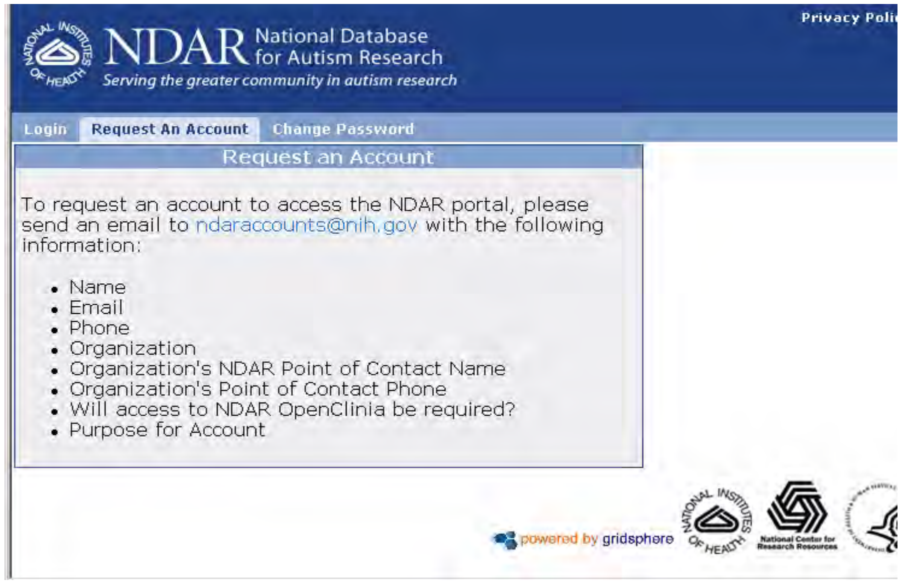


Figure 93. NDAR portal, requesting a user account.

SETTING MIPAV SRB OPTIONS AND OPENING FILES

To set up the MIPAV SRB options:

1. Open MIPAV;
2. Call Help>MIPAV Options menu;
3. In the MIPAV options dialog box that appears, navigate to the SRB tab. See Figure 94;
4. Complete the following tab options:
    - **Jargon version:** a version of Jargon API library, which should be used to communicate to the SRB server. Select one from the list;
    - **Transfer Mode** (parallel or sequential) - specify the file transfer protocol. There are SRB parallel transfer protocol and SRB sequential transfer protocol available;
    - **Temporary Directory** is your local directory, where you keep your files which need to be transferred FROM or TO the SRB server.

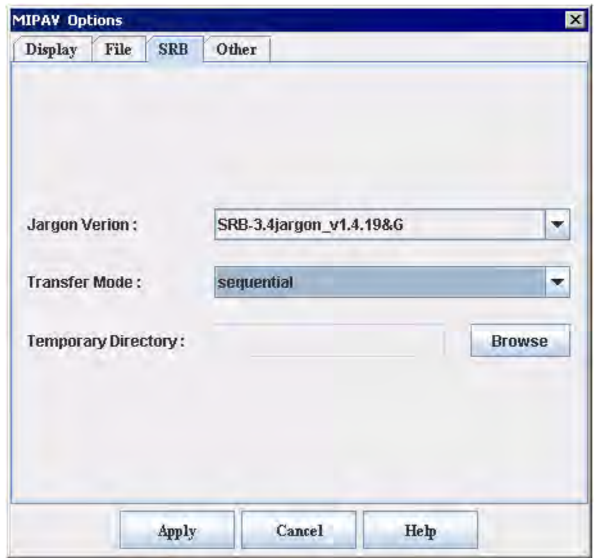


Figure 94. The MIPAV options dialog – SRB options tab.

SETTING THE SRB CONNECTION PARAMETERS

The connection parameters that must be used to connect to SRB BIRN from inside MIPAV are slightly different from the parameters which you used to connect to the NDAR portal. To obtain these parameters,

1 From the NDAR portal page, navigate to the Data Management tab;

2 Click the GridInfo icon. See Figure 97;

3 The info page appears displaying different sets of parameters;

4 You need those which appear under General SRB User Parameter;

5 Open MIPAV, and then call File>SRB BIRN>Open Image from SRB;

6 The Connect to dialog box appears;

7 Enter parameters from General SRB User Parameter into the dialog box. Refer to Figure 95;

8 In the Authentication list box, select ENCRIPT1;

9 Press Connect, to connect to SRB server. The SRB File Chooser dialog should appear displaying the list of catalogs.

10 Open the desired catalog and select a file of interest. Press Open to open the file. See Figure 96.

11 The file now should be opened in MIPAV.

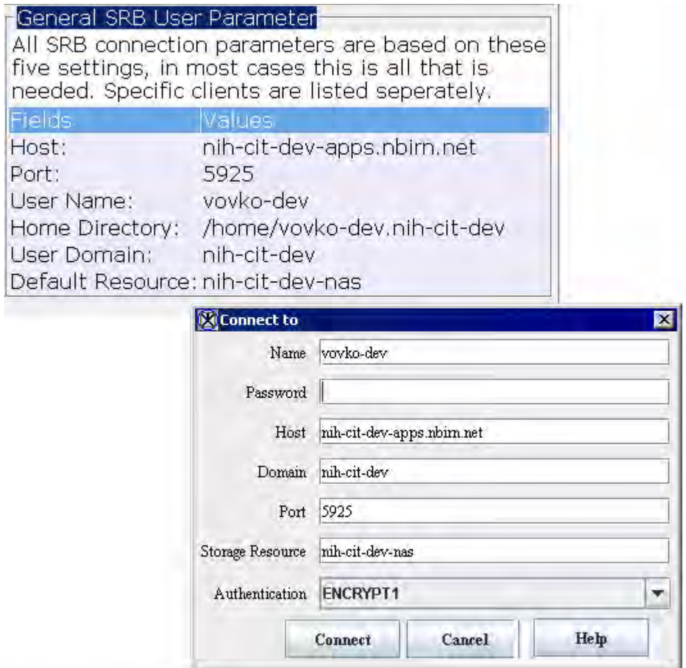


Figure 95. Parameters used to connect to SRB BIRN from inside MIPAV.

As shown in Figure 95 the following parameters are used to connect to SRB:

General SRB User Parameter	MIPAV
Host	Host
Port	Port
User Name	Name
Home Directory	
user Domain	Domain
Default Resource	Storage Resource

Figure 96. The SRB FIle Chooser dialog box.

UPLOADING FILES TO SRB BIRN

To test your access to SRB BIRN from inside MIPAV, you might choose to upload a couple of files to your home catalog, first. In order to do that,

1. From the NDAR portal page, navigate to the Data Management tab;
2. Navigate to the Upload File box;
3. Use the Browse button to select a file for upload;
4. Then, click Upload File;
5. The file should appear under SRB File/Folder List. See Figure 97.

Later, you can upload your image files to SRB BIRN from MIPAV using the SRB Transfer and Save to SRB menus.

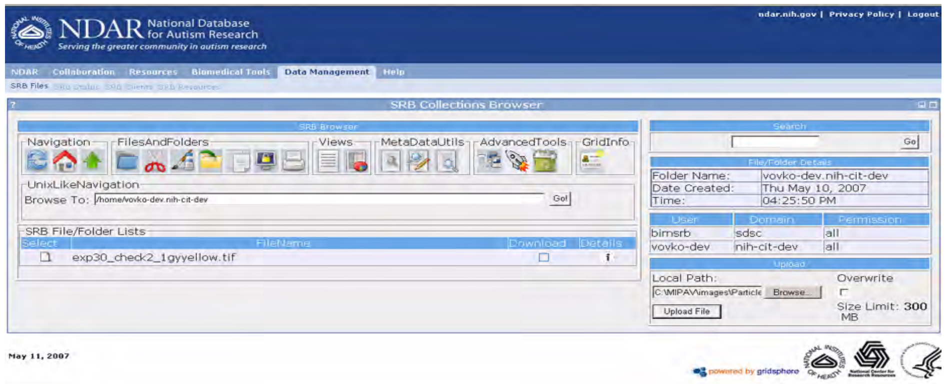


Figure 97. Uploading files to the NDAR portal.

TRANSFERRING FILES FROM MIPAV TO SRB BIRN AND VISE VERSA

To transfer files from MIPAV to SRB BIRN and back to MIPAV:

1. Use the MIPAV File>SRB BIRN>Open Image from SRB menu to connect to SRB BIRN;
2. Call File> SRB BIRN > SRB Transfer menu;
3. In the Transfer Files dialog box that appears, use the list boxes to specify the Source and Target directory (SRB and your local directory). Then, use the Browse buttons to select files and catalogs.
4. Press Transfer to send files. Or press Cancel to cancel. See Figure 98.

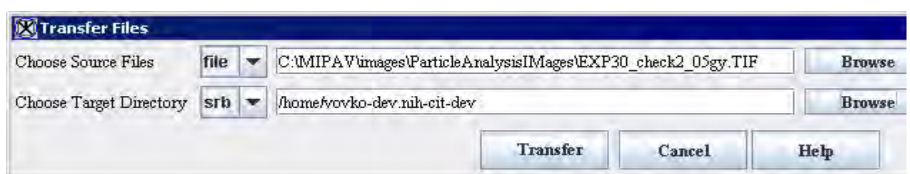


Figure 98. Transferring files between SRB BIRN and MIPAV.

Set up the NDAR pipeline

You can run MIPAV scripts on the DICOM images stored on the DICOM server and automatically save results to the chosen SRB BIRN catalogue.

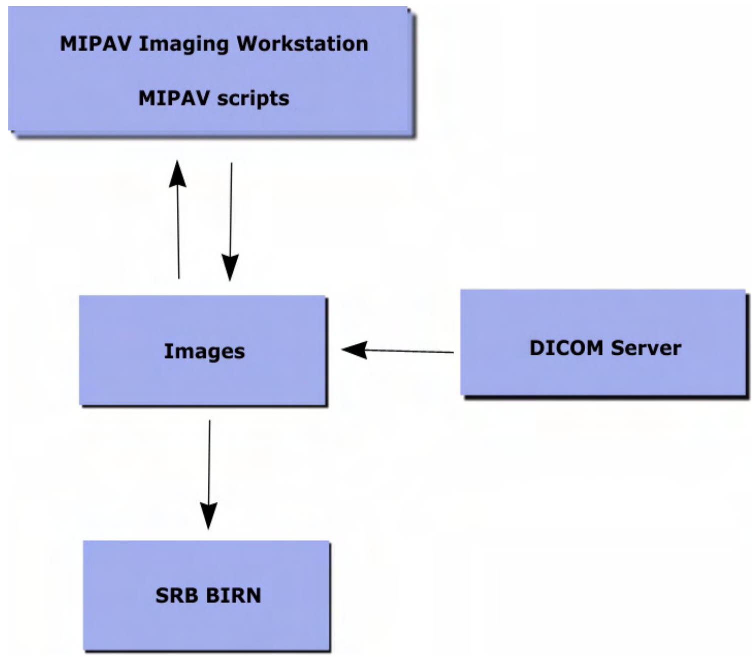


Figure 99. Running MIPAV scripts on the DICOM images and saving results to the SRB BIRN.

In order to be able to do that, you should establish the NDAR pipeline, first.

1. Navigate to the File> SRB BIRN menu;
2. Here, check the Enable Auto SRB Upload option;
3. The Setup the NDAR Pipeline dialog box appears;
4. In the dialog box, first, select the MIPAV script which you would like to run;
5. Select the target SRB directory and press Open to confirm your selection.
6. Press OK to sun the script. Press Cancel to discard.

The script you choose should be run on the DICOM files and the results stored in the chosen SRB directory.

Note: that, you might be asked to login to SRB BIRN before you select the target SRB catalog. In that case, login to SRB as it was described in Section "Requesting the SRB account" and Figure 95.

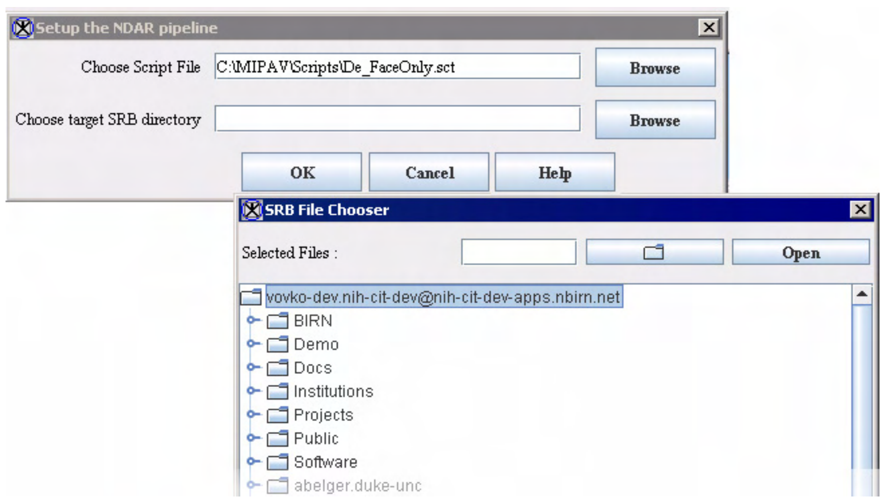


Figure 100. NDAR pipeline.

SAVING FILES TO SRB

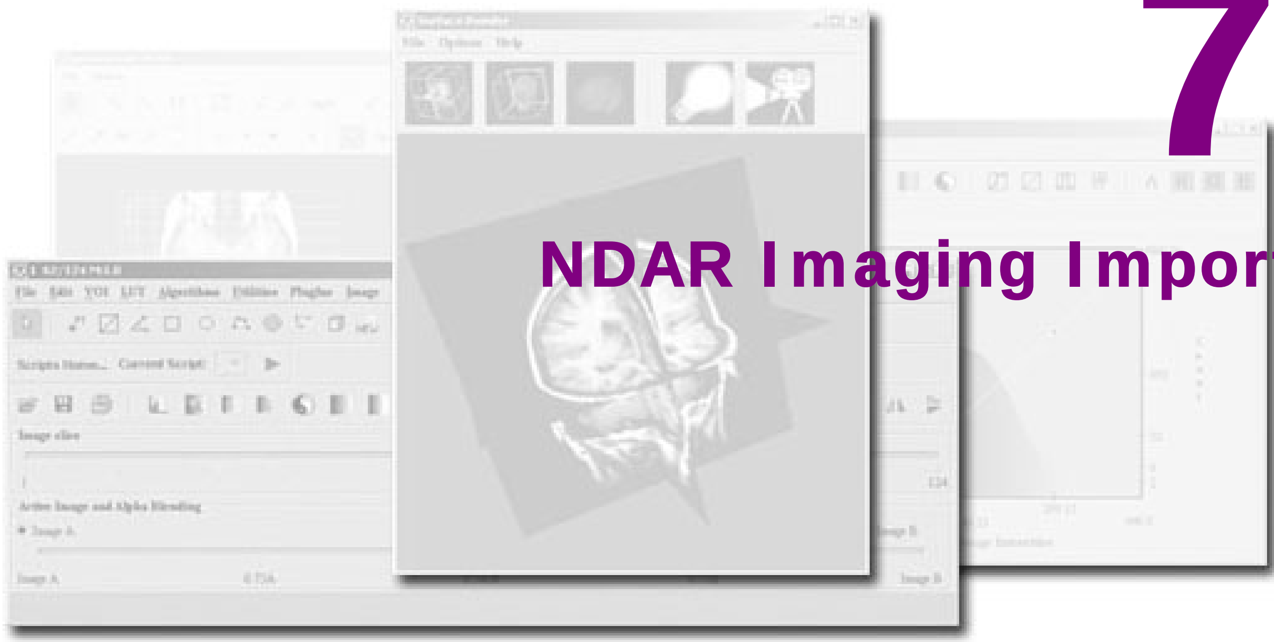
To save your image file directly to SRB:

1. Select the image file;
2. Call File>SRB-BIRN>Save Image to SRB menu;
3. The Connect to dialog box appears asking you to log in to SRB-BIRN;
4. Fill out the dialog box options as it was described in Figure 95. You need to enter only you password to SRB. All other information should be saved in the dialog box;
5. Press Connect;
6. In the SRB File Chooser dialog box that appears, select the catalog where you would like to save your file. See Figure 101;

7 Press Save.

The file should now appear in the chosen catalog.

Figure 101. Saving an image to the SRB BIRN catalogue.



7

NDAR Imaging Import

In this chapter . . .

This document provides the help for the NDAR Imaging Import tool, which is designed to assist uses in moving data from the MIPAV XML output to the NDAR shared data repository for the benefit of the greater autism community.

Pre requisites

Before you run the NDAR Imaging Import tool, the following five conditions must be met:

1. Java 1.5 should be installed on your PC.
2. All identifying information should be removed from the image (i.e., patient meta-data in the image header).

3 The image file format should be one which can be read by MIPAV, e.g.: Analyze, AFNI, DICOM 3.0, MINC, MIPAV XML, RAW, TIFF, among many others. For more information about file formats supported by MIPAV, refer to the MIPAV User Manual Volume 1, APPENDIX C: Supported formats (http://mipav.cit.nih.gov/documentation.php).

4 Required dataset meta-information, i.e. PI information, IRB number, Abstract, Publication level of the data (Public, Private to Organization, Private to Investigator), and NDAR GUIDs for each image dataset, should be obtained.

5 A user account on the NDAR Storage Resource Broker (SRB) data federation system should be set.

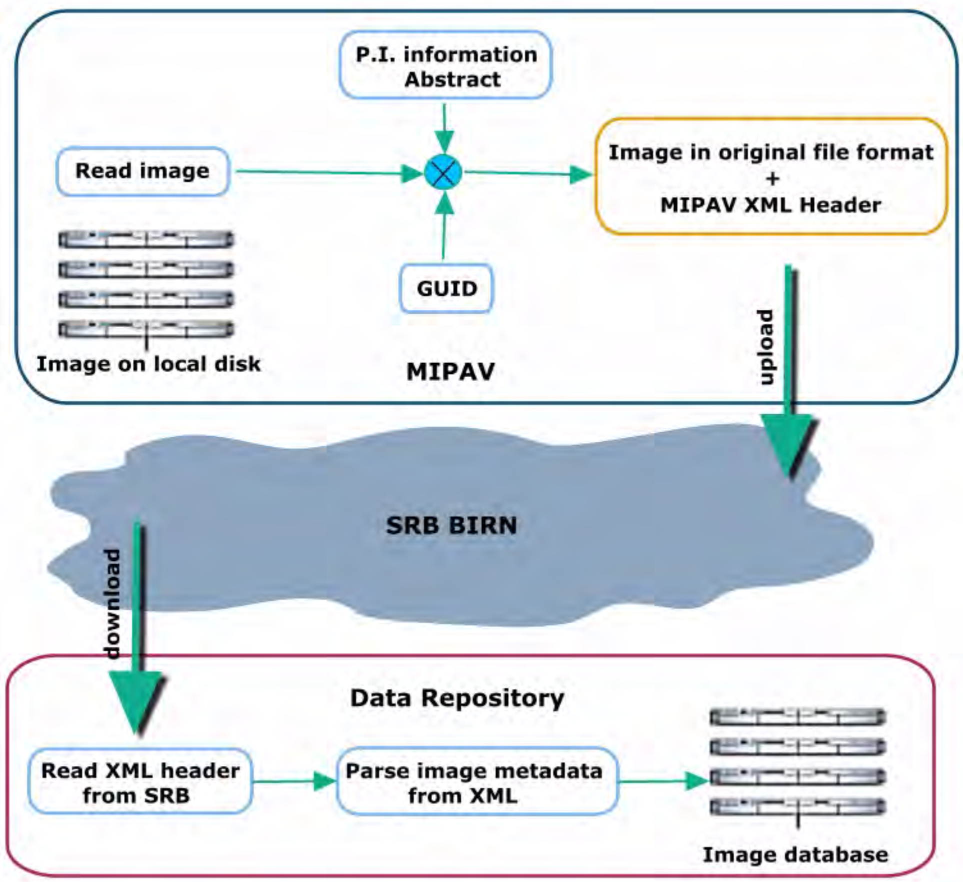


Figure 102. This scheme illustrates the process of moving data from the MIPAV XML output to the NDAR shared data repository.

Running NDAR Imaging Import tool

To run the tool,

1. Open MIPAV, and then call the File>SRB BIRN>NDAR Image Import menu.
2. The NDAR Imaging Import Tool dialog box appears.
3. **In the first Main tab** of the dialog box, read the statement, and then check the "I agree to the above statement" box. The Next button becomes available, press it to proceed further. See also "Appendix A: MIPAV Privacy Statement" on page 212.
4. **In the Principal Investigator (P.I.) tab**, enter the P.I. information including Name, Title, Email address, Phone, and IRB number. Press Next.
5. **In the Abstract tab** that appears next, enter the abstract title and summary. You also can use the Load from File option to load the summary from the file. Press Next.
6. **In the Source tab**, use the Add files button to select image datasets from a disk. Press Next, and then confirm adding datasets.
7. **In the GUIDs tab** that appears next, assign a GUID to each dataset. You can either do that manually or use the Load GUIDs option to load GUIDs from a file. Press Next. Note that if the image dataset has a valid NDAR GUID it its name, the software will recognize it, and it will appear in the GUID textbox.
8. **In the Destination tab**, select the location from the list and then specify the publication level for the imaging datasets, i.e. Public, Private to Organization, or Private to Investigator. Press Next. The upload statistics appear in the Output Log window.
9. For each dataset selected for processing and upload:
   - MIPAV reads in the dataset from a disk.
   - For each image in the dataset, MIPAV creates an MIPAV XML file containing the NDAR GUID, basic dataset metadata from the original image and the meta-information entered by the user. By default, the file is stored in the mipav\temp subdirectory of user directory, e.g.

```
C:\Documents and Settings\YourUserName\mipav\temp
```

- The MIPAV XML header file is uploaded to an Import Directory on the SRB, to be imported into the Shared Repository at a later time.
- The original image dataset files are uploaded to the SRB in a location determined by the user name, organization, and publication level entered by the user earlier.

Figure 103 shows the MIPAV XML file created based on the information you provided in the NDAR Imaging Import Tool dialog box. The file is opened in Internet Explorer.

See Figure 104 for more information about the dialog box options.

Figure 103. The MIPAV XML file created using the NDAR Imaging Import tool.

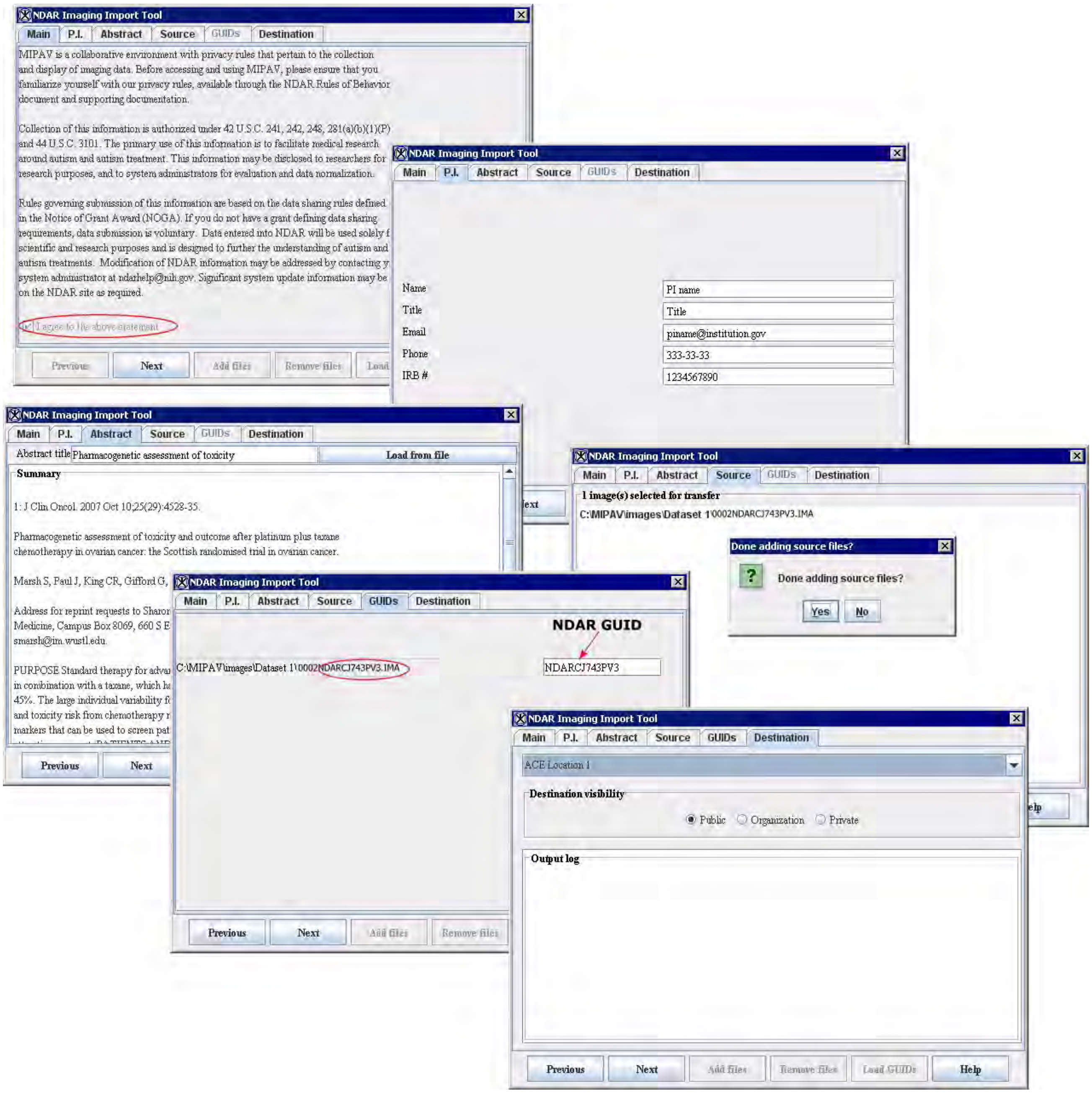


Figure 104. NDAR Imaging Import Tool Dialog box

PostConditions

After processing a series of image datasets with the MIPAV NDAR Imaging Import tool:

- The original image dataset files will be in a directory on the SRB determined based on the meta-information entered by the user.
- A MIPAV XML file will be in the Import Directory on the SRB describing the dataset.

Data Migration

PreConditions

The data migration pre-conditions are listed below.

1. Each MIPAV XML file contains data for only one image.
2. Java 1.5 or greater is installed and configured on the machine where the data transformation will be performed.
3. The NDAR shared repository database, tables, sequences, etc. are configured on the appropriate NDAR database server.

Data Transformation

1. Execute the following command to transform the XML document into a file ready for migration to the NDAR shared repository. The transformer used is Saxon and the required library file can be found in the imaging-transformer utility under the lib directory. The XSL file can be found in the imaging-transformer utility at `xsl\ndar_imaging_transform_sql.xsl`.
2. `java -jar <DIR>\lib\saxon8.jar -o <OUTPUT>.sql <INPUT>.xml <DIR>\xsl\ndar_imaging_transform_sql.xsl`
3. Open the output SQL file and validate a sample of the contents against the input XML document.

Data Load

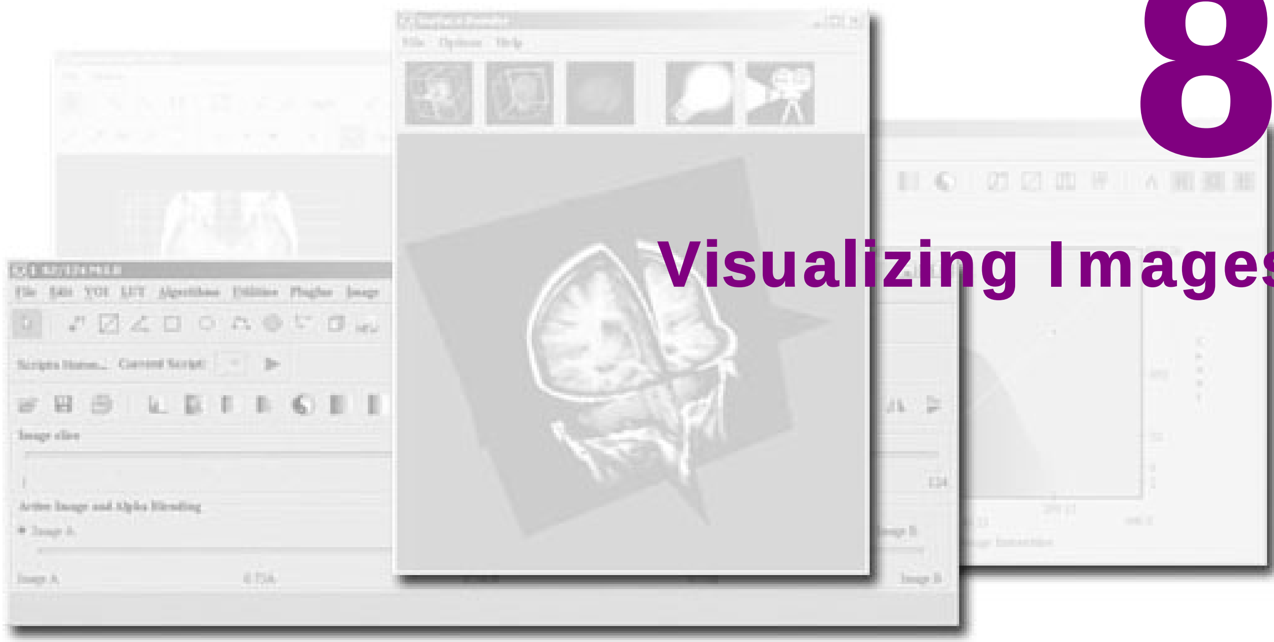
1. Using a PostgresSQL database load utility of choice (such as PGAdmin), execute the output SQL file on the NDAR shared repository development server.
2. Validate the script execution using the log files and/or console output to ensure proper submission to the database.
3. Validate the data load by comparing the XML document with the data stored in the NDAR shared repository.

Appendix A: MIPAV Privacy Statement

MIPAV is a collaborative environment with privacy rules that pertain to the collection and display of imaging data. Before accessing and using MIPAV, please ensure that you familiarize yourself with our privacy rules, available through the NDAR Rules of Behavior document and supporting documentation.

Collection of this information is authorized under 42 U.S.C. 241, 242, 248, 281(a)(b)(1)(P) and 44 U.S.C. 3101. The primary use of this information is to facilitate medical research around autism and autism treatment. This information may be disclosed to researchers for research purposes, and to system administrators for evaluation and data normalization.

Rules governing submission of this information are based on the data sharing rules defined in the Notice of Grant Award (NOGA). If you do not have a grant defining data sharing requirements, data submission is voluntary. Data entered into NDAR will be used solely for scientific and research purposes and is designed to further the understanding of autism and autism treatments. Modification of NDAR information may be addressed by contacting your NDAR system administrator at **ndarhelp@nih.gov**. Significant system update information may be posted on the NDAR site as required.



8

Visualizing Images

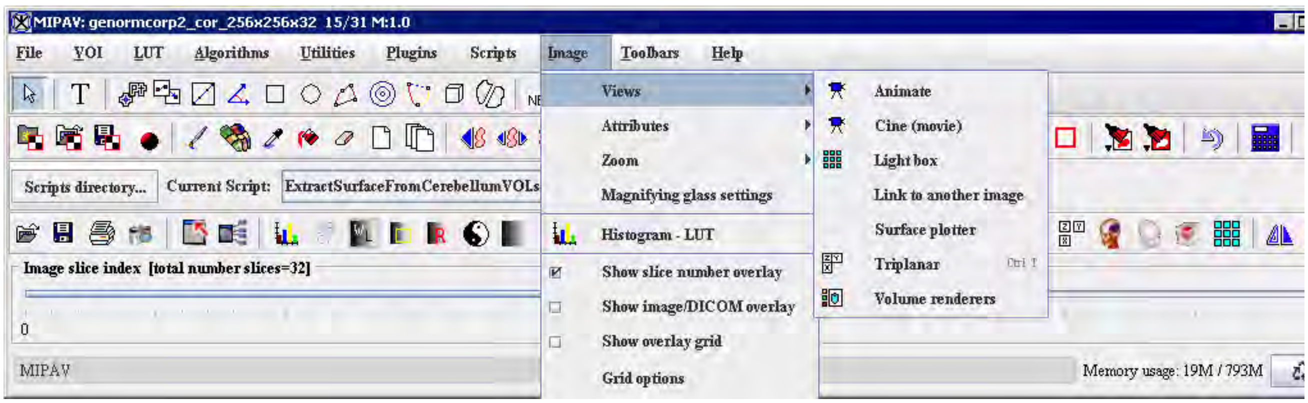
In this chapter . . .

Visualization of image datasets is an important aspect of image analysis. Researchers must often visualize the orientation, locality, or progression (time) of structures in clinical and non-clinical datasets. This chapter explains how to use MIPAV to visualize medical images using views.

Views specify the way image datasets are displayed. Views indicate the number of images in a dataset that are displayed at one time and whether images in a dataset are manually or automatically advanced.

In the *default view*, one image is displayed at a time. If a dataset contains more than one image, you can manually advance through the image stack. The default view can also display 2D through 5D datasets. For image datasets with more than two dimensions, additional views are available: animate, cine (movie), lightbox, link to another image, surface plotter, surface render, triplanar, and triplanar-dual. The lightbox view displays two or more images, or slices, at once. Others—animate and cine—automatically advance the images and display them in succession, much like a cinema film.

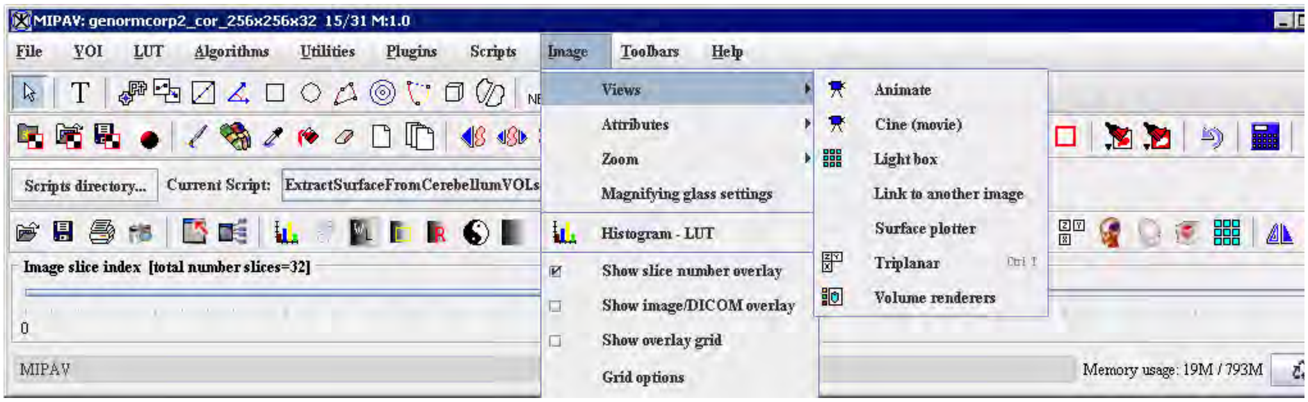
To gain access to views other than the default view, you use the Image > Views menu (Figure 105) in the MIPAV window. Figure 1 lists the types of views and the number of images each view displays at one time and indicates whether images are manually or automatically advanced in the view and the number of dimensions an image must have to be displayed in the specified view. For example, you can view an image dataset that has three or more dimensions in cine view, which displays only one image at a time and advances images in the forward direction only.



View	Number of images displayed	Manual or automatic image advancement	Dimensionality
Default	1	Manual	All (2D, 3D, 4D, 5D)
Animate	1	Automatic (forward, reverse, continuous looping)	Images with more than two dimensions (2D)*
Cine (Movie)	1	Automatic (forward)	Images with more than two dimensions

*MIPAV can display a 4D dataset that overlays a 3D dataset in the views indicated. If the 4D dataset has dimensions that differ from the 3D dataset, the 4D dataset is automatically resampled and interpolated into the same dimension space as the 3D dataset.

Figure 105. Types of views as shown on the Image > Views menu



View	Number of images displayed	Manual or automatic image advancement	Dimensionality
Lightbox	All images in the dataset	Manual	Images with more than two dimensions*
Link to another image	2	Manual	Images with more than two dimensions*
Surface plotter	1	Automatic and Manual	Images with more than two dimensions*
Surface renderer	All images in the dataset	Manual	3D images*
Triplanar	2	Manual	3D and 4D images*
Triplanar-dual	2 images loaded together	Manual	3D and 4D images*
Volume render (shear)	In development		
Volume renderer	In development		

*MIPAV can display a 4D dataset that overlays a 3D dataset in the views indicated. If the 4D dataset has dimensions that differ from the 3D dataset, the 4D dataset is automatically resampled and interpolated into the same dimension space as the 3D dataset.

Figure 105. Types of views as shown on the Image > Views menu (continued)

Tip: You can display different views of the same dataset on the desktop at the same time.

Displaying images using the default view

When you initially open or load an image dataset, it appears in the image window showing the default view (Figure 2).

If you display a 2D image dataset, the title bar only displays the name of the file and the magnification level only. If you open an image dataset that has more than two dimensions, the title bar displays the name of the file, the position of the image in the dataset, the total number of images in the dataset, and the magnification level.

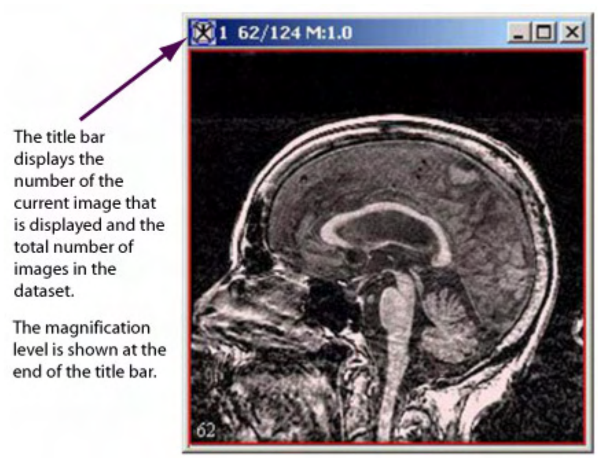


Figure 106. Image window showing the default view for a 2D image

Datasets with more than two dimensions typically contain many image slices. To manually advance through datasets that contain more than one image, you can use the image slice slider or , the Decrements image slice icon, or , the Increments image slice icons in the MIPAV window (Figure 4). You can also use the lightbox view to find and display a particular slice in the image dataset in the default view (refer to "Using the lightbox view to update the image slice displayed in the default view" on page 47).

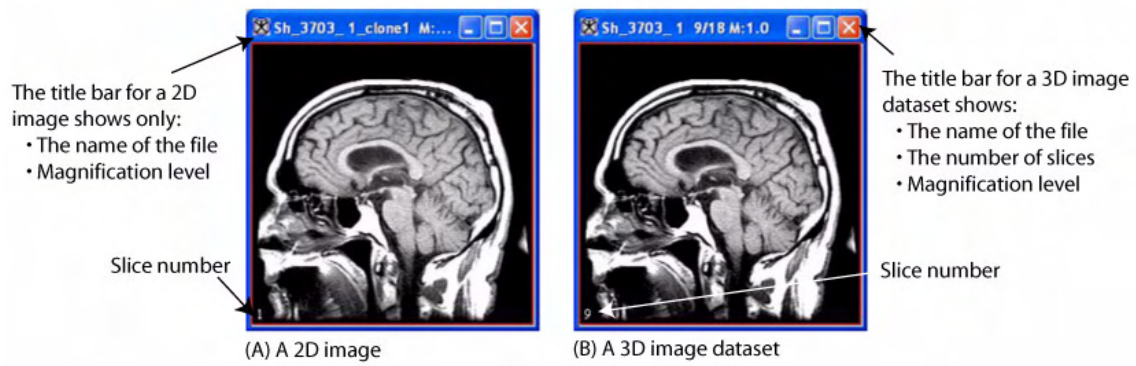


Figure 107. Image window showing (A) a 2D image, or slice, and (B) an image dataset that has more than two dimensions.

Magnifying and minifying images

In the default view you can magnify or minify, or reduce, displayed images. When you display an image, the magnification level is indicated on the title bar of the window. For example, M:4.0 indicates that the image was magnified by a factor of four times the original size. You can also use the magnifying glass to temporarily magnify or reduce a region of the image. You can magnify images to 32 times their original size, reduce them to one-fourth their original size, or restore them to their original size.

Figure 108. MIPAV window

In the MIPAV window, there are three methods for magnifying or minifying images:

- Automatically using three preset commands on the Image > Zoom menu
- Manually using the Custom command on the Image > Zoom menu, which allows you to create other magnification levels
- Automatically using the icons on the image toolbar

To set the magnification level automatically using the preset commands

1 Open an image.

2 Select Image > Zoom.

3 Select one of the following commands on the Zoom menu:

-  *0.5X*—To reduce the image by one-half of its current size
- 

1X—To restore the image to its original size
-  *2X*—To enlarge the image by twice its original size

The software immediately changes the magnification level of the image.

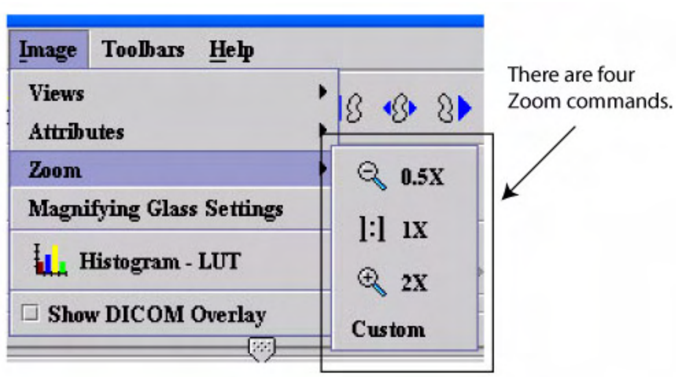


Figure 109. Zoom commands on the Image menu

To set the magnification manually using the Custom command

To manually adjust the magnification level to a specific level that is not available through the preset commands or the icons on the image toolbar, do the following:

1. Open an image.
2. Select Image > Zoom > Custom. The Custom Magnification dialog box (Figure 6) opens.
3. Slide the marker in the Magnification slider to the specific level of magnification that you want to apply.
4. Select the interpolation method if desired.

Interpolation is the addition of one or more voxels between two other voxels. The interpolation mode is particularly important when an image is magnified (and voxels are seen more easily). Three interpolation modes are available: nearest, bilinear, and cubic (refer to Figure 6 for more information).

5. Click Apply. MIPAV applies the magnification level and interpolation mode to the image.

Magnification	Slide the marker left to decrease the magnification or right to increase the magnification. You can reduce the image to one fourth of its current size or enlarge it up to 32 times its current size. Notice that the number below the center of the slider changes as you slide the marker to indicate the magnification level. 
Interpolation	Select one of the following methods: **Nearest**—Short for *nearest neighbor.* The replicated voxel is assigned the color of the closest voxel. This mode yields more accurate results because other modes because the original values are retained. However, if applied, the image may have a choppy "stair-step" effect. It is also possible that some data may be lost or duplicated. **Bilinear**—In this mode, the average of the four nearest voxels (to the newly interpolated voxel) is weighted. The interpolated voxel is then assigned the resulting color. If this mode is applied, the "stair-step" effect is reduced. However, because the view of the data is altered, it is not as accurate as the nearest neighbor mode. **Cubic**—Short for *cubic convolution.* This mode is similar to the bilinear mode, except that the weighted average of the nearest sixteen voxels is used instead of the nearest four. If this mode is applied, the stair-step effect is greatly reduced. However, because the view of the data is altered, it is not as accurate as the nearest neighbor or bilinear modes.
Apply	Applies the magnification and interpolation you specified.
Cancel	Disregards any changes that you made in this dialog box and closes this dialog box.
Help	Displays online help for this dialog box.

Figure 110. Custom Magnification dialog box

To set the magnification automatically using icons on the image toolbar

The three magnification icons on the image toolbar work the same way as the three preset commands on the Image > Zoom menu do. Do one or more of these tasks:

- To magnify an image, click , the Magnify image 2.0x icon. Each time you click Magnify, the image doubles in size. As the magnification level increases, it may be necessary to manually adjust the size of the window so you can see the entire image at one time. Scroll bars appear if the image is too big for the window.
- To minify an image, click , the Magnify image 0.5x icon. Each time you click this icon the image is reduced to half size.
- To return the image to the original level of magnification, click , the Magnify image 1.0x icon.

Tip: These icons appear on other windows beside the MIPAV window, and they work in the same way on each window.

WORKING WITH THE MAGNIFYING GLASS

The magnifying glass is much like a traditional, real-world magnifying glass. It allows you to view a portion of the image at a specific magnification level. You can select the magnification level, which can be anywhere from 1 to 32 times the original image size. For example, an image may be the original size; however, to examine a specific area more closely, you may want to set the magnification level of the magnifying glass to 16 times the original size.

Conversely, if an image uses a high-level of magnification, you may want set the magnifying glass to a lower magnification level so that you can gain perspective on where a section fits with the whole image.

To use the magnifying glass

1 Open an image.

2 Click , the Magnify Region icon, on the Image toolbar in the MIPAV window.

3 Move the cursor to the image. The cursor shape changes to the magnifying glass shape, a box outlined in red with the level of magnification in the lower left corner (Figure 7). The area inside the box is magnified.

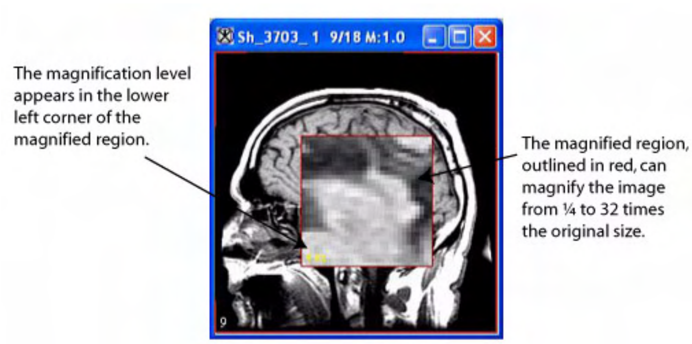


Figure 111. Magnified region

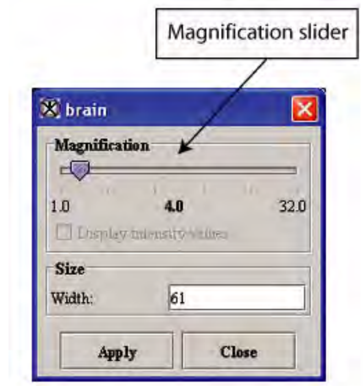
You can move the magnifying glass to different places on the image. This allows different regions to be magnified without affecting magnification level of the entire image. When you want to return to normal magnification, click the Default Mode icon.

To adjust magnifying glass settings

1 Do either of the following:

- Select Image > Magnifying Glass Settings.
- Right-click on the image.

The Magnification Settings dialog box (Figure 8) appears.



Magnification	Specifies the level of magnification in the magnified region. You can select a magnification from 1 to 32 times the original size of the image.
Display intensity values	Shows the intensity values of the voxels in four areas in the magnified region. This check box is only available for magnifications 21 to 32.
Width	Indicates the size of the magnified region in voxels; in other words, of the magnifying glass.
Apply	Applies the parameters in this dialog box to the magnifying glass.
Close	Disregards any changes that you made in this dialog box and closes this dialog box.
Help	Displays online help for this dialog box.

Figure 112. Magnification Settings dialog box

2 Move the Magnification slider to select the level of magnification. The number in bold in the middle of the slider indicates the current level of magnification.

3 Specify the size of the magnifying region (box) in the Width text box. (The width and the length are the same.) Values are in voxels.

4 Select Display intensity values if you want to see the intensity values of the voxels in the magnifying glass (Figure 9).

5 Click Apply. The new settings are applied to the magnifying glass.

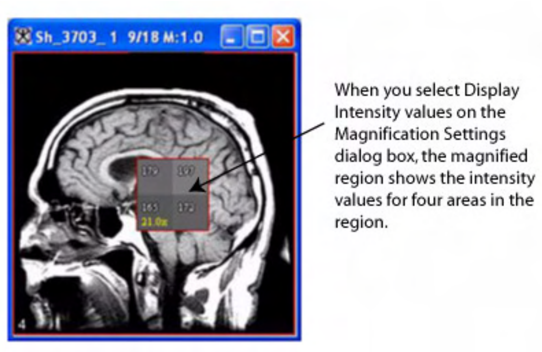


Figure 113. Magnified region showing intensities

Viewing two images together

When two separate image files are loaded into the same window (for directions on how to load an image, refer to Chapter 3, "Getting Started Quickly with MIPAV"), the window displays a blending of both images. MIPAV refers to the first image file as "Image A" and to the second image file as "Image B." By default, Image A is active and Image B is inactive. To change which image is active, simply select, as appropriate, the Image A or Image B radio button.

By default, Image A is active and Image B is inactive. Thus, even if two image files are loaded into the same image window, only Image A—which is the designation MIPAV assigns to the first image that was opened in the window—is displayed.

You can view the inactive image using one of the following methods:

- Adjust the alphablending value so that you can see the inactive and active images simultaneously
- View the part of Image B that is directly under that portion of Image A by using , the Window region of Image B icon.
- View portions of the image using the checkerboard tool

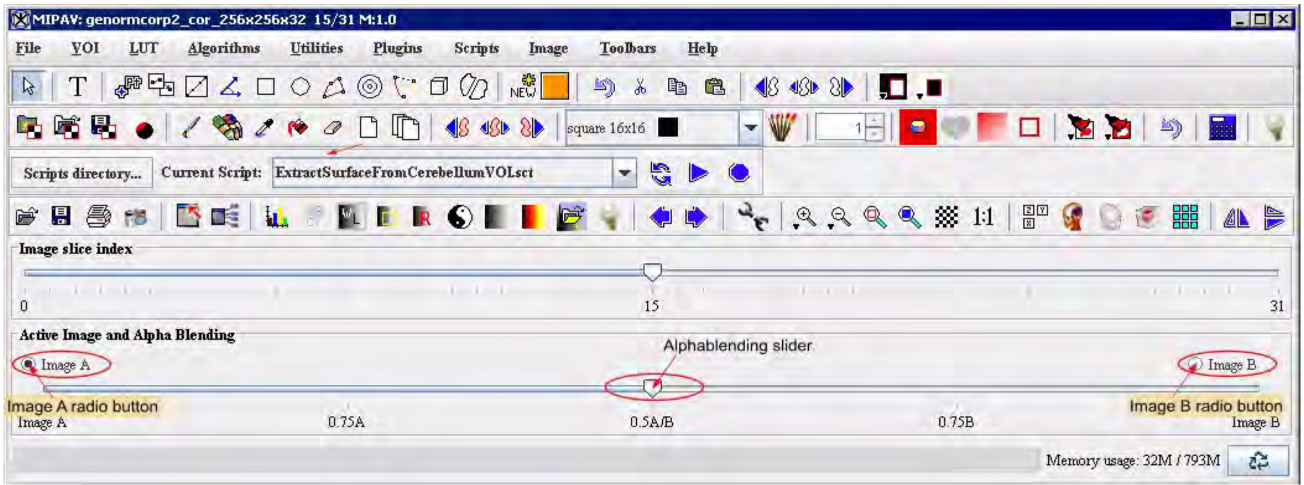


Figure 114. MIPAV window showing the Alphablending slider

COMPARING IMAGES USING ALPHABLENDING

Alphablending is a technique that adds transparency information to translucent objects. It is used to blend what is currently rendered on the display (i.e., Image A) with the contents of the frame buffer (i.e., Image B). Each colored voxel is assigned an alpha value, which represents transparency. The alpha value is used to blend the intensities, voxel by voxel. When two images share a window, you can adjust the alphablending settings so that you can see a blend of both images and can compare overlapping regions in two datasets. See Figure 114.

To use the alphablending tool and Window Region tool

1 Open two images.

2 Load the second image into the image window of the first image. The Active Image and Alphablending slider appears at the bottom of the MIPAV window (Figure 11).

The level of translucency for one image is inversely proportional to the other. Thus, if Image A is 75 percent transparent and 25 percent opaque; Image B is 75 percent opaque and 25 percent transparent ("Alphablending: An Example" on page 20).

Note: In the MIPAV window, notice that an additional magnifying glass is now available: , the Window Region of Image B icon, which you use to view the portion of Image B that is directly under that portion of Image A.

3 Move the Active Image and Alphablending slider between the Image A and Image B radio buttons.

Alphablending: An Example

For a very clear example of how alphablending works, open two images and then apply , the Hot Metal LUT icon, to one of the images before you load them together. After the images are loaded, move the Active Image and Alphablending slider from the left to the right. The following images show what you see at five points of the slider:

(A) The image on the far left shows 100 percent of Image A. It does not display a blending of the images.

However, for any of the images below, you can use the , Window Region of Image B icon, to inspect the portion of Image B that lies underneath that same portion of Image A.

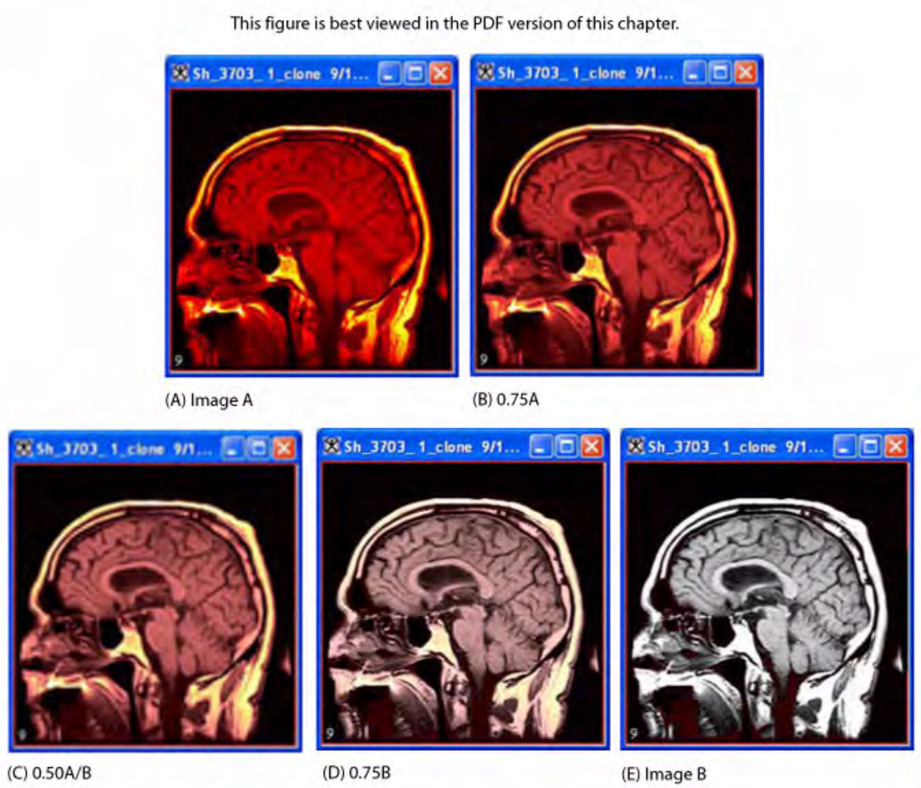
(B) The next image displays a blending of 75 percent Image A and 25 percent of Image B.

Again, use to see the portion of Image B beneath Image A.

(C) The middle image displays 50 percent blending of both Image A and Image B. You can use to inspect Image B.

(D) The fourth image displays 75 percent of Image B and only 25 percent of Image A. You can use still to inspect Image B.

(E) The final image on the right displays 100 percent of Image B. Since all of Image B is shown, you no longer have a need to use .



Depending on the direction in which you are moving the slider (that is, toward Image A or toward Image B), one image becomes more visible. To examine each image in more detail, you can use the magnifying glasses.

As you move the slider, the images in the image window are updated instantly. If not, release the mouse button to update the image.

Move the slider all the way to the right or to the left depending on which image you want to see. For example, to see Image B, move the slider to the far right. To see Image A, move the slider (Figure 10) to the far left. The closer the slider is to the middle, the more blended are the images.

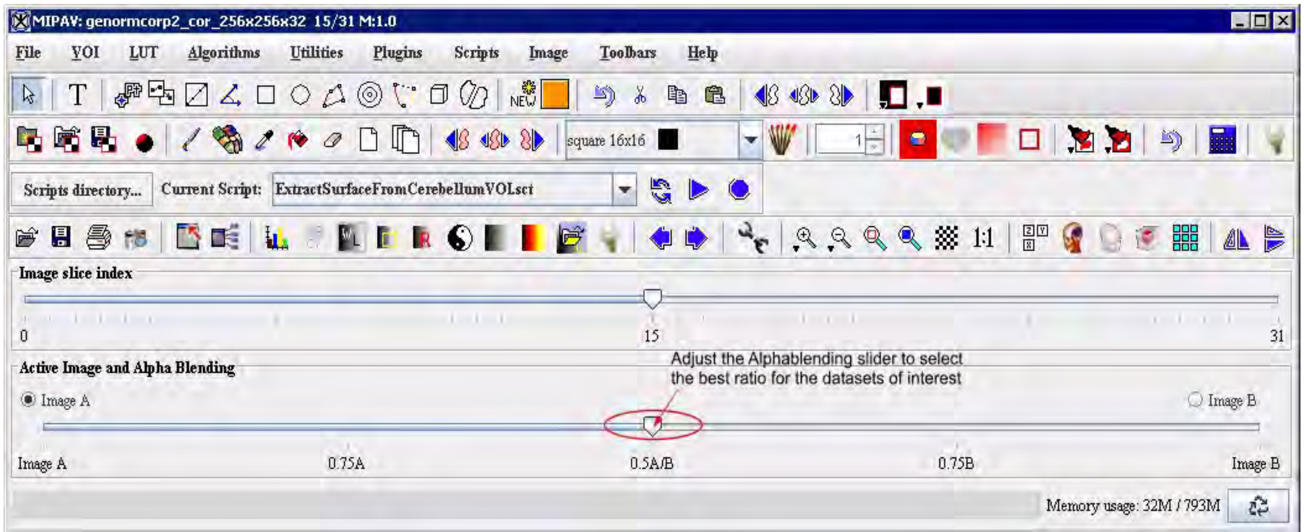


Figure 115. MIPAV window

To make images active or inactive

1 Open two images.

2 Load the second image into the first image window. The MIPAV window expands to include the Active Image and Alphablending slider (Figure 115).

3 Notice that Image A is currently the active image.

4 Select Image B in the Active Image and Alphablending slider to make Image B the active image.

5 Move the slider to the far right. As you are moving the slider, more of Image B appears in the image window. When the slider reaches the far right, all of Image B appears in the image window.

6 Click , the Default Mode icon.

If you want to take an action on one of the images—for example, adjusting the contrast or running an algorithm—make sure that it is the active image. If the inactive image is Image B, you also have the option of using the Image B region tool to view portions of the Image B. If the active image is currently Image A and you want to view Image B, simply select the Image B radio button. To make Image A the active image, select the Image A radio button.

VIEWING PORTIONS OF IMAGES USING THE CHECKERBOARD

The checkerboard tool displays both portions of both images in an alternating fashion, much like the squares on a checkerboard. A traditional checkerboard has a grid of alternating dark-colored and light-colored squares. Portions of Image A appear where the light-colored squares would appear on the checkerboard; portions of Image B appear in place of the dark-colored squares. You can adjust the number of rows and columns displayed.

To use the checkerboard tool

1 Open two images.

2 Load the second image into the first image window. The MIPAV window expands to include the Active Image and Alphablending slider.Click . The Checkerboard Pattern dialog box (Figure 12) appears.

Use checkerboarding	Specifies to use checkerboarding.
Rows	Indicates the number of rows you want to display in the checkerboard. You can display from 1 to 50 rows.
Columns	Indicates the number of columns you want to display in the checkerboard. You can display from 1 to 50 columns.
Apply	Applies the parameters that you specified to the images.
Close	Closes this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not apply checker boarding.
Help	Displays online help for this dialog box.

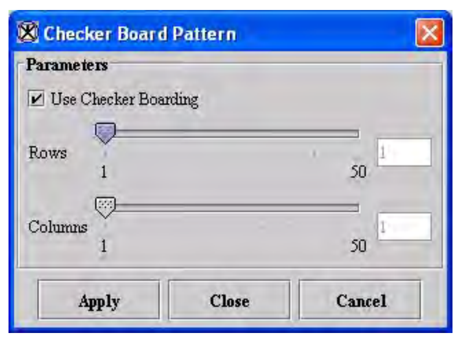


Figure 116. Checkerboard Pattern dialog box

3 Select Use checkerboarding if it is not already selected.

4 Adjust the Rows slider to indicate the number of rows that should be in the checkerboard pattern.

5 Click Apply when finished. The checkerboard pattern appears in the image window (Figure 13).

When the rows and columns intersect, a grid is created. Adjacent squares in the grid alternatively display the active and inactive images.

6 Click Close to close the Checkerboard Pattern dialog box.

Note: If you select another image—that is, an image that is not loaded into the same window as Image A and Image B, the Active Image and Alphablending slider disappears from the MIPAV window. The slider only appears when the selected image window contains two images.

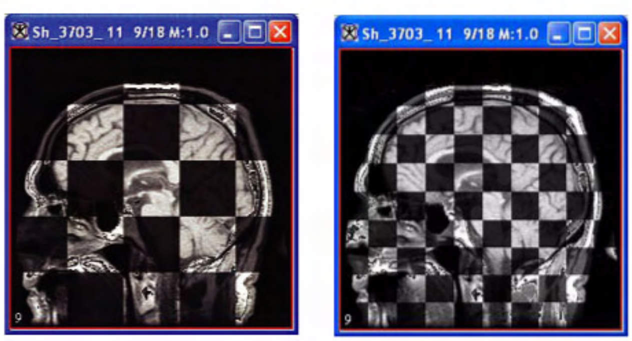
To remove the checkerboard from the image

1 Click . The Checkerboard Pattern dialog box (Figure 12 on page 22) appears.

2 Clear Use checkerboarding.

3 Click Apply. MIPAV removes the checkerboard from the image.

4 Click Close.



(A) Example of 5 rows and 5 columns

(B) Example of 10 rows and 10 columns

Figure 117. Examples of the checkerboard pattern

Changing image brightness and contrast using LUTs

Generally, computer systems have brightness display values written in the display hardware. These values are known as the *physical color map;* they are hard coded in your monitor. When you open an image, the image file contains data that indicates the intensity of each voxel in the image. These data are passed to the physical color map and displayed on the monitor. Additionally, MIPAV provides a logical color map, which allows you to remap the original intensities to other intensities. Although technically the term *look-up table* (LUT) can be used for the physical and logical color maps, in this guide look-up table refers to the logical color map only. You can apply predefined, pseudo color or inverse LUTs, or you can manually manipulate the transfer function used to map the image data to the LUT. The LUT then translates the remapped values so that they can be interpreted by the physical color map and displayed on your monitor.

To adjust the look-up table using the Quick LUT

To adjust the LUT, you open the Look-up Table window to modify the LUT's values and transfer function. However, you can use the Quick LUT icon to modify the LUT without opening the Look-up Table window. Quick LUT allows you to easily choose the highest and lowest values for the intensity levels in a user- defined area.

To do this, complete the following steps:

1 Open an image.

2 Click (Quick LUT) in the MIPAV window.

3 Move the cursor to the image window and draw a box around an area that has the highest and lowest intensities you want the image to display. These values are used to remap the image data to the LUT. The net effect is increased contrast in the area of interest (Figure 14).

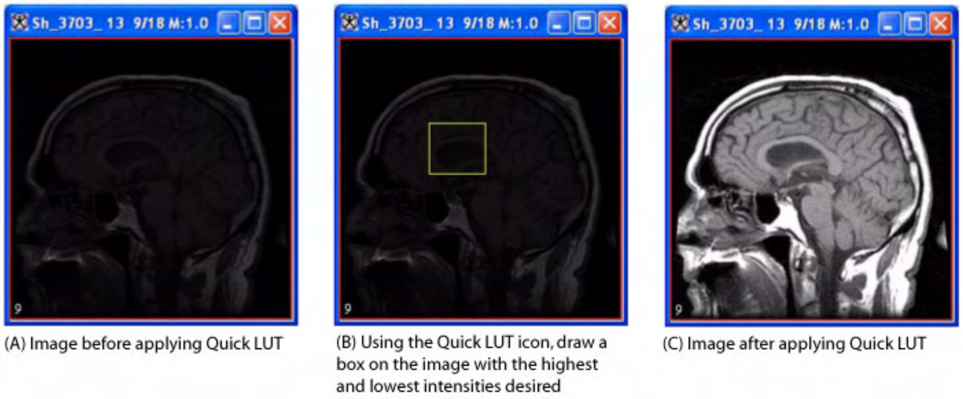


Figure 118. An image before and after applying Quick LUT

To generate a histogram and look-up table

A *histogram* is a graphic representation of the intensity level distribution in an image or VOI region. It displays the number of voxels at each intensity level. The histogram and LUT appear in the Look-up Table window.

To generate a histogram, and view the LUT, complete the following steps:

1 Open an image. The image appears in an image window.

2 Create a VOI on the image (optional step).

3 Do either of the following:
- Click Look-up Table.
- Select LUT > Histogram –LUT.

If the image contains a VOI, the Histogram window appears (Figure 15). Go to the next step.

Figure 119. Histogram dialog box

If there are no VOIs on the image, the Look-up Table window (Figure 16) appears.

4 Choose one of the following:

- *Whole image*—To generate a histogram for the whole image
- *VOI region(s)*—To generate a histogram for the VOI region of the image

5 Click OK. A progress message appears briefly. After a few moments, the Look-up Table window appears (Figure 16).

File	*Open LUT*—Opens a previously saved LUT file. LUT files have a *.LUT* extension. *Save LUT*—Saves the LUT displayed in this window in a LUT file. *Open Transfer Functions*—Opens a previously saved transfer function. Transfer function files have a *.FUN* extension. *Save Transfer Functions*—Saves the transfer function displayed in this window to a file. *Close LUT*—Closes the LUT window.
Utilities	*Change number of colors*—Allows you to change the number of colors displayed in the image. Valid values are 2 to 256. *CT function*—Allows you to select a preset LUT that is appropriate for the image content. Values are abdomen, head, lung, mediastinum, spine, and vertebrae. *Invert LUT*—Creates a negative of the image. *Reset histogram and LUT A*—Returns image A to its original values. *Reset histogram and LUT B*—Returns image B to its original values. This command is only available if two images are open.
LUT toolbar	Provides tools that allow you to manipulate the displayed image. Refer to Figure 19.
Update (real-time)	Changes the image as you make changes to the LUT, which allows you to see the effect of your changes immediately on the image.
Log scale (histogram)	Displays the image's histogram count in log scale along the *Y* axis.
Interpolate image	Displays image using interpolation, which reduces pixilated image to appear more smooth. **Caution:** Depending on the memory resources of your workstation, interpolation can be very lengthy.

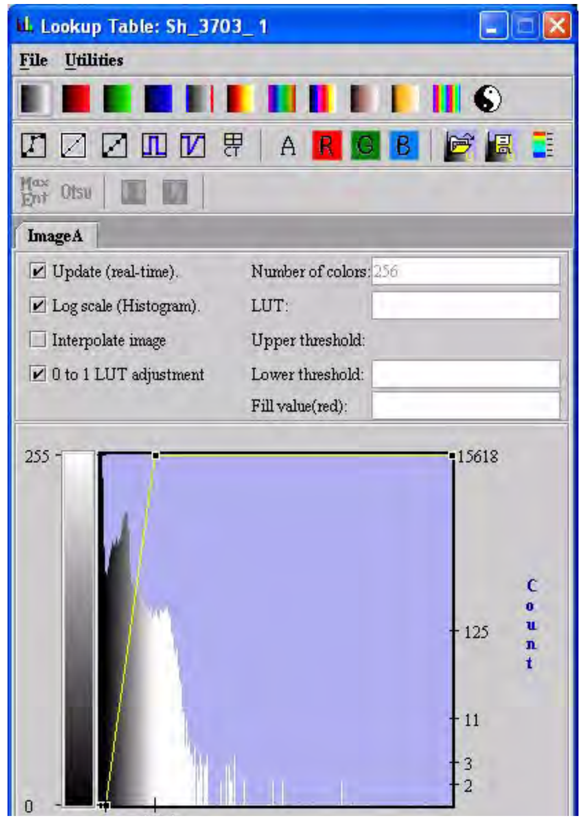


Figure 120. Look-up Table window

Number of colors	Allows you to change the number of colors displayed in the image.
LUT	Displays the image intensities.

Figure 120. Look-up Table window (continued)

The Look-up Table window consists of three sections: a menu bar, a toolbar, and one or more pages containing histograms. A tab appears for each image that is opened in the image window. For example, if only one image is in the image window, then only the Image A tab appears. If you generated the histogram for an image window that contains two images, a tab for Image A and a tab for Image B appear. Each of these tabbed pages contain a histogram for the applicable image. If you generated the histogram for a VOI, the window does not display a tab and only the applicable icons and buttons in the toolbar appear.

The toolbar allows you to manipulate the displayed image. You can apply pseudocolor LUTs, adjust the image contrast with the transfer function, and apply preset window and level settings for CT slices. You can also edit the red, blue, green, and alpha channels of a LUT.

look-up table (LUT)—Indicates the intensity of each voxel in the image and, in MIPAV, allows you to remap the original intensities to other intensities.

transfer function—Reflects the relationship between the original image intensity values and how they are mapped into the LUT. The line in the LUT represents the transfer function.

Note: You can generate a histogram for two image datasets that are loaded together. In this case, the Look-up Table window (Figure 17) shows two tabs—one for Image A and one for Image B.

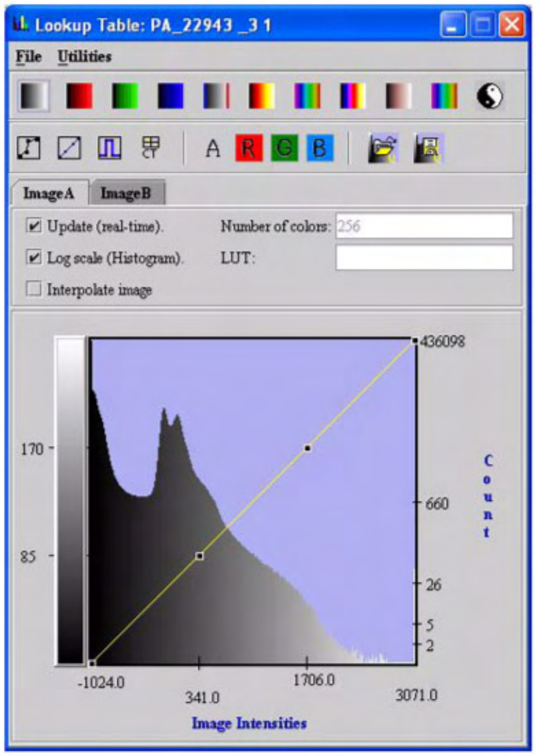


Figure 121. Look-up Table window showing Image A and B histograms

To update images in real time

When you modify the LUT, be sure to select the Update (real time) check box. The image in the image window is then updated in real time.

To change the number of intensities displayed in the LUT

You can change the number of intensities displayed in the LUT. To do this, do the following:

1. Select Utilities > Change Number of Colors in the LUT window. The Change Number of Colors dialog box opens.

Figure 122. Change Number of Colors dialog box

2. Type the number of colors you want in the Number of colors box. You can specify any whole number between 2 and 256.
3. Select Apply to apply the changes.

   Notice that the Number of colors box in the LUT window now displays the number you specified and the histogram changes to display the new colors.
4. Click Close or Cancel to close the dialog box.

APPLYING PREDEFINED LUTS TO IMAGES

You can use MIPAV's predefined LUTs to apply pseudocolor, create a negative of the image, and apply preset CT window and level settings to an image.

To apply pseudocolor LUTs to images

As you examine an image, you may need to observe small changes in intensity values or identify the same intensity values in different portions of an image. This can be difficult if the image is rendered in grayscale because the human eye can only see about 100 shades of gray. However, because varied colors are often easier to distinguish, MIPAV allows you to use various pseudocolor maps to elucidate objects of interest. Thus, MIPAV provides a variety of pseudocolor LUTs. If you apply a pseudocolor LUT, the grayscale intensity values are remapped to color intensity values. Note that the original image data is not changed; only the displayed image file (hence the term *pseudocolor*).

Figure 123. LUT toolbar

To apply a pseudocolor LUT, click one of the following icons:

- Red LUT
- Green LUT
- Blue LUT
- Gray blue/red LUT
- Hot metal LUT
- Spectrum LUT
- Cool hot LUT
- Striped LUT
- Invert LUT

The grayscale intensity values in the image dataset are remapped to color intensity values.

To invert intensities

To invert the intensities so that a negative of the dataset appears, click , the Invert LUT icon. The Invert LUT icon is in both the Look-up Table window (Figure 16 on page 26) and the MIPAV window. Figure 19 on page 30 shows the location of this icon in the Look-up Table window.

To apply CT level presets to images

There are six CT window and level presets: abdomen, head, lung, mediastinum, spine, and vertebrae. To apply a preset level to the image, complete the following steps:

1 Click , the CT Preset icon, in the LUT window. The CT Presets dialog box appears (Figure 20).

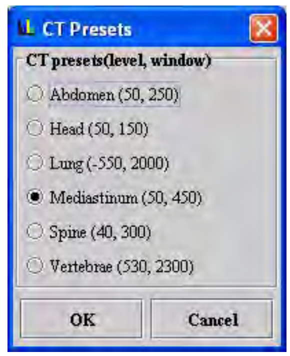


Figure 124. CT Presets dialog box

2 Select the desired CT preset. As you select the CT preset option, the colors in the image's histogram or LUT change, and, if you chose to update images in real time, the image changes.

3 Click OK to save the change.

ADJUSTING CONTRAST USING THE TRANSFER FUNCTION

The transfer function reflects the relationship between the original image intensity values and how they are mapped into the LUT. An example of how adjusting the transfer function affects the display of an image appears in Figure 16. In this example, the top image is generated by applying the linear transfer function (slope = 1) to produce display values that are evenly distributed over the range of the LUT (see Figure 125A). This results, in this case, in a low-contrast image (see Figure 125B). The contrast of the image can be improved by adjusting the transfer function in a manner shown in Figure 125A (e.g., changing a low-contrast image into a high-contrast image). The image scalar values between –175 and 275 are remapped as a function of the modified transfer function and distributed across the full

LUT range. The values above 275 are remapped to white and the values below –175 are remapped to black. The effect can be readily seen in Figure 125B.

To modify transfer functions

1 Open an image. The image appears in the default image window.

2 Click, the Displays Look-up Table (LUT) icon. The Look-up Table dialog box opens.

3 Click the transfer function. A new node may appear.

4 Drag the node to the new location.

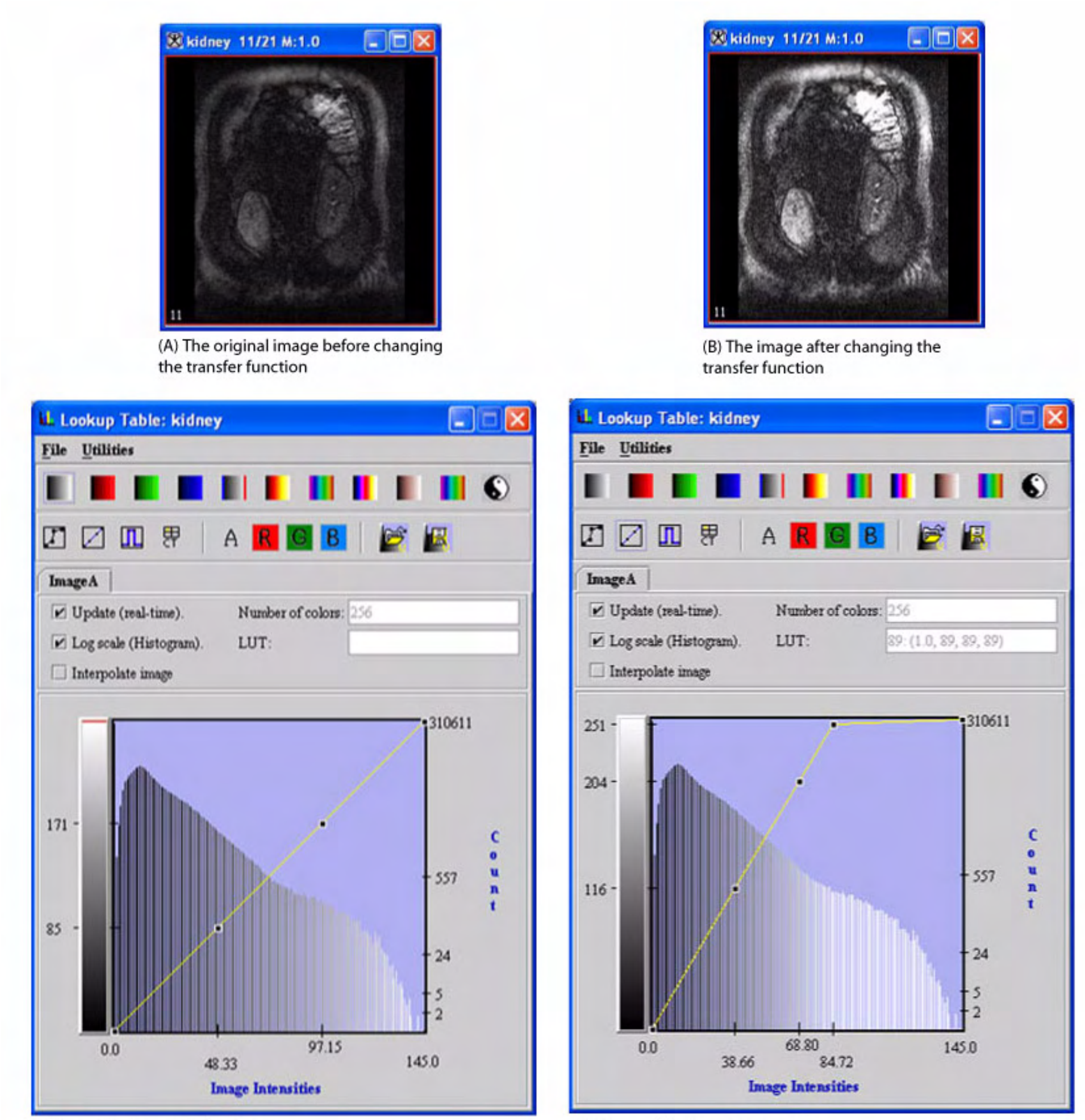
You can also adjust the transfer function for the alpha, red, green, and blue channels in an image.

Example: You might want to use these icons to highlight certain intensities in a particular color.

To do this, click the appropriate one of the following icons:

- A, the Edit Alpha icon
- R, the Edit Red icon
- G, the Edit Green icon
- B, the Edit Blue icon

When you click on one of these icons, the transfer function for that channel appears on the histogram and a node appears on that transfer function. Drag the node to the desired position. To adjust another channel, you must click on the icon and drag the node to the appropriate position.



(A) The original image before changing the transfer function

(B) The image after changing the transfer function

(C) The original transfer function

(D) The modified transfer function

Figure 125. An image before and after modifying the transfer function

To save transfer functions

To save a transfer function to a file, complete the following steps:

1 In the Look-up Table dialog box, select File > Save Transfer Functions (Figure 126) or press Ctrl+S. The Save dialog box (Figure 127) appears.

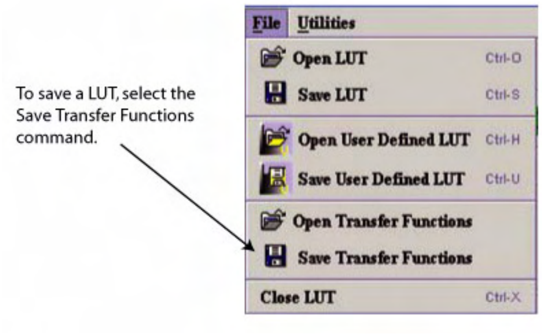


Figure 126. Open and Save commands in the File menu

2 Type a name to the transfer function in the File Name box. Be sure to add the .fun extension to the file name.

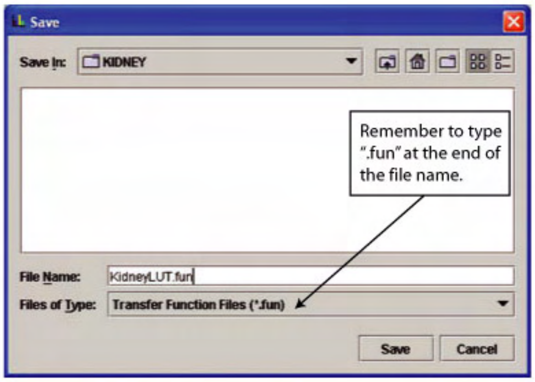


Figure 127. Save dialog box

3 Click Save. The program saves the transfer function under the name you specified.

To apply previously saved transfer functions

To open a transfer function file and apply it to an image, complete the following steps:

1. Select File > Open Transfer Functions in the Look-up Table window. The Open dialog box appears.
2. Select the desired file. LUT files have a `.fun` extension.
3. Click Open. The program applies the transfer functions file to the current image.

To save LUTs for later use

1. Select File > Save LUT in the Look-up Table window, or press Ctrl S. The Save dialog box (Figure 128) appears.

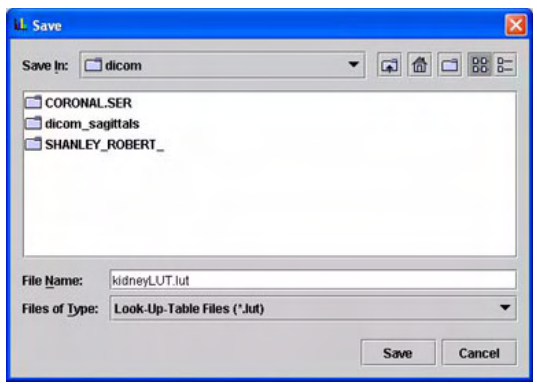


Figure 128. Save dialog box

2. Type a name for the LUT in the File Name box. Be sure to add the .lut extension to the file name.
3. Click Save. The program saves the LUT under the name you specified.

To open and apply previously saved LUTs to images

1. Select File > Open LUT in the Look-up Table window, or press Ctrl O. The Open dialog box (Figure 25) appears.
2. Select the desired file. LUT files have a .lut extension.

3 Click Open. The program applies the LUT settings from the LUT file you specified to the current image file.

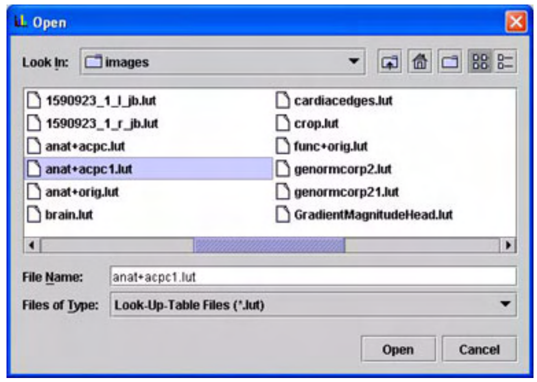


Figure 129. Open dialog box

To open, save, and apply frequently used LUTs

For a LUT that you defined and expect to use frequently, MIPAV provides a simple method for saving, opening, and applying it without needing to use the commands on the File menu. You use two icons on the toolbar in the Look-up Table window:

- , the Save User-Defined LUT icon, allows you to save the LUT.
- , the Open User-Defined LUT icon, provides a very quick way of opening and applying the user-defined LUT

Recommendation: Because these icons only apply to one user-defined LUT, it is recommended that you select the LUT that is used most frequently.

To reset original LUTs to images

Click , the Grayscale icon in the Look-up Table window (refer to Figure 16 and Figure 19). Alternatively, you can click , the Reset LUT icon, in the MIPAV window.

To adjust the threshold

1. Open an image. The image appears in the default image window.
2. Click , the Displays Look-up Table (LUT) icon. The Look-up Table dialog box opens.
3. Click , the Dual threshold function icon. The Threshold icon becomes active and the transfer function of the histogram changes.
4. Select Algorithms > Threshold. The Threshold dialog box (Figure 26) opens.

Set lower limits between (1.0–3774.0)	Threshold limit for the lowest image intensities.
Set threshold between (1258.3334–3774.0)	Threshold limit for the highest image intensities.
Produce binary image	Produces a binary image (Boolean).
Set values outside of limits to	Specifies the intensity value to assign to values outside the threshold limits.
New image	Shows the results of the algorithm in a new image window
Replace image	Replaces the current active image with the results of the algorithm.
Whole image	Applies the algorithm to the whole image.
VOI regions	Applies the algorithm to the volumes (regions) delineated by the VOIs.
OK	Applies the changes you made in this dialog box and closes the dialog box.

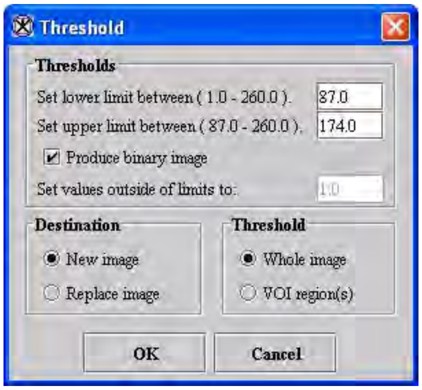


Figure 130. Threshold dialog box

Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not change the threshold.
Help	Displays online help for this dialog box.

Figure 130. Threshold dialog box (continued)

5 Complete the dialog box.

Note: You can choose to generate a binary image (Boolean) by selecting the Produce binary Image check box. Alternatively, you can clear the binary option and enter a threshold value. If you still want to generate a Boolean image, select the check box again. Note that, if you generate a Boolean image, MIPAV does not allow you to reapply the threshold or to generate either a histogram or LUT for a Boolean image.

6 Click OK to apply the threshold.

Displaying images using the animate view

If you select the *animate* view, the software makes a copy of the image dataset and loads it into an image window showing the animate view. In animate view, like cine view, the software automatically advances through the entire image dataset. However, the animate view allows you to adjust the speed at which images are advanced. You can specify that images be displayed in forward or reverse order. You can also pause or stop the sequence.

To view an image dataset in animate view

1 Select Image > Views > Animate in the MIPAV window. The Animate Parameters dialog box (Figure 27) appears.

2 Type the scale factor that you want to use in the Scale factor box.

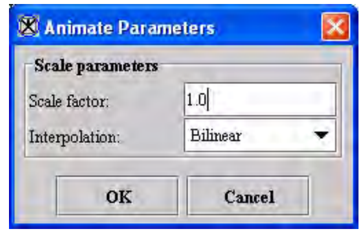
Scale factor	Changes the size of the dataset by the factor specified. Unlike magnification, the number of voxels used to replicate the image may change. For example, if you scale a dataset by a factor of 2, the image becomes twice the original size, and the number of voxels is squared.	
Interpolation	Determines the intensity of the additional voxels using one of the three following methods: **Bilinear—**Weighs the average of the four nearest voxels (to the newly interpolated voxel). The interpolated voxel is assigned the resulting intensity. **Bspline 3rd order—**Weighs the average of the 27 nearest voxels (to the newly interpolated voxel). **Bspline 4th order—**Weighs the average of the 64 nearest voxels (to the newly interpolated voxel).	
OK	Applies the scale factor and interpolation to the image dataset according to the specifications in this dialog box.	
Cancel	Disregards any changes that you made in this dialog box and closes this dialog box.	
Help	Displays online help for this dialog box.	

Figure 131. Animate Parameters dialog box

3 Select one of the three interpolation methods.

4 Click OK.

MIPAV transforms the dataset using the scale factor and interpolation method you indicated. A progress message (Figure 28) appears while the program constructs the animated file. It then loads a copy of the dataset in the Animate window.

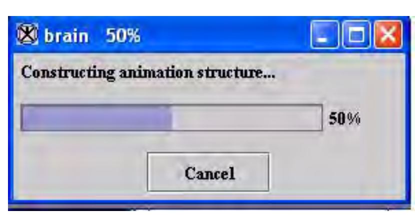
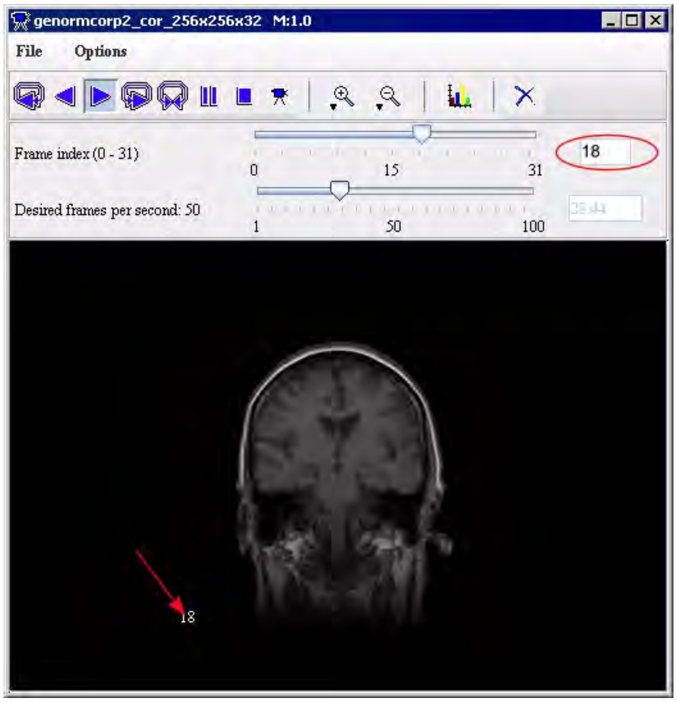


Figure 132. Animate progress message



File	**Save Image as .avi**—Saves the dataset as an AVI, or digital video, file.
	Close Animate—Closes the Animate window.
Options	**Show Z Slice Numbers**—Displays, if selected, the slice number, or the frame number, on the lower left side of the image.
Continuous reverse	Steps through all of the images in the dataset continuously in reverse without stopping at the speed specified in Desired frames per second.
Reverse	Steps through all of the images in the dataset once in reverse at the speed specified in Desired frames per second.
Forward	Steps through all of the images in the dataset once in sequence at the speed specified in Desired frames per second.
Continuous forward	Steps through all of the images in the dataset continuously without stopping at the speed specified in Desired Frames per second.
Backward and forward	Steps through all of the images in the dataset in sequence and then in reverse continuously without stopping at the speed specified in Desired Frames per second.

Figure 133. Image window shown in the Animate view

Pause	Pauses the animation at the frame currently shown.
Stop	Halts the animation and redisplays the first image in the dataset.
Magnify image 2.0X	Magnifies the image by 2.
Magnify image 0.5x	Magnifies the image by one-half.
Set brightness/ contrast	Specifies the brightness and contrast of the image. When you select this icon, the Brightness/Contrast dialog box opens. To change the brightness, slide the Brightness slider to the level desired and then click Apply. To change the contrast, slide the Contrast slider to the level desired and click Apply.
Delete current slice	Removes the slice currently shown in this window from the image dataset. **Caution:** The program does not allow you to undo slice deletion. Before you delete slices, you may want to have a backup of the complete image dataset to avoid losing images.
Frame number	Sets the first image slice shown to the frame, or slice, number that you specify. Notice that the frame number appears in the box to the right of the slider.
Desired frames per second	Specifies how many frames, or slices, should appear in 1 second. The read-only box to the right shows the *actual* number of frames per second your computer has achieved. The number of frames per second depends on the amount of memory accessible by the computer and CPU time.

Figure 133. Image window shown in the Animate view (continued)

ADJUSTING DATASET DISPLAY

You can magnify or reduce the display of your dataset using the magnification icons. Use , the Magnify image 2.0x icon, to enlarge the image to twice its current size.

Adjusting the brightness and contrast in images

To adjust the brightness and contrast in the image, do the following:

1. Click , the Brightness/Contrast icon, in the Animate window. The Brightness/Contrast dialog box (Figure 30) opens.
2. Move the Brightness slider to the position you want. Notice that the level of brightness changes on the image as you move the slider.
3. Move the Contrast slider to the desired contrast level. The contrast in the image changes as you move the slider.
4. Click Apply when you are finished. The image in the Animate window reflects the changes in brightness and contrast that you made.
5. Click , the close button at the top right of the dialog box, to close the dialog box.

Brightness	Specifies the level of brightness to apply to the image.
Contrast	Specifies the level of contrast to apply to the image.
Apply	Applies the brightness or contrast you specified to the image.
Cancel	Disregards any changes that you made in this dialog box and closes this dialog box.
Help	Displays online help for this dialog box.

Figure 134. Brightness/Contrast dialog box

Removing slices from image datasets

In the Animate window, you can also remove slices from the dataset. First, you need to find the slice that you want to delete. To find a slice number, do either of the following:

- Select Options > Show *Z* Slice Numbers, which shows the slice number at the lower left of the image (refer to Figure 29 on page 40).
- Move the Frame Number slider to the slice that you want to delete. The number appears in the box on the right of the slider.

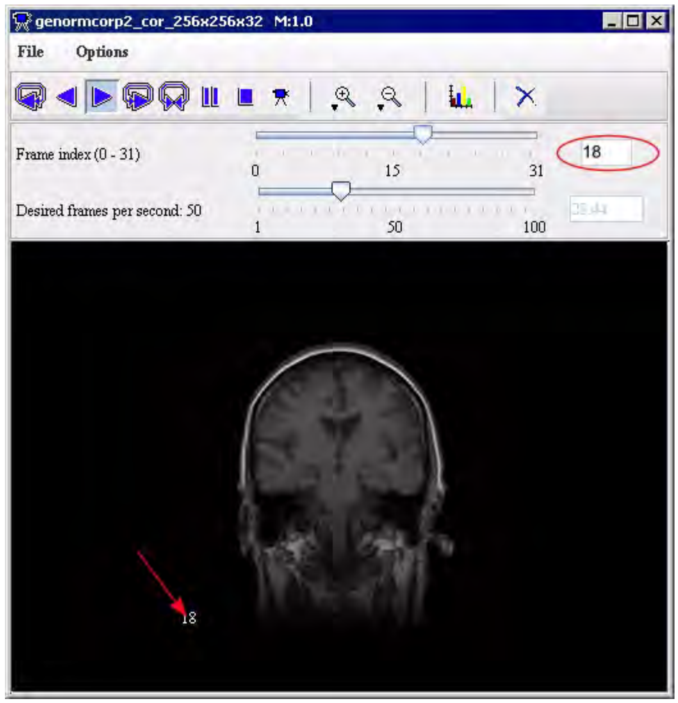


Figure 135. In the Animate dialog box, the slice number appears in the information box beside the Frame Number slider, and the Desired Frames per Second information box shows the actual number of slices per second.

Once you locate the slice that you want to remove, click , the Delete icon. The slice is removed from the dataset.

Caution: The program does not allow you to undo slice deletion. Before you delete slices, you may want to have a backup of the complete image dataset to avoid losing images.

PLAYING A DATASET "FILM"

Using the icons and buttons on the toolbar in the Animate window (Figure 29 on page 40), you can produce and play a dataset "film." To do so, complete the following steps:

1. Move the Frame number slider to the slice that you want to see first.
2. Move the Desired frames per second slider to indicate the speed at which images should be advanced or reversed.
3. Click the following buttons as appropriate:

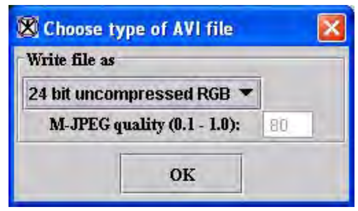
- Click , the Forward icon, to advance the image dataset forward through all of the images one time (one loop).
- Click , the Reverse icon, to advance the image dataset backward through all of the images one time (one loop).
- Click , the Continuous Forward icon, to advance the image dataset forward through all of the images multiple times. To stop image advancement, you must press , the Pause icon, or , the Stop icon.
- Click , the Continuous Reverse icon, to advance the image dataset backward through all of the images multiple times. You must press the press , the Pause icon, or , the Stop icon, to stop image reversal.
- Click , the Pause icon, to pause the image advancement or reversal.
- Click , the Stop icon, to stop the image advancement or reversal.

Note: As the images advance or reverse, you may notice that the desired frames per second and the actual frames per second differ. Although you can type any number in the desired frames per second, the actual speed at which the images advance depends primarily on your computer's processor speed.

4 Save the animate viewer file by selecting File > Save Image as .avi (AVI, or audio video interleaved, is a Microsoft multimedia format). The Choose Type of AVI File dialog box (Figure 32) opens.

5 Select a type of file.

6 Click OK. The program runs the animation sequence that you created, closes this dialog box, and then displays the Save dialog box.



24-bit uncompressed RGB	An uncompressed RGB file, which uses 8 bits per color (red, green, blue).
8-bit RLE compressed	A run length encoded (RLE) file that is converted to 8 bits and is compressed.
Quicktime movie	An Apple format.
Motion JPEG	A format that consists of motion JPEGs.
MPEG-4 version 2	Another compression format.
OK	Plays back the animated sequence that you created and then displays the Save dialog box for you to save the sequence.
Cancel	Disregards any changes that you made in this dialog box and closes this dialog box.
Help	Displays online help for this dialog box.

Figure 136. Choose Type of AVI File dialog box

7 Specify a name for the animation sequence in the File Name box.

8 Click OK. The dialog box closes and the Choose File Type dialog box (Figure 33) opens.

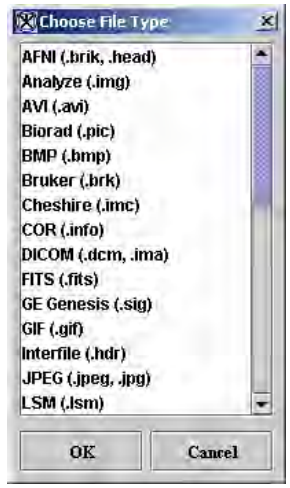


Figure 137. Choose File Type dialog box

9 Select a file type.

10 Click OK. The Save Range of Slices dialog box (Figure 34) opens.

11 Close the Animate window by either selecting File > Close Animate or pressing Ctrl X. The Animate window closes.

Save image to separate files	Saves the animation to separate files. Selecting this check box makes the First file starting number text box and the File name number of digits text box become active.
First file starting number	Indicates which frame, or slice, to begin the first file saved. To make this box active, you must first select Save image to separate files.
File name number of digits	Specifies how may digits you want used for the file name. To make this box active, you must first select Save image to separate files.
First slice	Specifies the first slice that should appear in the animation file.
Last slice	Specifies the last slice that should appear in the animation file.
First time period	Specifies the time period of the images that should start the file. This box only applies to 4D images.
Last time period	Specifies the time period of the images that should end the file. This box only applies to 4D images.
Save with packed bits compression	Indicates that the file should be saved as compressed TIFF files. This check box only applies to 4D images.
OK	Saves the animation according to the specifications in this dialog box and closes the dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes this dialog box.
Help	Displays online help for this dialog box.

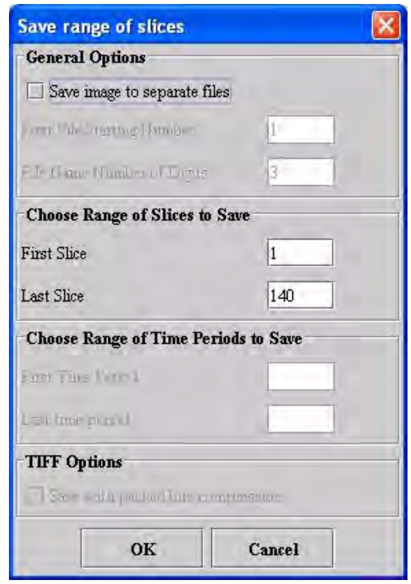


Figure 138. Save Range of Slices dialog box

Displaying images using the cine (movie) view

The *cine* view, like the default view, displays images one at a time in the default image window. However, in cine view, the software automatically advances the images.

To display a 3D or 4D dataset in this view, select Image > Views > Cine

(Movie). The images appear sequentially in the default image window. All images in the dataset are displayed only once. The cine view does not loop images, nor does it show images in reverse order.

Displaying images using the lightbox view

You can use the *lightbox* view to display datasets that have more than two dimensions (2D). Like its real-world counterpart, the lightbox view displays several images, or slices, in the dataset at one time in an image window. Through the lightbox settings, you can easily control the number of slices that appear.

To view an image dataset in lightbox view

1. Open an image dataset that has more than two dimensions. It appears in the image window in default view (Figure 35).
2. Select Image > Views > Lightbox in the MIPAV window. The image window for the lightbox view appears (Figure 35). The image number, or slice number, appears in the lower left corner of each image.

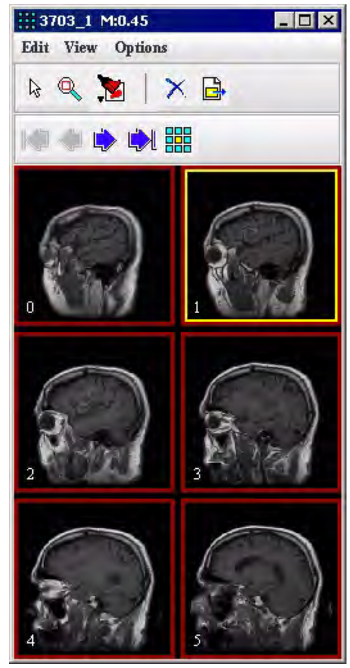
   Notice that the image you originally opened in step 1 remains on your desktop. So you not only see the image dataset in the lightbox view, but you also see the image in the default view.

Using the lightbox view to update the image slice displayed in the default view

The MIPAV window allows you to step through all of the slices of an image dataset shown in the default view one at a time using the Image slice slider,  , the Decrements image slice icon, and  , the Increments image slice icon. However, the default view only shows one slice at a time, but the lightbox view can show two or more slices at one time depending on the lightbox settings (that is, you can control the number of image slices that appear in lightbox view; refer to "Adjusting the lightbox view" on page 50).

Suppose it is easier for you to find a particular slice in the dataset by viewing several slices at the same time in order to compare them against each other. To do so, you would open an image dataset in default view and then display the dataset in lightbox view. The lightbox view displays the number of image slices that you had already specified.

At this point your desktop shows both the image dataset in default view and the dataset in lightbox view. The lightbox view allows you to quickly look through the entire dataset until you locate the correct slice. Once you find it, you may want to perform some actions, such as running algorithms, on that slice. To update the image in the default view to display the slice you located, first select and then right-click the image slice in the lightbox view. The default image window now shows the image slice you found.



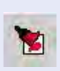
Edit	**Select All**—selects all of the slices in the dataset. **Select None**—deselects all of the selected slices. **Invert Selections**— deselects selected slices and select those that were not selected. **Delete**—deletes the selected slices from the dataset. **Extract**—copies the selected slices to a separate image window, which you can save as a separate dataset.
View	**First Page**—displays the first set* of slices in the dataset. **Previous Page**—displays the previous set* of slices in the dataset. **Next Page**—displays the next set* of slices in the dataset. **Last Page**—displays the last set* of slices in the dataset.
Options	**Settings**—allows you to change settings for the number of rows and columns used to display images in the dataset, change the grid size and the size of the frame border, change the colors of the borders and frames, and the level of magnification. **Save Settings**—saves the settings currently set in the Lightbox Settings dialog box. **Continuous Update**—updates all of the displayed images when you make a change to one of the images.
 **Default mode**	Displays the images according to the standard settings used by MIPAV when it is initially installed.
 **Magnify region**	Magnifies a portion of the image from 1 to 32 times. To change the level of magnification or the size of the magnifier, right-click on the image, and the Magnification dialog box opens.
 **Repaints images**	Updates the images displayed in the lightbox reflecting the changes made to the lightbox settings and to the dataset.

Figure 139. Image window showing the lightbox view

*The set of slices depends on the number of slices currently being displayed. For example, if four slices are currently displayed, then View > Previous Page shows the previous set of four slices. If seven slices are currently displayed, then View > Previous Page displays the previous seven slices in the dataset. View > Next Page works similarly. It displays the next set of slices in the dataset.

Delete selected slices	Removes the selected slices from the image dataset.
Extract selected slices to a new image	Makes a copy of the selected slices in the image dataset to a new image window.
First page	Displays the first set* of slices in the dataset.
Previous page	Displays the previous set* of slices in the dataset.
Next page	Displays the next set* of slices in the dataset.
Last page	Displays the last set* of slices in the dataset.
Go to active slice	Displays the currently selected slice.
Go to slice	Specifies which slice you want to display. Type a number in this box and then select Enter to display the slice. The slice then appears in the lightbox view.
Image window	Displays an individual slice in the dataset. To select a slice, click once on the slice. A thin red line appears around the slice inside the border.
Border	Outlines each slice.
Background	Provides the grid for the images in the window.

Figure 139. Image window showing the lightbox view (continued)

*The set of slices depends on the number of slices currently being displayed. For example, if four slices are currently displayed, then View > Previous Page shows the previous set of four slices. If seven slices are currently displayed, then View > Previous Page displays the previous seven slices in the dataset. View > Next Page works similarly. It displays the next set of slices in the dataset.

Adjusting the lightbox view

By default, the lightbox view shows only one column of images. Each image in the dataset is surrounded by frame borders and background (Figure 36). However, you can adjust the lightbox view so that you can see several columns and rows of images at a time (Figure 36) in a grid. You can also adjust the borders and the default magnification level of the images.

To change the grid configuration

1. Select Options > Settings in the image window for the lightbox view. The Lightbox Settings dialog box (Figure 140) appears.
2. Select Row.
3. Type the number of rows you want to appear in No. rows.
4. Select Column.
5. Type the number of columns in No. columns.

Row	Selects the row settings.
Column	Selects the column settings.
No. rows	Specifies the number of rows that should exist in the lightbox view.
No. columns	Specifies the number of columns that should exist in the lightbox view.
Grid size	Indicates the spacing in pixels used for the grid size.
Frame border slice	Indicates the spacing in pixels used in the grid.
Background color	Allows you to choose the color of the background of the lightbox. When you select this icon, the Pick Background Color dialog box opens.
Border color	Allows you to choose the color of the border that surrounds each image displayed in the lightbox. When you select this icon, the Pick Border Color dialog box opens.
Magnification	Controls the level of magnification.
Apply	Applies the parameters that you specified to the lightbox view.

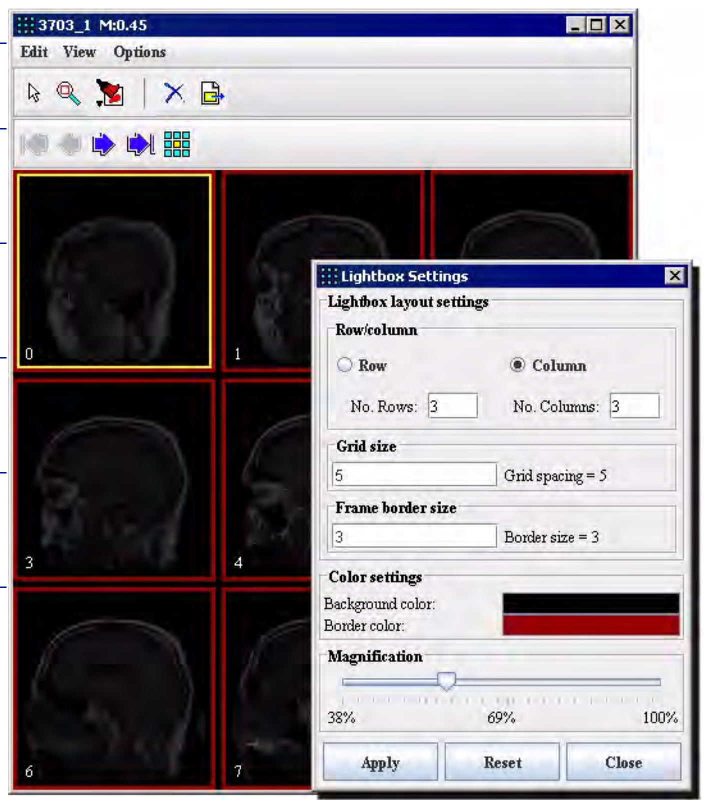


Figure 140. Lightbox Settings dialog box

Reset	Returns all of the parameters to their default values.
Close	Closes this dialog box without making changes to the lightbox view. If you changed the parameters in this dialog box, you must first select Apply to put those changes into effect.
Help	Displays online help for this dialog box.

Figure 140. Lightbox Settings dialog box (continued)

6 Click Apply. Notice that the number of rows and columns in the lightbox view changed to the numbers you specified.

7 Continue to make changes to other lightbox settings.

8 Click Close to close the Lightbox Settings dialog box.

9 Click Options > Save Settings to save the settings as the default number of rows or columns for the next time you display the lightbox view.

To change the grid size

1 Select Options > Settings in the image window for the lightbox view. The Lightbox Settings dialog box (Figure 37) appears.

2 Type a number from 0 to 20 in Grid size (Figure 38).

Note: On the right of the Grid size text box, Grid spacing shows the current grid size.

Figure 141. Grid size box

3 Click Apply. Notice that the grid size in the lightbox view changed to the size that you specified.

4 Continue to make changes to other lightbox settings, or click Close to close the Lightbox Settings dialog box. The image window for the lightbox view appears.

5 Click Options > Save Settings (Figure 39) to save the settings as the grid size for the next time you display the lightbox view.

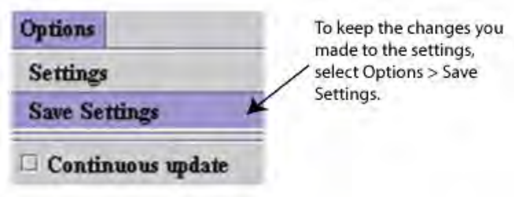


Figure 142. Save Settings command on the Options menu in the Lightbox Settings dialog box

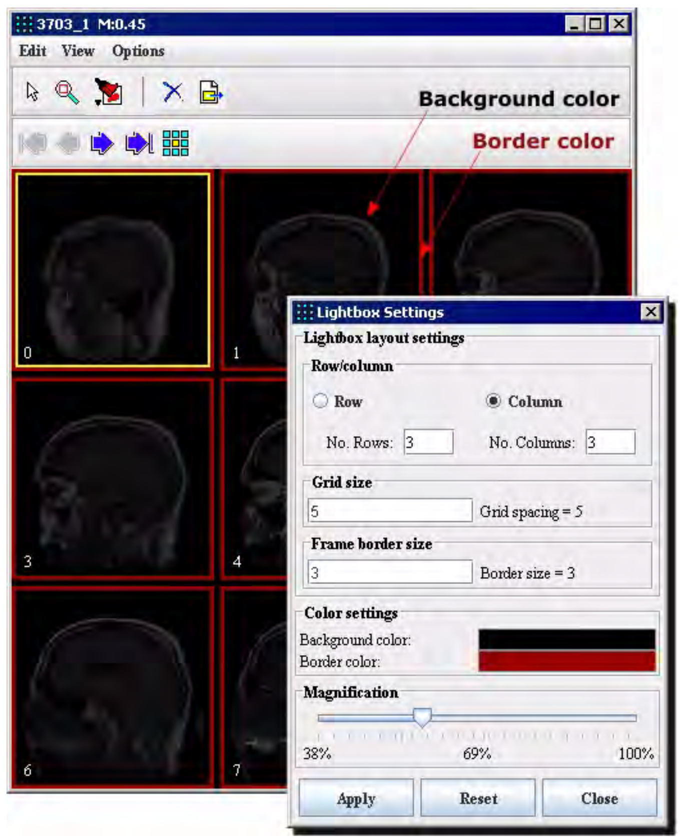


Figure 143. Grid and frame border

To change the border size

1 Select Options > Settings in the image window for the lightbox view. The Lightbox Settings dialog box (Figure 37 on page 51) appears.

2 Type a number from 0 to 10 in Frame border size (Figure 41). The unit of measurement for the number is in pixels.

Figure 144. Frame border size box

3 Click Apply. On the right of Frame border size, Border size changed to the size you specified.

4 Continue to make changes to other lightbox settings, or click Close to close the Lightbox Settings dialog box.

5 Click Options > Save Settings to save the settings as the default border size for the next time you display the lightbox view.

To change background and border colors

1 Select Options > Settings in the image window containing the lightbox view. The Lightbox Settings dialog box (Figure 37 on page 51) appears.

Notice that Background color displays the color of the current grid border, and Border Color shows the color of the current frame border.

2 Click the background color bar (Figure 42).

The Pick Background Color dialog box (Figure 43) opens.

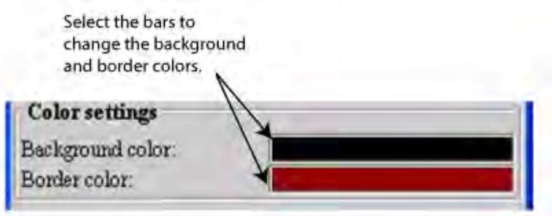


Figure 145. Background color and border color bars

This dialog box includes three tabbed pages:

- Swatches page (Figure 43), which, by default, is always displayed first
- HSB page (Figure 43)
- RGB page (Figure 43)

You can change the color of the background on any of these pages

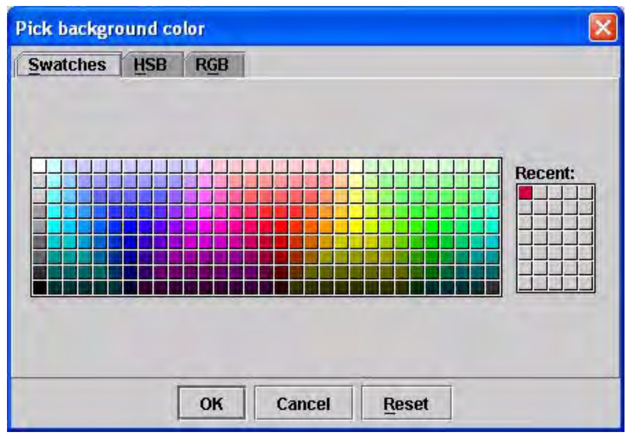


Figure 146. Swatches page in the Pick Background Color dialog box

3 Select a color from the Swatches, HSB, or RGB page. (For instructions on how to select colors, refer to "Manually creating a mask using paint" on page 332 in Chapter 6, "Segmenting Images Using Contours and Masks").

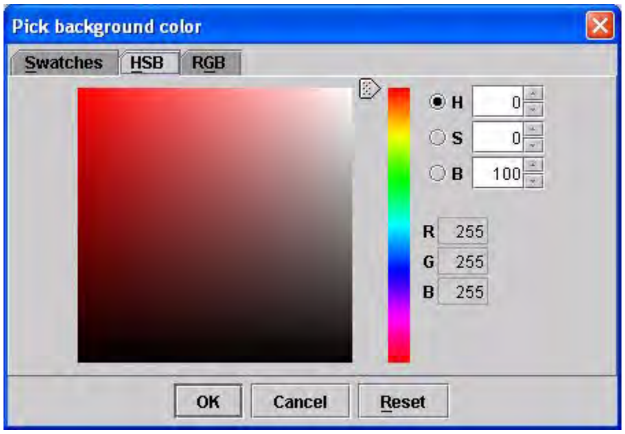


Figure 147. HSB page in the Pick Background Color dialog box

4 Click OK to apply the color to the background or border.

The Pick a Background Color dialog box closes, and the Lightbox Settings dialog box appears.

5 Click Apply.

Notice that the color of the background or border colors in the lightbox view changed to the colors you specified.

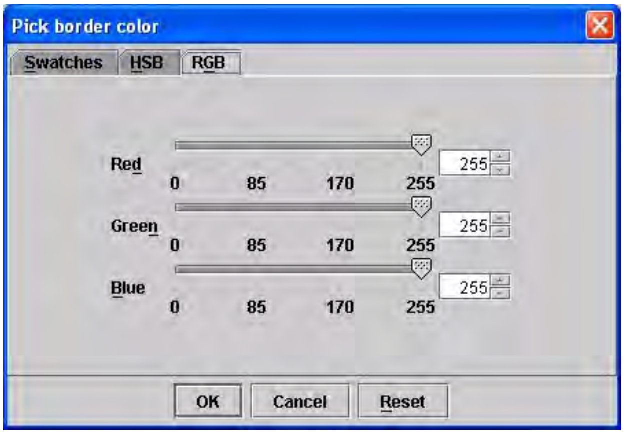


Figure 148. RGB page in the Pick Background Color dialog box

6 Continue to make changes to other lightbox settings, or click Close to close the Lightbox Settings dialog box.

7 Click Options > Save Settings in the lightbox view to save the settings as the default background and border colors for the next time you display the lightbox view.

To change the default magnification level for one slice in the dataset

1 Click .

2 Right-click on the image. The Magnification dialog box (Figure 46) opens.

3 Change the magnification level by sliding the marker on the Magnification slider to the level you want.

4 Select Display intensity values if you would prefer to see the intensity values in the magnified region rather than the magnified image.

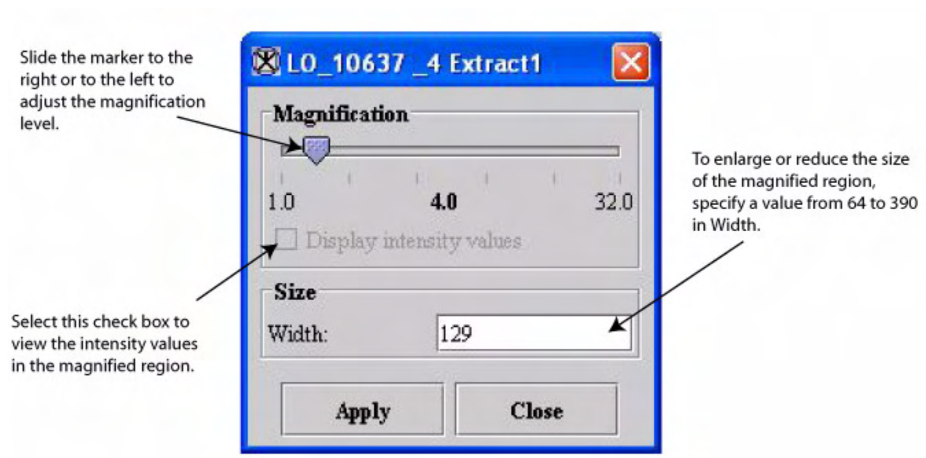


Figure 149. Magnification dialog box

5 Type a number from 64 to 390 to enlarge or reduce the size of the magnified region.

6 Click Apply. The settings specified in the Magnification dialog box take effect immediately and only for that one image, or slice.

To enlarge or reduce the size of all images

1 Select Options > Settings in the image window containing the lightbox view. The Lightbox Settings dialog box opens (Figure 37 on page 51).

2 Select the level of magnification by moving the slider right or left to the appropriate value.

3 Click Apply. Notice that the image magnification changed to the magnification you specified.

4 Continue to make changes to other lightbox settings, or click Close to close the Lightbox Settings dialog box.

5 Click Options > Save Settings in the lightbox view to save the settings as the default magnification for the next time you display the lightbox view.

Tip: To return to the original settings for the lightbox view, click Reset in the Lightbox Settings dialog box.

In the example shown in Figure 47, the researcher changed the original lightbox view (Figure 47A) in several ways. The number of rows was increased from two to four and the number of images in each row from two to three. The frame border color was changed from yellow to gray and the grid color from blue to black. Also the magnification level of the images was reduced to 38 percent of the original image size. Figure 47B shows the final lightbox view.

Repainting (updating) the lightbox

You can simultaneously display the same dataset in different views on your desktop. For example, you can display the dataset in default view and in lightbox view. If you perform an action on the dataset that is visualized in default view, such as contour a structure, you can update the dataset display so that the contours also appear in lightbox view. You can do this automatically or manually.

To automatically update the dataset in the image window

Select Options > Continuous Update in the lightbox view. The dataset is refreshed whenever a change occurs in the default image window.

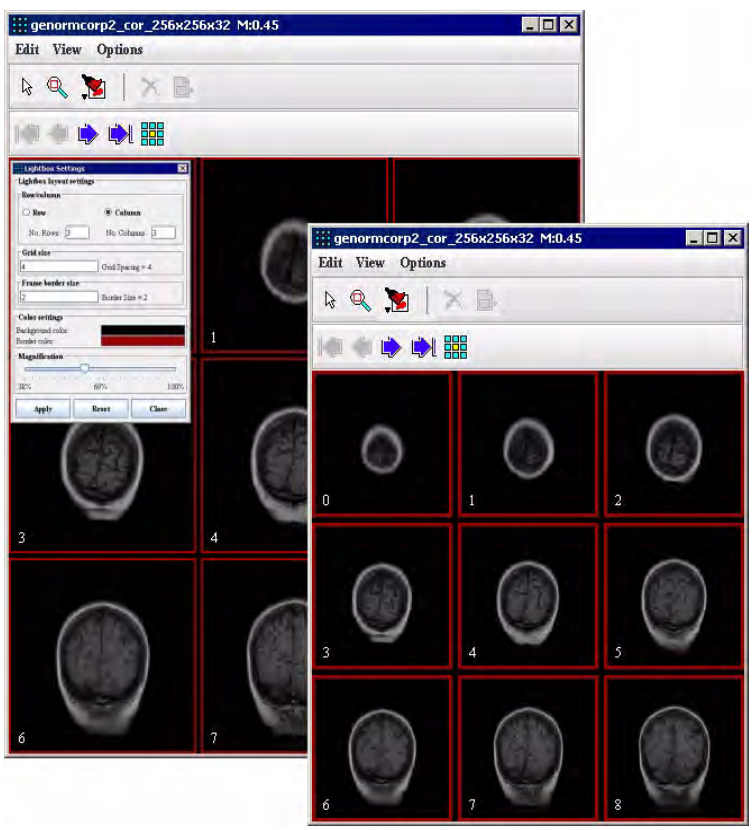
Caution: Although this option is convenient, it may slow software response time if you are working with a large dataset.

To manually update the dataset in the image window

Click , the Repaint icon, from the toolbar in the lightbox view.

To close the Lightbox window

Click in the upper right corner of the Lightbox window. A confirmation message (Figure 48) appears. To close the window, click Yes and the Lightbox window closes. If you change your mind about leaving the lightbox view, click No.



(A) original lightbox view (image on the left) and (B) changed lightbox view (image on the right)

Figure 150. Some of the display options available in lightbox view:

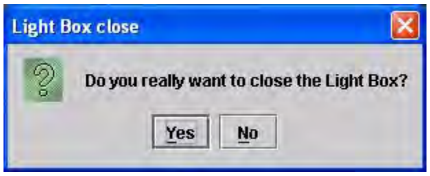


Figure 151. Lightbox Close message

Displaying images using the link to another image view

At times you may want to compare two images slice by slice side by side. Linking one image to another allows you to make this type of detailed comparison.

To compare images side by side

1. Open the two images you want to compare.
2. Select one of the images. This image becomes known as Image A, and the second image becomes known as Image B.
3. Select Image > Views > Link to Another Image. The Image Frame Linker dialog box opens.
4. Select the name of the second image, or Image B, in the Image B box.
5. Click Link. MIPAV immediately connects the images.
6. Move the Image slice slider in the MIPAV window. Notice that the image slices change on both images so that Image B shows the exact same slice as shown on Image A.

Tip: Select Image B, and move the Image slice slider. Although the slices change in Image B, they do **not** change in Image A. The reason is that only Image A controls the simultaneous movement of slices for both images. Therefore, when you select an image to be Image A, make sure that the image is the one you want to control the movement of both images.

Image A	Indicates the name of the Image A file. This field is a read-only field.
Image B	Allows you to choose the open image that you want to use as Image B.
Link	Links the two image files.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not link the images.
Help	Displays online help for this dialog box.

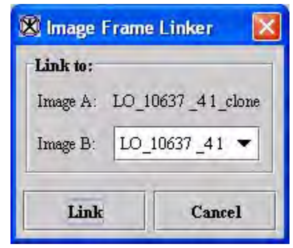


Figure 152. Image Frame Linker dialog box

Displaying images using the triplanar view

The *triplanar* view allows you to see three orthogonal planes of 3D or 4D datasets. The three orthogonal views enhance the visualization of important aspects of the anatomy and, in some cases, aids some manual segmentation processes.

Displaying image datasets in triplanar view

To view an image dataset in triplanar view, open an image and then select Image > View > Triplanar in the MIPAV window. The triplanar view (Figure 50 on page 62) appears in an image window.

The image window for the triplanar view includes a File menu, an Options menu, a toolbar, and three views of the dataset. The bottom right corner of the window displays the currently selected coordinate. Two guidelines (cross hairs) also intersect at the selected point. Additionally, the axis appears in the upper left corner in each view of the dataset.

You can use icons and buttons on the toolbar to change the magnification level, draw a point or line, apply paint, and apply a new intensity values to selected voxels in the image.

Designating new center points for image datasets

You can designate a new center point for the dataset. To do this, complete the following steps:

1. Click , the Volume Center icon, to display the center of a volume. The cross hairs intersect where the center of the volume has been calculated.
2. Drag the cross hairs to the point that you want to designate as the new center.
3. Release the mouse button.
4. Click Apply. The Apply Transformation Matrix dialog box (Figure 51 on page 65) opens.
5. Complete the dialog box.

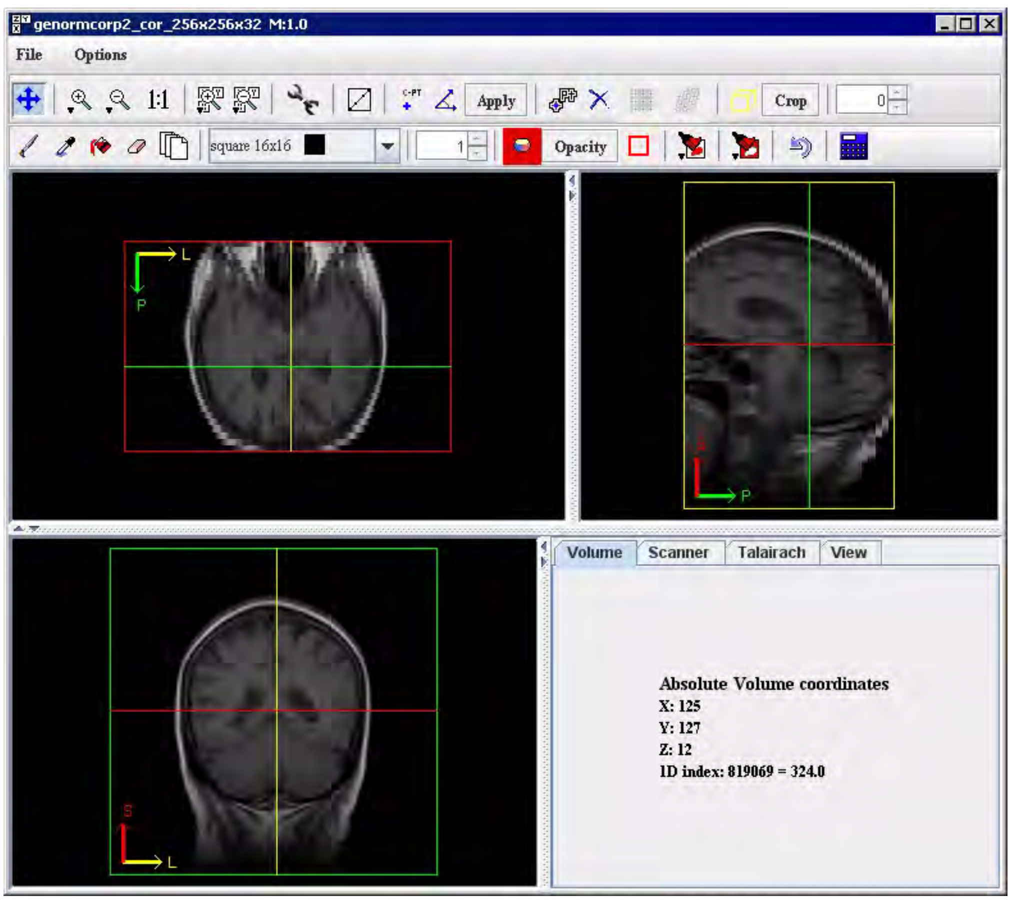


Figure 153. Image window showing the triplanar view

File	**Close frame**—Closes this window.
Options	**Show Axes**—Displays the *XY* (top left), *ZY* (top right), and *ZX* (bottom left) axes on the images.
	Show Cross Hairs—Displays cross-hair lines on the images. When you move these lines on one image, it also moves to the corresponding points on the other two images. Notice that the Absolute and Position values at the bottom right of the screen also change when you move the cross hairs.
	Show Talairach Grid—Displays a Talairach (stereo tactic) grid on the images.
	Show Talairach Position—Lists the Talairach position.
	Link to Another TriImage—Links this image to another image in triplanar view.
Traverse image	Aligns two images.
Magnify	Magnifies the image. It might be necessary to manually resize the window and use scroll bars to view all of the image.
Minify	Minifies the image (reduces the magnification level).
Original magnification	Displays image at the original magnification level.
Volume center	Reassigns a new center to the dataset. The location that you select appears in the center of the image window.
Alignment tool	Allows you to apply a transformation matrix to an image dataset.
Apply	Applies the rotations and translations made to the image.
Point	Draws a point on an image.
New Contour	Assigns the subsequently drawn contour to a new group. If this icon is not selected, all subsequent contours are assigned to the existing group.

Figure 153. Image window showing the triplanar view (continued)

Delete	Deletes a single contour or a group of contours. **Caution:** When you select this icon, the contour is not copied to the clipboard. It is permanently deleted.
Crop Volume	Indicates the volume that should be cropped.
Crop	Crops the specified volume.
Paint Brush	Selects paint brush tool. You can use the paint brush to fill specific voxels with color.
Eyedropper	Changes the default intensity to the one sampled by the eyedropper. Click this icon, then move the pointer to the image. The sampled intensity becomes the default intensity.
Paint Grow	Applies the paint grow segmentation method. This method uses voxel aggregation to group voxels into larger regions. The effect is much like a fill operation; when you click a voxel, this algorithm is used to determine whether adjacent voxels should be filled with the default color or intensity.
Erase	Deletes painted sections of the image. To delete, click the icon and move the pointer on painted areas of the image.
Global Erase	Deletes all paint from the image.
Small Tip	Used with the paint brush and erase icons. Click this icon to indicate the size of the eraser or the paint brush tip. Click this icon to paint or erase one voxel each time the mouse button is clicked.
Medium Tip	Used with the paint brush and erase icons. Click this icon to indicate the size of the eraser or the paint brush tip. Click this icon to paint or erase 16 voxels (4 x 4 square) each time the mouse button is clicked.
Large Tip	Used with the paint brush and erase icons. Click this icon to indicate the size of the eraser or the paint brush tip. Click this icon to paint or erase 100 voxels (10 x 10 square) each time the mouse button is clicked.
Paint Intensity	Indicates the intensity value.
Change Paint Color	Changes the color of the paint used. This icon is often used in conjunction with the Paint Brush and Paint Grow icons. Click this icon and select the desired color from the window that appears.

Figure 153. Image window showing the triplanar view (continued)

Opacity **Opacity**	Indicates the opacity level of the paint. When this icon is clicked, a window appears that allows you to indicate the opacity of the paint: 0 is transparent, 1 is opaque. By default the paint is translucent (0.3), which allows you to see the original image under the paint.
Commit **Commit**	Permanently applies intensity level to the image.

Figure 153. Image window showing the triplanar view (continued)

6 Click Apply. A progress message appears as the image is transformed. In a few moments, the new image either appears in a new window or replaces the image in the current window depending on your choice.

7 Select C-PT again. Notice that the image center was moved to the center you designated.

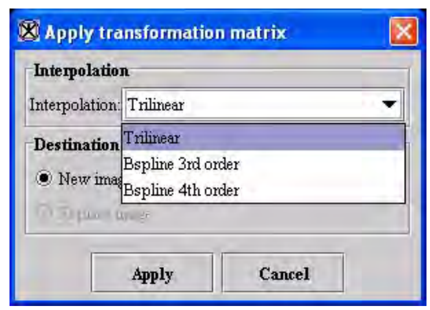
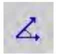
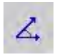
Interpolation	Determines the intensity of the additional voxels using one of the three following methods: **Bilinear**—Weighs the average of the four nearest voxels (to the newly interpolated voxel). The interpolated voxel is assigned the resulting intensity. **Bspline 3rd order**—Weighs the average of the 27 nearest voxels (to the newly interpolated voxel). **Bspline 4th order**—Weighs the average of the 64 nearest voxels (to the newly interpolated voxel). 
New image	Indicates where the results of the transformation appear. If you select this option, the transformed image appears in a new image window.
Replace image	Indicates where the results of the transformation appear. If you select this option, the transformed image replaces the current active image.
Apply	Applies the transformation to the image dataset according to the specifications in this dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes this dialog box.
Help	Displays online help for this dialog box.

Figure 154. Apply Transformation dialog box showing choices in the Interpolation list

Aligning image datasets

You can use , the Alignment Tool icon, to create a new transformation matrix for the dataset (refer to Figure 52).

The following instructions explain how to use the alignment tool:

1 Click . A closed angle, which looks like a line that is slightly thicker on one end, appears on each of the three images displayed in triplanar view.

2 Click the angle. Arrows appear at the end of each ray. The measurement of the angle appears beneath the angle.

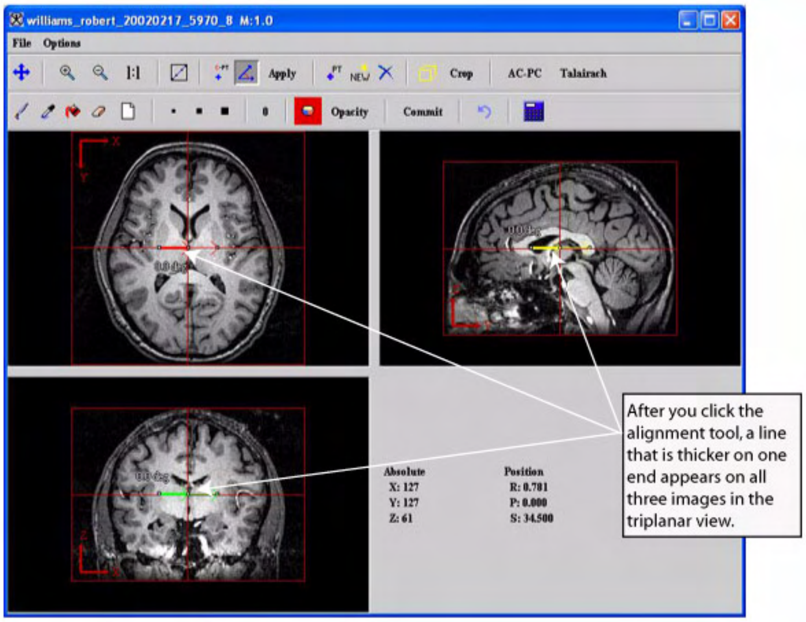


Figure 155. Using the alignment tool in triplanar view

3 Open the angle. The angle degree indicates the direction and distance to rotate the image. You can open the angle with or without constraints. (If an angle is opened with constraints, it only opens in 90-degree increments.) Do the following:

- *To open the angle in 90-degree increments:* Position the pointer over the shorter ray of the angle. The pointer changes to a cross-hair shape. (The cross-hair pointer looks much like a plus sign.) Drag the shorter ray to the desired location. The new angle measurement appears below the angle.

- *To open an angle without constraints:* Position the pointer over the longer ray of the angle. The pointer changes to a cross-hair shape. The cross-hair pointer looks much like a plus sign. Drag the longer ray to the desired location. The new angle measurement appears below the angle.

4 Click Apply. The Apply Transformation Matrix dialog box (Figure 51 on page 65) appears.

5 Complete the dialog box.

6 Click Apply.

A progress message appears as the image is transformed. In a few moments, depending on your choice in the dialog box, the new image either appears in a new window or replaces the image in the current window.

7 Click , the Traverse Image icon, to return the cursor to normal mode.

Creating, modifying, and deleting point VOIs

You can create, modify, and delete a point VOI. For more information on how to do this, see Chapter 7.

Cropping images

To crop an image, complete the following steps:

1. In an image window that is displaying the image dataset in triplanar view, click , the Crop Volume tool.
2. Draw a rectangle around the area of the image that you want to remain. The area outside of the rectangle will be cropped.
3. Click . A confirmation box appears.
4. Click Apply. The cropped area appears in a separate window.

Changing the intensity

For information on how to change the intensity of a portion of the image, refer to "To change the number of intensities displayed in the LUT" on page 29 and "To invert intensities" on page 31. In addition, refer to "Generating masks" on page 196.

Displaying images using the triplanar-dual view

The *triplanar-dual view* displays two image datasets (referred to as Image A and Image B) and a blended version of both Image A and Image B from three orthogonal planes.

To view image datasets using this view

1. Open two 3D or 4D images, which become known as Image A and Image B.
2. Load Image B into Image A.
3. Select Image > Views > Triplanar-Dual. The triplanar-dual view appears in an image window.

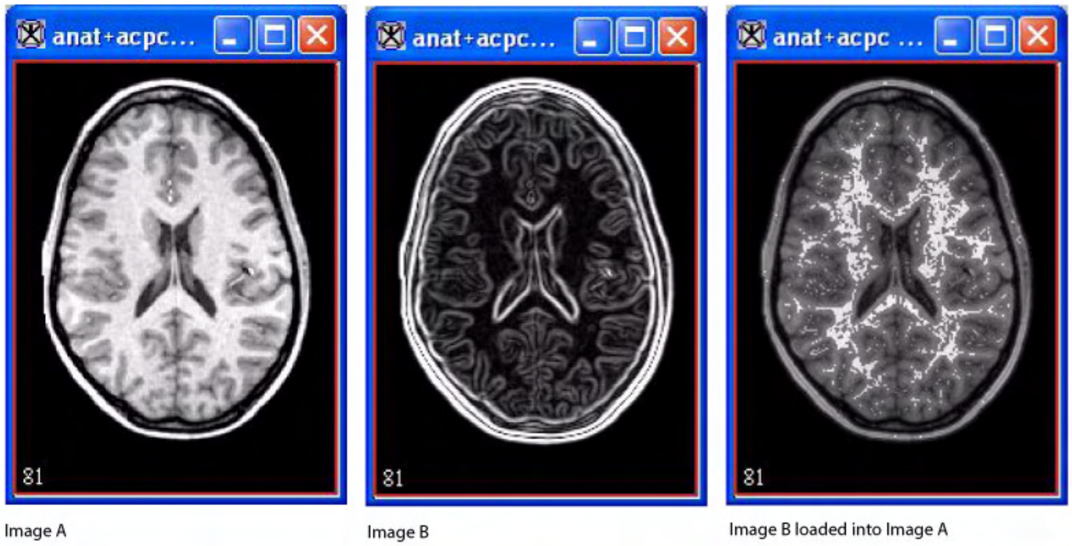


Figure 156. Image A, Image B, and Image B loaded into Image A

The image window for the triplanar-dual view is quite similar to the image window for the triplanar view. The File and Options menus contain the same commands except for one command—the Options menu for the triplanar view contains the command Link to Another TriImage. Since the triplanar-dual view shows two datasets in the image window and allows you to manipulate them together, linking to another image is not necessary. Most of the icons and buttons on the toolbars for both views are the same with one primary difference: The triplanar-dual view allows you to create VOIs on the images; the triplanar view does not.

Like the triplanar view, you can use the icons and buttons on the toolbar to change magnification, draw, change colors, change opacity, and apply a new intensity to selected voxels in the images.

Showing and hiding axes, cross hairs, and Talairach grid

By default, the triplanar-dual view automatically shows the axes and cross hairs on Image A, Image B, and the blended dataset.

To show or hide axes on images

To hide the axes, select Options > Show Axes (Figure 54). The check mark disappears from the check box in front of the command on the menu, and the axes disappear from all of the images.

Figure 157. Show Axes command on the Options menu in the Triplanar-Dual window

To display the axes on the images again, simply select Options > Show Axes. The check mark reappears in the check box in front of the command.

To show or hide cross hairs

Showing or hiding cross hairs works just as showing or hiding axes. To hide the cross hairs, select Options > Show Cross Hairs. To show them, select Options > Show Cross Hairs again.

To show or hide the Talairach grid

By default, the Talairach grid does not automatically display on the images unless you specifically select it to do so. To show the grid, select Options > Show Talairach Grid. The grid appears on all of the images shown in the triplanar-dual view. When you want to hide the grid, select Options > Show Talairach Grid again.

Creating VOIs on images

You can create point or line VOIs on the images.

To create point VOIs

1 Click .

2 Click in the image where you want to create a point.

3 Repeat steps 1 and 2 for all other desired points.

To draw line VOIs

1 Click .

2 Click in the image where the line should begin and drag the line to where it should end.

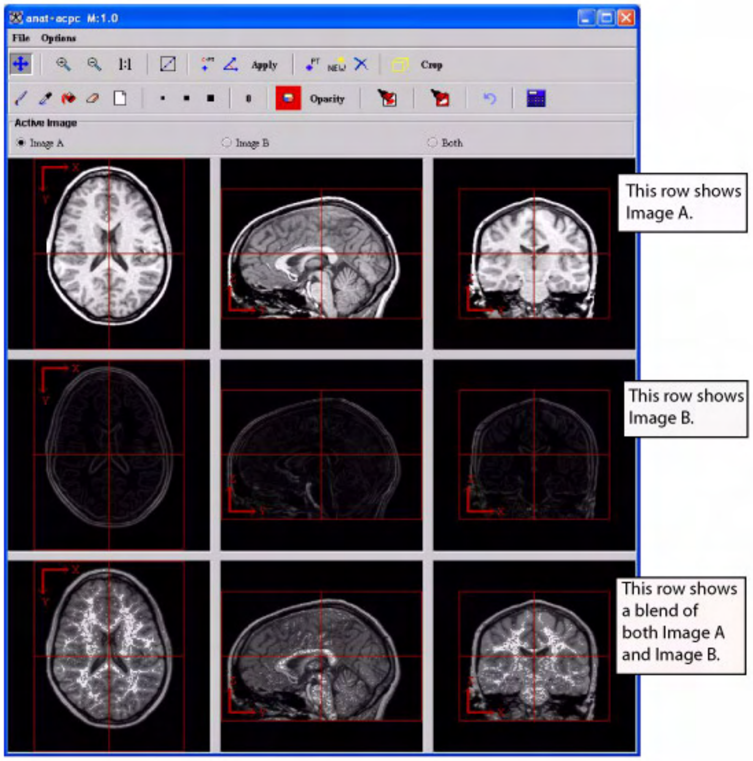
The numbers beneath the line indicate the length of the line in millimeters.

To change the line VOI, simply click on the VOI and then drag it to another position in the image.

Note: You can only create one line VOI per orthogonal plane, although you can create a line VOI in each one of the three orthogonal planes.

To delete line VOIs

You cannot delete a line VOI. However, you can remove the display of the line VOI on the image. To do so, click . The line VOI disappears.



File	**Close Frame**—Closes this window.
Options	**Show Axes**—Displays the *XY* (top left), *ZY* (top right), and *ZX* (bottom left) axes on the images.
	Show Cross Hairs—Displays the cross-hair lines on the images. When you move these lines on one image, it also moves to the corresponding points all of the other images.
	Show Talairach Grid—Displays a Talairach (stereo tactic) grid on the images.
 **Traverse image**	Aligns two images.
 **Magnify image 2.0x**	Enlarges the images to twice their current size. It might be necessary to manually resize the window and use scroll bars to view all of the images.

Figure 158. Triplanar-Dual window

Magnify image 0.5x	Reduces the images to half of their current size.
Magnify image 1.0x	Restores the images to their original size.
Draw line VOI	Draw a line VOI.
Identify center of volume	Reassigns a new center to each dataset. The location that you select appears in the center of each image.
Alignment tool	Allows you to apply a transformation matrix to an image dataset.
Applies rotations and translations	Applies the rotations and translations made to the images.
Add point	Draws a point on an image. This point is only on the image you select.
Initiate new VOI	Indicates that the next VOI is a new VOI rather than a change to the currently existing VOI.
Delete point VOI	Removes the point VOI from the image.
Identify crop volume	Displays the bounding box, which is outlined in red, on each image.
Crops image delineated by the bounding cube	Displays in a separate image window only the area enclosed by the bounding box. When you click this icon, the Crop Image dialog box opens. 
Draw using a brush	Allows you to draw with a brush on the image.

Figure 158. Triplanar-Dual window (continued)

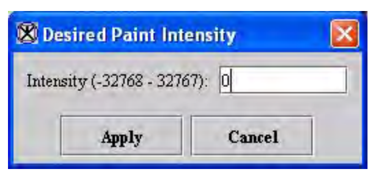
Picks up a color from the image	Changes the color of the paint used to the color selected on the image.
Fills an area with desired color	Allows you to fill an area with color. When you select this icon, the Paint Grow dialog box opens.
Erases a portion of the image	Removes the paint from the specific spot that you select.
Erase all paint	Removes all paint from all of the images.
Draw using small size	Indicates the size of the paint brush tip. Click this icon to paint one voxel each time the mouse button is clicked.
Draw using medium size	Indicates the size of the paint brush tip. Click this icon to paint 16 voxels (4 x 4 square) each time the mouse button is clicked.
Draw using large size	Indicates the size of the paint brush tip. Click this icon to pain 100 voxels (10 x 10 square) each time the mouse button is clicked.
Change intensity level of paint	Allows you to change the intensity value of the paint. When you click this icon, the Desired Paint Intensity dialog box opens.  You can specify an intensity between -32,768 and 32,767. After you type an intensity level, the icon displays the level you entered.
Change paint color	Changes the color of the paint used. This icon is often used in conjunction with the Draw icons. When you click this icon, the Pick Paint Color dialog box opens. After you change the color, this icon displays the color.

Figure 158. Triplanar-Dual window (continued)

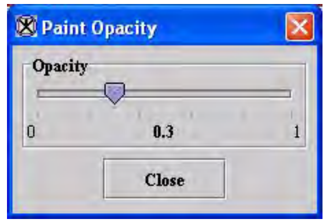
Opacity **Opacity**	Allows you to change the opacity of the paint. When you click this icon, the Paint Opacity dialog box opens. 
Commit **Changes image where painted**	Permanently applies the intensity level to the images.
Undo last region paint	Removes paint from the last area to which it was applied.
Calculate volume of paint	Determines the volume, or number of pixels, in the painted area of the image.
Image A **Image A**	Performs the action only on Image A.
Image B **Image B**	Performs the action only on Image B.
Both **Both**	Performs the action on both Image A and Image B.

Figure 158. Triplanar-Dual window (continued)

To delete point VOIs

1. Click the first point on the image aligning the middle of the cursor with the center of the point.
2. Click . The point disappears.
3. Repeat the first two steps for every point you want to delete.

Aligning the datasets

The Alignment Tool icon, , allows you to create a new transformation matrix for the dataset.

1 Click .

A closed angle appears on each of the images in the triplanar-dual view. On Image A, the angle is red; on Image B, it's yellow; on the blended image, the angle is green.

2 Move the cursor to the arrow on the thin line to the right, and align the cursor with the end of the arrow. At this time, the cursor should be a red cross.

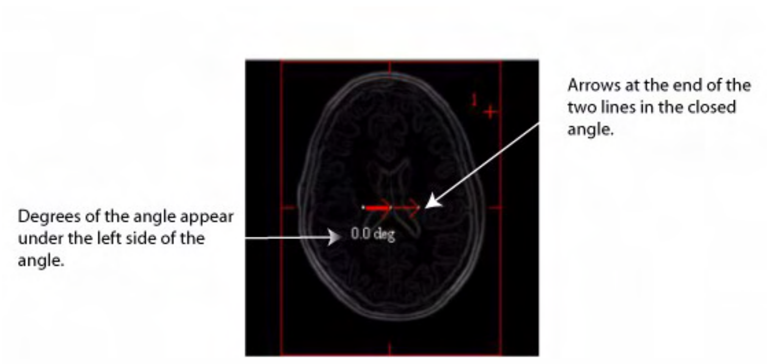


Figure 159. Closed angle that appears on each of the images

3 Change the size of the angle by dragging the arrow on the thin line up or down depending on the type of angle you want to create. The exact number of degrees appear on the left below the angle.

4 Release the mouse button when you reach the desired number of degrees.

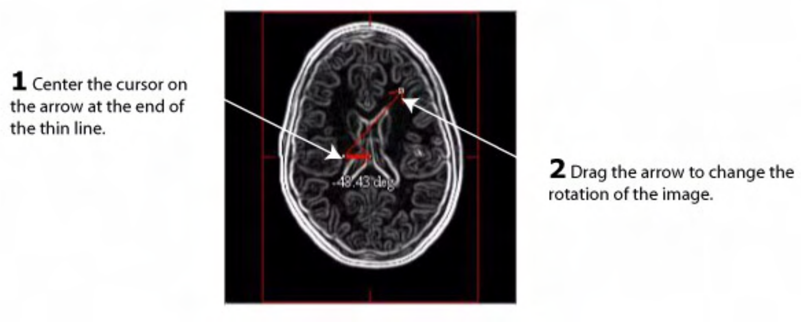


Figure 160. Changing the angle by dragging the arrow on the thin line

5 Click on the image or on any other area of the triplanar-dual view. The arrows disappear from the angle.

6 Click on the angle again to be able to change the angle again. The arrows reappear and you can repeat the previous steps.

Finding the center of image datasets

To find the center of the dataset, click . The center of each image dataset, shown as , appears in each image in the triplanar-dual view.

Erasing part or all of drawings

If you used the paint brush to draw on the image, you may at times find a need to erase a part of the drawing or the entire drawing.

Caution: Make sure that you want to erase a part of the drawing or the entire drawing. There is no undo.

To erase a part of the drawing

1 Select the image that contains the drawing you want to erase. In other words, select Image A, Image B, or Both.

2 Click .

3 Click on the part of the drawing that you want to erase.

4 Repeat step 3 for as long as needed.

To erase all of the drawing

1 Select the image that contains the drawing you want to erase. Specifically, select Image A, Image B, or Both.

2 Click . MIPAV removes all of the paint from the image.

Cropping images

MIPAV allows you to extract the crop volume as a separate image, which is helpful if you need to work with or use only that portion of the image. The first step in cropping images is to identify what portion of the image is in the crop volume. Then you need to use the bounding cube for cropping the image.

To select the crop volume

1 Click . A red box appears on each image in the triplanar-dual view.

2 Align the cursor with one of the corner points of the box, and enlarge or reduce the size of the box.

3 Repeat the previous step on each corner point as required to enclose the portion of the image you want to be included in the cropped image.

The numbers on the sides of the box indicate the length and width of the box in millimeters.

Note: The size of the box changes proportionally on all of the images shown in the triplanar-dual view.

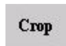
4 Release the mouse button when you are finished making changes to the box.

To crop images

1 Select one of the following:

- Image A if you want only the cropped area in Image A

- Image B if you want only the cropped area in Image B
- Both if you want both the cropped area in Image A and the cropped area in Image B

2 Click . The Crop Image dialog box appears.

Apply	Applies the transformation to the image dataset according to the specifications in this dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes this dialog box.
Help	Displays online help for this dialog box.

Figure 161. Crop Image dialog box

3 Click Apply.

The cropped area in the image you selected appears in a separate image window. So, if you selected Image A only or Image B only, one image appears. If you selected Both, two image windows appear: one for Image A and one for Image B.

If you selected . . .	Then . . .
Image A	The cropped area appears in a separate window.
Image B	The cropped area appears in a separate window.
Both	Two small windows—one displaying the cropped area of Image A and the other displaying the cropped area of Image B—appear.

After you generate the cropped images, you can run algorithms on the images, apply a quick LUT on the images, or use any other image processing that's available in the MIPAV window on them.

Magnifying or minifying images

Like the image toolbar on the MIPAV window, the Triplanar-Dual window includes the following magnification icons:

- **, the Magnify image 2.0x icon**—To magnify images to twice their current size
- **, the Magnify image 0.5x icon**—To reduce images to half their current size
- **, the Magnify image 1.0x** —To restore their original size

For information on how to use these icons, refer to "magnify or reduce a region of the image. You can magnify images to 32 times their original size, reduce them to one-fourth their original size, or restore them to their original size." on page 13.

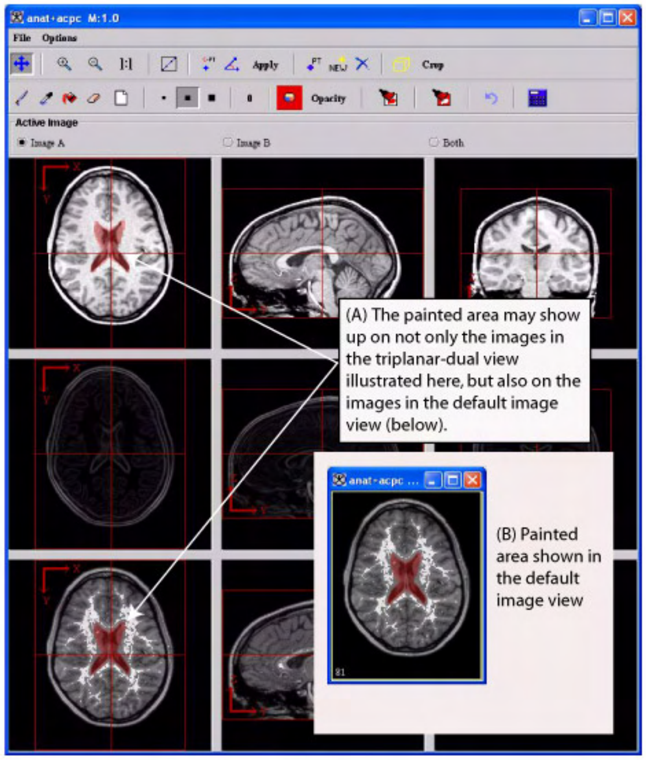
Drawing on images

You draw on images in the triplanar-dual view in the same way as you do in the MIPAV window. However, any of the changes you make on the triplanar-dual view are also made on the original images.

For example, if you highlight a particular area in the image in the triplanar-dual view, the same change is made on all of the images in the triplanar-dual view, but also on the images in the default view.

To change the paint color

Click , the Change Paint Color icon. The Pick Paint Color dialog box (Figure 43 on page 54) opens. For instructions on how to use this dialog box to change the paint color, refer to "To change background and border colors" on page 54.



Painting an area on images in the triplanar-dual view affects images in the default image view

Figure 162. Painting an area on images in the triplanar-dual view

To change paint opacity

1 Click 

. The Paint Opacity dialog box opens.

2 Move the Opacity slider to the level of opaqueness you want. For example, to make the paint color more opaque, or solid, move the slider to the right. To make the paint color more transparent, move the slider to the left.

3 Click Close when you have finished. The Paint Opacity dialog box (Figure 60) closes.

Opacity	Indicates the opacity of the paint. The further right you move the slider, the more opaque the paint color. The further left you move the slider, the more transparent the paint color.	
Close	Closes this dialog box.	

Figure 163. Paint Opacity dialog box

To calculate the volume of paint

The software can calculate the volume of a painted area in the number of pixels. To do so, do the following:

1. Paint the area of the image in which you are interested.
2. Click  to discover the volume of paint. The software lists the number of pixels in each image and the volume in millimeters on the Data page of the Output window.
3. Amend and save, as an option, the information on the Data page.

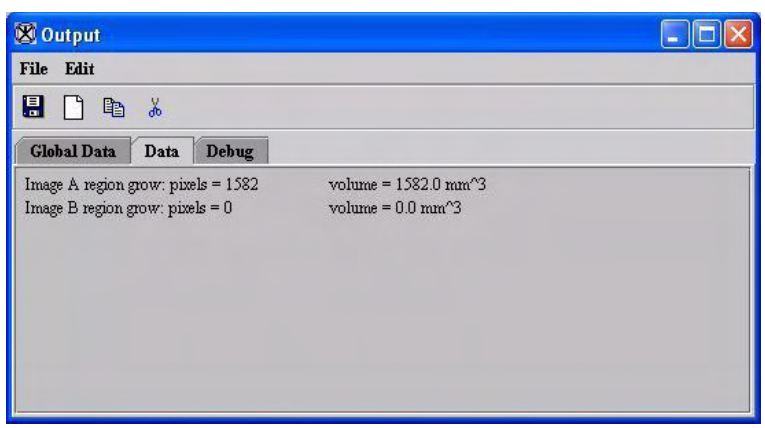


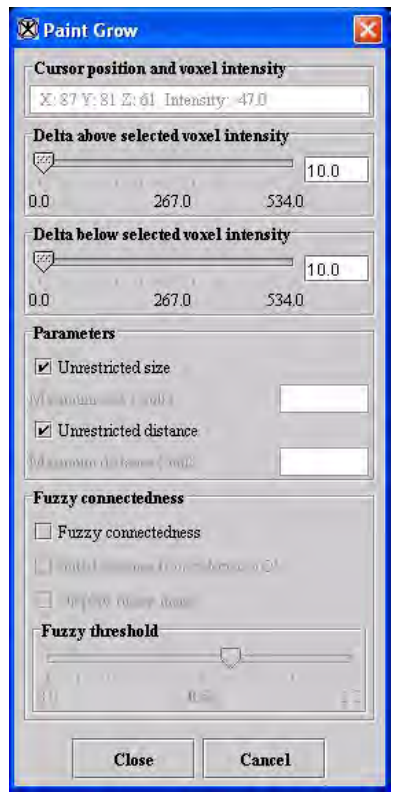
Figure 164. Output window after calculating the volume of paint

To fill an area with color

1 Click . The Paint Grow dialog box opens.

2 Complete the dialog box.

3 Click Close.



Cursor position and voxel intensity	Indicates the location of the cursor and the intensity value at that point
Delta above selected voxel intensity	Indicates the range of paint beyond
Delta below selected voxel intensity	Indicates the range of paint
Unrestricted size	Limits the total volume in 3D or 4D images or area in 2D images
Maximum size (null)	In development.
Unrestricted distance	Limits the distance from the seed point
Maximum size (null)	In development.
Fuzzy connectedness	Applies the fuzzy algorithm
Initial variance from selected VOI	In development.
Display fuzzy image	In development.
Fuzzy threshold	In development.
Close	Applies the parameters you specified
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box
Help	Displays online help for this dialog box

Figure 165. Paint Grow dialog box

To erase an area of paint from images

1 Click .

2 Click the area of the paint that you want to remove.

3 Repeat the previous step for each area of paint to be removed.

4 Click when done to return the cursor to the default state.

To erase all painting from images

Click . MIPAV removes all of the paint from all of the images.

Displaying images using the surface plotter view

The *surface plotter* view visualizes the intensity levels for a chosen image using the height of the image surface. The surface plotter takes a 2D image and renders it into a 3D object that is similar to a raised-relief map. Higher intensities are peaks and lower intensities are valleys. The colors that the surface plotter used are the same as those used in the LUT, and when the LUT for the image is updated, so is the LUT for the plotted surface. The image itself is also a part of the scene, and it can be slid along, so that the peaks of the intensity levels match the image.

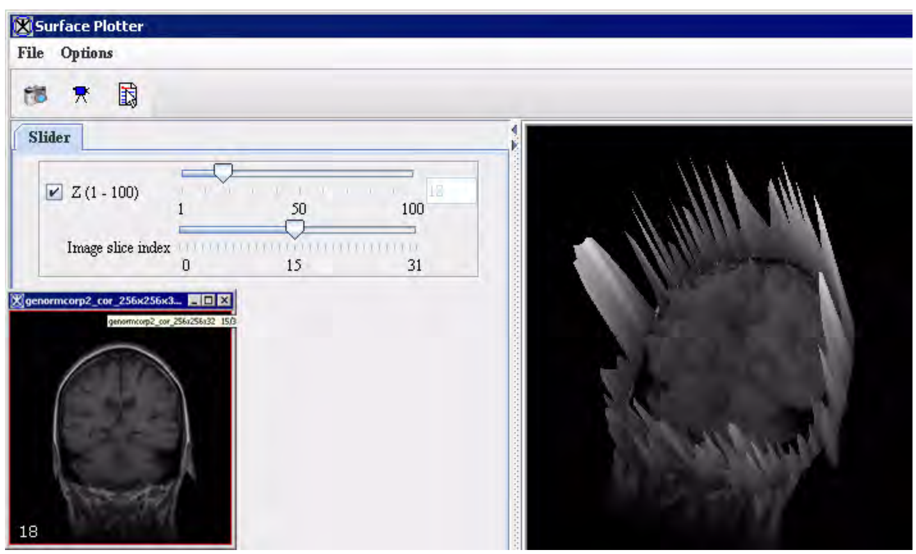


Figure 166. An image open in MIPAV and in the MIPAV Surface Plotter default view. At each point of the surface, the height indicates the intensity value for that point.

- The Z slider can be used as a cut off for pixel intensities. The read-only box on the right indicates the actual height of the cutoff.
- The image slice slider allows to select the image slice.

Surface Plotter window

Menu	
File	**Exit**—closes the Surface Plotter window.
Options	**View Mode**— opens the View tab, see "View tab" **Mouse Recorder**— opens the Mouse tab, see "Mouse tab" **Line Mode**—displays the surface plotter image as lines. **Resample**— improves the speed of animating of the image by resampling and forming the image with fewer triangles. **Display Options**—allows you to add a bounding box, change colors of the bounding box and background, show an orientation cube, and choose the projection type: perspective or orthographic. See also "Box tab" below.
Help	Displays help for this dialog box.
Toolbar	– opens the Camera tab, see "Camera tab" – opens the Mouse tab, see "Mouse tab" – opens the Box tab, see "Box tab"
Image area	Displays the plotted image.
Slider tab	

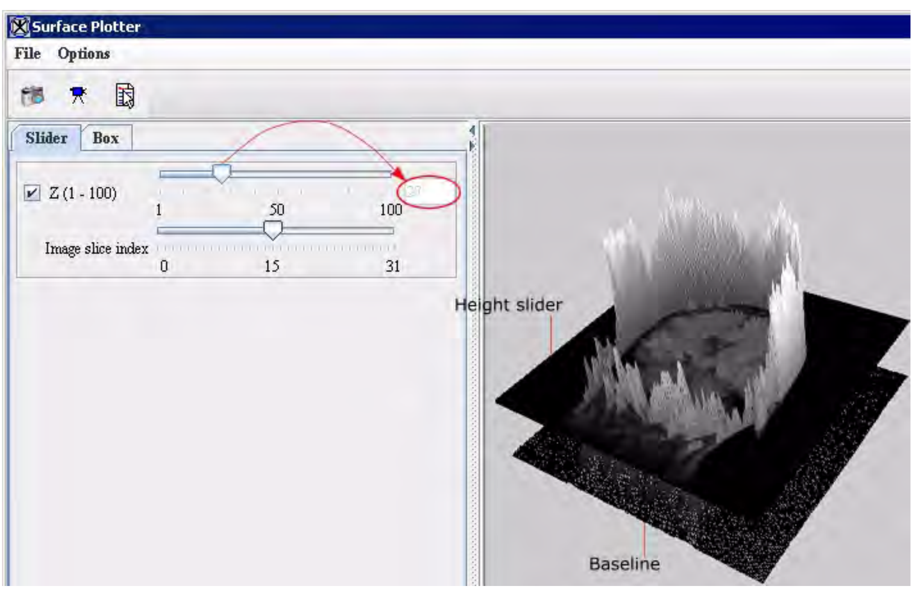
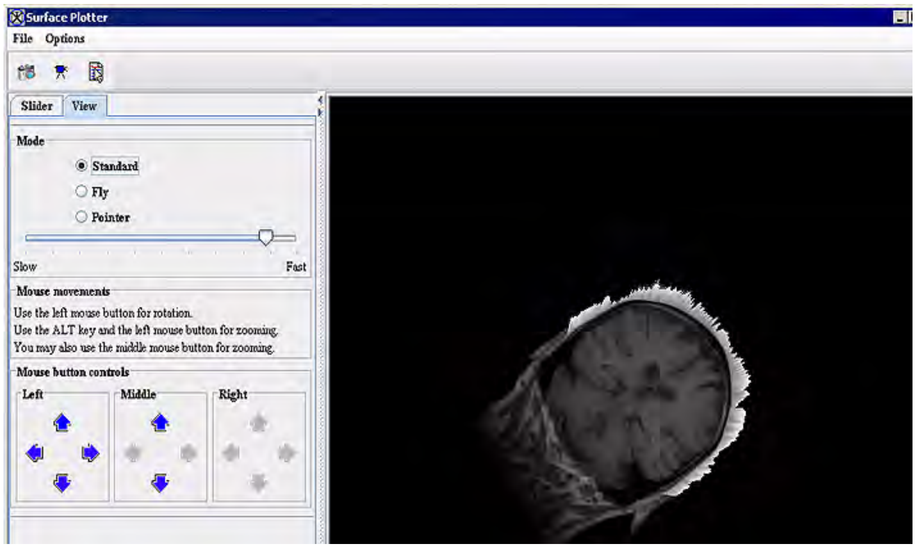


Figure 167. Surface Plotter window

Z(1-100)	Turns on the ability to visualize the pixel intensities. When this box is selected, the height slider is available. It shows the height of the intensities in the image. Move the slider to the left to show the wider range of intensities: from the lower level to the higher level. Move the slider to the right to set up the cutoff and show only the higher level intensities. The read-only box on the right indicates the actual height of the cutoff.
Image Slice slider	Allows to select a particular slice and visualize the pixel intensities for that slice.
View tab	



Mode	Different view modes offer different opportunities for using the mouse controls. **In Standard mode**, you can use left mouse button controls for rotation and the ALT+ left mouse button controls combination of keys for zooming, you can also use middle mouse button controls for zooming. **In Fly mode**, left, middle and right mouse button controls are available for zooming, rotation, and also pitch and roll rotation of the image. **In Pointer mode**, you can use the mouse pointer for rotation, translation and zooming. See also "Viewing an image dataset in animate view" on page 298
Slow-fast slider	use this slider to adjust the speed of the mouse actions.
Mouse tab	

See also: "Recording and playing mouse actions" on page 301, "Saving as AVI" on page 305.

Figure 167. Surface Plotter window (continued)

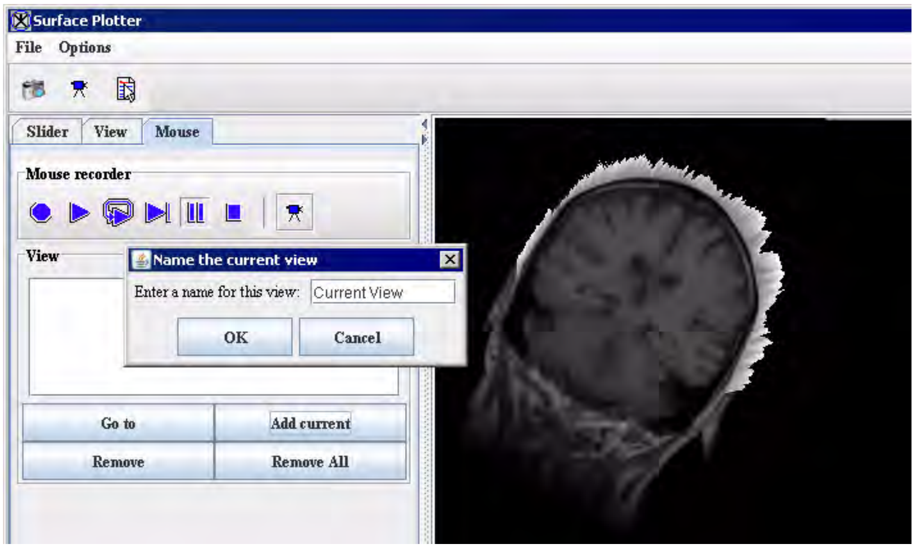
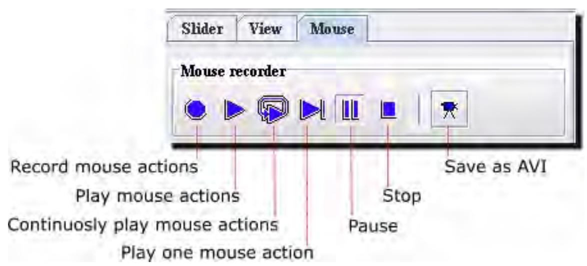
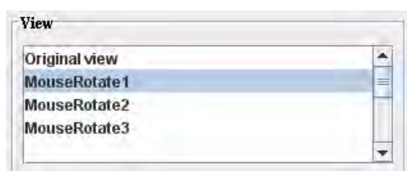
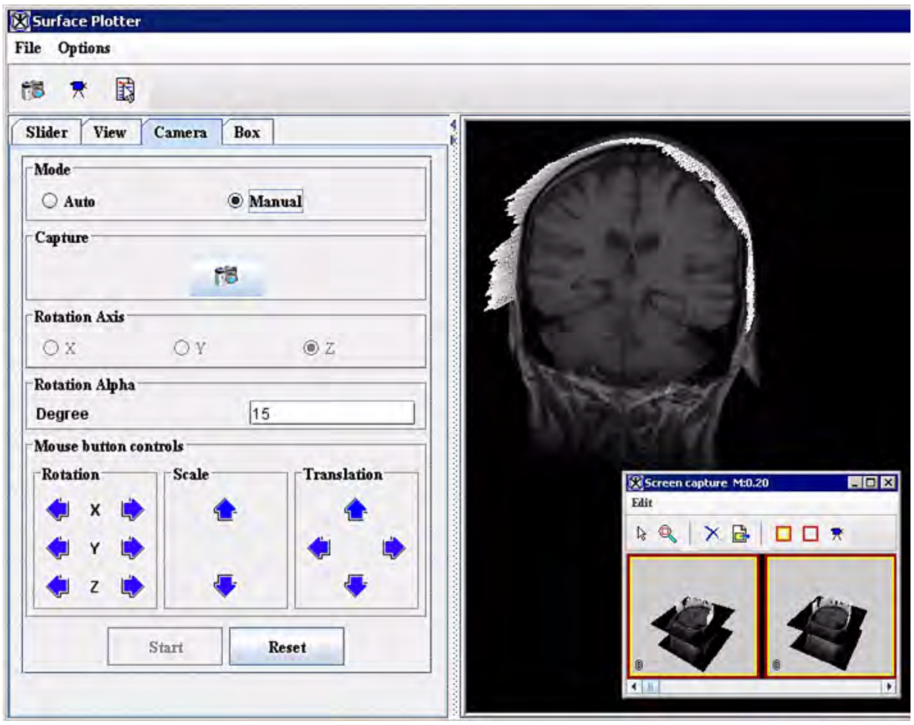
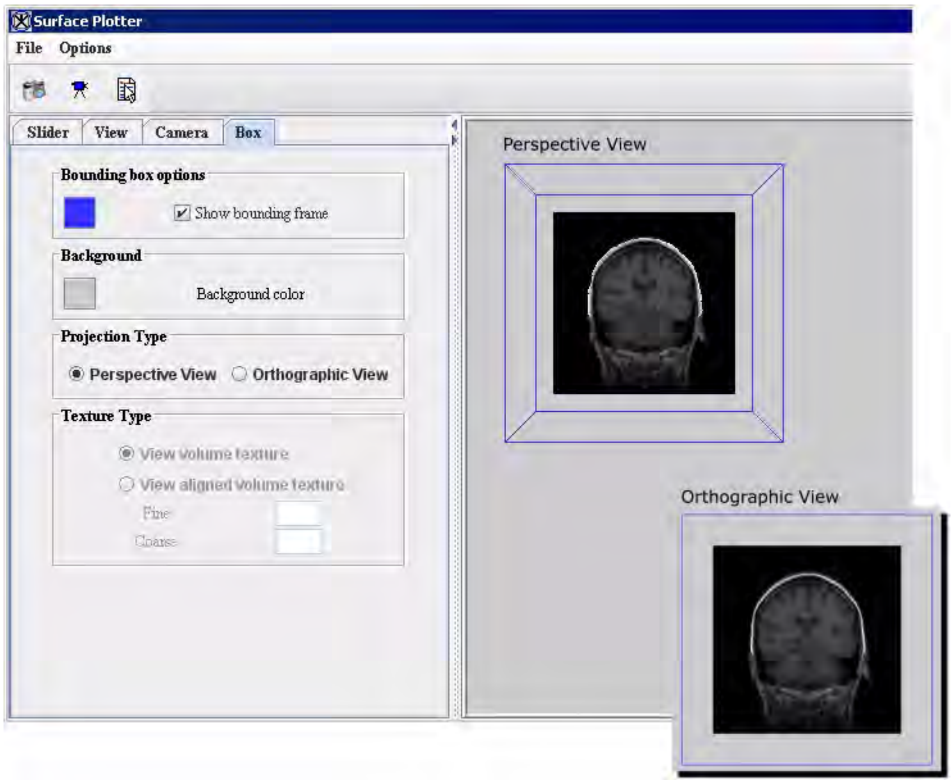
	
Mouse recorder controls	
View	This box displays a list of recorded mouse actions. You can select any of mouse actions from the list as a starting point and start playing the movie from that point. 
Go to	– directs the viewer to the selected view.
Add current	– adds a current view to the sequence of views shown in the View box.
Remove	– removes selected view.
Remove All	– removes all views.
Camera tab	
See also: "Capturing snapshots" on page 303, "Saving as AVI" on page 305.	

Figure 167. Surface Plotter window (continued)



Mode	Auto mode allows the automatic rotation and capturing of the image. Manual mode allows the manual rotation and capturing of the image.
Capture	Press the Camera button to capture the current image view. This option is only available if Manual mode is activated.
Rotation Axis	Use the X, Y, or Z radio buttons to select the rotation axis. This option is only available if Auto mode is activated.
Rotation Alpha Degree	Enter a value for the rotation angle. By default, it is set to 15 degrees. This option is available in both Auto and Manual modes.
Mouse button controls	Use the mouse button controls to rotate, scale and translate the surface plotter view. The mouse button controls are only available in Manual mode.
Start	– starts capturing. Only available in Auto mode.
Reset	– resets the Surface Plotter view.
Box tab	

Figure 167. Surface Plotter window (continued)



Bounding box and Background	Use these options to change the background and bounding box colors and also to display the bounding box frame.
Projection type	Use the radio buttons to switch between Perspective and Orthographic view.
Texture type	TBD.

Figure 167. Surface Plotter window (continued)

Viewing an image dataset in animate view

In animate view, like cine view, the software automatically advances through the image dataset. However, animate view allows you to adjust the speed at which images are advanced using the Slow-Fast slider.

To view an image dataset in Standard mode

1. Select Image > Views > Surface Plotter in the MIPAV window. The Surface Plotter window opens.
2. In the Surface Plotter window, call Options > View Mode to open the View tab.
3. Select Standard mode, and then adjust the mouse action speed slider.
4. Use either the mouse controls, or hold down the left mouse button and drag the mouse to the left, right, top, and bottom. This allows you to move the image in all directions.
5. If you release the left mouse button, and then hold it down again while you move the mouse, you can turn the image upside down.
6. Hold down the middle mouse button (or use the mouse controls) and push the mouse forward. This makes the image smaller as if it has been seen from a distance.
7. Hold down the middle button and push the mouse toward you. This enlarges the image so that you can examine it closely.

You can also use the controls for the middle mouse button.

---

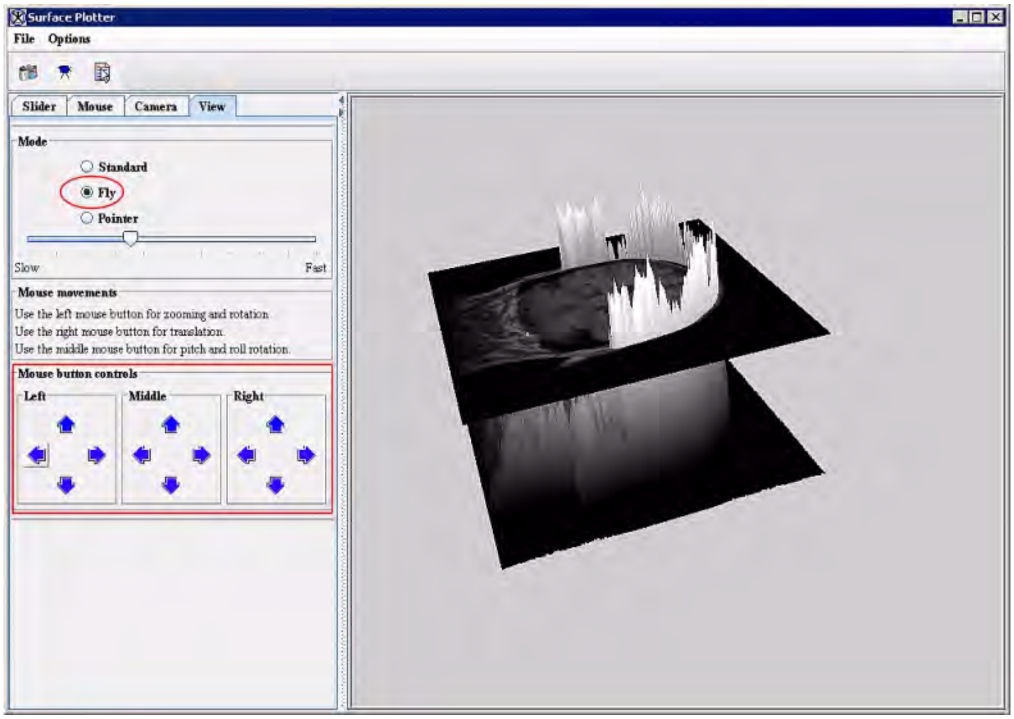
8. Return to the Slider tab and slide the height slider to determine the highest intensity in the image.

Example: One way of determining the highest intensity in the image is to use the mouse to drag the image to its side, enlarge it using the middle mouse button and pushing the mouse toward you, and then sliding the height marker to the top of the tallest intensity peak. The level of the intensity appears in the box to the right of the height slider.

To view an image dataset in Fly mode

1. Select Image > Views > Surface Plotter in the MIPAV window. The Surface Plotter window opens.
2. In the Surface Plotter window, call Options > View Mode to open the View tab.
3. Select Fly mode, and then adjust the mouse action speed slider. See Figure 168.
4. Use the left mouse button controls (upper and lower) to drag the image up and down.
5. Use the left mouse button controls (right and left) to rotate the image along the X axis.
6. Use the middle mouse button controls (upper and lower) to rotate the image along the Y axis.
7. Use the middle mouse button controls (right and left) to rotate the image along the Z axis.
8. Use the right mouse button controls (right and left) to zoom in or zoom out the image.
9. Use the right mouse button controls (upper and lower) to translate the image.

You can also use the mouse buttons instead of the mouse button controls.



Manipulating of the plotted image in the Surface Plotter window in Fly mode

Figure 168. Surface Plotter window – Fly mode

Recording and playing mouse actions

Surface plotter allows you to record mouse actions (e.g., rotations, translations, and zooming) you performed on the image and create a movie from them. This is available via the Mouse tab. When you record mouse actions, the software makes a copy of the image dataset and loads it in the animate image window. Before starting recording, you can adjust the speed at which images are advanced (the View tab). After recording is completed, you can specify that images be displayed in forward or reverse order, and you can also pause or loop the sequence.

To record the mouse actions:

1. In the Surface Plotter window, go to the Slider tab and adjust the intensity height slider.
2. Then call Options > View mode and choose Pointer mode. Use the Slow-Fast slider to adjust the speed of the mouse actions.

3 Open the Mouse tab (use Options > Mouse Recorder).

4 In the Mouse tab, press the Start Mouse Recording button to start recording the mouse actions. The list of actions appears in the View box.

5 To pause recording, press Pause.

6 To stop recording, press Stop.

7 To save the recorded movie as an AVI file, press Save as AVI. See Figure 169. See also "Saving as AVI" on page 305.

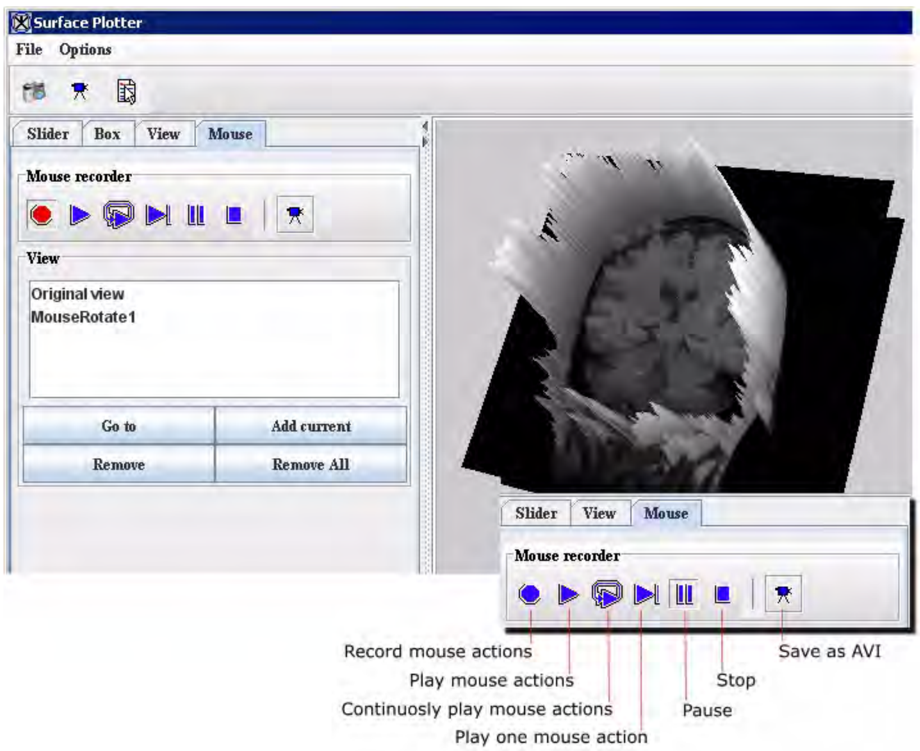


Figure 169. Recording and playing the mouse actions

To play recorded mouse actions:

1 In the Mouse tab, press the Continuously Play Mouse Actions button. This starts playing the recorded actions.

2 To stop playing, press Stop.

3 To start playing the movie from a particular mouse action, select that action in the View box, and then press the Continuously Play Mouse Actions button. The recorded mouse actions will play starting from that action. See Figure 169.

To edit recorded mouse actions:

1 In the Mouse tab, select the mouse action(s), and then press Remove. this will remove the selected action(s) from the list of recorded actions.

2 To remove all recorded actions and clean up the View box, press Remove All. See Figure 169.

See also: "Saving as AVI" on page 305.

Capturing snapshots

Surface Plotter dialog also offers the additional Auto Snapshot tool for automatic and manual capturing snapshots, which allows the user to reproduce the movements of the image. These snapshots can be worked on afterwards as slides or the video can be compiled in AVI format. See "Saving as AVI" on page 305.

To run the Auto Snapshot tool, click the Auto Snapshot Screen button on the Surface Plotter toolbar. This opens the Camera tab where you can set up the tool options and run the tool. For more information, refer to "Auto Snapshot tool" .

AUTO SNAPSHOT TOOL

The Auto Snapshot tool can be run in Auto and Manual modes.

- In Auto mode, the surface plotter image is rotated along the chosen axis (X, Y, or Z) using the user defined rotation angle step, and the snapshot is taken at each rotation step. The snapshots are displayed in the Screen Capture window. See also "To take snapshots in Auto mode"
- In Manual mode, the surface plotter image is rotated manually (using the mouse button controls) using the user defined rotation angle step, and the snapshots are also taken manually at each rotation step which is interesting for the user. The snapshots are then displayed in the

Screen Capture window. See also "To take snapshots in Manual mode" on page 304.

To take snapshots in Auto mode

1. In the Surface Plotter window, go to the Slider tab and adjust the intensity height slider.
2. Open the Camera tab (use the Auto Snapshot Screen button on the Surface Plotter toolbar).
3. In the Camera tab, select Auto mode. This is the default mode for capturing snapshots.
4. Then, choose the rotation axis and define the rotation angle. The default value for the rotation angle is set to 15 degrees.
5. Press the Start.

The image starts rotating and the snapshots appear in the Screen Capture window. After completing the whole 360 degrees rotating circle, the image stops rotating. See also Figure 167.

For more information refer to "Displaying images using the animate view" on page 246.

To take snapshots in Manual mode

1. In the Surface Plotter window, go to the Slider tab and adjust the intensity height slider.
2. Open the Camera tab (use the Auto Snapshot Screen button on the Surface Plotter toolbar).
3. In the Camera tab, select Manual mode.
4. The mouse button controls and Capture Screen icon become active.
5. Then, define the rotation angle. By default, the value for the rotation angle is set to 15 degrees.
6. Use the mouse button controls to rotate the image. Note that you can rotate the image in any direction.
7. Press the Capture Screen icon each time you want to take a snapshot.

The snapshots appear one by one in the Screen Capture window.

You can stop capturing the snapshots at any time by pressing the Reset button. See also Figure 167.

SCREEN CAPTURE WINDOW

After you've completed capturing snapshots, they appear in the Screen Capture window. This window provides you with the tools for further editing of the snapshot sequence and saving it as an AVI file. The following options are available:

– Default mode	– Select all
– Magnify region	– Unselect all
–Delete selected slices	– Capture to AVI
Extract selected slices to a new image	

SAVING AS AVI

The Save as AVI or Capture as AVI option saves the sequence of recorded mouse actions or captured snapshots as an AVI file. To call the Save as AVI option, click the corresponding icon on the Mouse tab. To call the Capture as AVI option, click the icon on the Screen Capture window toolbar. The follow the steps provided by the dialogs.

1. Click the Capture to AVI or Save to AVI icon.
2. In the Choose Type of AVI File dialog, specify the AVI type and JPEG quality, press OK.
3. In the Save Range of Slices dialog that appears next, press OK, then enter the file name.
4. The saved AVI file appears in the MIPAV AVI player.

For more information refer to "Displaying images using the animate view" on page 246.

Displaying images using the surface renderer view

In development.

Displaying images using the volume shear view

In development.

Displaying images using the volume renderer view

In development.

9

Segmenting Images Using Contours and Masks

In this chapter

Segmentation is the process of identifying selected regions of images as members of a common group. Physicians routinely segment structures in medical images to facilitate the treatment of patients. The quantification of important attributes, such as the volume of various tissue types, enables researchers to better understand, diagnose, monitor, and treat neurobehavioral disorders.

MIPAV provides several segmentation tools that allow you to delineate a volume of interest (VOI). You can indicate a VOI by drawing a contour or by creating a mask. You can also calculate predefined statistics on the indicated VOI.

For more information, refer to the MIPAV web site: {http://mipav.cit.nih.gov/documentation/HTML%20Algorithms/MIPAV_Segmentation.html}.

Note: Once a VOI is segmented, you can also apply an algorithm to the volume bound by the contours. MIPAV provides a number of algorithms, such as Gaussian blur, Laplacian, threshold, and watershed. These algorithms and others are addressed in detail in volume 2 of the *User's Guide*.

Understanding contours and masks

Before you indicate a volume of interest, it is helpful to determine whether to delineate the VOI by drawing a contour around a structure or to generate a mask.

A *contour* is an array of points. If the contour is a closed polygon, the area inside defines the VOI. Contours are scalar and can be grouped to form a single multicontoured VOI. Contours are particularly useful if you want to save a VOI that covers a large area because the resulting file is generally smaller than a comparable mask file.

A *mask* is a filter that selectively includes or excludes certain values. You can create a mask that includes the VOI areas and excludes all other areas. Masks are particularly helpful if you use other image analysis programs that require you to supply a mask. Masks can also be used to indicate a complex area of interest. MIPAV allows you to generate binary and short image type masks. Also, you can manually create masks using paint tools.

Using contours to segment a VOI

A VOI can consist of a single contour or several contoured structures that are assigned to a group. You can automatically, manually, or interactively draw contours on any image that is shown in the default view. You can modify the shape and size of a contour and trim anchor points.

Once contours are drawn and modified, you can copy them to other slices in the dataset or save them to a separate file for future use. The three overall tasks involved in using contours to segment a VOI are:

- Drawing contours ("Drawing contours" on page 309)

- Modifying contours (“Modifying contours” on page 318)
- Grouping and ordering contours (“Grouping and ordering contours” on page 323)

Drawing contours

You can draw contours using three different methods: manual, interactive, or automatic. With the *manual method*, you use the mouse to draw a contour. This method provides the greatest amount of freedom: you can choose one of MIPAV's predefined shapes or create a freehand shape. However, the manual method can be time consuming, particularly if the structure to be contoured is intricate. With the *interactive method*, you must also manually contour the structure. However, in this case the manually drawn contour need not be precise because you then apply an algorithm that generates a new contour based on both the manually drawn contour and the structure. With the *automatic method*, MIPAV generates contours as you move the mouse over different structures. The automatic method is generally accurate and quick; however, it does not provide the freedom of the manual method.

Recommendation. If you are unsure about which contour method to use, it is recommended that you try the automatic method first.

CONTOURING STRUCTURES AUTOMATICALLY

In this method, you move the mouse pointer over a structure. MIPAV first analyzes the intensity values and then contours what it perceives to the contour boundary.

Complete the following steps:

1. Select VOI > New VOI > Levelset in the MIPAV window.
2. Move the mouse pointer to the image.

   The pointer changes to the cross-hair shape. As you move the pointer, MIPAV determines the level of intensity of the pixel under the cross-hair pointer and uses the results from the Levelset algorithm to determine the probable boundary of the structure. Having determined the probable boundary, MIPAV generates a contour (Figure 170). Continue to move the pointer until the structure you wish to outline is contoured.

3 Click the contour to apply it to the structure. See Figure 170.

Figure 170. Automatically generated contour.

You can now do the following:

- Draw an additional contour
- Modify the contour
- Reassign an existing contour to a different group
- Order the contours
- Delete the contour

CONTOURING STRUCTURES MANUALLY

You can contour structures in images by selecting one of MIPAV's predefined shapes, such as a point, straight line, 2D or 3D rectangle, 2D ellipse, interslice VOI, etc. Alternatively, you can use the polygon and polyline tools to draw freehand shapes. For more information about VOIs available in MIPAV, refer to Figure 171.

 – Point VOI

 – Interslice Polyline

Figure 171. VOIs available in MIPAV

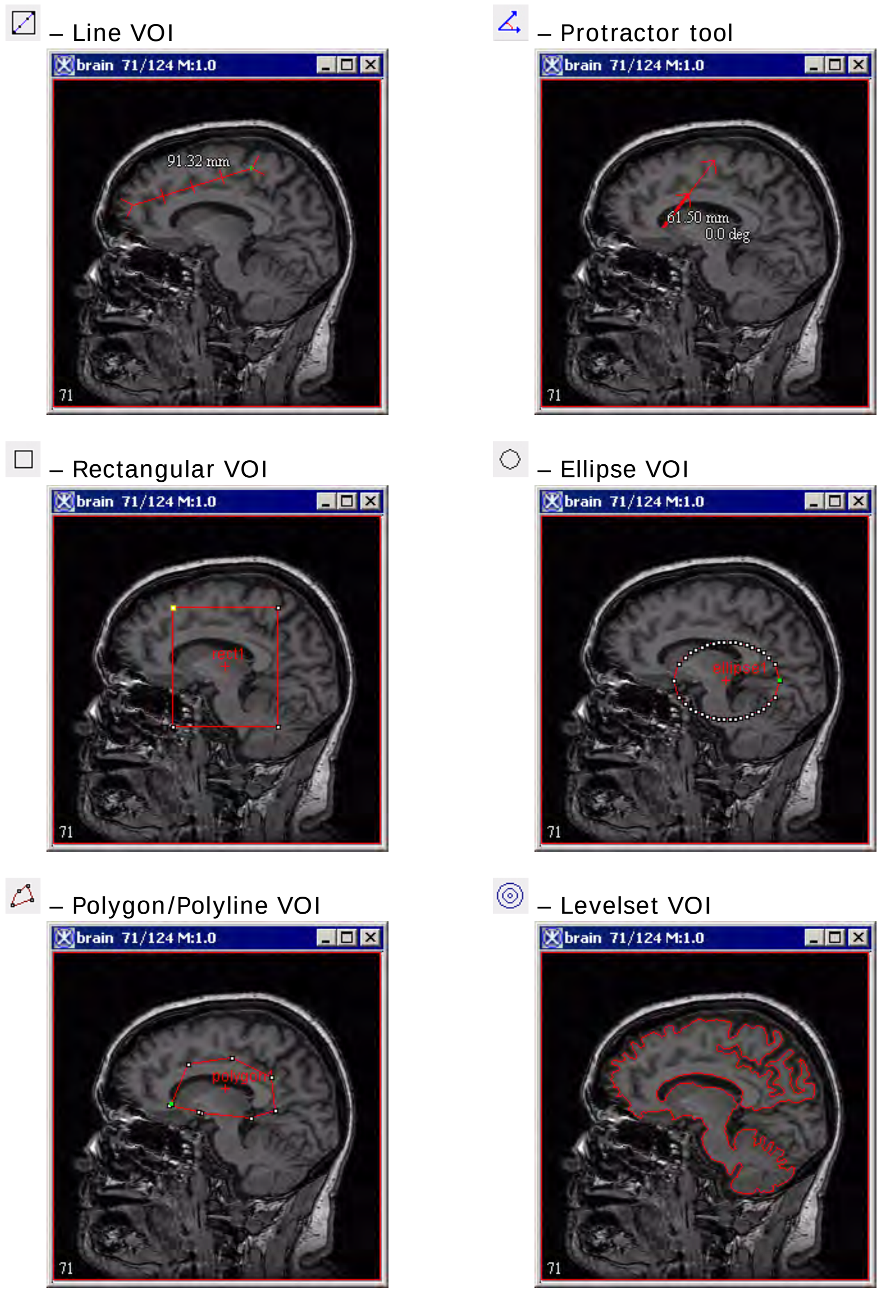
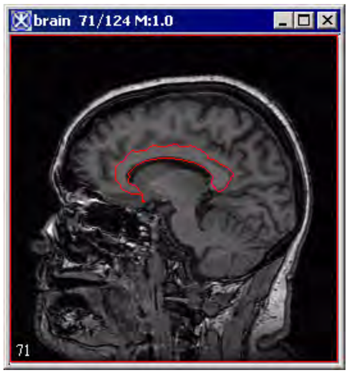


Figure 171. VOIs available in MIPAV (continued)

 – Live wire VOI



 – 3D rectangular VOI

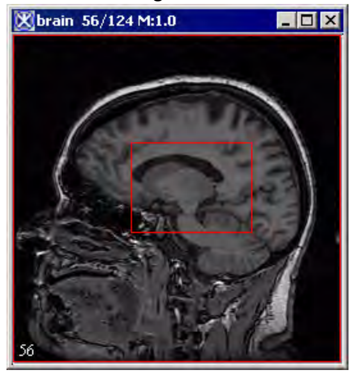


Figure 171. VOIs available in MIPAV (continued)

This section explains how to draw the following:

- Point VOIs
- VOIs with a straight line
- 2D rectangular VOIs
- 3D rectangular VOIs
- Ellipsoidal VOIs
- Polyline (segmented line) and closed polygon VOIs

Tip: Before you begin, decide whether you want to draw more than one of the same contour in succession. If you do, hold down the Shift key as you select the contour button and draw the contours.

To draw point VOIs

 – Draw Point VOI

1 Click the Draw Point VOI button, in the MIPAV window.

2 Move the pointer to the image.

3 Click the area where you want the point to appear. A numbered point appears on the image.

Note: For some algorithms, the sequence in which you draw the points is important (refer to Volume 2 of the *User's Guide*).

To create VOIs with a straight line

– Draw Line VOI

1 Click the Draw line VOI icon.

2 Click the area on the image where you want the line to begin.

3 Drag the pointer to the place where you want the line to end. As you drag the pointer, the line lengthens.

4 Release the mouse button. The line VOI appears on the image in red.

To create 2D rectangular VOIs

– Draw Rectangular VOI

1 Click the Draw Rectangle VOI icon.

2 Click the area on the image where you want a corner of the rectangle to appear.

3 Drag the pointer. As you drag the pointer, the rectangle changes in size.

4 Release the mouse button when the rectangle is the desired size.

Note: When you create a 2D rectangular VOI, the rectangles appear on only one slice in the dataset.

To create 3D rectangular VOIs

– Draw 3D Rectangular VOI

1 Click the 3D rectangular VOI icon.

2 Click the area on the image where you want a corner of the rectangle to appear.

3 Drag the pointer. As you drag the pointer, the rectangle changes in size.

4 Release the mouse button when the rectangle is the desired size.

Note: Unlike 2D rectangular VOIs, 3D rectangular VOIs appear on *all* slices in the dataset.

To create ellipsoidal VOIs

 – Draw Ellipsoid VOI

1. Click the Draw Ellipse VOI icon.
2. Click the area of the image where you want an edge of the ellipse to appear.
3. Drag the pointer. As you drag the pointer, the ellipse changes in size.
4. Release the mouse button when the ellipse is the desired size (Figure 171).

To create polyline (segmented line) VOIs or closed polygon VOIs

 – Draw Polygon/Polyline VOI

1. Click the Draw Polygon/Polyline VOI icon.
2. Click the area of the image where the polyline VOI should begin.
3. Decide whether to draw either a straight or freehand line segment. Do one of the following:
   - *To draw a straight line segment*
   - Make sure that you have released the mouse button. Move the pointer to the place where you want the segment to end. A straight line, which extends from the starting place to your pointer tip, appears. Click the mouse button to anchor the line segment.
   - *To draw a freehand line segment*
   - Drag the mouse button and move the pointer to the place on the image where you want the segment to end. A line, which matches your mouse movements, extends from the starting place to your pointer tip. Click the mouse button to anchor the line segment.
4. Repeat drawing straight line segments or freehand lines until you finish drawing all of the segments needed to contour the structure.
5. Decide whether to create a polyline VOI or a closed polygon VOI:

- *Polyline VOI:* Go to the next step.

- *Closed polygon VOI:* Click the first anchor point to connect the last line segment to the first, thus forming a closed polygon. Then go to the next step.

6 Click the Default icon, to return the pointer to the default mode.

7 Double-click the mouse button to complete the polyline VOI.

Tip: Depending on which is more comfortable, you can switch between drawing straight line segments or drawing freehand lines at any time.

Tip: To draw multiple VOI, hold Shift while pressing the corresponding VOI icon.

CONTOURING STRUCTURES INTERACTIVELY

In the interactive method, you first manually draw a contour that provides a general indication of the location and shape of the structure. You then select an algorithm that analyzes the manually drawn contour and generates a new one that more closely outlines the structure.

To use the interactive method to contour structures, complete the following steps:

1 Draw a contour using one of the manual methods detailed in "Contouring structures manually" on page 310. The contour need not be precise, but it should indicate the general location and shape of the structure you want to delineate.

2 Select the contour. White anchor points appear on the contour outline.

3 Select one of the following in the MIPAV window:

- *VOI > Evolve Boundary 2D > Active Contour*. This algorithm determines the structure's boundary more quickly than the 2D Spline Active Contour algorithm. It is also more sensitive to noise present in the image. MIPAV applies a gradient magnitude filter to determine the structure's boundary. When complete, the MIPAV-generated contour appears around the structure.
- *VOI > Evolve Boundary 2D > Active GVF*. The Evolve Boundary

dialog box uses a special type of active contours, or snakes to find the object boundaries. GVF or the *gradient vector flow snake*, which is used to calculate VOI boundaries, begins with the calculation of a GVF *field of forces* by applying generalized diffusion equations to both components of the gradient of an image edge map. Here, GVF fields, are dense vector fields derived from images by minimizing a certain energy functional in a variational framework. The minimization is achieved by solving a pair of decoupled linear partial differential equations that diffuses the gradient vectors of a gray-level or binary edge map computed from the image. See also Figure 172.

- *VOI > Evolve Boundary 2D > Spline active contour*. This algorithm determines the structure's boundary less quickly than the Active Contour algorithm, however it is less sensitive to noise. This algorithm fits a spline to the data. When complete, the MIPAV-generated contour appears around the structure.
- *VOI > Evolve Boundary 2D > Spline GVF*. This is the combination of the GVF force with the B-spline snake. B-spline snakes have several characteristics which make them well suitable for describing VOI boundary as well as snake evolution: the B-spline implicitly incorporates contour smoothness and avoids the *ad hoc* tension and rigidity parameters of the traditional GVF snake and also fewer sample points are required to implement contour evolution for the B-spline.

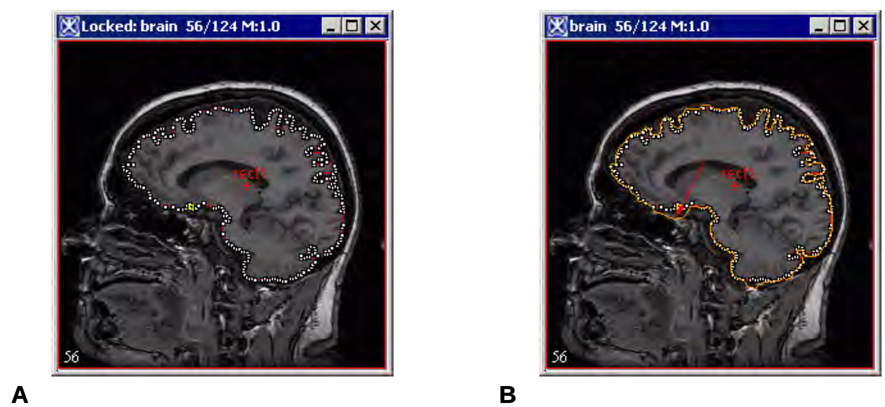
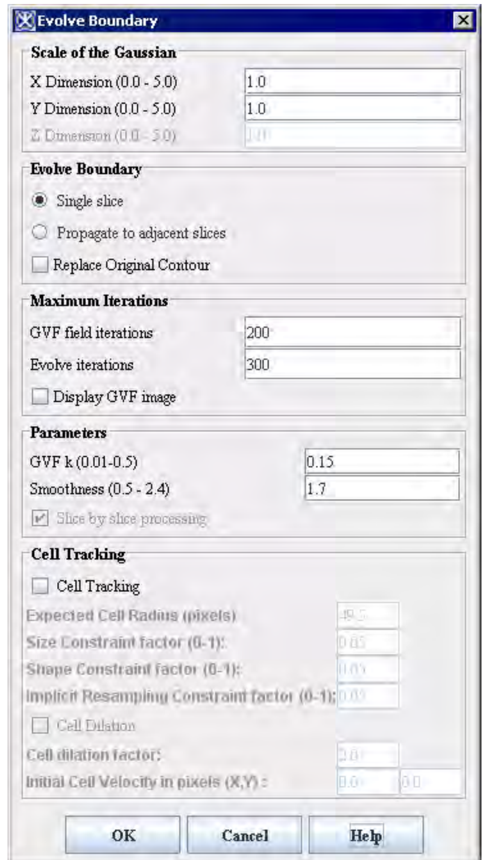


Figure 172. An initial VOI (A) and VOI after applying the Evolve Boundary > Active Contour algorithm with the default parameters (B)

EVOLVE BOUNDARY DIALOG BOX OPTIONS



Scale of the Gaussian	A large Gaussian scale slows the snake and causes it to conform to large scale structure. A small Gaussian scale causes the snake to conform to the small scale structure, which is therefore more sensitive to noise.
Evolve Boundary	Single Slice – applies the algorithm only on a current image slice. Propagate to adjacent slices – propagates boundaries to adjacent slices. Note that the algorithm used for 3D images version is a 2.5D dimensional algorithm where the resultant contour in a slice is projected into the adjacent slice and is used as an initialization to the evolution in the new slice. Replace original contour – if checked, replaces the original VOI.
Maximum Iterations	GVF field iterations – enter a desired number of GVF iterations here. Evolve iterations – enter a max number of iterations needed to generate a new VOI boundary. Display GVF image; if this box is checked, the GVF image appears in a new image frame.
Parameters	GVFk is a positive constant controlling the smoothness of the resulting VOI. Smoothness corresponds to the balance between internal and external forces. The internal force constrains the snake to be smooth while the external guides the snake to seek desirable image properties, such as edges. Slice by slice processing – if checked, this activates evolving boundary slice by slice.

Figure 173. The Evolve Boundary dialog box

Cell Tracking	Cell Tracking – if checked, this box activates the cell tracking algorithm; Expected Cell Radius (in pixels) – a user-defined cell radius Size Constraint and Shape Constraint factors are contributed towards the shape and size of the snake in order to make it circular with a user-defined cell radius. Implicit Resampling Constraint factor minimizes the number of points along the snake curve, thus they maintain an equal distance. Cell Dilation – if checked, allows dilation Cell dilation factor is multiple of the cell radius Initial Cell Velocity in pixels (X,Y)
OK	Applies the parameters that you specified.
Close	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

Figure 173. The Evolve Boundary dialog box (continued)

Modifying contours

Once contours are drawn, you can reposition, resize, or reshape them. You can also trim anchor points.

Repositioning contours

You can reposition a single contour or a group of contours. Follow the instructions below.

To reposition a single contour

Select the contour and drag it to the new location.

To reposition a group of contours

1 Hold down the Shift key as you click one of the contours in the group. All of the other contours in the group are also selected.

2 Continue to hold down the Shift key as you drag the contour group to the new location.

RESIZING CONTOURS

You cannot collectively resize a group that consists of multiple contours; each contour must be resized individually. Except for point VOIs, you can resize any contour shape. For lines, only the length can be changed; the width cannot. To resize lines, go to "To resize contours".

Before you can resize them, all contour shapes except lines must be surrounded by a bounding box. See also the following section "To turn the bounding box on".

To turn the bounding box on

1 Do either of the following:

- Select the contour and then select VOI > Properties in the MIPAV window.
- Right-click the contour. A menu appears. Select Properties.

The VOI Statistics dialog box opens. See also Figure 196 on page 361.

2 Select Show contour bounding box.

3 Change, if you want, any of the other information in the dialog box.

4 Click Apply.

5 Click Cancel to close the dialog box.

To turn the bounding box off

1 Do one of the following:

- Right-click the contour. Select Properties from the pop-up menu.
- Left-click the contour. Select Properties from the VOI menu in the MIPAV window.

The VOI Statistics dialog box appears. See also Figure 174.

2 Clear Show Contour Bounding Box.

3 Click Apply.

4 Click Cancel to close the window.

To resize contours

1 Select the contour. The bounding box appears (Figure 174) around the shape. (For rectangle contours, the bounding box and rectangle legs may overlap.)

2 Click Cancel to close the window.

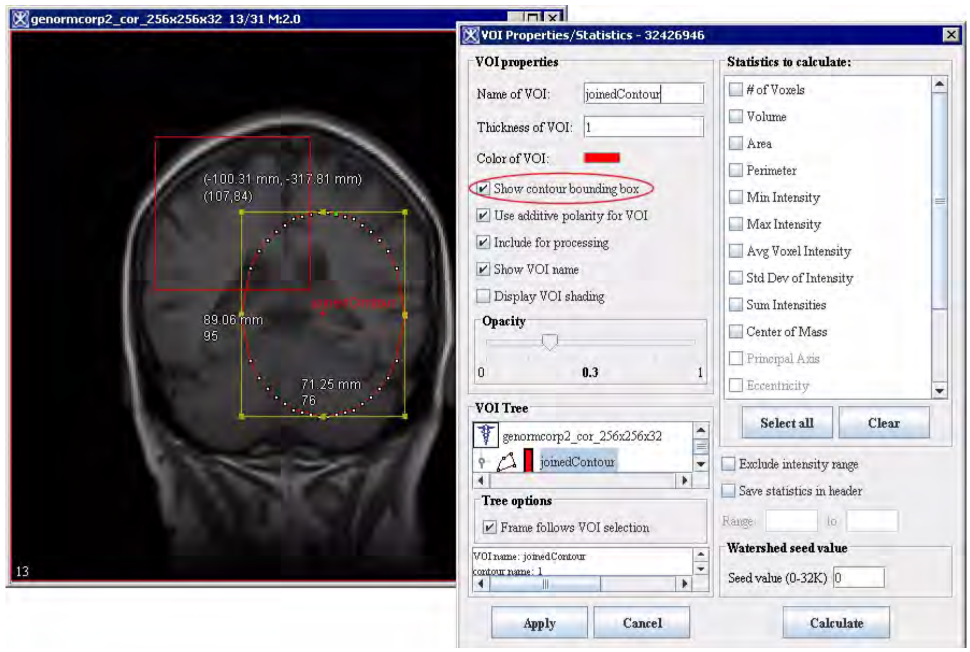


Figure 174. A VOI bounding box and the VOI Properties dialog box

3 Do one of the following:

- *To resize lines:* Click an edge or corner of the shape. Drag until the shape is the desired size.
- *To size other shapes:* Click one of the bounding box handles. Drag the handle until the shape is the desired size.

4 Choose whether to continue showing the bounding box.

Note: If you want to reposition or add an anchor point, turn the bounding box option off.

Reshaping a contour

You reshape the contour by adding or repositioning anchor points. Instructions for adding and repositioning anchor points follow.

To add anchor points

1. Click the contour to select it. The anchor points become visible.
2. Click the place where you want to add the new anchor point. The new anchor point appears.

To reposition anchor points

You can reposition a single anchor point at a time. If you need to redraw a portion of the contour, MIPAV allows you to reposition several contiguous anchor points with one motion of the mouse.

1. Select the contour. The anchor points become visible.
2. Position the pointer over the anchor point or points that you want to reposition. The pointer changes to the shape of a hand with the index finger extended.
3. Do either of the following:
   - *To reposition a single anchor point:* Drag the anchor point to the new location.
   - *To reposition several anchor points:* Hold down the Alt key as you drag the anchor points in the counterclockwise direction to their new locations.

Retracing a VOI contour

To retrace a VOI contour do the following (see Figure 175):

1. Select an existing VOI;
2. Hold the ALT key and use the mouse to pick up a point on the VOI;
3. While holding both the ALT key and left mouse button, move the mouse clockwise to redraw the VOI;

4 Stop moving when you reach the last point which you would like to redraw.

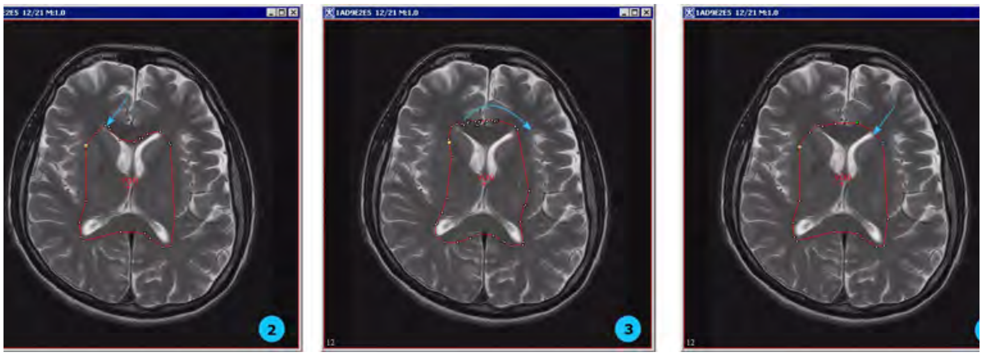


Figure 175. Retracing a VOI contour

TRIMMING VOI ANCHOR POINTS

MIPAV can automatically trim anchor points from the contour. In the trimming process, unnecessary anchor points are pruned. For example, if three anchor points are on the same line, the middle anchor point is deleted. Since only two anchor points are needed to indicate a line, the middle anchor point is extraneous. The trim parameter defines what is meant by the *same line*. Setting this value to zero causes only unnecessary anchor points on a straight 180-degree line to be pruned. Setting a higher value prunes all but the sharpest corners of the contour.

To trim anchor points

1 Select the contour you want to trim.

2 Select VOI > Trim Parameter in the MIPAV window. The VOI Trim Parameter dialog box (Figure 176) opens.

3 Move the slider to the desired number.

Trim parameter	Specifies the number of anchor points you want the software to trim from the contour. Setting this value to zero causes only unnecessary anchor points on a straight 180-degree line to be pruned. Setting a higher value prunes all but the sharpest corners of the contour.	
Trim adjacent points	Trims points that are adjacent.	
Apply	Applies the parameters that you specified.	
Close	Disregards any changes that you made in this dialog box and closes the dialog box.	
Help	Displays online help for this dialog box.	

Figure 176. VOI trim parameter dialog box

4 Click Apply. The software removes all unnecessary anchor points.

5 Click Close to close the dialog box.

Grouping and ordering contours

By default, when you draw multiple VOIs they appear grouped in a single group, and therefore all colored in the same color. To ungroup VOIs, select one VOI from the group, and then call VOI>VOI Grouping>Group VOIs. This will ungroup VOIs and they appear in the image in different colors. To group VOIs, select the VOIs which you would like to group (use Ctrl+Shift+ right mouse combination of keys for multiple selection), and then call VOI>VOI Grouping>Group VOIs. This will group selected VOIs and they all appear in the image colored in the same color.

Grouping contours

A group may consist of one or more contours. A contour is assigned to only one group. When a contour is initially drawn or generated, MIPAV assigns it to a default group. MIPAV also assigns default group properties, such as color and name. Each contour inherits the properties of its assigned group. At any time, you can change the properties of a group or its members (contours).

To change the group with which a contour is affiliated

1. Right-click the contour you wish to modify.
2. Select Properties. The VOI Statistics dialog box opens. See Figure 36.

   The name of the VOI appears in the Name of VOI text box. A colored box appears beside the text box.

To change group properties

1. Select the colored box on the VOI Statistics dialog box. The Pick VOI Color dialog box opens.
2. Click OK to apply the color to the grid/border. The Pick VOI color dialog box closes.
3. Click OK.
   - To change the name of the VOI, modify the name in the text box next to the Name of VOI colored box.
   - To adjust the opacity, move the slider in the Opacity panel. The number in the center indicates the current value; 0 is transparent and 1 is opaque.

Splitting VOI contours

A VOI can be divided into two or more contours using the Split VOI Contour icon. Any contour that is initially drawn or generated can be spitted into two contours using that tool. For result VOIs, MIPAV assigns default VOI properties, such as color and name. By default, the result VOIs have the following names: *firstVOI* and *secondVOI*.

Each result contour inherits the properties of its parent VOI. However, at any time, you can change the properties of result contours using the standard MIPAV tools (see "Contouring structures manually").

To split a VOI

1. Select the VOI.
2. Click the Split VOI Contour icon on the VOI toolbar, the mouse cursor changes to "+".

3 In the image, use the mouse to draw the line splitting the VOI into two.

4 In the Split VOI dialog box that appears, specify if you want to split the VOI in all slices or only in the active slice. Press Split.

The result VOIs appears in the image, see Figure 177.

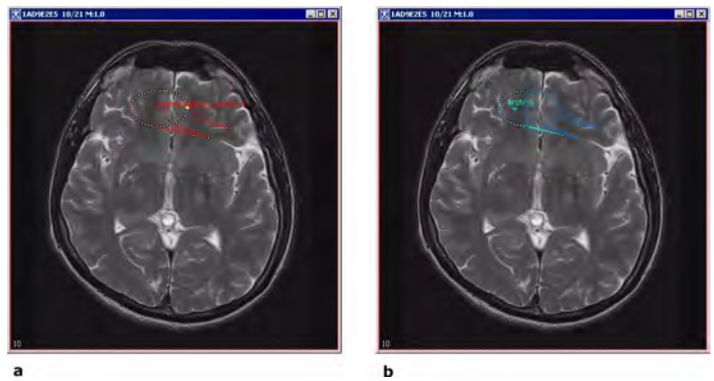


Figure 177. Splitting a VOI: (a) the initial VOI and (b) the two result contours

ORDERING CONTOURS

If two or more contours overlap, MIPAV organizes them so that they appear one on top of the other, much like a stack of papers on a desk. This is often not readily apparent because, by default, when contours are drawn, they are transparent and only the outlines are visible. Thus, if two contours overlap, all parts of each contour, except the areas where the outlines overlap, can be seen.

To fill contours

1 Right-click the contour.

2 Select Properties. The VOI Statistics dialog box opens. See also Figure 196 on page 361.

3 Clear the Boundary or Blended check box to remove the check mark, then click Apply.

4 Close the VOI Statistics dialog box. The contour is filled with the default paint color. To change the paint color, refer to “Manually creating a mask using paint” on page 332.

To arrange contours

1 Click the contour to be reordered. See Figure 178.

2 Select the Contour Order context menu. Then select one of the following commands:

- *Bring to Front:* Moves the contour in front of all of the other contours.
- *Send to Back:* Moves the contour behind all other contours.
- *Bring Forward:* Moves the contour forward one position. For example, if you select this command and the contour is the third contour in a stack of five contours, the contour is moved to the second position in the stack.
- *Send Backward:* Moves the contour backward one position. For example, if the contour is the third contour in a stack of five contours, the contour is moved to the fourth position in the stack.

The software reorders the contours according to the command.

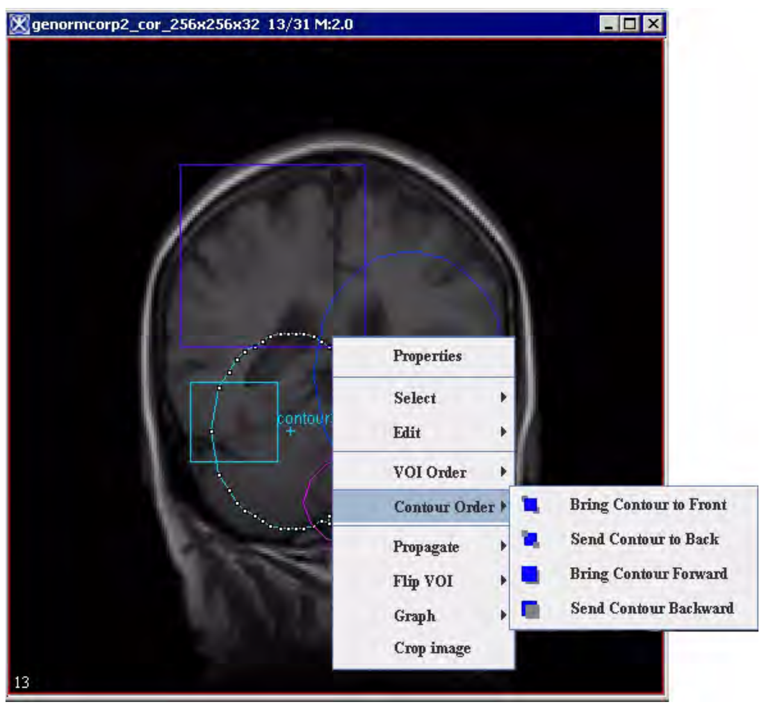


Figure 178. Arranging the VOIs

SMOOTH VOI

To smooth a VOI, select it first, and then call VOI > Smooth VOI. The Smooth VOI dialog box opens. This dialog sets the parameters and then calls the algorithm for smoothing the VOI. In the dialog box, you can specify the following:

- Whether or not to remove the original selected VOI;
- Whether or not to apply trimming to remove nearly collinear points;
- A number of interpolation points.

In 2D images all selected curves of a selected VOI are smoothed. In 3D images all selected curves in all slices of a selected VOI are smoothed. The original Z-slice information is only used so that the default number of interpolated points comes from a contour in a Z-slice.

If the original VOI was not removed, the new VOI has a new color. If the original VOI was removed, the new VOI has the same color as the original VOI had.

Replace Original Contour	If selected, this option allows a new smoothed VOI to replace the original VOI.
Trim Collinear Points	If selected, it allows to trim points that are adjacent.
Number of Interpolation Points	Enter the number of interpolation points that will used to calculate a new smoothed VOI.
OK	Applies the parameters that you specified.
Close	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

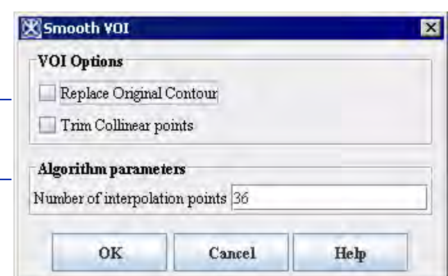


Figure 179. Smooth VOI dialog box

Allow VOI holes (XOR)

This option (if turned on) allows to preserve holes in VOIs when converting them to masks (see "Converting VOI contours to masks"), see Figure 180.

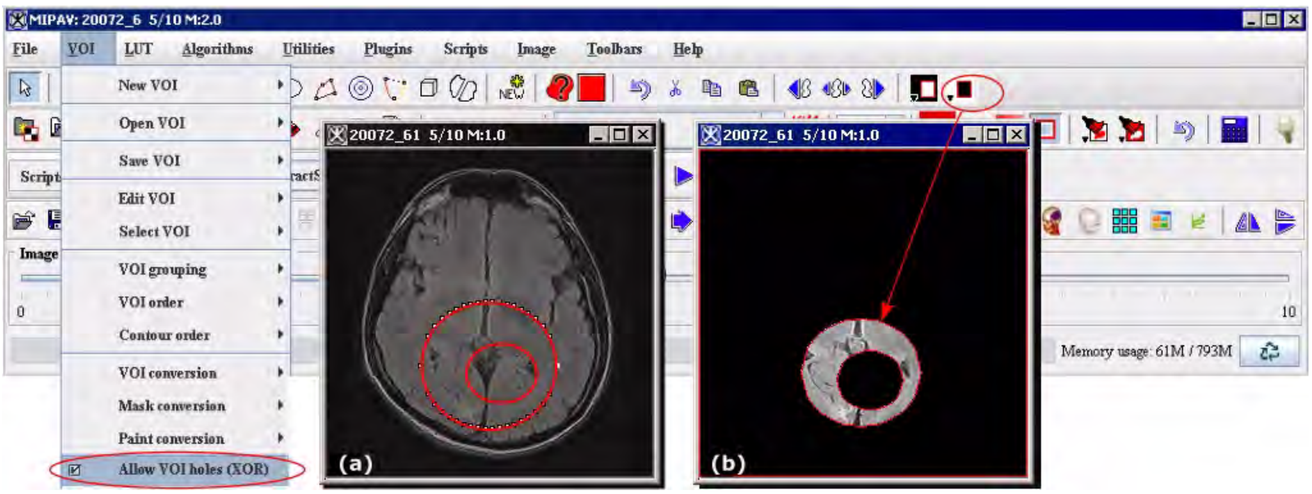


Figure 180. The Allow VOI holes (XOR) option is turned on

The algorithm uses the XOR operation, for more information, refer to "Morphology".

Copying, cutting, and pasting VOIs

You can undo, cut, copy, and paste a VOI within the image, to another slice in the image dataset, or to another image dataset.

To undo a VOI

1. Select the VOI.
2. Select Edit > Undo VOI in the MIPAV window. The software removes the VOI from the image.

To cut a VOI

1. Select the VOI.
2. Select Edit > Cut VOI in the MIPAV window. The software copies the VOI to the clipboard and deletes it from the window.

Tip: You can also remove a VOI by pressing the Del key after selecting the VOI.

To copy and paste a VOI

1 Select the VOI.

2 Select VOI> Edit VOI> Copy VOI. It is copied to the clipboard.

3 NOw, do one of the following:

- Navigate to another slice using the Image Slice Slider.
- Or open another image dataset.

4 Select VOI > Edit VOI > Paste VOI. The program copies the VOI from the clipboard and pastes it into the image.

You can only copy and paste a single, not multiple, VOIs into another slice using the Paste command. Also, note that the software pastes a copied VOI into the image at exactly the place from which it was copied.

To propagate multiple VOIs to the next or previous slice in the dataset

– Propagate Both Sides; – Propagate Left; – Propagate Right;

1 Select all of the VOIs that you want to copy.

2 Do one of the following:

- Click the Propagate VOI up icon, to copy the VOI to the next slice.
- Click the Propagate VOI down icon, to copy the VOI to the previous slice.
- Click the Propagate VOI to all slices icon, to copy it to every image in the dataset.

Deleting VOIs

You can easily delete a VOI or one of its contours.

To delete a VOI

1 Select the VOI or one of its contours.

To delete a VOI that consists of many contours, hold down the Shift key and then click one of the contours in the VOI group.

2 Do one of the following:

- Right-click on the VOI, and then select Edit > Delete.
- Select Edit > Delete VOI in the MIPAV window.

The software removes the VOI or contour.

Saving and opening VOI files

You can save a VOI to a text file. This file can then be opened and applied to the same dataset or a different one, or it can be manipulated using a word processor.

To save a single VOI as a text file

1 Select a contour or a VOI.

- To select a contour, click on it.
- To select a VOI, hold down the Shift key and select a contour. All contours that are part of the VOI are selected.

2 Do either of the following:

- If you want the software to automatically name the file, select VOI > Save VOI in the MIPAV window. The software saves the file with an.XML extension.
- If you want to assign the file name, select VOI > Save VOI as. The Save as dialog box appears.

a Type the name of the VOI file in the File Name box. Be sure to use either the VOI extension or the XML extension.

b Click Save. The software saves the file under the specified name.

To save multiple VOIs

Do either of the following:

- To allow the software to automatically name the file, select VOI > Save All VOIs.
- To assign a file name to the file, select VOI > Save All VOIs as.

  a Type the name of the VOI file in the File Name box. Be sure to use either the VOI extension or the XML extension.

  b Click Save. The software saves the file under the specified name.

To open saved VOI files

1. Select the image window of the dataset to which you want to apply the VOI.
2. Select VOI > Open VOI in the MIPAV window. The Open dialog box appears.
3. Select the file or files and then click Open. The VOI or VOIs appear in the image window.

Note: If you make a change to the VOI and want to make a change to the file, select VOI > Save VOI in the MIPAV window. The changes are saved to the VOI file.

Generating masks

A *mask* is a filter that selectively includes or excludes certain values. You can create a mask that includes the VOI areas and excludes all other areas. Masks are particularly helpful if you use other applications to further analyze images because those programs may require that you supply a mask. An example of a mask appears in Figure 181.

You can create masks using a variety of methods:

- Use paint tools to manually create a mask, refer to page 332
- Use Paint Grow Segmentation tool, refer to page 344
- Use binary or short mask utilities, refer to page 347
- Use the Advanced Paint tool, refer to page 352
- Use the Power Paint tool, refer to page 353

MIPAV also provides algorithms that you can use to generate masks. These algorithms are addressed in volume 2 of the *User's Guide*.

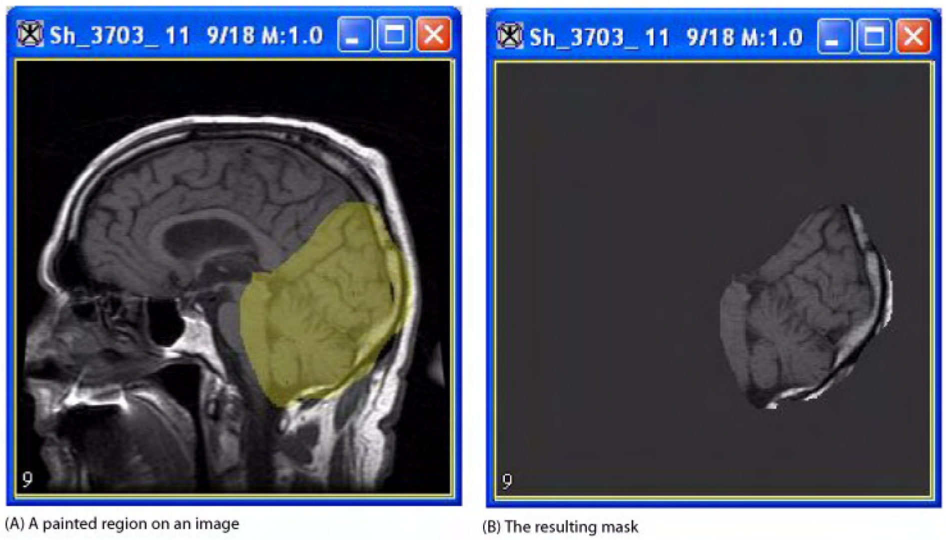


Figure 181. Image with (A) a painted region and (B) the resulting mask

Manually creating a mask using paint

You can manually create a mask using the tools on the Paint toolbar, which is located in the MIPAV window. To create the mask, you first select the intensity level of the paint, along with the paint color and opacity value. Next, you paint on the image using the paint brush. You can adjust the size of the paint brush tip. When complete, you then commit the paint by choosing one of the following icons:

 – Masks Inside the Painted Area

 – Masks Outside the Painted Area

When you click one of these icons, it applies the intensity permanently to the image. You can then choose to save the image.

If the Paint toolbar does not appear in the MIPAV window, select Toolbars. The Toolbar menu lists four commands: VOI Toolbar, Paint Toolbar, Scripting Toolbar, and Image Toolbar. If a toolbar command has a check mark next to it, the toolbar is displayed; otherwise, it is hidden. To display the Paint toolbar, select Paint Toolbar. See also "MIPAV toolbars" on page 71.

MANUALLY GENERATING A MASK INCLUDES THE FOLLOWING FOUR STEPS:

- Step 1, Selecting the paint intensity level
- Step 2, Selecting a paint color
- Step 3, Adjusting the opacity level of the paint
- Step 4, Applying the paint

STEP 1, SELECTING THE PAINT INTENSITY LEVEL

The first step in manually creating a mask is to select the intensity level of the masked area.You can select the intensity level by using the eyedropper tool to select an intensity value that is already present in an image or by typing a specific intensity value in the Desired Paint Intensity dialog box.

To select the intensity level using the Eyedropper tool

 – Eyedropper

The allows you to copy an intensity level that is already present in the image. To use this tool, complete the following steps:

1. Click the Eyedropper icon, on the paint toolbar in the MIPAV window.
2. Click the area of the image that has the intensity level you want to copy. The intensity shown at the bottom of the MIPAV window changes to the intensity level in the part of the image you've selected.

Example: Initially, the intensity level button shows a zero on the button. However, when you click an area of the image, it displays the number that is associated with that part of the image, such as 250.

You can enter a specific intensity level value in the Desired Paint Intensity window. To do this, complete the following steps:

1 Click the Intensity button.

Note: If you changed the intensity level using the eyedropper or this button previously in your current MIPAV session, the number on the Intensity button would have changed from 0 to the number representing that earlier intensity level.

The Desired Paint Intensity dialog box opens.

2 Notice that the minimum and maximum intensity values appear in parentheses. These values depend on the image type, or data type, of the image.

Tip: You can look at the exact intensity levels in the image using , the Magnify region icon.

Intensity	Specifies the intensity, or strength, of the paint.
Apply	Applies the intensity that you specified to the image.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

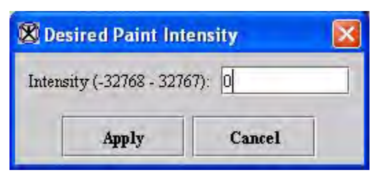


Figure 182. Desired Paint Intensity dialog box

3 Type the desired level of intensity.

4 Click Apply to apply the intensity level to the paint. Note that the number on the Intensity button changes to the number you entered.

5 Click  or Cancel to close the dialog box.

Step 2, Selecting a paint color

 – Change Paint Color

After you select the intensity level, the next step is selecting the paint color. The program applies this color to the image so that you can easily see where you applied the new intensity. When you commit the changes to the image, the grayscale intensity level that you indicated in step 1 is permanently applied to the image.

To select a color, use the Change Paint Color icon, to display the Pick Paint Color dialog box (Figure 184). This dialog box includes three tabbed pages: Swatches page, which is always displayed first; the HSB page; and the RGB page. You can select a color on any of the pages.

To select a color from one of the 310 color swatches

1. Click the Change Paint Color icon. The Pick Paint Color dialog box (Figure 183) opens.
2. Stay on the Swatches page, or, if you moved to HSB or RGB pages, click Swatches to return to the Swatches page.
3. Click the desired color in the larger grid. The color appears in the Recent color grid on the right.
4. Click Close when done. The Pick Paint Color dialog box closes. The color you selected becomes the current paint color.

To select a color using the HSB model

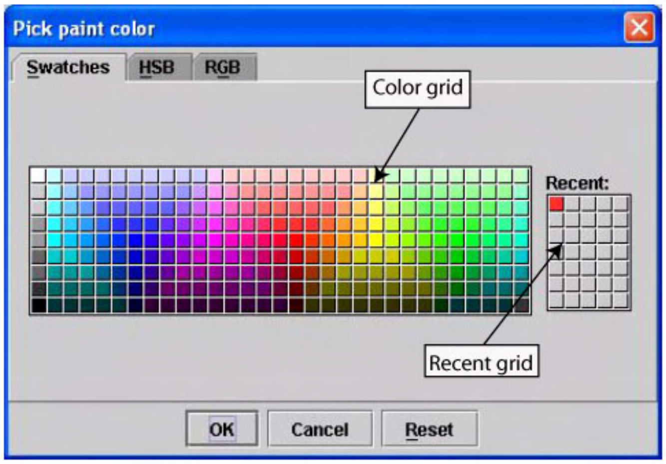
1. Click the Change Paint Color icon. The Pick Paint Color dialog box opens.
2. Click HSB. The HSB page appears (Figure 184B).
3. Specify a color by entering specific numbers for the hue, saturation, and brightness values, or select the color from the color graphic.
   - Enter specific values into the H, S, and B boxes:

- *Type the hue value in H.* The hue can be any number from 0 to 359. The numbers correspond to the degree numbers on the color square (Figure 185). For example, to specify a shade of blue, enter a number from 255 to 270. The slider automatically moves the correct hue on the color strip.
- *Type the saturation value in S.* The level of saturation can be anywhere from 0 (gray) to 100 (full-strength hue). When you type a value into this box, a circle appears in the color square around the color with the specified saturation level. Colors to the left of the color graphic are circled if the saturation level is high. Lower saturated colors are found to the left of the color graphic.
- *Type the brightness value in B.* The level of brightness can be anywhere from 0 (dark) to 100 (light). In the color graphic, a circle appears around the color with the specified brightness level. Colors near the top of the color graphic are circled if the saturation level is high. Darker colors are found near the bottom of the color graphic.

- Select the hue, saturation, and brightness on the color square or color strip:
  - Drag the slider to select the desired hue from the color strip.
  - Select the shade you wish to use in the color square.

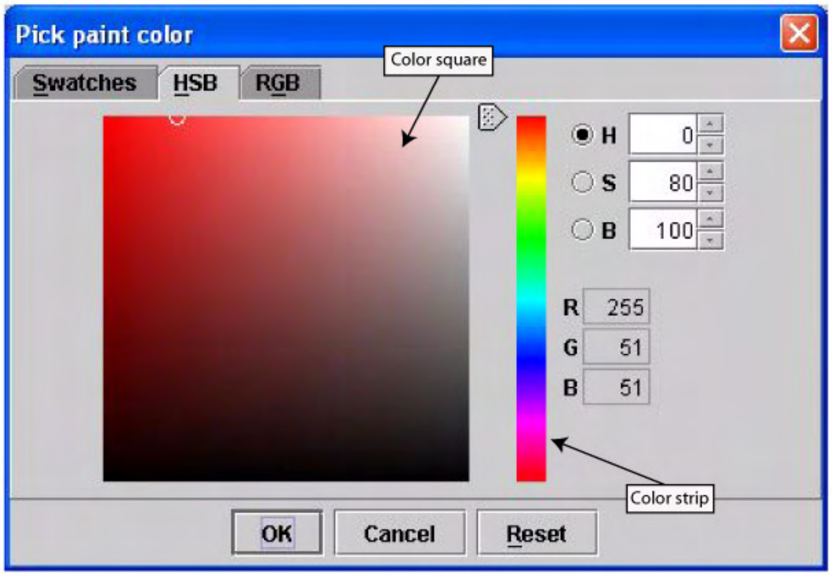
4 Click Close when done. The Pick Paint Color dialog box closes. The color you selected becomes the current paint color.

Pick Paint Color dialog box options



Color grid	Shows the available colors. Select one of these colors and it appears in the Recent grid on the right.
Recent grid	Indicates colors that you have recently selected from the color grid on the left. MIPAV uses the last color that you selected from the color grid and that appears in this grid as the color of the background in the lightbox view once you select OK.
OK	Applies the parameters that you specified to crop the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not crop the image.
Reset	Returns the color settings to their previous values.
Help	Displays online help for this dialog box.

Figure 183. Swatches page in the Pick Paint Color dialog box



Color square	Specifies a specific color to be used as the background color.
Color strip	Specifies a specific color to be used as the background color.
H	Specifies the number representing the hue of the color.
S	Specifies the number representing the saturation of the color.
B	Specifies the number representing the brightness of the color.
R	Specifies the number for the color used for the red channel.
G	Specifies the number for the color used for the green channel.
B	Specifies the number for the color used for the blue channel.
OK	Applies the parameters that you specified to crop the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not crop the image.
Reset	Returns the color settings to their previous values.
Help	Displays online help for this dialog box.

Figure 184. The HSB page in the Pick Paint Color dialog box

To select a color using the RGB model

1. Click the Change Paint Color icon. The Pick Paint Color dialog box opens.
2. Click RGB. The RGB page (Figure 184C) appears.
3. Specify the numbers for the three components (red, green, and blue), or use the sliders to select a color.
4. Click Close when done. The Pick Paint Color dialog box closes. The color you selected becomes the current paint color.

Using the Hue, Saturation, and Brightness Color Model

The HSB color model defines colors by a combination of their hue, saturation, and brightness.

The *hue* is the wavelength of light transmitted through or reflected from an object. The hue is more commonly known as *color* (such as magenta or green).

Below is the Munsell color wheel. A color strip, which is derived from the color wheel, appears as well. Hues are represented by the degrees in the color wheel. (Since colors are graduated, the degrees in the figure are approximations and not necessarily absolute values.

For example, yellow is approximately 45 degrees; 50 degrees is a different shade of yellow.

Primary colors (red, yellow, and blue) combine to create secondary colors (magenta, cyan, and green).Tertiary colors, such as orange, are also represented on the color wheel.

The color strip is simply a different representation of the color wheel. Degrees and corresponding hues were added to the figure.

Saturation is the strength or purity of the hue. To lessen the saturation, gray is added to the color. Thus, if pure orange is 70 percent saturated, the hue is a mixture of orange (70 percent) and gray (30 percent).

Brightness is the relative darkness and lightness of a color; 0% brightness is black; 100% brightness is white.

Figure 185. Munsell color wheel (left) and color strip (right)

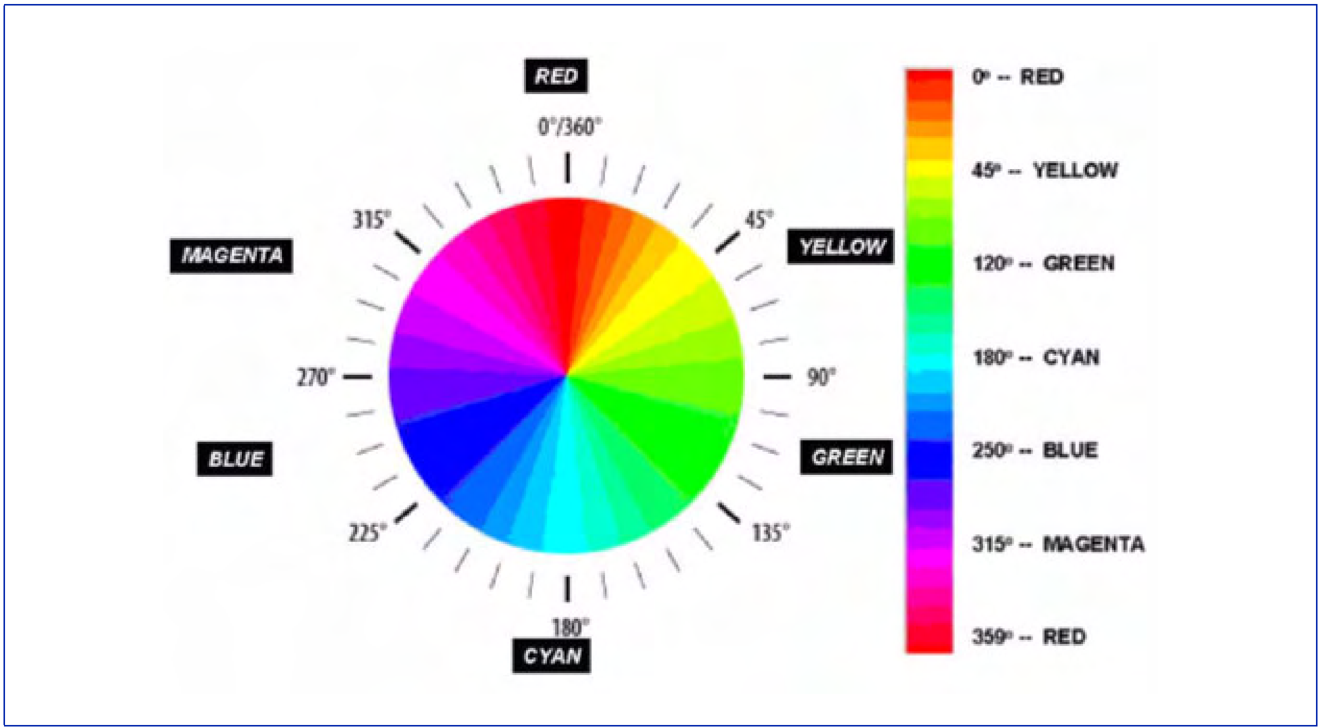
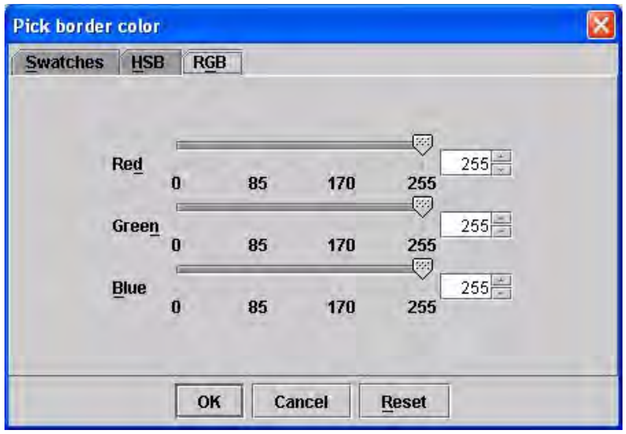


Figure 185. Munsell color wheel (left) and color strip (right)



Red slider and list box	Specifies the number representing the red channel. Either use the slider, or select the number in the list box.
Green slider and list box	Specifies the number representing the green channel. Either use the slider, or select the number in the list box.
Blue slider and list box	Specifies the number representing the blue channel. Either use the slider, or select the number in the list box.
OK	Applies the parameters that you specified to crop the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not crop the image.
Reset	Returns the color settings to their previous values.
Help	Displays online help for this dialog box.

Figure 186. RGB page in the Pick Paint Color dialog box

Step 3, Adjust the opacity level of the paint

You can further refine the appearance of the paint selecting its opacity level. Opacity levels can range from 0.0 (transparent) to 1.0 (opaque).

To change the opacity level

 – Change Opacity of Paint

1. Click the Change Opacity of Paint icon on the Paint toolbar. The Paint Opacity window appears (Figure 187).
2. Drag the slider to the desired level of opacity. The current level appears in the bold print underneath the slider.
3. Click Close to apply the opacity level. The opacity level on the image changes according to your specifications.

Opacity	Specifies the degree of transparency. To make the color more transparent, move the slider to left. To make it more opaque, move the slider to the right.
Close	Closes this dialog box.

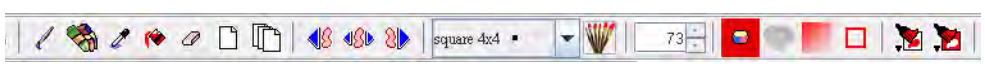
Figure 187. Paint Opacity dialog box

STEP 4, APPLY PAINT

Having selected the intensity level, paint color, and opacity level, you are now ready to paint on the image with the paintbrush.

 – Draw Using a Brush

To paint on the image



1. Click the Draw Using a Brush icon, on the Paint toolbar in the MIPAV window.
2. Specify the size of the paintbrush tip.
3. Click and drag the mouse button to paint on the drawing. You can select one pixel at a time, or drag the mouse button to draw paint strokes on the image.

Tip: You may need to use the magnification buttons to adjust the magnification level of the image.

If you make a mistake, you can erase or undo the painted area.

To erase paint from a specific area

– Erases Paint

1 Click the Erase icon.

2 Select the size of the eraser tip.

3 Use the mouse to begin erasing. You can select one pixel at a time, or you can drag the mouse.

Tip: When erasing the paint, it may be easier to magnify the size of the image using the magnification icons.

To resume painting, click the Draw Using a Brush icon.

To erase all paint

– Erase All Paint

– Undo

1 Do one of the following as appropriate:

- Click the Erase all paint icon, which removes all paint from the image.
- Click the Undo last region paint icon, which allows you to undo the last paint stroke.

2 Click the Default Mode icon, when you have finished painting the image.

STEP 5, COMMIT

When you visualize a dataset in MIPAV, a composite of several layers appears in the default image window. The image layer contains the original image. The next two layers contain the results of applied utilities, algorithms, and paint. The fourth layer contains VOI contour information.

When you click the commit icons. e.g. the Masks Inside Painted Area icon or Masks Outside Painted Area icon, the paint and utility/algorithm layers are merged with the image layer and saved as a new file.

To commit changes, select one of the following commit icons:

 – the Masks Inside Painted Area icon. This tool shows the image without the area you painted (Figure 188-C).

 – the Masks Outside Painted Area icon. This tool removes the painted area from the image and shows only the removed area (Figure 188-B).

The software redraws the dataset with the intensity levels indicated on the image. You may need to adjust the lookup table (LUT) to see the image more clearly, as it has been dome for the image shown in Figure 188-C. Instructions for adjusting the LUT appear in Chapter 5, "Visualizing Images."

a b c

Figure 188. The painted region (a) and the mask produced using the Mask the Outside Painted Area tool (b) and Mask the Inside Painted Area tool (c). Here, arrows point to the painted region.

Creating a mask using the Paint Grow Segmentation method

You can create a mask using the paint grow segmentation method, which uses voxel aggregation to group voxels into larger regions. You begin by selecting a voxel to serve as the *seed point*, or starting point. The software applies paint color to all voxels touching the seed point that fall within the intensity range that you specify.

Note: You cannot apply the paint grow segmentation method to RGB images.

To create a mask using the Paint Grow tool

 – Fill an Area with Desired Color

1. Click the Fill an Area with Desired Color icon. The Paint Grow dialog box appears (Figure 189).
2. Select the seed point, which is used as a starting point for the paint grow operation. To do this, move the pointer to the image. As you move the cursor, the location and intensity of the voxel under the pointer tip appears in the Cursor position and voxel intensity text box. Click the voxel you want to designate as the seed point.
3. Adjust the delta values and parameters.
4. Click Apply when complete to begin the paint grow method. The Paint Grow dialog box closes.

To correct the mask

If the results are not what you want, do the following:

1. Click the Paint Grow button. The Paint Grow dialog box appears.
2. Select the seed point, which is used as a starting point for the paint grow operation. To do this, move the pointer to the image. As you move the cursor, the location and intensity of the voxel under the pointer tip appears in the Cursor position and voxel intensity text box. Click the voxel you want to designate as the seed point.
3. Adjust the delta values and parameters.
4. Click Apply when complete to begin the paint grow method. The Paint Grow dialog box closes.

If the results are not what you want, do either of the following:

- Click the Undo last region paint icon, and start again.
- Click the Erase icon, or click, the Erase all paint icon, to erase all paint.

To commit the mask

Click one of the following commit buttons:

 – the Masks Inside Painted Area icon.

 – the Masks Outside Painted Area icon.

Paint Grow dialog box options

Cursor position and voxel intensity	Indicates the coordinates and intensity of the pixel under the mouse pointer tip. This pixel is the seed point.
Delta above selected voxel intensity	Once a seed point has been selected, MIPAV uses this value to determine whether to fill adjacent voxels with color. The voxels that are filled must have intensity levels that fall within the range of the seed point intensity minus the lower delta value and the upper delta value. For example, if seed point has an intensity of 100.0, and the *Delta Above Selected Pixel Intensity* value is 10 and the *Delta Below Selected Pixel Intensity* value is 15, MIPAV color-fills adjacent voxels whose intensities range from 85.0 to 110.0.
Delta below selected voxel intensity	Once a seed point is, MIPAV uses this value to determine whether to fill adjacent voxels with color. The voxels that are filled must have intensity levels that fall within the range of the seed point intensity minus the lower delta value and the upper delta value.

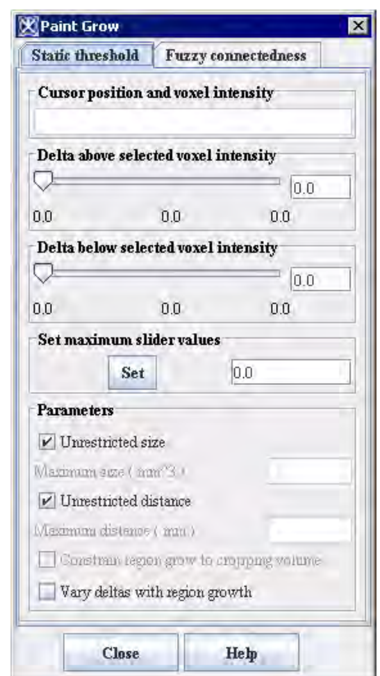


Figure 189. Paint Grow dialog box

Parameters: **Unrestricted size** **Maximum size** **Unrestricted distance** **Maximum distance**	Constrains the growth of the paint grow operation. Select the **Unrestricted size** and **Unrestricted distance** check boxes to allow the paint grow operation to be applied without restraint. If the **Unrestricted size** check box is not selected, type the **maximum size** (in cubic meters) of the paint grow region in the text box. If the **Unrestricted distance** check box is not selected, type the maximum distance from the original seed point in the text box.
Fuzzy connectedness	
Fuzzy connectedness	Check tis box to use the fuzzy connectedness coefficient instead of static threshold. Here, Fuzzy connectedness represents the idea of connection or "hanging-togetherness" of image elements in an object by assigning a strength of connectedness to every possible path between every possible pair of image elements. A fuzzy connected object is defined with a fuzzy threshold or the strength of connectedness.
Initial variance from selected VOI	Uses the initial intensity values from the selected region of interest (VOI).
Display fuzzy image	Displays the result image in a separate frame.
Fuzzy threshold	is a threshold on the strength of connectedness of image elements.
Close	Closes this dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

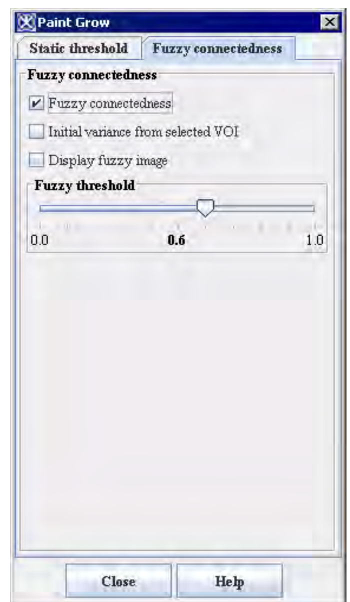


Figure 189. Paint Grow dialog box (continued)

EXAMPLES OF USING THE PAINT GROW TOOL

Here, is a step-by-step guide for selecting a region(s) of interest using the Paint Grow tool options. In this example we will use an image of the human eye and we will try to segment the blood vessels located on the retina.

First, make sure to adjust the contrast of your image so that the different tissues appear visually distinctive. For images with more than 8 bits per voxel you may want to use the various LUT available in MIPAV.

To segment blood vessels using the static threshold

1 Open the Paint Grow dialog box.

2 Use the mouse cursor to select the point on the image (on a blood vessel) which has the desired intensity value.

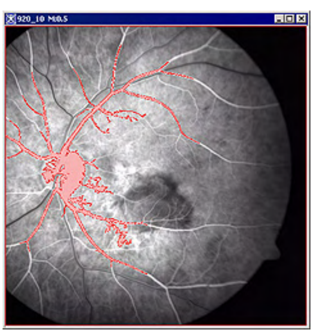
3 Use the Change Paint Color option to select the color (other than red) which will be used for painting.

4 On the Static Threshold tab, adjust the intensity thresholds so that the tissue you want to preserve is highlighted.

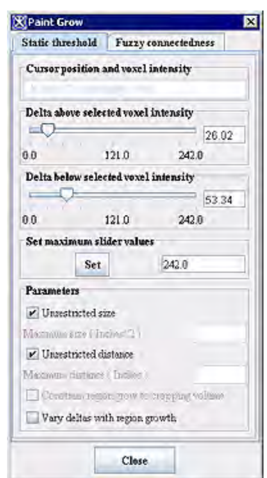
5 Check the Unrestricted size and Unrestricted distance options. This will allow the paint grow operation to be applied without restriction.

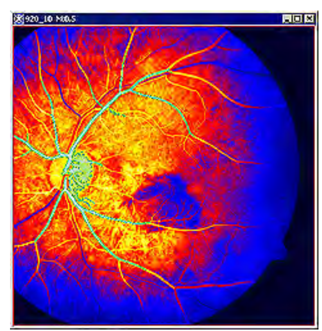
6 Watch the paint region growing.

For example, when selecting the blood vessels, the image might look like the panel A for grayscale images or panel B for images after applying Blue LUT in the Figure 190 below.



A – the painted region appears in red (which is the default color)





B – the painted region appears in green, because we selected it as a color contrast to LUT colors

Figure 190. The Paint Grow tool was used to locate the blood vessels on the grayscale image (A) first, and then on the same image after applying the Blue LUT (B).

To segment blood vessels using the Fuzzy Connectedness option

1 Open an image of interest.

2 You might consider to delineate a VOI on a region of the image which is of your interest, first. This is optional.

1 Open the Paint Grow dialog box, and then open the Fuzzy Connectedness tab.

2 Check the **Fuzzy Connectedness** check box to activate the tool.

3 Check the **Initial variance from selected VOI** box (optional).

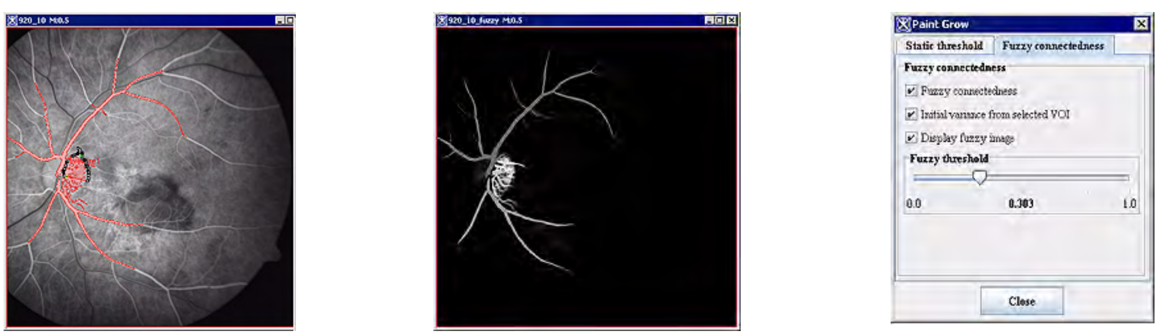
4 Check the **Display fuzzy image option** to view the result in a new frame.

5 Use the mouse cursor to select the point on the image (on a blood vessel) which has the desired intensity value.

6 Adjust the Fuzzy thresholds so that the tissue you want to preserve is highlighted.

7 Watch the paint region appeared in a new frame.

For example, when selecting the blood vessels, the image might look like the panel A or panel B for images in the Figure 191 below.



a – the painted region appears in red

b –the painted region also appears in a new frame

c

Figure 191. The Fuzzy Connectedness option.

Converting VOI contours to masks

You can convert a contoured VOI to a binary, short, or paint mask using the options found in the VOI menu in the MIPAV window.

To convert a contoured VOI to a binary mask

In a binary mask, the value 1 is assigned to the area inside the VOI; 0 is assigned to the area outside the VOI. Thus, typically, the area that represents the VOI is white, while the area outside the VOI is black.

1. Select **VOI > Mask > VOI(s) to Binary Mask**. The binary mask appears in a separate image window.
2. Save the image (optional).

To convert a contoured VOI to a short mask

In a short mask, the values 1 to 65,535 are assigned to the area inside the VOI; 0 is assigned to the area outside the VOI. Thus, typically, the area that represents the VOI is several shades of gray and the area outside is black.

1. Select **VOI > Mask > VOI(s) to short mask** option. The short mask appears in a separate image window.
2. Save the image (optional).

To convert a contoured VOI to a paint mask

1. Contour the VOI. For instructions on how to contour VOIs, refer to "Using contours to segment a VOI" on page 308.
2. Select the color, opacity, and intensity level of the paint.
3. Select **VOI > Mask > VOI(s) to Paint Mask**. The contours are filled with the default paint color.
4. Delete the contours by right-clicking the VOI and selecting Edit > Delete.
5. Save the image (optional).

Converting masks to VOI contours and paints

To convert a mask to an image with VOIs

1 Create a paint mask. For information on how to do this, refer to "Generating masks" on page 331.

2 Select the VOI > Mask conversion > Mask-> VOI.

Contours replace the painted areas.

To convert a mask to an image with paint(s)

1 Open an image of interest.

2 Create a mask. For information on how to do this, refer to "Generating masks" on page 331. Alternatively, you can load the previously saved mask into the image frame using the File > Load image B > From frame menu.

3 Select the VOI > Mask conversion > Mask-> Paint.

Paints replace the masked areas.

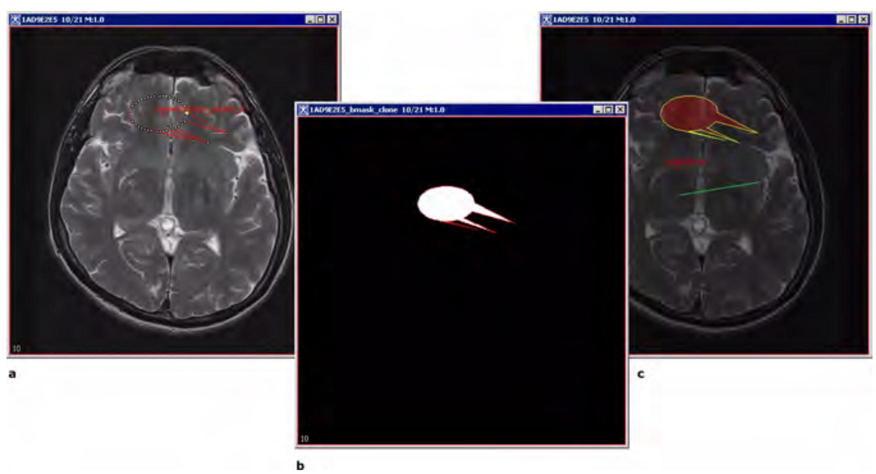


Figure 192. Converting a VOI to, a mask and then to a paint: (a) –the initial VOI, (b) – the mask created from the VOI, and (c) – the paint

Advanced Paint and Power Paint tools

Paint masks are a way of marking specific parts of an image. For example, a mask could define the specific region in a patient brain scan or location of kidney in a healthy adult. Masks could have multiple uses: for work with neurological patients, masks can help map the location and extent of an injury or some abnormality; in addition, masks can be used to understand the role of a specific area of the healthy brain when applied to functional brain images (e.g. fMRI). In MIPAV, masks are created by mapping the regions on each 2D slice of a 3D volume using a brush tool. This section describes how a user can map 2D and 3D regions of interest or create paint masks in a few simple steps using the Advanced Paint and Power Paint tools.

–Load Advanced Paint Tools

–Load Power Paint Tools

Advanced paint

This tool has been designed to provide labeled masks to associate with structures to delineate, and to perform as seamless as possible the switch between paint and masks.

Power Paint tool

The Power Paint tool has three options

- Object processing, to delete entire regions of paint, and fill any hole (in 2D or 3D images, with any specified connectivity)
- Morphology, to erode or dilate the paint mask (in 3D, 2.5D, and 2D images)
- Import/Export, to quickly transform paint to VOIs or masks

The Power Paint tool works either on the original image or its triplanar viewer.

To run the Advanced Paint tool,

1. Open an image of interest.
2. Go to the image slice that you want to edit.
3. On the Paint toolbar, click the Load Advanced Paint Tools icon;
4. This opens the Multiple Paint Tools dialog box.
5. Here, use the dialog box options to create a one or multiple paint masks (paints). For dialog box options, refer to Figure 193 on page 354.
6. Draw the paint mask by dragging the mouse cursor around the region of interest.
7. If you make a mistake, press the Erase Paint from Current Frame icon located on the Paint toolbar. This will erase your mask from the current slice only.
8. Repeat steps 5–7 until you have drawn the mask on the region of interest on all of the slices.
9. Choose Save masks.
10. Open the next image you wish to view. Repeat until you have collected region of interest information for all of the images you want to process.

MULTIPLE PAINT TOOLS DIALOG BOX OPTIONS

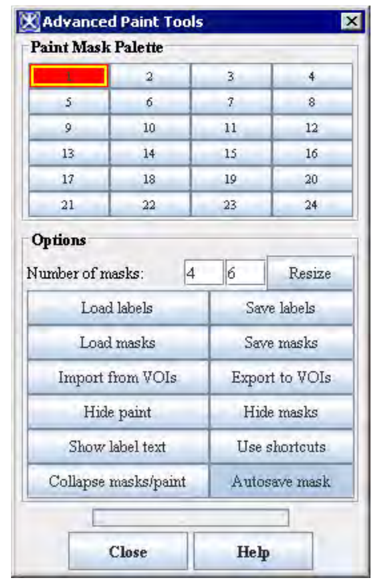
Paint mask palette	Advanced Paint uses a color palette that contains a limited number of colors and each color is described explicitly in the palette. The palette colors themselves are the same as MIPAV uses for stripped LUT. By default, a value from 1 to 24 assigned to each palette button identifies which of the colors in the palette to use. When you click any Paint Mask Palette button, e.g. the button 1, both - the button and the paint brush - appear highlighted with the color assigned to that button. This might remind you of painting by numbers.	
Options		
Number of masks	By default, the Paint Mask Palette shows 24 buttons which represent 24 predefined colors (from 1 to 24). And these 24 colors can be applied to 24 masks.	
	You can change the number of displayed colors by changing the size of the color button. In order to do that, simply enter desired numbers in the Number of Masks control, and then press Resize. For example,  will display 4*6=24 palette buttons  will display 4*8=32 palette buttons	
Changing a mask number	You can change the number assigned to the palette button or the mask number by right clicking on the palette button. This opens the Change Mask Number dialog box. In the dialog box, enter the mask number which you would like to assign to the mask, and then press Change. The mask with the new number appears in the same place on the Paint Mask Palette.	

Figure 193. The Multiple Paint dialog box options

Show Label text	Expands the dialog box so that it shows paint colors and corresponding labels. You can edit the label text (by default, all labels are marked as "Label n" where n is a number of a color) and also select which labels to save. And you can also lock the paint, so that it cannot be overwritten by the Paint to Mask operation. 
Save masks	This option allows to save a paint mask(s) to a file. It is useful to give the paint mask the name that is similar to the image file name (e.g. if the image file is x.img, then choose for the mask the name like x*.xml).
Load masks	Loads the saved mask(s) from a file.
Import from VOIs	Creates mask(s) from VOI(s) delineated on the image.
Export to VOIs	Exports mask(s) to VOI.
Hide paint	Hides the current paint.
Hide masks	Hides masks.
Collapse mask/ paint	This button allows you to quickly change a mask to a paint and vise versa.
Use shortcuts	This option activates hot keys which you can use to quickly access the dialog box options. The hot keys are as follows: • Numbers from **1** to **9** can be used to switch between the first 9 paint masks • Use **t** to show the label text • Use **c** to show or hide the current paint • Use **v** to hide or display mask(s) painted on the image • Use **s** to save the mask(s)
Autosave mask	If activated, this option automatically saves (every 5 minutes) an active paint in the file named **active_mask_autosave.xml**; this file is stored in the image catalogue. It also saves all paints (masks) that you painted on the image in the **multipaint_mask_autosave.xml** file; this file is also stored in in the image catalogue.This allows you to preserve all paints and masks that you put onto the image. Note that a paint become a mask automatically as soon as you choose another paint color in the Palette.
Close	Closes the dialog box.

Figure 193. The Multiple Paint dialog box options (continued)

Help	Displays the help for the dialog box.

Figure 193. The Multiple Paint dialog box options (continued)

Tip: You can paint over any pre-existing mask without changing the base mask if you choose another color number on the palette, and save it as a separate mask.

To run the Power Paint tool,

1. Open an image of interest.
2. For 3D images, use the Image Slice Slider to display the slice that you want to edit.
3. On the Paint toolbar, click the Load Power Paint Tools icon.
4. This opens the Paint Power Tools dialog box.
5. Draw the paint mask by dragging the mouse cursor around the region of interest.
6. If you make a mistake, press the Erase Paint from Current Frame icon located on the Paint toolbar; or use the Erosion button. This will erase your mask from the current slice only.
7. Complete the Paint Power Tools dialog box options.
8. Repeat steps 5–7 until you have drawn all the masks that you need;
9. Choose Save masks.
10. Open the next image you wish to view. Repeat until you have collected region of interest information for all of the images you want to process.

Paint Power Tools dialog box options

Object processing

Figure 194. The Paint Power Tools dialog box

The object processing can be done for 2D and 3D images. To specify the type of the image, use the corresponding radio button.

Grow Region option can be used to grow paint inside a region of interest (e.g., VOI) of a mask. This tool works similar to the usual region growing tool, but it uses only one intensity value and the specified connectivity.

Fill Background – allows to fill the background.

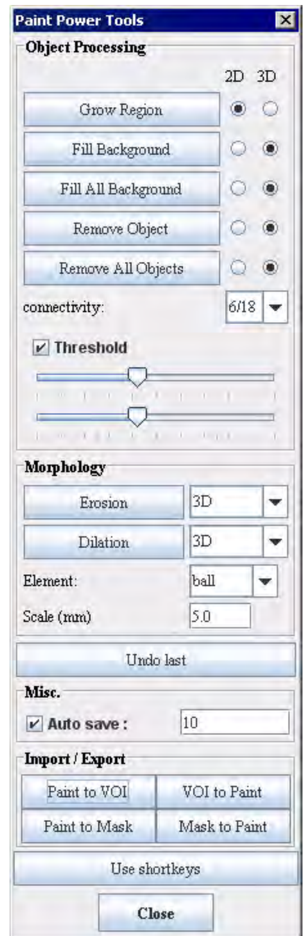
Fill All Background –TBD.

Remove object – removes a selected object.

Remove All Objects – removes a selected object.

Connectivity - TBD.

Threshold specifies the upper and lover threshold for the image intensities to paint on. By default, you can paint everywhere, but if you move the sliders further in, you can see that any new paint you add will not affect the lowest or highest image intensities.



Morphology	Morphology operations include erosion and dilation. Dilation causes objects to dilate or grow in size; erosion causes objects to shrink. The amount and the way that they grow or shrink depend upon the choice of the structuring element (ball, diamond, or cube) which you can select from the Element list box. You must also specify the dimensionality of the morphological operation - 3D, 2D or 2.5D.
Erosion and dilation options	3D, 2.5D, 2D, triplanar, 2.5D(XY), 2.5D(XZ), 2.5D(YZ),2D(XY), 2D(XZ), 2D(YZ)
Element	Choose among a ball, diamond, and cube.
Scale (mm)	TBD.
Undo last	Undoes the last operation.
Misc.	Auto Save – allows auto saving every specified time period.
Import/Export	
Paint to VOI	Transforms a paint to a VOI.
VOI to Paint	Transforms a VOI to paint.

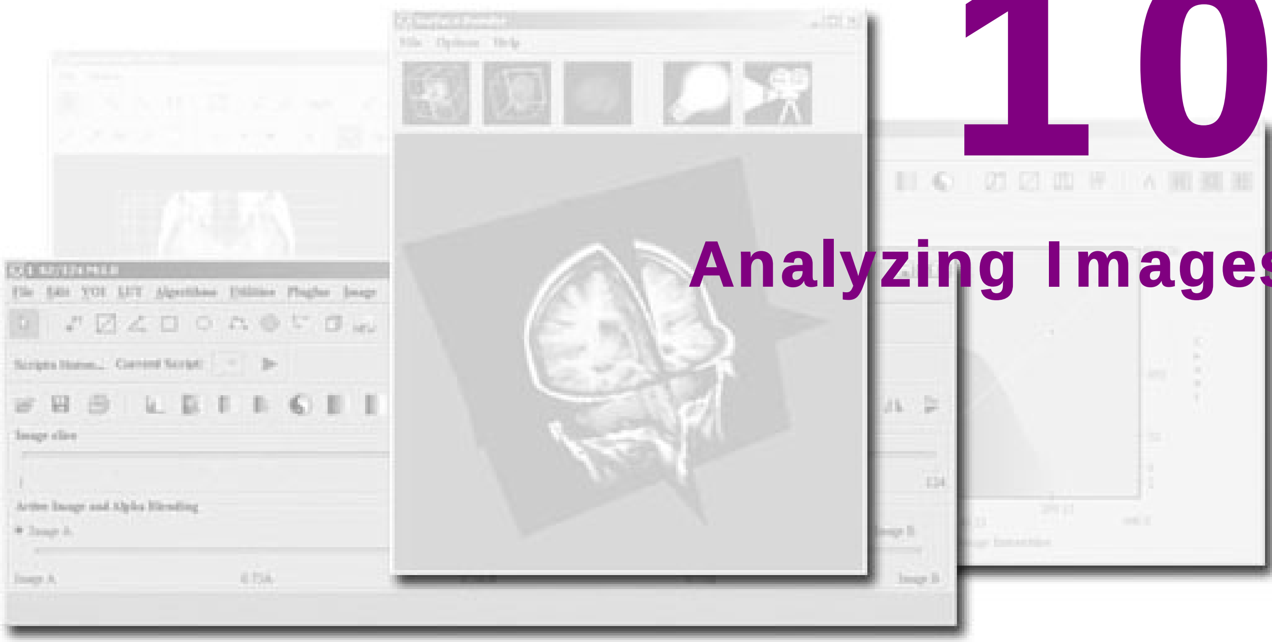
Figure 194. The Paint Power Tools dialog box (continued)

Paint to Mask	transforms a paint to mask
Mask to Paint	transforms a mask back to paint
Use short-keys	Allows using short-keys. The short-keys are **g** for grow, **f** for fill, **r** for remove, **d** for dilate, and **e** for erode.

Figure 194. The Paint Power Tools dialog box (continued)

10

Analyzing Images



In this chapter . . .

You can use MIPAV to generate statistics on contoured volume of interest (VOI) regions and to calculate the volume of painted pixels and voxels. You can also use MIPAV’s algorithms to perform more sophisticated image analysis. For information about algorithms, refer to Volume 2, *Algorithms*.

Calculating statistics for contoured VOIs

Once you have contoured structures, you can generate statistics on the VOI.

To select the type of statistics to calculate for a single VOI

1 Open an image.

2 Contour a VOI. An example of a contoured VOI is shown below (Figure 195).

3 Select the VOI.

4 Do one of the following:

- Select VOI > Properties.
- Right-click inside the VOI, which automatically selects it. Then select Properties on the menu (Figure 195).

The VOI Statistics dialog box (Figure 196) opens.

5 Choose the types of statistics that you want the program to calculate by selecting the appropriate check boxes in the Statistics to calculate group.

Refer to Figure 196 for information on each statistic you can select.

6 Click Calculate.

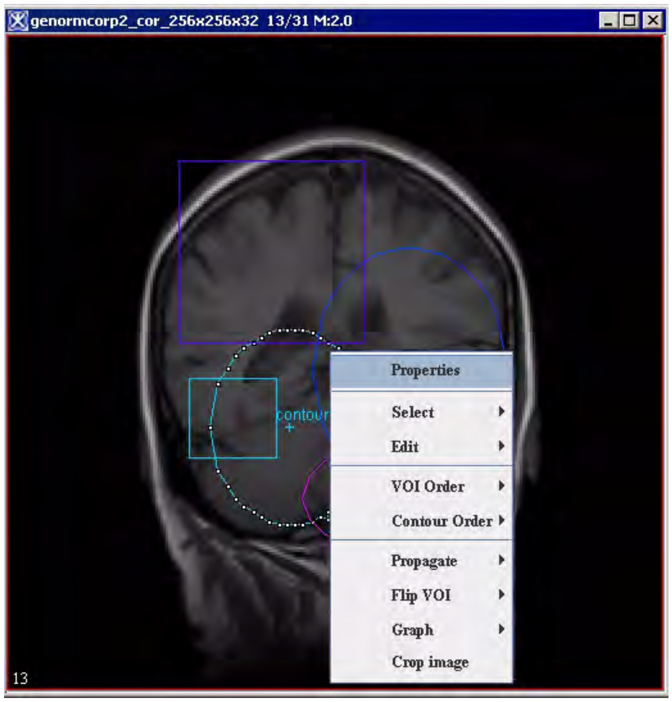


Figure 195. VOI properties context menu

Name of VOI	Indicates the color of the VOI outlines and the name of the VOI. To use this to group two contours, refer to Chapter 7 for more details.
Show bounding box	Indicates whether to show the bounding box. If this check box is selected, the bounding box appears around the VOI.
Use Additive polarity for VOI	TBD.
Include for processing	Indicates whether to include the VOI in the processing when running an algorithm. If selected, the VOI is included.
Boundary or blended	Indicates whether the contour is filled with color (blended) or transparent (only the boundary appears). If selected, the VOI is filled with color.
Opacity	Indicates whether a VOI that is filled with color is transparent, translucent, and opaque: 0 is transparent, and 1 is opaque.
Statistics to calculate	**# of voxels**—Indicates the number of voxels, including voxels that span frames in an image stack, that are enclosed in the VOI.

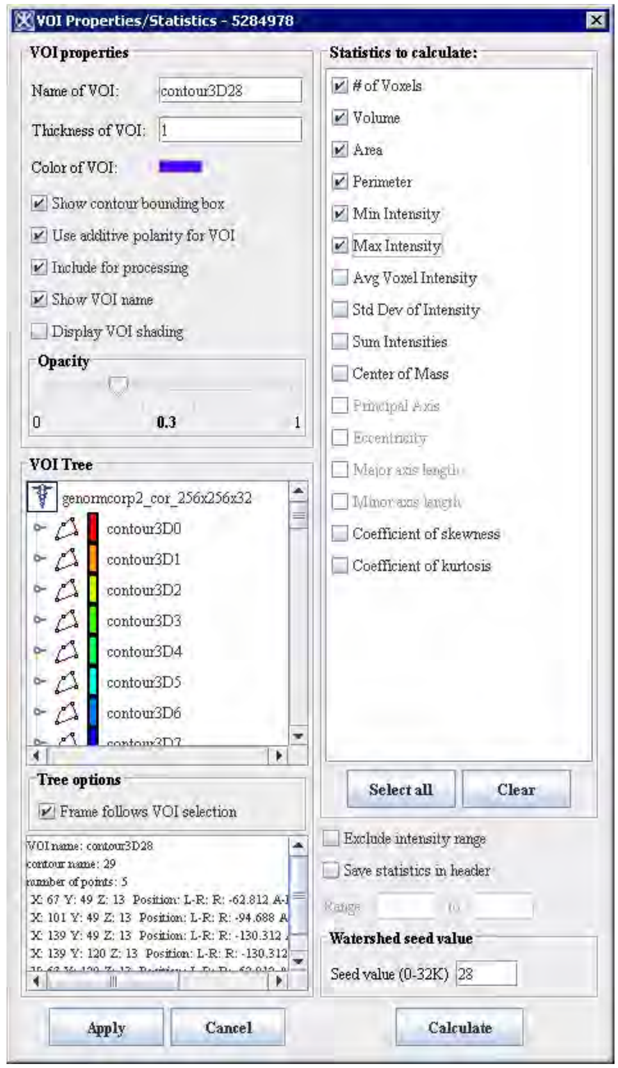


Figure 196. VOI Statistics dialog box

	Volume—Measures the amount of space occupied by a 3D VOI. To calculate the volume, the software multiplies the number of pixels by the resolution of each dimension. **Area**—Measures the surface of the VOI. To calculate the area, the software multiples the number of pixels by the resolutions of the *x* and *y* dimensions. **Perimeter** - measures a perimeter of VOI. **Min. Intensity** – shows the min voxel intestacy. **Max. Intensity** – shows the max voxel intensity. **Average voxel intensity**—Calculates the average intensity of the voxels in the VOI by adding the intensity of all voxels in the VOI and dividing the result by the sum of the voxels. **Std. dev. of voxel intensity**—Calculates the standard deviation of the intensity of the voxels in the VOI. **Center of mass**—Indicates the point at which the whole mass of the VOI is concentrated. It is calculated as the sum of all *x* coordinates divided by the number of points and the sum of all *y* coordinates divided by the number of points. **Principal axis (only 2D)**—Calculates the principal axis for 2D images only. **Eccentricity (only 2D)**—Describes the geometric shape of the VOI as an ellipse, with 0 indicating a circle and 1 indicating a straight line. **Major axis length** — calculates the length of the major axis for an elliptical VOI. **Minor axis length** — calculates the length of the minor axis for an elliptical VOI.
Select all	Selects all of the statistical measures in Statistics to calculate.
Clear	Clears all of the statistical measures in Statistics to calculate.
Exclude intensity range	Excludes a specific range of intensity values, which you specify in the Range boxes, in the calculation. When this check box is selected, the Range boxes become available.
Range	Specifies a particular range of intensities to exclude from the calculations. These boxes become available only when the Exclude intensity range check box is selected.
Watershed seed value (0-32K)	Indicates the watershed seed value.
Apply	Applies the changes you made to this dialog box.
Cancel	Disregards any changes you made in this dialog box and closes the dialog box.
Calculate	Runs all selected statistics according to the specifications in this dialog box.
Help	Displays online help for this dialog box.

Figure 196. VOI Statistics dialog box (continued)

To view the statistics for a single VOI

1 Select the Output window.

2 Select the Data tab on the Output window to view the information. The Data page of the Output window (Figure 197) appears showing the results of the calculations.

To save the statistics for a single VOI

1 Select File > Save Messages in the Output window.

The Save dialog box opens.

2 Specify a name for the messages file and select a location for storing it.

3 Click Save. The software saves the file and places it in the specified location.

Tip: MIPAV provides you with the ability to type directly onto the Data page. So, for example, you could add a description of when the statistics were calculated or any other such meaningful information (Figure 197).

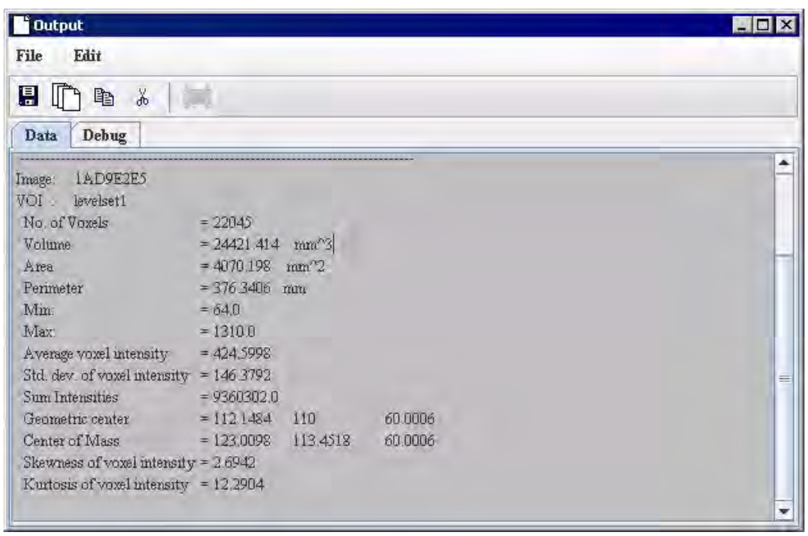


Figure 197. Data page in the Output window showing statistics

Calculating statistics on VOI groups

The Statistics Generator command on the VOI menu in the MIPAV window opens the Calculate Statistics on VOI Groups window, which allows you to obtain statistics on one VOI or on multiple grouped VOIs. You can save the statistics to a tab-delimited file, which can then be incorporated into a database.

The Calculate Statistics on VOI Groups dialog box includes three tabs:

- *VOI Selection page*—On this page you can select the VOIs on which to calculate the statistics. It also allows you to name and save the logging file at the path you choose.
- *Statistics Options page*—This page provides a choice of the types of statistics that may be calculated as well as the conditions under which they may be run.
- *Logging page*—This page provides the results of the statistics in a tabular form. Depending on the number of VOIs included in the calculation, the results may include one or more lines of text.

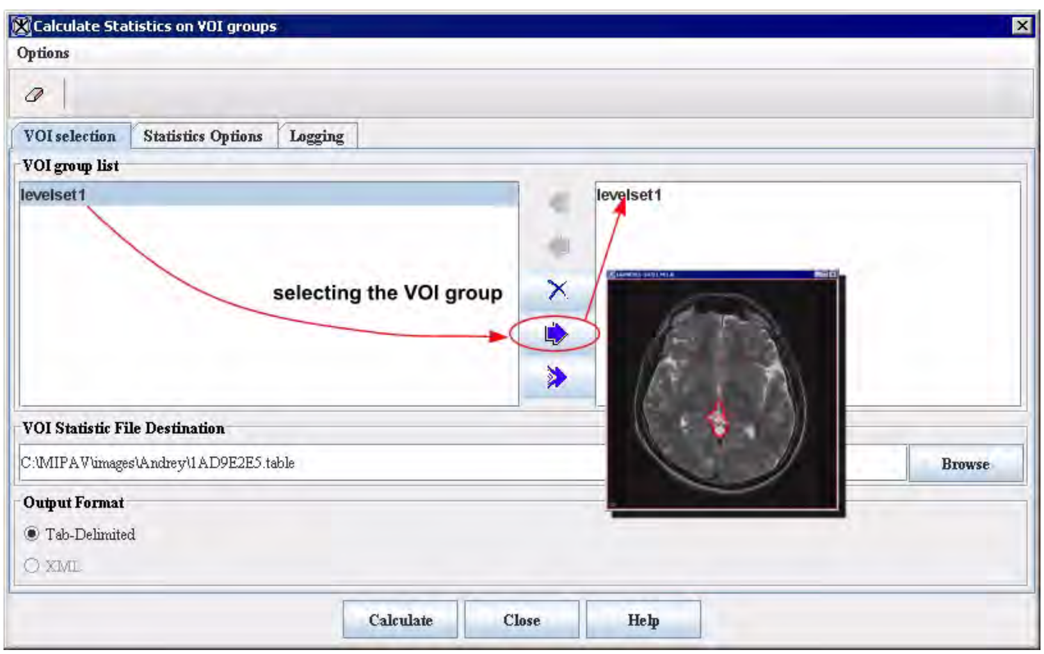


Figure 198. VOI Selection page in Calculate Statistics on VOI groups window

VOI groups list—left	Lists all of the VOIs found on the image.
VOI groups list—right	Lists the VOIs on which you plan to calculate statistics.
Send all left	Moves all of the VOIs that appear in the VOI groups list on the right to the VOI groups list on the left.
Send selection left	Moves the selected VOI that appears in the VOI groups list on the right to the VOI groups list on the left.
Delete selection	Deletes the selected VOI in either the VOI groups list on the left or the VOI groups list on the right.
Send selection right	Moves the selected VOI in the VOI groups list on the left to the VOI groups list on the right on which you plan to calculate statistics.
Send all right	Moves all of the VOIs, whether they are selected or not, in the VOI groups list on the left to the VOI groups list on the right on which you plan to calculate statistics.
VOI statistics file destination	Specifies the file path, file name, and file extension to which you want to save the file.
Browse	Allows you to navigate to the directory on your workstation or attached disks where you want to store the file.
Tab-delimited	Saves the statistics in a comma-separated tab-delimited file. The extension for this file is.TABLE.
XML	In development.
Calculate	Runs all statistics that are selected on the Statistics Options page according to the options specified.
Close	Closes this window.
Help	Displays online help for this window.

Figure 198. VOI Selection page in Calculate Statistics on VOI groups window

To calculate statistics on VOIs

1. Open an image.
2. Delineate one or more VOIs on the image (Figure 198).
3. Select one of the VOIs by clicking it.
4. Hold down the Shift key and select the next VOI.

5 Continue holding down the Shift key and selecting VOIs until all of the VOIs to be included in the calculation are selected.

6 Select VOI > Group VOIs.

7 Select VOI > Statistics Generator. The Calculate Statistics on VOI Groups window opens showing the VOI Selection page.

8 Select the VOI group, which appears in the left VOI groups list.

9 Click the Send selection right button. The group appears on the right VOI groups list (Figure 199).

10 Do one of the following:

- Accept the default path and file name for the logging file, where the statistics are stored, in VOI statistics file destination.
- Update the path and file name for the logging file.

11 Select the format of the logging file in Output format.

12 Select Statistics Options. The Statistics Options page (Figure 200) appears.

13 Select the statistics you want to include in the calculation in the Statistics to calculate list by doing one of the following:

- Click on the individual check boxes for each specific statistic.
- Click Select all to run all of the statistics.

Tip: If you, for example, clicked Select all and then decide that you don't want the software to calculate all of the statistics, it may be faster to click Clear to remove the check marks from *all* of the check boxes. You can then select only the statistics you want to calculate.

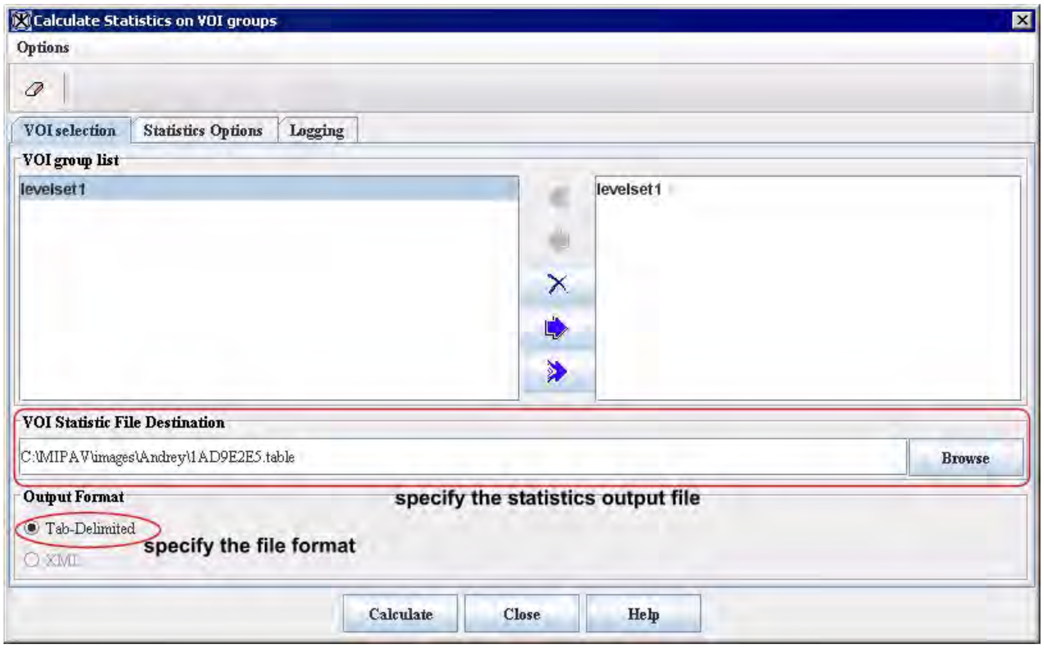


Figure 199. Specifying the output file

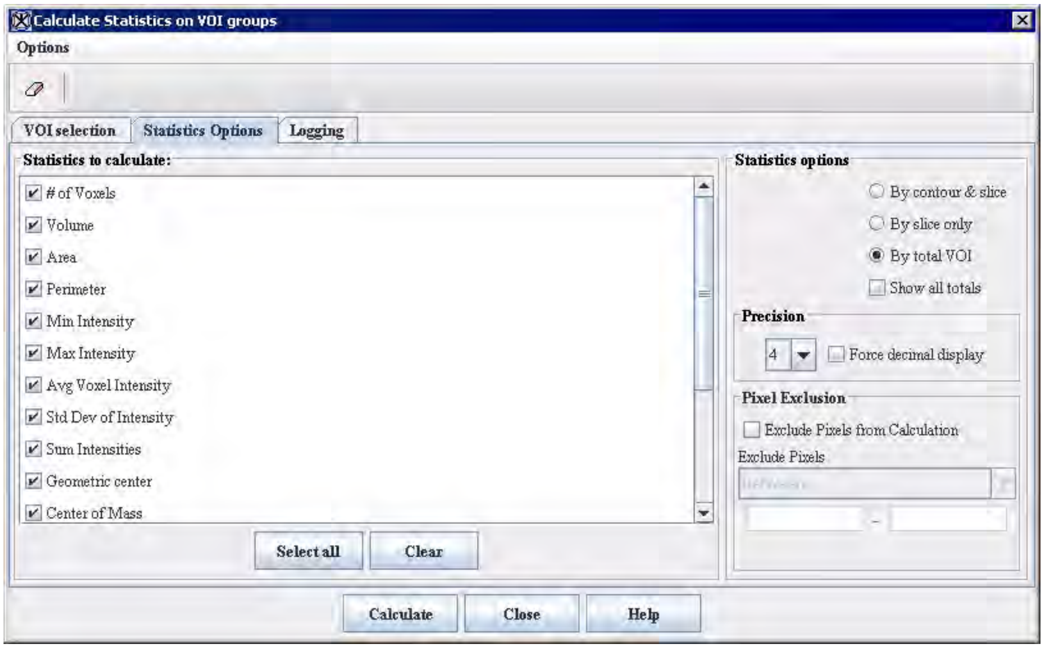
14 Select one of the following options in the Statistics options group:

- By contour & slice
- By slice only
- By total VOI

15 Select Show all totals if appropriate.

16 Click Calculate. The Logging page (Figure 201) appears with the results of the statistics you selected.

17 Click Close when finished to close the Calculate Statistics on VOI Groups window.



Statistics to calculate	**No. of voxels**—Indicates the number of voxels—including voxels that span frames in an image stack—enclosed in the VOI.
	Volume—Measures the amount of space occupied by a 3D VOI. To calculate the volume, the software multiplies the number of pixels by the resolution of each dimension.
	Area—Measures the surface of the VOI. To calculate the area, the software multiplies the number of pixels by the resolutions of the *x* and *y* dimensions.
	Average voxel intensity—Calculates the average intensity of the voxels in the VOI by adding the intensity of all voxels in the VOI and dividing the result by the sum of the voxels.
	Std. dev. of voxel intensity—Calculates the standard deviation of the intensity of the voxels in the VOI.
	Center of mass—Indicates the point at which the whole mass of the VOI is concentrated. It is calculated as the sum of all *x* coordinates divided by the number of points and the sum of all *y* coordinates divided by the number of points.
	Principal axis (only 2D)—Calculates the principal axis for 2D images only.
	Eccentricity (only 2D)—Describes the geometric shape of the VOI as an ellipse, with 0 indicating a circle and 1 indicating a straight line.

Figure 200. Statistics Option page in Calculate Statistics on VOI Groups window

Statistics options	**By contour & slice**—Runs the selected statistics on both the contour and slice.
	By slice only—Runs the selected statistics on only the slice.
	By total VOI—Runs the selected statistics on the entire VOI.
	Show all totals—Specifies to display all of the totals for __________.
Pixel exclusion	**Exclude pixels from calculation**—Indicates to exclude the pixels in the Pixel range boxes from the calculation. When you select this check box, the Exclude pixels button and the Pixel range boxes become available.
	Exclude pixels—Indicates to exclude the pixels between, above, below, or outside the range shown in the Pixel range boxes.
	Pixel range—Lists a specific range of pixels between, above, below, or outside that should be excluded in the calculations.
Select all	Selects all of the statistics listed in the Statistics to calculate group.
Clear	Clears all of the statistics listed in the Statistics to calculate group.
Calculate	Runs all statistics that are selected on the Statistics Options page according to the options specified.
Close	Closes this window.
Help	Displays online help for this window.

Figure 200. Statistics Option page in Calculate Statistics on VOI Groups window

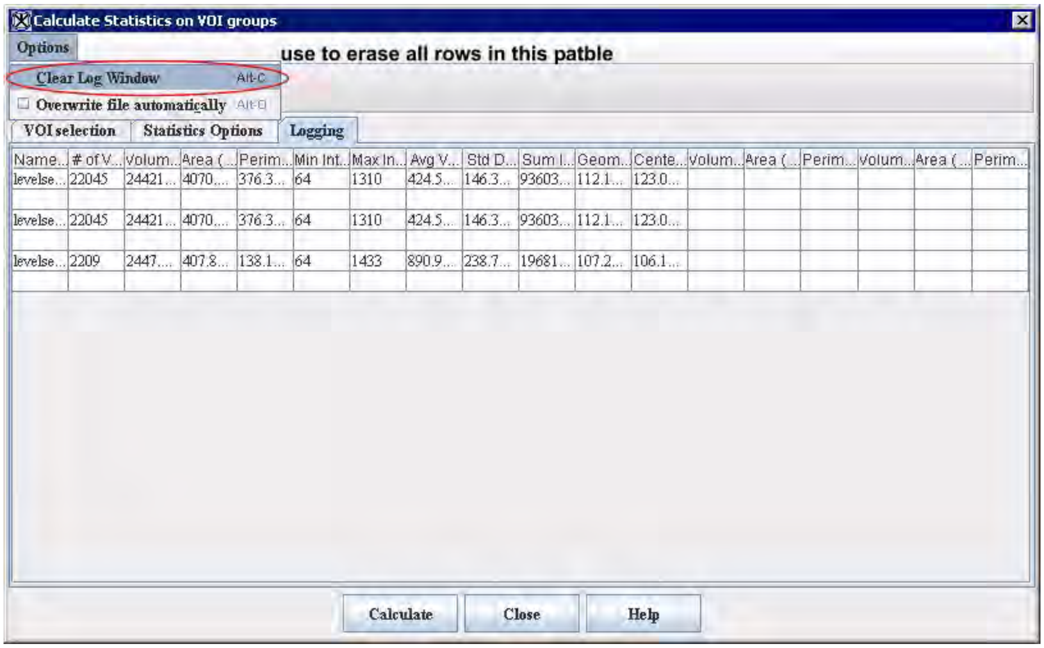
To exclude a range of pixels from the calculations

When you run statistics on VOIs in an image but want to exclude a specific range of pixels from the calculations, make sure that you do the following:

1. Select Exclude pixels from calculation on the Statistics Options page in the Calculate Statistics on VOI Groups window. The Exclude pixels list and the Pixels range boxes become available.
2. Select Between, Above, Below, or Outside in the Exclude pixels list.
3. Type a range of pixels in the Pixel range boxes.

4 Continue to select statistics and other options for the calculation.

5 Click Calculate.



Statistics table	Lists all of the statistics that were calculated according to the selected statistics and the options specified on the Statistics Options page.
Calculate	Runs all statistics that are selected according to the options specified on the Statistics Options page.
Close	Closes the window.
Help	Displays online help for this window.

Figure 201. Logging page in the Calculate Statistics on VOI Groups window

To save the calculations to a specific file

By default, the software saves the results of the calculations to the following path and file name:

```
c:\Documents and Settings\[username]\voistatistics.table
```

However, you can easily change the file name by typing over the default file name in the VOI statistic file destination box on the VOI Selection page in the Calculate Statistics on VOI Groups window. To change the path, type over the current path, or click Browse to select another path.

To use the calculations in a database application

Saving the statistics in a tab-delimited file allows you to import them into a database application.

1. Make sure—before running the calculations—that you select the Tab-delimited check box on the VOI Selection page in the Calculate Statistics on VOI Groups window.
2. Change the path and file name, if necessary, in the VOI Statistics File Destination box on the VOI Selection page.
3. Click Calculate. The resulting statistics appear on the Logging page and in the file you specified.

To rearrange the columns in the logging table

You can adjust the order in which the columns in the logging table appear as well as their width. To change the order, simply click on the title of a column and drag the column to the new position. To make a column wider, click on the line between it and the next column and drag it to the desired width.

To compare statistics

If you calculate statistics on an image more than once in a MIPAV session, the Logging page shows the previously run calculations for the VOIs in that image. For example, suppose you calculate statistics on the VOIs in an image and then save the image and close it. Later, you reopen the image and decide to run additional statistics on the previous VOIs or on any new VOIs. As long as you did not exit MIPAV and are therefore in the same session, the Logging page in the Calculate Statistics on VOI Groups window still displays the results of the statistics you previously ran on the VOIs in the image.

When you perform additional calculations on the same or new VOIs on that image or on VOIs in another image, the results of those calculations appear below the earlier run statistics. This allows you to compare the results of both.

To clear, or delete, all of the statistics on the Logging page

When you no longer need the statistics on the Logging page or after you saved them to a file, you can clear, or delete, all of the statistics on the Logging page. To do so, do one of the following:

- Select Options > Clear logging page in the Calculate Statistics on VOI Groups window.
- Press Alt C.
- Select Clear on the Statistics Options page.

The software deletes all of the statistics on the Logging page.

To overwrite logging files automatically

If you ran calculations previously in your current MIPAV session and then select Calculate in the Calculate Statistics on VOI Groups window, the File Exists message appears asking whether to overwrite the previously saved logging file or cancel the action. If you select Overwrite, the software replaces the previously saved file with the new statistics. Selecting Cancel means that MIPAV does not perform any calculations.

Figure 202. File Exists message

Recommendation: It is recommended that you decide in advance on a standard procedure for handling statistics files. That is, do you need to keep them for historical purposes? Do you need to keep statistics on individual images in separate files? If so, you may want to assign unique names to the statistics file for each image. If not, you may want to simply overwrite the statistics file each time you calculate statistics of VOI groups.

If you do not want to keep the statistics in separate files, it may become tiresome to always receive and need to respond to the File Exists message. To this end, the software offers an option for always overwriting the logging files. To use this option, simply select Options > Overwrite File Automatically in the Calculate Statistics on VOI Groups window.

Tip: To turn the overwrite command on, press Alt O on the keyboard. To turn it off, press Alt O again.

Calculating the volume of masks

You can calculate the volume of painted voxels, or mask, in an image and view the information about the volume on the Data page of the Output window.

To calculate the volume of masks

1. Open an image, and paint the voxels in a desired area of the image.
2. Follow the instructions for manually creating a mask in "Generating masks" on page 331 in Chapter 9, "Segmenting Images Using Contours and Masks."
3. *Do not* commit the paint, which permanently merges the paint layer with the image layer.
4. Click the Calculate Volume of Paint icon, in the Paint toolbar.

   The region grow information (how many voxels or pixels are painted) and the volume of all painted voxels appear on the Data page of the Output window (Figure 203).

To view the information

1 Select the Output window.

2 Select the Data tab on the Output window to view the information. The Data page of the Output window (Figure 203) appears.

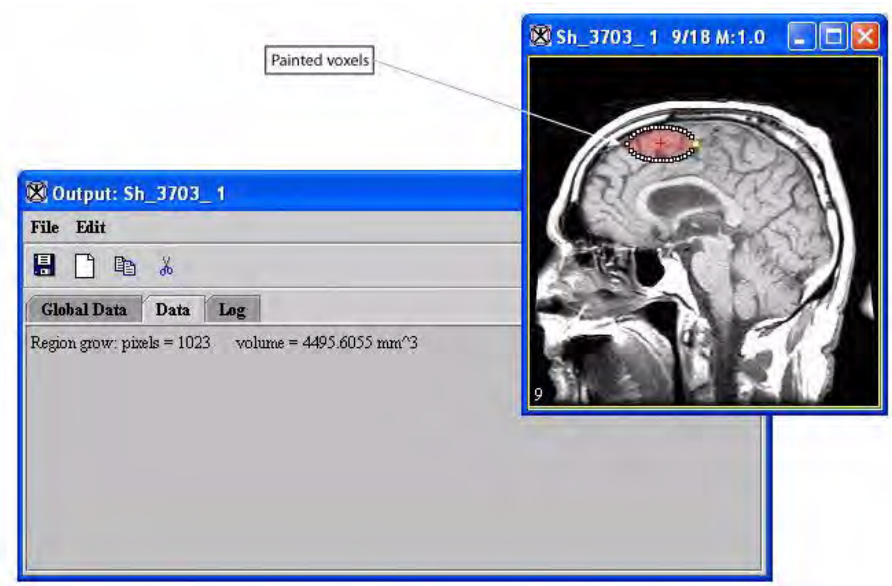


Figure 203. Data page in the Output window (left) listing the region grow and volume information from the painted voxels in the image (right)

Remember: You can type directly onto the Data page if you want to record such information as the date and time the calculations were run.

To save the information

1 Select File > Save Messages in the Output window.

The Save dialog box opens.

2 Specify a name for the messages file and select a location for storing it.

3 Click Save. The software saves the file and places it in the specified location.

Generating graphs

MIPAV allows you to generate intensity profiles, or contour VOI graphs, for VOI contours. For delineated VOIs, you can generate 2D, 3D, or 4D intensity graphs. You can also generate a 3D intensity graph at a specific point across all slices in a dataset. For information on how to contour a VOI, refer to Chapter 9, "Segmenting Images Using Contours and Masks."

Generating contour VOI graphs

Contour VOI graphs display the intensity values of the selected contour's boundary in the Contour VOI Graph window (Figure 204). You can generate either 2D or 3D contour VOI graphs.

To generate 2D contour VOI graphs

1 Open an image.

2 Delineate a 2D VOI on the image using one of the 2D icons in the MIPAV window.

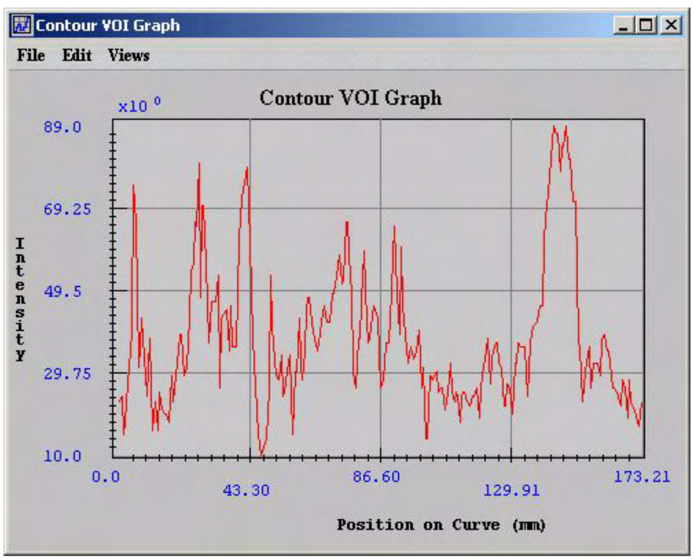


Figure 204. Contour VOI Graph window

File	**Open Graph**—Opens a PLT file that contains graph data. When you select this command or press Ctrl O on the keyboard, the Open Graph Data dialog box appears.
	Save Graph—Saves the graph data in a PLT file. When you select this command or when you press Ctrl S on the keyboard, the Save dialog box opens.
	Print Graph—Allows you to print the graph. When you select this command or press Ctrl P, the Print dialog box opens.
	Close Graph—Closes the Intensity Graph window. To close the window, you can also press Ctrl X on the keyboard.
Edit	**Delete Function**—Allows you to delete a specific function. However, you cannot delete a function if it is the only function displayed in the window.
	Copy Function—Copies a function that is currently displayed in the window.
	Paste Function—Pastes a previously copied function into the window. The pasted function has a different color than the first function displayed in the window.
Views	**Modify Graph Features**—Allows you to customize the appearance of the graph.
	Reset Range to Default—[TBD]
	Reset Graph to Original—[TBD].
Help	**Help Topics**—Displays online help topics.

Figure 204. Contour VOI Graph window (continued)

3 Select the VOI.

As an option, copy the VOI to other slices in the dataset by selecting VOI > Propagate and one of the following commands:

- To Next Slice
- To Previous Slice
- To All Slices

4 Do one of the following:

- Select VOI > Graph > Boundary Intensity in the MIPAV window.
- Right click on the VOI and then select Graph > Boundary Intensity.

The Contour VOI Graph window (Figure 204) opens.

To generate 3D contour VOI graphs

1 Open an image.

2 Delineate a VOI on the image using the 3D rectangular VOI icon, in the MIPAV window.

3 Select the VOI.

As an option, copy the VOI to other slices in the dataset by selecting VOI > Propagate and one of the following commands:

- To Next Slice
- To Previous Slice
- To All Slices

4 Do one of the following:

- Select VOI > Graph > Boundary Intensity in the MIPAV window.
- Right-click on the VOI and then select Graph > Boundary Intensity.

The Contour VOI Graph window (Figure 204) opens. This window displays a graph of the intensity values of the selected contour's boundary.

Generating intensity graphs

Intensity profiles, or graphs, present information on the intensity values of the VOI region in an image. The intensity graph appears in the Intensity Graph window (Figure 205).

To generate 2D intensity graphs

1 Open an image.

2 Delineate a 2D VOI on the image using one of the 2D icons in the MIPAV window.

3 Select the VOI.

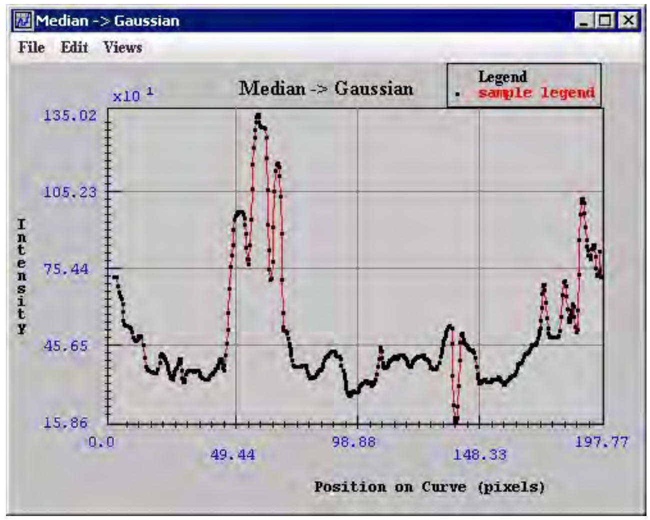
As an option, copy the VOI to other slices in the dataset by selecting VOI > Propagate and one of the following commands:

- To Next Slice
- To Previous Slice
- To All Slices

4 Do one of the following:

- Select VOI > Graph in the MIPAV window and either of the following:
  - *2.5D Total Intensity*—To generate a graph of the sum of the intensity values of the VOI region.
  - *2.5D Average Intensity*—To generate a graph of the average of the intensity values of the VOI region.
- Right-click on the VOI and then select Graph and one of the following commands:
  - *2.5D Total Intensity*—To generate a graph of the sum of the intensity values of the area delineated by the VOI per slice.
  - *2.5D Average Intensity*—To generate a graph of the average of the intensity values of the VOI region.
  - *2.5D Total Intensity with Threshold*—TBD.
  - *2.5D Average Intensity with Threshold*—TBD.

The Intensity Graph window (Figure 205) opens.



File	**Open Graph**—Opens a PLT file that contains graph data. When you select this command or press Ctrl O on the keyboard, the Open Graph Data dialog box appears.
	Save Graph—Saves the graph data in a PLT file. When you select this command or when you press Ctrl S on the keyboard, the Save dialog box opens.
	Print Graph—Allows you to print the graph. When you select this command or press Ctrl P, the Print dialog box opens.
	Close Graph—Closes the Intensity Graph window. To close the window, you can also press Ctrl X on the keyboard.
Edit	**Delete Function**—Allows you to delete the function that you select. However, you cannot delete a function if it is the only function displayed in the window.
	Paste Function—Pastes a previously copied function into the window. The pasted function has a different color than the first function displayed in the window.
Views	**Modify Graph Features**—Allows you to customize the appearance of the graph.
	Reset Range to Default—TBD.
	Reset Graph to Original—TBD.
Help	**Help Topics**—Displays online help topics.

Figure 205. Intensity Graph window

To generate 3D intensity graphs of all slices in a dataset at a specific point

1 Open an image.

2 Draw a point VOI on the image (Figure 206).

3 Select the VOI.

4 Do one of the following:

- Select the Propagate VOI to all slices icon.
- Select VOI > Propagate > To All Slices.
- Right-click on the VOI, then select Propagate > To All Slices (Figure 206).

5 Right-click on the VOI and select Show VOI Graph (Figure 206).

The Intensity Graph window (Figure 205 on page 379) opens.

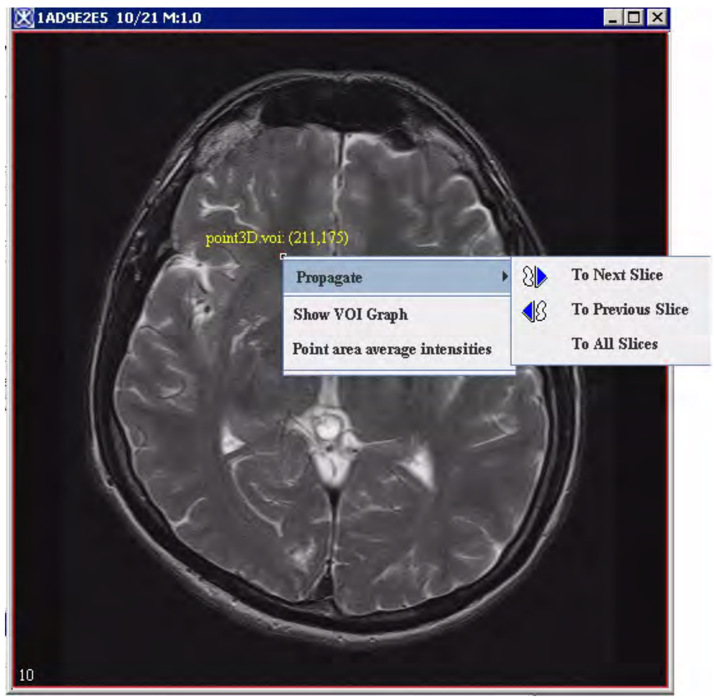


Figure 206. Point VOI

To generate 3D intensity graphs of specific areas

1 Open an image.

2 Delineate a VOI on the image using the 3D rectangular VOI icon.

3 Select the VOI. Then, do one of the following:

a Select VOI > Graph and either of the following in the MIPAV window:

- *2.5D Total Intensity*—To generate a graph of the sum of the intensity values of the area delineated by the VOI per slice.

- *2.5D Average Intensity*—To generate a graph of the average of the intensity values of the VOI region.

b Right-click the VOI, and then select Graph and one of the following commands in the MIPAV window:

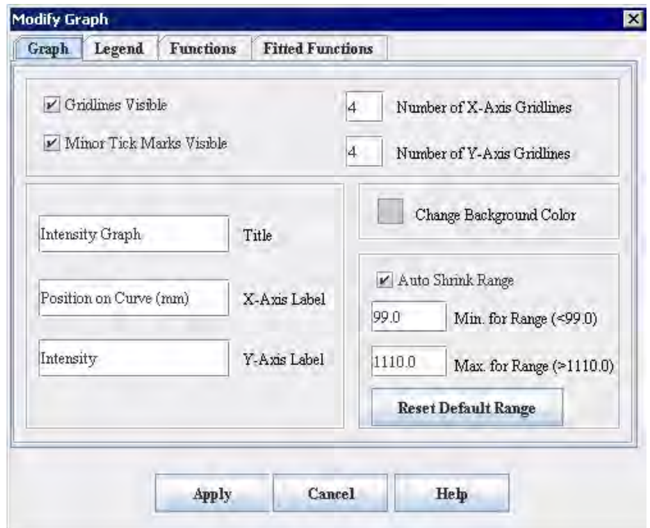
- *2.5D Total Intensity*—To generate a graph of the sum of the intensity values of the area delineated by the VOI per slice.
- *2.5D Average Intensity*—To generate a graph of the average of the intensity values of the VOI region.
- *2.5D Total Intensity with Threshold*—TBD.
- *2.5D Average Intensity with Threshold*—TBD.

The Intensity Graph window (Figure 205 on page 379) opens.

Customizing the appearance of graphs – Modify graph dialog box

You can adjust the appearance of graphs to interpret the information more clearly or easily. To customize graphs, you use the Modify Graph dialog box (Figure 207), which is accessible through Views > Modify Graph Features in both the Contour VOI window (Figure 204) and the Intensity Graph window (Figure 207). The Modify Graph dialog box includes the following four tabbed pages:

- *Graph tab* (Figure 207), which allows you to show or hide the gridlines and tick marks, change the number of gridlines and background color of the graph, change the labels on the graph, and change the range values.
- *Legend tab* (Figure 215), which lets you determine whether a legend should appear on the graph and allows you to assign a specific name to each function.
- *Functions tab* (Figure 217), which allows you to change the appearance of the functions in the graph.
- *Fitted Functions tab* (Figure 219), which allows you to modify the functions.



Graph tab	
Gridlines visible	Displays, if selected, gridlines on the graph in the Intensity Graph window.
Minor tick marks visible	Displays, if selected, the tick marks on the *X* and *Y* axes of the graph in the Intensity Graph window.
Number of X-axis gridlines	Indicates the number of gridlines that appear on the *X* axis of the graph. The default number of gridlines is 4. You can specify from 1 to 50 gridlines.
Number of Y-axis gridlines	Indicates the number of gridlines that display on the *Y* axis of the graph. The default number of gridlines is 4. You can specify from 1 to 50 gridlines.
Title	Specifies the name of the graph. By default, the name is *Intensity Graph.* However, you can replace this name with any name you choose.
***X* axis label**	Specifies the name that appears for the *X* axis. By default, the name is *Position on curve (mm).*
***Y* axis label**	Specifies the name that appears for the *Y* axis. By default, the name is *Intensity*.
Change background color	Allows you to choose the color of the background of the graph. When you select this icon, the Pick Background Color dialog box opens. By default, the background color for graphs is light gray. The HSB values are 0 hue, 0 saturation, and 100 brightness and the RGB values are 255 red, green, and blue.
Auto Shrink Range	Check/uncheck this box to expand or shrink the graph.
Min. for range	The default minimum range is 207,437.0. If you specify another range, it must be less than the default range.
Max for range	The default maximum range is 595,910.0. If you specify another range, it must be more than the default range.

Figure 207. The Graph page of the Modify Graph dialog box

Reset default range	Erases the current minimum and maximum ranges and replaces the values with the default minimum and maximum ranges.
Apply	Applies the changes you made in this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not change the threshold.
Help	Displays online help for this dialog box.

Figure 207. The Graph page of the Modify Graph dialog box (continued)

To display or hide the points, or tick marks, on graphs

1. Open an image.
2. Delineate a VOI on the image.
3. Generate an intensity profile (refer to "Generating graphs" on page 375). Either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379) opens.
4. Select Views > Modify Graph Features in either the Contour VOI Graph window or the Intensity Graph window. The Graph page of the Modify Graph dialog box (Figure 207 on page 382) appears.
5. Do one of the following:
   - If you want to make the points appear on the graph, select Minor tick marks visible. A check mark appears in the check box.
   - If you want to make the points invisible, clear Minor tick marks visible.

6 Click Apply.

- If you selected **Minor tick marks visible**, the tick marks along the *X* and *Y* axes in the graph appear.
- If you cleared **Minor tick marks visible**, the tick marks disappear from the graph.

7 Do one of the following:

- Close the Modify Graph dialog box.
- Keep the Modify Graph dialog box open to continue modifying the graph.

To display or hide the gridlines on graphs

1 Open an image.

2 Delineate a VOI on the image.

3 Generate an intensity profile (refer to "Generating graphs" on page 375). Either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379) opens.

4 Select Views > Modify Graph Features in either Contour VOI Graph window or the Intensity Graph window. The Graph page of the Modify Graph dialog box (Figure 207 on page 382) appears.

5 Do one of the following:

- To make the gridlines appear on the graph, select, if not already selected, Gridlines visible. A check mark appears in the check box.
- To make the gridlines disappear from the graph, clear Gridlines visible. The check box is empty.

Figure 208. Displaying or hiding the gridlines and tick marks on the graph

6 Click Apply.

- If you selected Gridlines visible, horizontal and vertical gridlines appear in the graph.
- If you cleared the check box, the gridlines disappear from the graph.

7 Do one of the following:

- Close the Modify Graph dialog box.
- Keep the Modify Graph dialog box open to continue modifying the graph.

To change the number of gridlines in graphs

1 Select Views > Modify Graph Features in either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379). The Graph page of the Modify Graph dialog box (Figure 207 on page 382) appears.

2 Change the number of *X*-axis gridlines in Number of *X*-axis gridlines (Figure 209) by specifying a value from 1 to 50. The default number of gridlines is 4.

3 Change the number of *Y*-axis gridlines in Number of *Y*-axis gridlines (Figure 209) by specifying a value from 1 to 50. The default number of gridlines is 4.

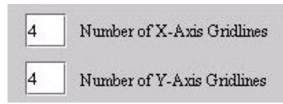


Figure 209. Number of *X*-axis and *Y*-axis gridlines

4 Click Apply.

The number of horizontal and vertical gridlines that you specified appear in the graph in the Intensity Graph window.

Tip: If you specify many gridlines, to see the gridlines more clearly, resize the Modify Graph dialog box.

5 Do one of the following:

- Close the Modify Graph dialog box.
- Keep the Modify Graph dialog box open to continue modifying the graph.

To change the graph title and labels on the *X* and *Y* axes

1 Open an image.

2 Delineate a VOI on the image.

3 Generate an intensity profile (refer to "Generating graphs" on page 375). Either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379) opens.

4 Select Views > Modify Graph Features in Contour VOI Graph window or the Intensity Graph window. The Graph page of the Modify Graph dialog box (Figure 207 on page 382) appears.

5 Type the new title and axes labels in Title, *X*-axis label, and *Y*-axis label boxes (Figure 210).

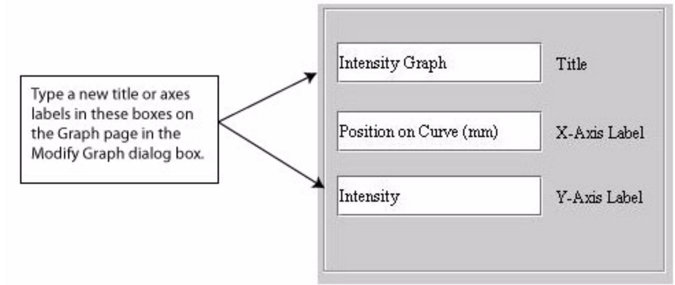


Figure 210. Title, *X*-axis label, and *Y*-axis label boxes in the Modify Graph dialog box

6 Click Apply.

The new title appears number of horizontal and vertical gridlines that you specified now appear in the graph (Figure 211).

Tip: If you specify many gridlines, you might want to resize the Modify Graph dialog box to see the gridlines more clearly.

---

7 Do one of the following:

- Close the Modify Graph dialog box.
- Keep the Modify Graph dialog box open to continue modifying the graph.

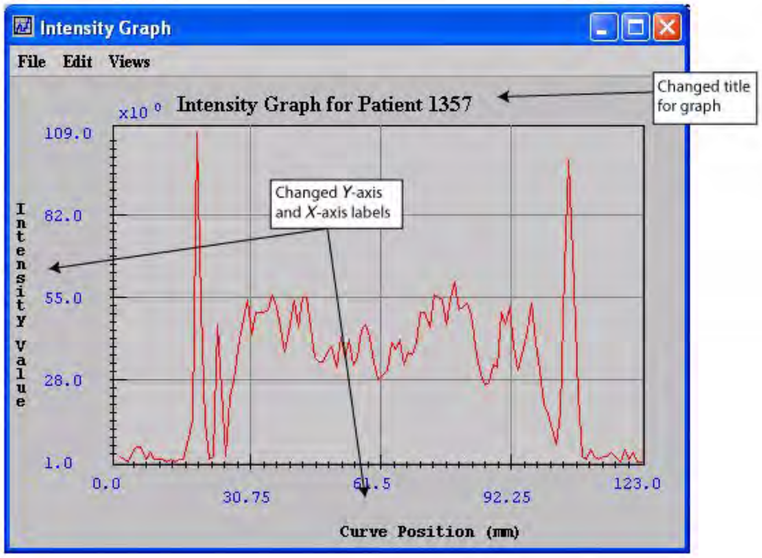


Figure 211. Changed title and axes labels in the Intensity Graph window

To change the background color of graphs

1 Open an image.

2 Delineate a VOI on the image.

3 Generate an intensity profile (refer to "Generating graphs" on page 375). Either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379) opens.

4 Select Views > Modify Graph Features in either the Contour VOI Graph window or the Intensity Graph window. The Graph page of the Modify Graph dialog box (Figure 207 on page 382) appears.

5 Select Change background color (Figure 212).

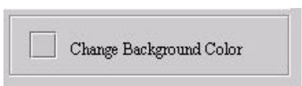


Figure 212. Change background color button

The Pick Background Color dialog box (Figure 213) opens.

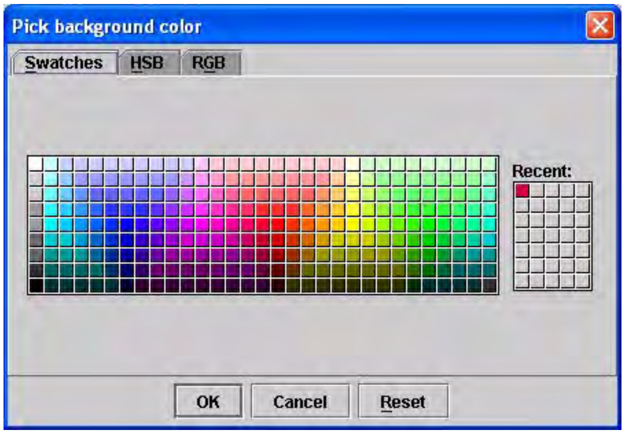


Figure 213. Pick Background Color dialog box

6 Select a color from one of the three pages in this dialog box. Refer to "To change background and border colors" on page 263 for information on how to select a color. Refer to "To change background and border colors" for information on how to select a color.

7 Click OK to apply the color to the graph background. The Pick Background Color dialog box closes, and the graph background (Figure 214) changes to the color you selected.

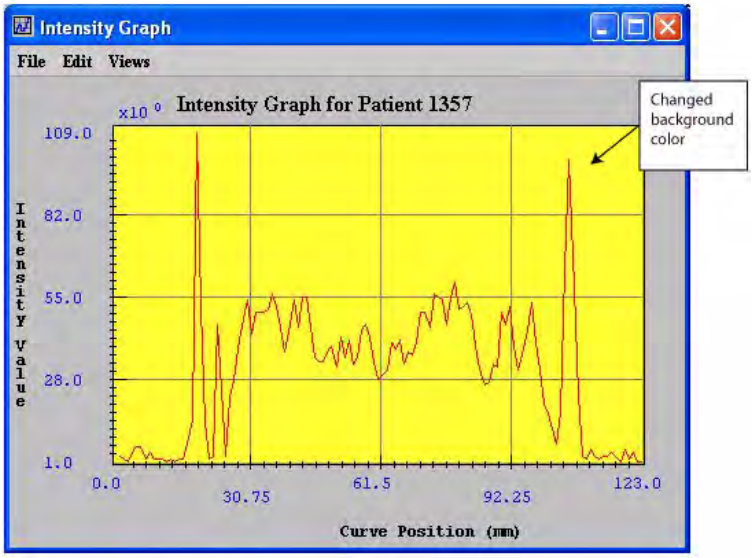


Figure 214. Changed background color for the graph

To reset graphs to their original background colors

To return either the intensity graph or the contour VOI graph back to its default colors, select Views > Reset Graph to Original or press Ctrl Z in either the Contour VOI Graph window or Intensity Graph window as appropriate.

CHANGING THE LEGENDS FOR FUNCTIONS

The Legend page controls whether a legend appears on the contour VOI graph or intensity graph and allows you to assign a specific name to each function. By default, the legend does not appear on the graph; however, when you choose so, the legend appears in the upper right corner of the graph.

To display or hide legends

1. Open an image.
2. Delineate a VOI on the image.
3. Generate an intensity profile (refer to "Generating graphs" on page 375). Either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379) opens.
4. Select Views > Modify Graph Features in either the Contour VOI Graph window or the Intensity Graph window. The Modify Graph dialog box (Figure 207 on page 382) appears.
5. Select the Legend tab. The Legend page (Figure 215) appears.
6. Do one of the following:
   - To display the legend on the graph, select Show legend. A check mark appears in the check box.
   - To hide the legend, clear Show legend. The check mark disappears from the check box.

7 Click Apply.

- If you selected Show legend, the legend appears in the upper right of the graph (Figure 216).
- If you cleared Show legend, the legend disappears from the graph.

8 Do one of the following:

- Close the Modify Graph dialog box.
- Keep the Modify Graph dialog box open to continue modifying the graph.

Legend tab	
Show legend	Displays, if selected, the legend on the upper right side on the graph in either the contour VOI graph or the Intensity Graph window.
Function *N* name	Specifies the name of the function. By default, the name is *Function 1,* but you can replace this name with any name you choose. This page allows you to specify up to five function names as long as those functions exist.
Apply	Applies the parameters that you specified.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

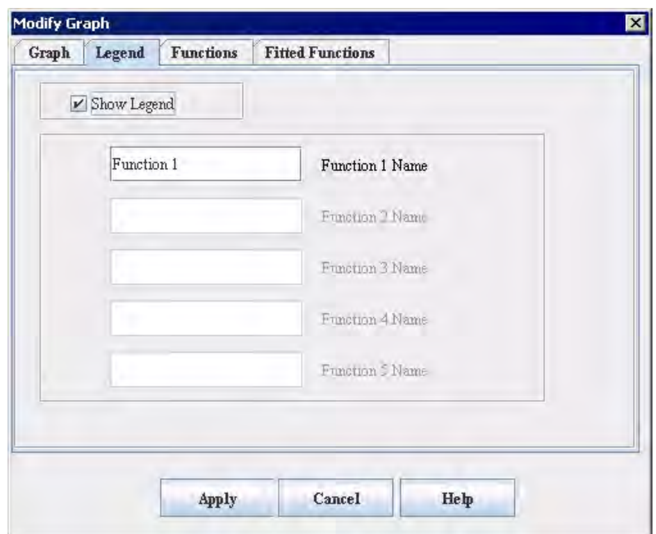


Figure 215. Legend page of the Modify Graph dialog box

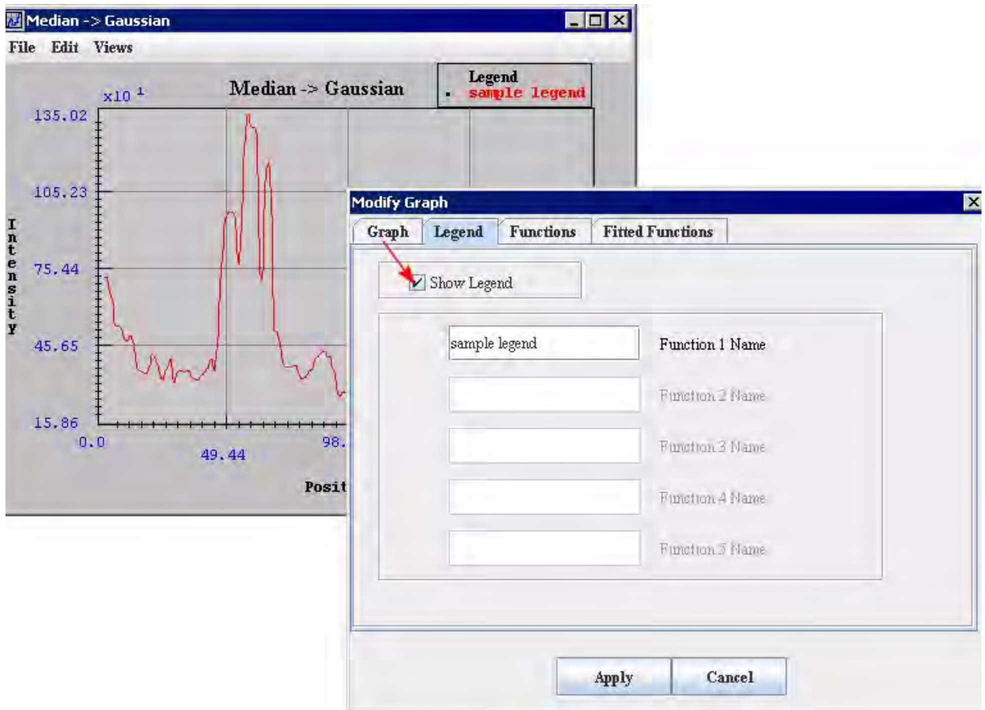


Figure 216. Legend at the upper right of the graph

CHANGING THE APPEARANCE OF FUNCTIONS

The Functions page in the Modify Graph dialog box allows you to display or hide the points on functions, display or hide from one to five functions, and change the color of functions.

Functions tab	
Points visible	Displays, if selected, all of the points on the functions.
Function *N* visible	Displays, if selected, function #*N* on the graph. You can display from one to five functions. This dialog box allows you to select only those functions that exist. Otherwise, they are dimmed.
Change function *N* color	Allows you to choose the color to use for displaying function *N* (listed on the left). When you select this icon, the Pick Background Color dialog box opens.
Apply	Applies the parameters that you specified.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

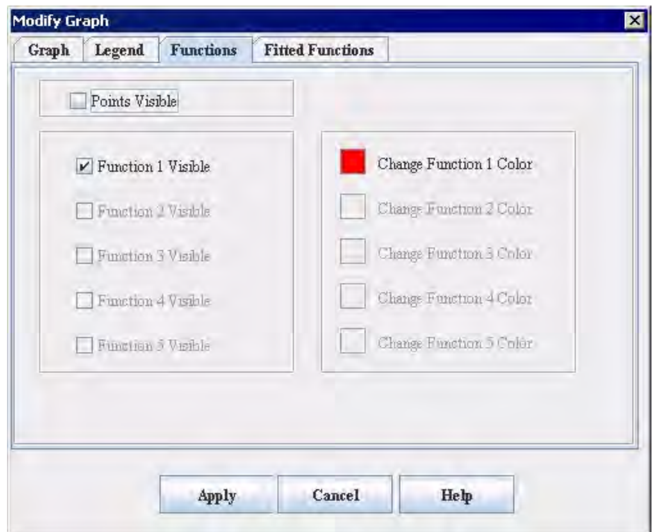


Figure 217. Functions page in the Modify Graph window

To display or hide the points on functions

1. Open an image.
2. Delineate a VOI on the image.
3. Generate an intensity profile (refer to "Generating graphs" on page 375).

   Either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379) opens.

4 Select Views > Modify Graph Features in either the Contour VOI Graph window or the Intensity Graph window.

The Modify Graph dialog box (Figure 207 on page 382) opens.

5 Select the Functions tab. The Functions page (Figure 217) appears.

- To display the points, select Points visible. A check mark appears in the check box.
- To make the points invisible, clear Points visible. The check mark disappears from the check box.

6 Click Apply.

- If you selected Points visible, the points appear on the functions (Figure 218).
- If you cleared Points visible, the points disappear from the function.

7 Do one of the following:

- Close the Modify Graph dialog box.
- Keep the Modify Graph dialog box open to continue modifying the graph.

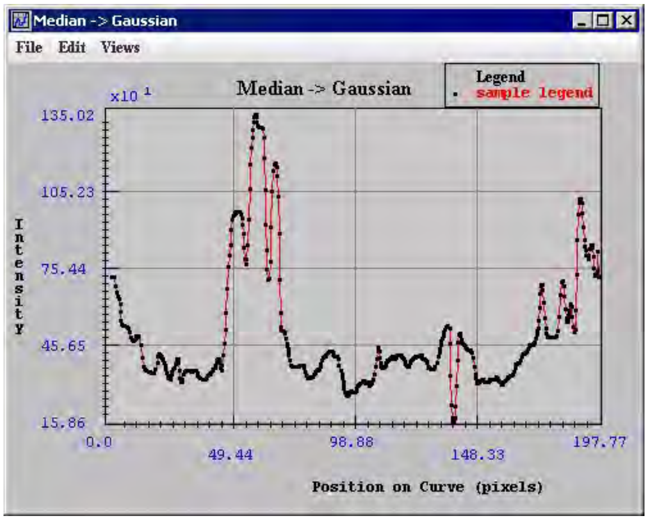


Figure 218. Points on a function whose color changed from red to blue

To display or hide functions

1 Open an image.

2 Delineate a VOI on the image.

3 Generate an intensity profile (refer to "Generating graphs" on page 375).

Either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379) opens.

4 Select Views > Modify Graph Features in either the Contour VOI Graph window or the Intensity Graph window.

The Modify Graph dialog box (Figure 207 on page 382) opens.

5 Select the Functions tab. The Functions page (Figure 217) appears.

6 Do one of the following:

- To display the function, select Function *N* visible. A check mark appears in the check box.
- To remove the function from the graph, clear Function *N* visible. The check mark disappears from the check box.

7 Click Apply.

- If you selected Function *N* visible, the function appears on the graph.
- If you cleared Function *N* visible, the function disappears from the graph.

8 Do one of the following:

- Close the Modify Graph dialog box.
- Keep the Modify Graph dialog box open to continue modifying the graph.

To change the color of functions

1 Open an image.

2 Delineate a VOI on the image.

3 Generate an intensity profile (refer to "Generating graphs" on page 375).

Either the Contour VOI Graph window (Figure 204 on page 375) or the Intensity Graph window (Figure 205 on page 379) opens.

4 Select Views > Modify Graph Features in either the Contour VOI Graph window or the Intensity Graph window.

The Modify Graph dialog box (Figure 207 on page 382) opens.

5 Select the Functions tab. The Functions page (Figure 217) appears.

6 Do one of the following:

- To change the color of the function, select Change Function *N* color. A check mark appears in the check box.
- To keep the color the same, clear Change Function *N* color. The check mark disappears from the check box.

7 Click Apply.

- If you selected Change Function *N* color, the function appears in the new color.
- If you cleared Change Function *N* color, the color of the function remains the current color.

8 Do one of the following:

- Close the Modify Graph dialog box.
- Keep the Modify Graph dialog box open to continue modifying the graph.

To reset functions to their original colors

To return the functions on either the intensity graph or the contour VOI graph back to their default colors, select Views > Reset Graph to Original or press Ctrl Z in the Intensity Graph window or the Contour VOI Graph window as appropriate.

MODIFYING FUNCTIONS ON GRAPHS

The graph modifying or *fitting* is designed for fitting of statistical functions used in parameters' evaluation. Available functions are currently various statistical functions with linear or exponential autocorrelation functions. In the Fitted Functions tab, you can select the autocorrelation function and then fit the data.

As the result you will obtain the fitted curve and the set of its parameters. The fit report will appear in the Fitting Data tab and can be saved into a file using Save Messages button.

Fitted functions	
Fit linear	Use the linear autocorrelation function (a1*x+a0).
Fit exponential	Use the exponential autocorrelation function.
None	Do not use the autocorrelation function.
Fitted function *N* visible	If checked, adds the fitted curve to the graph. If this is not desirable, uncheck the box.
Apply	Applies the parameters that you specified.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

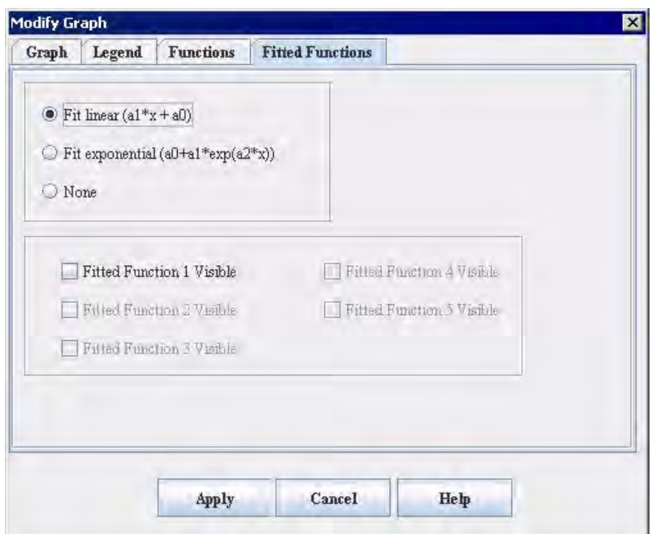


Figure 219. Fitted Functions page in the Modify Graph dialog box

Opening, saving, printing, and closing graphs

The File menu in the Contour VOI Graph window and the Intensity Graph window allows you to open, save, print, and close graphs.

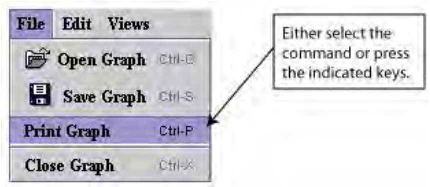


Figure 220. File menu in the graph windows showing the Open Graph, Save Graph, Print Graph, and Close Graph commands

To open previously saved graphs

1 Select File > Open Graph in either the Contour VOI Graph window or the Intensity Graph window, or press Ctrl+C. The Open dialog box appears.

2 Navigate to the directory where the graph was stored.

3 Type or select the file name in File name.

4 Click Open. The graph opens.

To save contour VOI graphs or intensity graphs

1 Select File > Save Graph in either the Contour VOI Graph window or the Intensity Graph window, or press Ctrl+S. The Save Graph dialog box opens.

2 Navigate to the directory where the graph was saved.

3 Type or select the file name in File name. Use *.PLT for the extension.

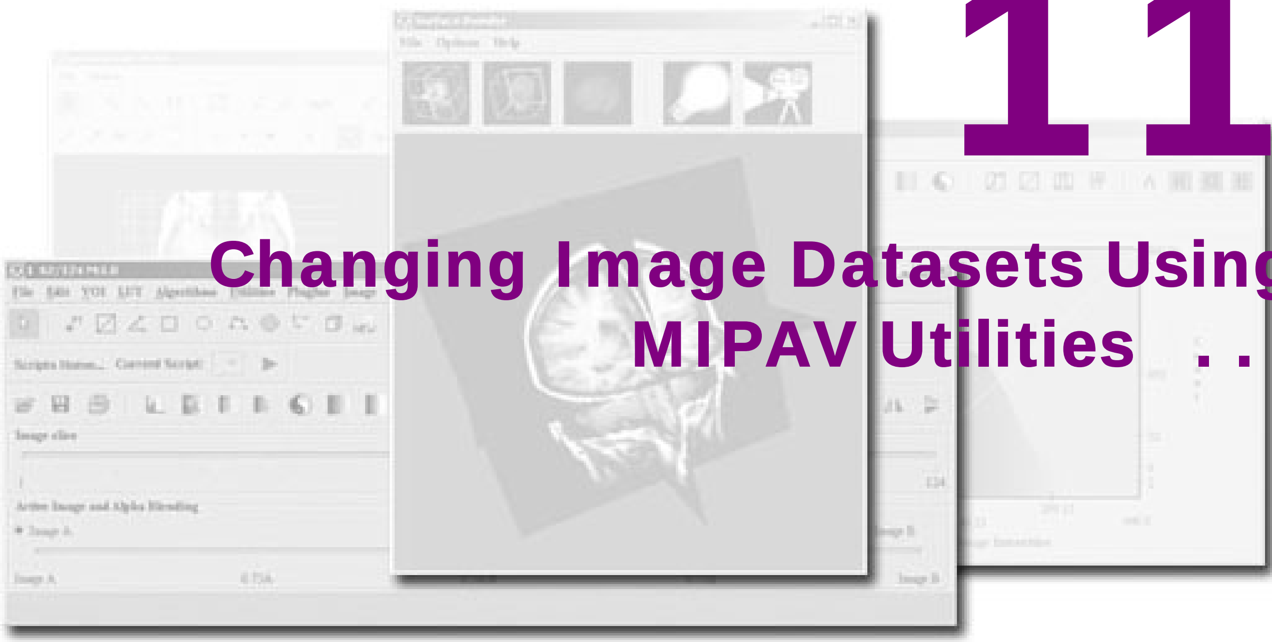
4 Click Save. The graph is saved in the specified directory.

To print contour VOI graphs or intensity graphs

1. Select File > Print Graph in the Contour VOI graph window or Intensity Graph window, or press Ctrl+P. The Print dialog box opens.
2. Select the printer and number of copies you want to print.
3. Click OK. The printer prints the graph.

To close graphs

Select File > Close Graph in either the Contour VOI Graph window or in the Intensity Graph window, or press Ctrl+X. The graph closes.



11

Changing Image Datasets Using MIPAV Utilities . . .

In this chapter

Standard tasks provided through commands on the Utilities menu

MIPAV provides a number of utilities that you can use to perform standard image-processing tasks such as converting image datasets to another image type, transforming images by copying, cropping, or rotating them, and changing image datasets by adding, removing, or reordering slices. To perform more complex tasks, MIPAV provides algorithms, which are covered in Volume 2 of the *User's Guide*. This chapter explores the standard tasks.

To access the standard image-processing tasks, you use the commands (refer to Table 2) on the Utilities menu in the MIPAV window.

Table 2. Standard tasks provided through commands on the Utilities menu in the MIPAV window

Task	Command	Scalar*			RGB		
		2D	3D	4D	2D	3D	4D
4D							
"Converting 3D to 4D images and vise versa"	Convert 3D to 4D	N	Y	N	N	Y	N
	Convert 4D to 3D	N	N	Y	N	N	Y
"Convert 4D to RGB"	Convert 4D to RGB	N	N	Y	N	Y	Y
"Extracting 3D subset from 4D"	Extracting 3D Subset from 4D	N	N	Y	N	N	Y
"Removing time volumes"	Remove Time Volumes	N	N	Y	N	N	Y
"Swapping the third and fourth dimensions"	Swap DIMS 3 <-> 4	N	N	Y	N	N	Y
"Adding image margins"	Add Image Margins	Y	Y	Y	Y	Y	Y
Copying images	Clone (copy)	Y	Y	Y	Y	Y	Y

*Scalar includes the following image types: boolean, byte, unsigned byte, short, unsigned short, integer, long, float, and double.

Table 2. Standard tasks provided through commands on the Utilities menu in the MIPAV window

Task	Command	Scalar*			RGB		
		2D	3D	4D	2D	3D	4D
Conversion Tools							
"Converting image datasets to different data types"	Convert Type	Y	Y	Y	N	N	N
"Converting grayscale images to RGB images"	Grays to RGB	Y	Y	Y	N	N	N
"Converting RGB datasets to grayscale datasets"	RGB to Gray RGB to Grays RGB to HSB	N	N	N	Y	Y	Y
"Correcting image spacing"	Correct Image Spacing	N	Y	Y	N	N	N
"Cropping images"	Crop Using Parameters Crop using VOI	Y	Y	Y	Y	Y	Y
"Masking (filling) images"	Fill Image	Y	Y	N	Y	Y	N
"Flipping images"							
	Flip > Horizontal	Y	Y	Y	Y	Y	Y
	Flip > Vertical	Y	Y	Y	Y	Y	Y
	Flip>In Depth (invert slice order)	N	Y	Y	N	Y	Y
"Image Calculator"	Image Calculator	Y	Y	Y	Y	Y	Y
"Image Calculator–Bulk Images"	Image Calculator–Bulk Images	Y	Y	Y	Y	Y	Y
"Image Math"	Image Math	Y	Y	Y	Y	Y	Y
"Inverting the image"	Invert	Y	Y	Y	Y	Y	Y
"Matching images"	Match Images	Y	Y	N	Y	Y	N

*Scalar includes the following image types: boolean, byte, unsigned byte, short, unsigned short, integer, long, float, and double.

Table 2. Standard tasks provided through commands on the Utilities menu in the MIPAV window

Task	Command	Scalar*			RGB		
		2D	3D	4D	2D	3D	4D
"Adding noise to images"	Noise	Y	Y	N	Y	Y	N
"Pad"	Pad	Y	Y	N	Y	Y	N
"Quantify Mask"	Quantify Using Mask Quantify Mask(s)	Y	Y	Y	N	N	N
"Replacing pixel/voxel value in images"	Replace Pixe/Voxel Value	Y	Y	Y	Y	Y	Y
"Rotating images"	Rotate -> X axis +90	Y	Y	Y	Y	Y	Y
	Rotate -> X axis -90						
	Rotate.> Y axis -90	Y	Y	Y	Y	Y	Y
	Rotate -> Y axis +90	Y	Y	Y	Y	Y	Y
	Rotate -> Z axis +90	Y	Y	Y	Y	Y	Y
	Rotate -> Z axis -90	Y	Y	Y	Y	Y	Y
Slice Tools							
"Concatenating images"	Concatenate	Y	Y	Y	Y	Y	Y
"Extracting slices/ volumes"	Extract Slices/ Volumes	N	Y	Y	N	Y	Y
"Insert Missing Slices"	Insert Missing Slices	N	Y	Y	N	Y	Y
"Inserting slices into image datasets"	Insert Slice	N	Y	Y	N	Y	Y

*Scalar includes the following image types: boolean, byte, unsigned byte, short, unsigned short, integer, long, float, and double.

Table 2. Standard tasks provided through commands on the Utilities menu in the MIPAV window

Task	Command	Scalar*			RGB		
		2D	3D	4D	2D	3D	4D
"Pad slices to power of 2"	Pad Slices to power of 2	N	Y	Y	N	N	N
"Randomizing image (slice) order"	Randomize Slice Order	N	Y	Y	N	Y	Y
"Removing images (slices) from datasets"	Remove Slices	N	Y	Y	N	Y	Y
"Replace blanks with averages"	Replace blanks with averages	N	Y	Y	N	N	N
"Replace Slices"	Replace Slices	N	Y	Y	N	Y	Y
"Subtract VOI Background"	Subtract VOI background	Y	Y	Y	Y	Y	Y

*Scalar includes the following image types: boolean, byte, unsigned byte, short, unsigned short, integer, long, float, and double.

Recording utilities usage with the history feature

MIPAV provides a way for you to record the actions, whether with algorithms or utilities, that you perform on images. You use the MIPAV Options dialog box to turn this feature on. Refer to "Saving a history of actions on images (TBD)" on page 126 in Chapter 3,"Getting Started Quickly with MIPAV" of this *User's Guide* for more information.

4 D tools

Converting 3D to 4D images and vise versa

To convert 3D images to 4D images

1 Open a 3D image or image dataset.

2 Select Utilities > Convert 3D to 4D. The Convert 3D to 4D dialog box opens.

3 Do either of the following in the Number of slices in the 3rd dimension, Resolutions: 3rd dimension, and Resolutions: 4th dimension boxes:

- Accept the default numbers.
- Enter different numbers.

4 Select in each of the Resolution units: 3rd dimension and Resolutions units: 4th dimension lists one of the following: millimeters (the default value), unknown, inches, centimeters, angstroms, nanometers, micrometers, milliliters, meters, kilometers, miles, nanoseconds, microseconds, milliseconds, seconds, minutes, hours, or Hz.

5 Click OK. The program applies all of the specification in this dialog box to the image or image dataset. See Figure 221 on page 405.

Number of slices in the 3rd dimension	Specifies how many slices are in the 3rd dimension. The default number is 2.
Resolutions: 3rd dimension	Indicates the resolution for the third dimension. The default number is 5.0.
Resolutions: 4th dimension	Indicates the resolution for the fourth dimension. The default number is 1.0.
Resolution units: 3rd dimension	Indicates the voxel resolution in the 3rd dimension.

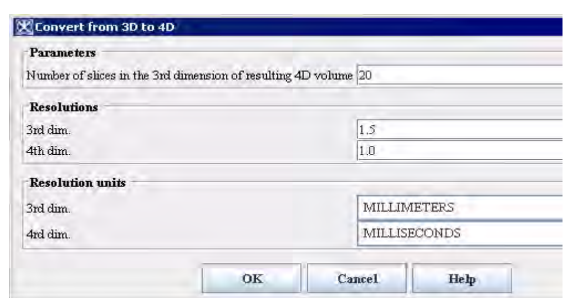


Figure 221. Convert from 3D to 4D dialog box

Resolution units: 4th dimension	Indicates the voxel resolution of the 4th dimension.
OK	Applies the specified parameters to convert a 3D image to a 4D image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not convert the image.
Help	Displays online help for this dialog box.

Figure 221. Convert from 3D to 4D dialog box (continued)

To convert 4D to 3D images

1 Open a 4D image or image dataset.

2 Select Utilities > Convert 4D to 3D.

The program immediately begins processing the image.

Extracting 3D subset from 4D

This utility allows you to remove one of the *x*, *y*, *z*, or *t* dimensions from 4D images to produce a 3D image. Refer to Figure 222 on page 408.

To remove an X dimension slice

1 Open a 4D image.

2 Select Utilities >4D Tools> Extract 3D Subset from 4D. The Extract 3D Subset dialog box (Figure 222) opens.

3 Select X.

4 Type the index number of the slice you want to extract in the Select index from. <*N*> to <*N*> box.

5 Click Remove. The program removes the slice you specified from the dataset.

To remove a Y dimension slice

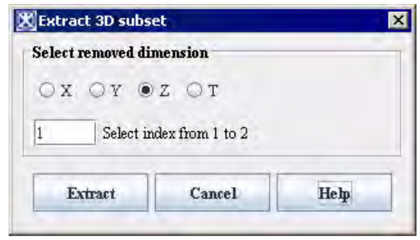
1. Open a 4D image.
2. Select Utilities > Extract 3D Subset from 4D. The Extract 3D Subset dialog box (Figure 222) opens.
3. Select Y.
4. Type the index number of the slice you want to extract in the **Select index from <*N*> to <*N*>** box.
5. Type the index number of the slice you want to extract in the Select index from. <*N*> to <*N*> box.
6. Click Remove. The program removes the slice you specified from the dataset.

To remove a Z dimension slice

1. Open a 4D image.
2. Select Utilities > Extract 3D Subset from 4D. The Extract 3D Subset dialog box (Figure 222) opens.
3. Select Z.
4. Type the index number of the slice you want to extract in the Select index from. <*N*> to <*N*> box.
5. Click Remove. The program removes the slice you specified from the dataset.

To remove the T dimension

1. Open a 4D image.
2. Select Utilities > Extract 3D Subset from 4D. The Extract 3D Subset dialog box (Figure 222) opens.
3. Select T.
4. Type the index number of the slice you want to extract in the Select index from. <*N*> to <*N*> box.
5. Click Remove. The program removes the slice you specified from the dataset.



X	Specifies that the algorithm should remove the *X* (width) dimension.
Y	Specifies that the algorithm should remove the *Y* (height, or length) dimension.
Z	Specifies that the algorithm should remove the *Z* (depth) dimension.
T	Specifies that the *T* (time) dimension should be removed.
Index from <N> to <N>	Specifies the index number of the volume or slice that you want to extract from the dataset.
Extract	Removes the specified dimension.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not remove any dimensions.
Help	Displays online help for this dialog box.

Figure 222. Extract 3D Subset dialog box

Removing time volumes

If an image dataset contains blank images or unusable images, which might be caused by patients blinking their eyes during tests, the Remove Time Volumes command on the Utility menu in the MIPAV window allows you to remove unusable images from the dataset.

To remove time volumes

1 Open an image dataset that contains time volumes.

Note: The Remove Time Volumes command only becomes active for 4D image datasets.

2 Use the image slider to look through the dataset one time volume, or image, at a time (Figure 223). If you find an unusable volume—one that is blank, or totally black, or that contains unusable information—stop moving the slider on that volume.

Note: The volume number appears in the title bar of the MIPAV window and correlates to the slice indicated by the image slider.

3 Select Utilities > Remove time volumes. The Remove Time Volumes dialog box (Figure 282 on page 486) opens.

4 Select the time volume or volumes (slices) that you want to remove in the list at the top of the dialog box, or type the number of the slice or range of volumes in the **Check the time slices to remove** box.

5 Do either of the following:

- Click Select all to remove all of the slices.
- Click Remove to remove the slices you selected.

  A status message appears. Then the MIPAV window refreshes.

6 Use the image slider again, if you'd like, to look through the slices in the dataset to verify that the appropriate slices were removed. See Figure 223 and Figure 224.

Swapping the third and fourth dimensions

Swapping the third and fourth dimensions refers to how image datasets are stored. Datasets may be stored using the following two methods:

- *xytz* (horizontal, vertical, time, third dimension)
- *xyzt* (horizontal, vertical, third dimension, time)

Because MIPAV requires that datasets be stored using the *xyzt* method, it provides the Swap Dims 3 <-> 4 command for those users whose datasets may be stored using the *xytz* method.

To swap the third and fourth dimensions

1 Open an image that is stored using the *xytz* method.

2 Select Utilities>4D Tools>Swap Dims 3 <-> 4. A progress message appears briefly while the program changes the storage method of the image and replaces the image with one that is stored using the *xyzt* method.

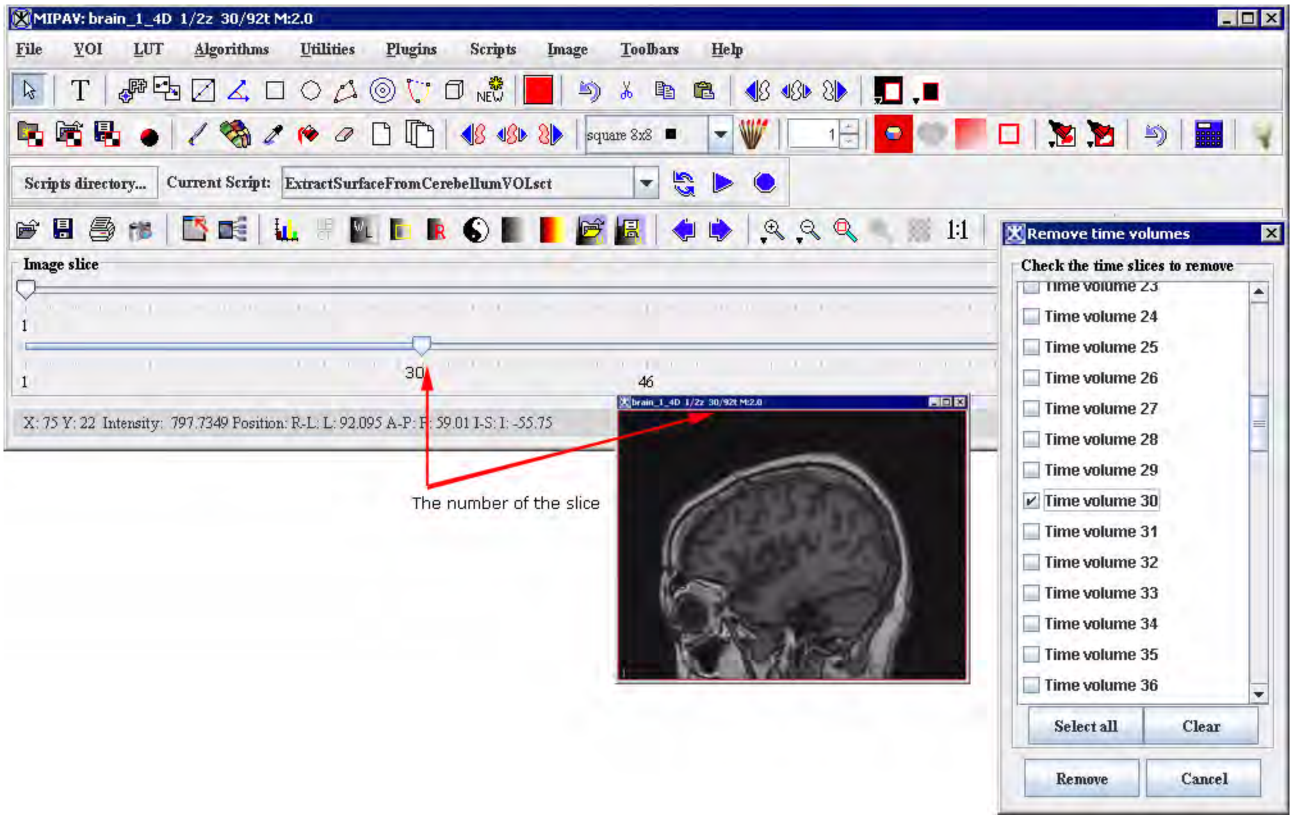


Figure 223. Using the image slider to look through an image dataset

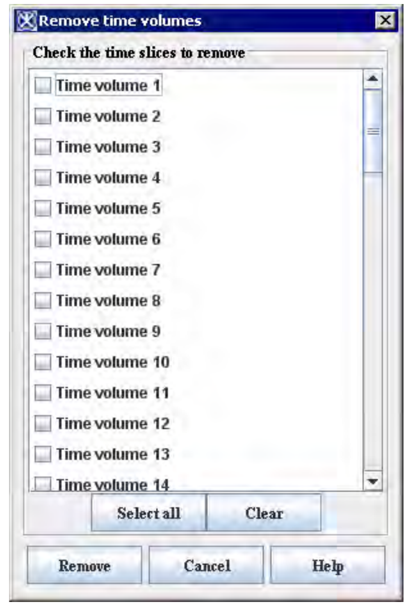
Select all	Selects all time volumes.	
Clear	Clears selection.	
Remove	Removes selected slices.	
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not remove the time volumes.	
Help	Displays online help for this dialog box.	

Figure 224. The Remove Time Volumes dialog box

Convert 4D to RGB

This utility converts 4D datasets that have three or less slices (volumes) into RGB 3D images, where each slice is represented by one channel – red, green or blue.

To convert 4D to 3D images

1 Open a 4D image or image dataset.

2 Select Utilities >4D Tools>Convert 4D to RGB.

3 The Convert 4D to RGB dialog box appears. Complete the dialog box, use the list boxes to assign a chosen volume to the corresponding color channel. Refer to Figure 225.

4 Press OK. The new 3D RGB image appears in a new image frame. See Figure 226.

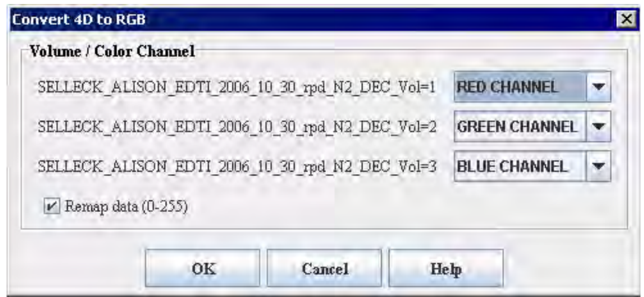
Volume/color channel Specify the image volume and the corresponding color channel. Use the list boxes to select Red, Green, and Blue channel; or use blank.	
Remap data	Check this box, if you would like to remap RGB data to 0–255 range.
OK	Applies the specified parameters to convert a $D image to a RGB image.
Cancel	Disregards any changes you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

Figure 225. The Convert 4D to RGB dialog box

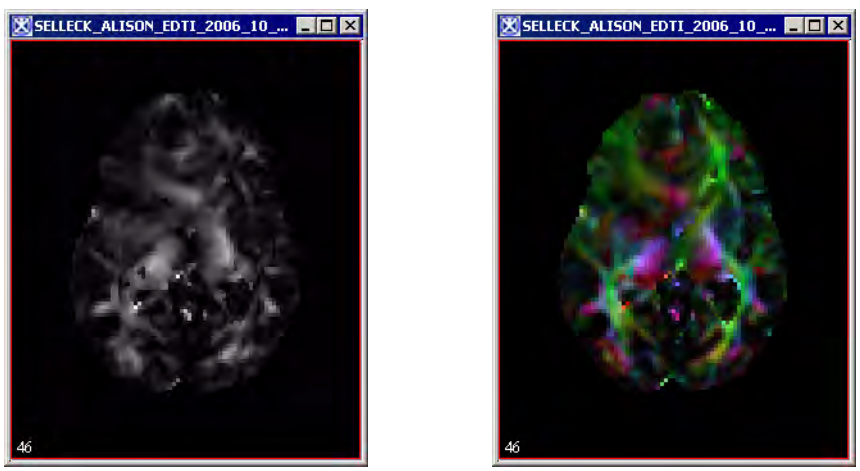


Figure 226. The original 4D image and the same image converted to 3D RGB

Adding image margins

The Add Image Margins command allows you to add a border of pixels or a specific intensity along the edges of an image dataset.

To add image margins

1 Open an image.

2 Select Utilities > Add Image Margins. The Add Image Borders dialog box (Figure 227) opens.

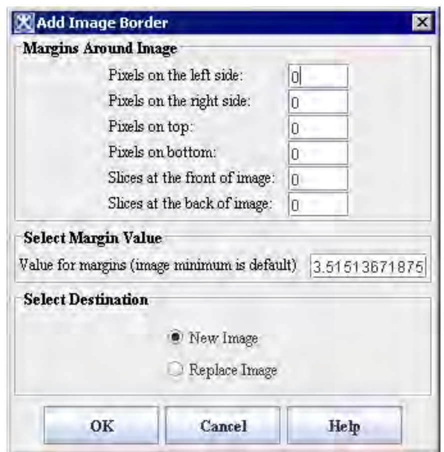
3 Enter the number of pixels to add on the left and right sides of the image, the top and bottom, and to the front and back of the image.

4 Enter the intensity value for the margins.

5 Select one of the following destinations:

- *New image*—The software applies the margins to another instance of the dataset in a new image window.
- *Replace image*—The software adds the margins to the existing instance of the dataset.

6 Click OK. The image margins are applied to the dataset.



Margins Around Image	**Pixels on the left side:** Specifies the number of pixels that should appear on the left side of the image. **Pixels on the right side:** Specifies the number of pixels that should appear on the right side of the image. **Pixels on top:** Specifies the number of pixels that should appear on the top of the image. **Pixels on bottom:** Specifies the number of pixels that should appear at the bottom of the image. **Slices at the front of image:** Specifies the number of slices that should appear at the front of the image. **Slices at the back of image:** Specifies the number of slices that should appear at the back of the image.
Value for margins	Specifies the intensity of the border around the image. As a default, the intensity of the border is the same intensity as that for the image.
New image	Shows the image with the additional or adjusted margins in a new image window.

Figure 227. Add Image Border dialog box

Replace image	Replaces the current active image with the results of the image to which margins were added or adjusted.
OK	Applies the parameters that you specified to add margins to this image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not add image margins.
Help	Displays online help for this dialog box.

Figure 227. Add Image Border dialog box (continued)

Copying images using the Clone command

Suppose you need to copy an image dataset. To do so, you would use the Clone command on the Utility menu. This command generates a duplicate of the dataset and any information stored on the image, utility, algorithms, VOI, and paint layers.

Note: The Clone utility copies VOIs as well as the image, but it does not copy LUT information. Although we can observe its effects, it is not stored in an image layer. The Clone utility doesn't work for 4D images!

If two datasets are loaded into one image window, only the active dataset is cloned. When the dataset is duplicated, the duplicate dataset appears in its own image window.

To copy an image dataset, you simply select Utility > Clone (copy) in the MIPAV window. In a moment or two, a duplicate of the dataset appears in a new image window.

Converting image datasets to different data types

In MIPAV you can convert image datasets to different data types. For example, you might want to convert a Boolean type dataset to an integer type dataset. MIPAV also allows you to simultaneously alter the input and output values. This utility is particularly helpful if you want to apply an algorithm to a dataset but cannot do so because the original images are the wrong image type.

data type—A set of values from which a variable, constant, function, or expression may takes its value. MIPAV automates the following data types: Boolean, signed byte, unsigned byte, signed short, unsigned short, integer, long, float, double, and color 24.

CONVERT TYPE

To convert a dataset to a different image type

1. Select Utilities >Conversion Tools > Convert type. The Convert Image Type dialog box (Figure 228) opens.
2. Select the desired image type in the Image Type group.
3. Indicate the start and end input ranges in the Range of input values group.
4. Indicate the start and end output ranges in the Range of output values group.
5. Specify either Little endian or Big endian in the Endianess group.
6. Click OK. The dataset is converted to the new image type.

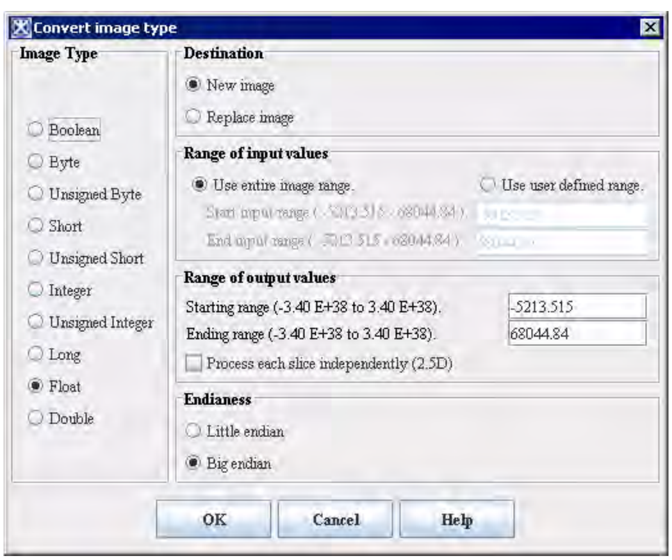


Figure 228. Convert Image Type dialog box

Image Type	Specifies the data type. Select one of the following:	
	Boolean	Indicates whether a condition if true or false.
	Byte	Primitive 8-bit data type. Valid values range from -127 to 128.
	Unsigned byte	Primitive 8-bit data type. Unsigned byte is a variation of the integer data type. The unsigned byte data type signifies that valid values must fall within a specified range of positive whole number values. Valid values range from 0 to 255. Negative values are not valid, hence the term *unsigned byte*.
	Short	Primitive 16-bit data type. Short is a variation of the integer data type. Short accommodates values that are whole numbers. Valid values range from 0 to +32,768.
	Unsigned short	Primitive 16-bit data type. Unsigned short is a variation of the integer data type. The unsigned short data type signifies that valid values must fall within a specified range of positive whole number values. Valid values range from 0 to 65,535. Note that negative values are not valid, hence the term *unsigned byte*.
	Integer	Primitive 32-bit data type. Integer is sometimes abbreviated as int. Integer accommodates values that are whole numbers. Valid values range from -2,147,483,648 to +2,147,483,648.
	Unsigned integer	Primitive 32-bit data type.
	Long	Primitive 64-bit data type. Long is a variation of the integer data type. Long accommodates values that are whole numbers. Valid values range from -9,223,372,036,854,775,808 to +9,223,372,036,854,775,808.
	Float	Primitive 32-bit data type. Float is a floating point data type that accommodates decimal values, up to 6 or 7 significant digits of accuracy. Valid values can range from -3.4×10^{38} to 3.4×10^{38}.
	Double	Primitive 64-bit data type. Double is a floating point data type that accommodates decimal values, up to 14 or 15 significant digits of accuracy. Valid values can range from -1.7×10^{308} to 1.7×10^{308}.
Destination	*New image*	Shows the converted dataset in a new image window.
	Replace image	Replaces the current active dataset with the converted dataset.

Figure 228. Convert Image Type dialog box (continued)

Range of input values	*Use entire image range*	Converts all intensity values to the result image range when converting the image to a different type.
	Use user-defined range	Converts only the intensity values in the user-defined range when converting the image to a different type.
	Start input range	Specifies the intensity value at the beginning of the input range. The default value is the image minimum.
	End input range	Specifies the intensity value at the end of the input range.
Range of output values	*Starting range*	Specifies the intensity value at the beginning of the output range. The default value is the image minimum.
	Ending range	Specifies the intensity value at the end of the output range.
Process each slice independently (2.5D)	Converts each image slice independently as if it was a separate image. This options works for 2.5D images.	
Endianess	Data organization strategy. Refers to the way computer processors store data in memory.	
	Little endian	Stores the least significant byte (LSB) first.
	Big endian	Stores the most significant byte (MSB) first.
OK	Applies the parameters that you specified to convert the dataset.	
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not convert the dataset.	
Help	Displays online help for this dialog box.	

Figure 228. Convert Image Type dialog box (continued)

CONVERTING GRAYSCALE IMAGES TO RGB IMAGES

You can convert grayscale image datasets to RGB. RGB images have three channels (red, green, and blue) that contain image data.If you open two datasets in one image window, you can create a composite image that contains a mixture of the red, green, and blue channels.

To convert grayscale images to RGB images

1 Open an image.

2 Select Utilities >Conversion Tools> Grays -> RGB. The Concatenate -> RGB dialog box opens.

3 Select the image to which you want to apply the red channel in the Image (red) list. If you loaded two images into the image window, two file names should appear.

4 Do the same for the Image (green) list and the Image (blue) list.

5 Indicate whether you want to remap to current intensity values to the full standard 0–255 RGB values.

Note: Generally, each color is defined as one of the 256 intensities. If you remap the grayscale intensities and select Remap data, the system defines the intensities as one of the 256 values. If one or all of the input images have values that exceed 255 and you do *not* select Remap data, then data truncates to 255 since a color channel can only represent values 0–255.

6 Click OK. A status message appears. When filtering is complete, the new RGB image appears in a separate image window.

Image (red)	Identifies the image to be added to the red channel of the resulting image.
Image (green)	Identifies the image to be added to the green channel of the resulting image.
Image (blue)	Identifies the image to be added to the blue channel of the resulting image.
Remap data (0-255)	Indicates, when selected, that you want image intensities to be remapped to values 0-255, which is the standard for RGB images.
Destination	
New image	Sends the output to a new image frame.
Replace image	Replaces the existing image.

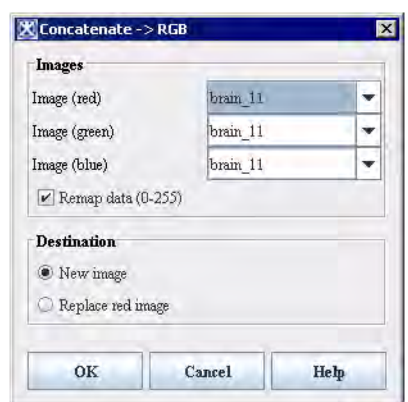


Figure 229. Concatenate -> RGB dialog box

OK	Applies the parameters that you specified to create an RGB image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not create an RGB image.
Help	Displays online help for this dialog box.

Figure 229. Concatenate -> RGB dialog box (continued)

CONVERTING RGB DATASETS TO GRAYSCALE DATASETS

On the Utilities > Conversion Tools menu in the MIPAV window, the RGB conversion utility offers both a manual and an automatic method of converting RGB datasets to grayscale RGB images.

- **Manual conversion**—In the manual method of conversion, you select Utilities >Conversion Tools > RGB -> Gray to display the RGB -> Gray dialog box (Figure 230). The dialog box provides three weighting methods:*Equal weights*—This method assigns equal weights to each of the three color channels. When you select this option, you can also select the **Only average RGB values greater than** check box and specify a threshold value. For example, if you specify a threshold value of 50, the program ignores any pixel in that channel that contains a value over 50.
  - *Computer graphics weights*—This method assigns the weights to each channel that are typically used to display computer graphics.
  - *User-specified weights*—In this method, you enter the specific weights for each color channel.
- **Automatic conversion**—If, instead, you select Utilities > >Conversion Tools > RGB -> Grays, the program automatically converts the RGB image to three grayscale images: one each for the red, blue, and green channels.

Note: You can only specify a threshold when you choose to use the equal weights method of conversion.

RGB datasets have four channels: red, green, blue, and alpha. When you convert RGB datasets to grayscale. RGB images, the intensities are combined to form a single grayscale value.

For example:

$$N = R \times RW + G \times GW + B \times BW$$

Where,

N = New grayscale pixel

R = Red channel

G = Green channel

B = Blue channel

RW = Weight assigned to the red channel

GW = Weight assigned to the green channel

BW = Weight assigned to the blue channel

Equal weights	Assigns the same weight (0.3333) to each channel in the image. When you select this option, the Only average RGB values great than becomes available.
Computer graphics	Assigns the weighting factors typically used in computer graphics to each channel in the image: • Red, 0.299 • Green, 0.587 • Blue, 0.114
User specified	Specifies a weight that you determine for each channel. When you select this check box, you must type a specific weight for each channel in the Red, Green, and Blue boxes.
Red	Specifies the weight assigned to the red channel in the image. You can only specify a weight in this box if you selected the User specified option.

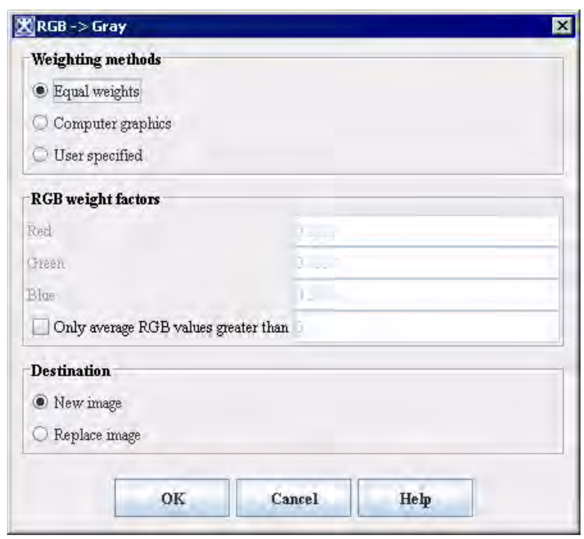


Figure 230. RGB -> Gray dialog box

Green	Specifies the weight assigned to the green channel in the image. You can only specify a weight in this box if you selected the User specified option.
Blue	Specifies the weight assigned to the blue channel in the image. You can only specify a weight in this box if you selected the User specified option.
Only average RGB values greater than	Excludes any voxel in the image over the threshold value that you specify in this box. This check box is only available when you select the Equal Weights option.
New Image	Sends the output to a new image frame.
Replace Image	Replaces the existing image.
OK	Applies the parameters that you specified to create a grayscale image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not create a grayscale image.
Help	Displays online help for this dialog box.

Figure 230. RGB -> Gray dialog box (continued)

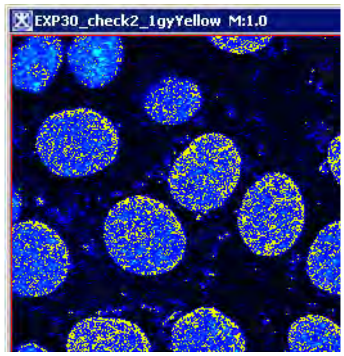
To manually convert RGB datasets to grayscale

1 Select Utilities > RGB > RGB -> Gray: The RGB -> Gray dialog box (Figure 230) opens.

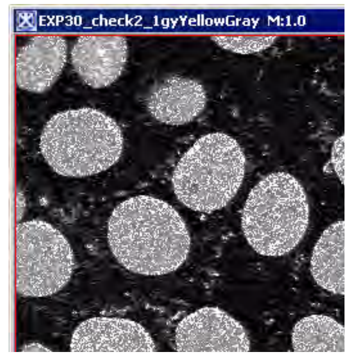
2 Select one of the following weighting methods: equal weights, computer graphics, or user specified.

- If you selected Equal Weights, decide whether to apply a threshold to the RGB channels. If you want to apply a threshold, select Only average RGB values greater than and enter the threshold value. Then go to the next step.
- If you selected Computer graphics or User specified, go to the next step.

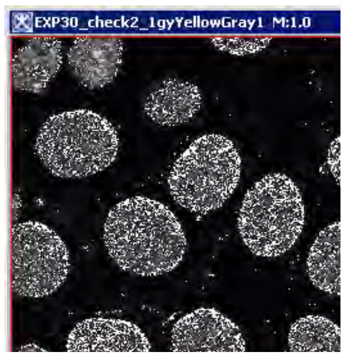
3 Click OK. A status message appears. In a few moments, the image is replaced by the new grayscale dataset.



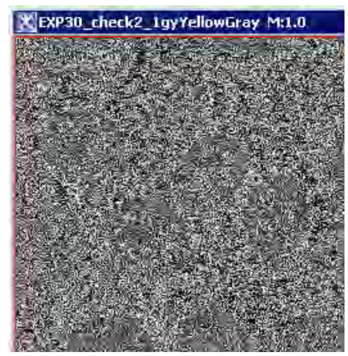
A



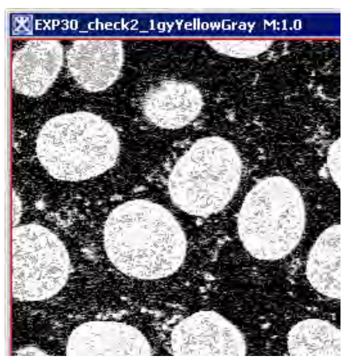
B



C



D



E

A – the original image

B – the image converted to grayscale using Equal Weights

C – the image converted to grayscale using Computer Graphics

D – the image converted to grayscale using the User Specified RGB weight factors (0.33 for each channel)

E – the image converted to grayscale using combined options – Equal Weights and average values greater than 50

Figure 231. Manually converting RGB datasets to grayscale

To automatically convert RGB datasets to grayscale image

Select Utilities > RGB > RGB -> Grays. The program briefly displays a status message during the conversion and then generates three new grayscale datasets, one for each channel.

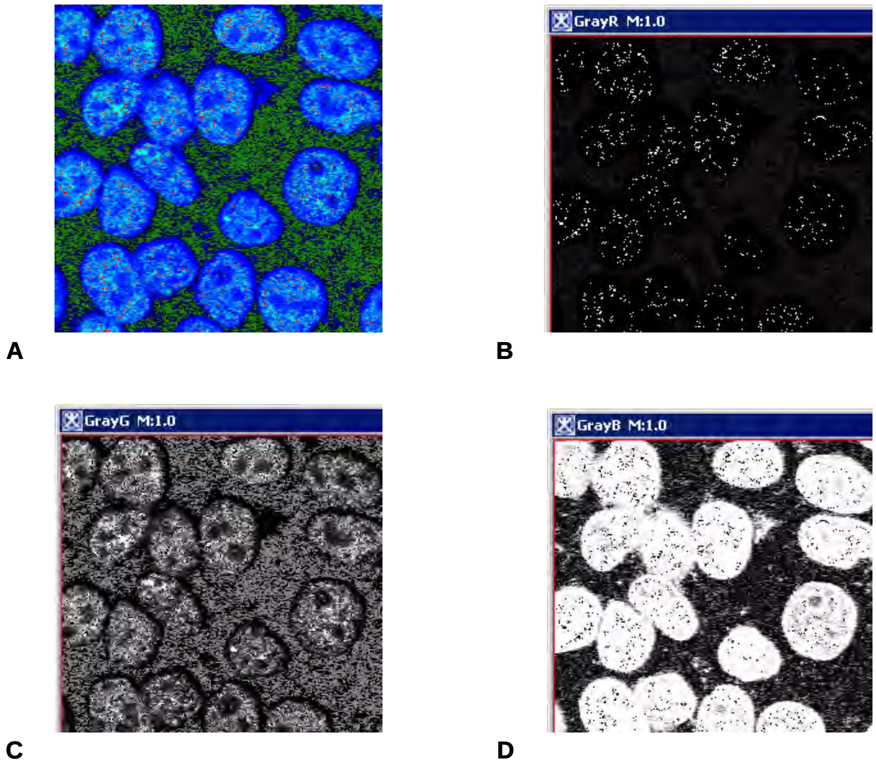


Figure 232. Example of three grayscale datasets generated automatically from an RGB dataset. Here, A is an original RGB image, B – gray red channel image, C – gray green channel image, and D –gray blue channel image.

Correcting image spacing

The Correct Image Spacing command on the Utilities menu corrects images in which slice thickness and slice spacing are unequal, which may distort images. To understand how this problem can occur, the following section presents some background information.

Understanding contiguous planes' effect on image scanning

Image scanning is usually done in contiguous planes. For example, if the first slice is centered at position 5.0 and the slices are 2 mm thick, the second slice is centered at 7.0. In such a case, the slice thickness, 2, and the slice spacing, 2, are equal. There are, however, two scenarios when the slice spacing does not equal the slice thickness:

- **Slice spacing is less than the slice thickness**—In this case, although the images are, for example, 2 mm thick, they are spaced only 1 mm apart (Figure 233). This scenario arises when the ZIP x 2 feature is used during MRI scanning, which enables the acquisition of slices only 1 mm apart but uses signals from a 2-mm slab to increase the signal-to-noise ratio.
- **Slice spacing is greater than the slice thickness**—This scenario (Figure 233), which is more common than the first, occurs when the operator chooses to acquire images at intervals greater than the slice thickness (i.e., there is a gap between successive image slices) in order to cover a deeper field of view.

MIPAV assumes that successive images are contiguous. So, for example, if users chose the triplanar view to display an image that had slice spacing larger than slice thickness, the image would appear shortened in the out-of-plane direction. The Correct Image Spacing utility corrects both situations in which the slice thickness and spacing are not equal. Once the correction is applied, the slice thickness and slice spacing for the given image dataset are the same.

Understanding how MIPAV solves the image spacing problem

When the spacing between slices is *less* than the slice thickness, the Correct Image Spacing utility assigns the slice spacing to the slice thickness (refer to Figure 233 on page 427 for triplanar views before and after correction). When the slice spacing is *larger* than the slice thickness (refer to Figure 233 on page 427 for triplanar views before and after correction), the utility inserts blank slices between the existing images.

Since all slices within an image volume must have the same slice thickness, in many cases more than one slice must be inserted and the original images must be repeated so that a new value can be found for the slice thickness that fits evenly into the original slice spacing. Since there are an infinite number of combinations of slice spacing and thickness, not all could be handled. Table 3 displays the most common combinations and the algorithms MIPAV uses for handling them.

T = Original slice thickness

S = Original space between slices

G = Gap = $S - T$

M = Number of original images

O = Original image set origin

N = New slice thickness

To use the Correct Image Spacing utility, DICOM images must first be saved in XML format. The following DICOM tags (Figure 233 on page 427) are used:

- (0018, 0050) Slice Thickness
- (0018, 0088) Spacing Between Slices

You can also find this information listed in the Essential Image Information section of the image header (Figure 233 on page 427) as Pixel resolution 2 (i.e., slice thickness) and Slice Spacing.

To view image attributes

1. Open an image.
2. Select Image > Attributes > View Header. The Header dialog box (Figure 233) opens.
3. Find the tags under Essential Image Information to find the values for Pixel resolution 2 and Slice Spacing, or find the values of the DICOM tags Slice Thickness and Spacing Between Slices.

Table 3. Solutions for correct image spacing

Case	New slice thickness (*N*)	Number of images in new set for each original image	Number of blanks inserted for each original image	Total images in set now	New image set origin $\left(O - \frac{T}{2} + \frac{N}{2}\right)$
$T = S$	T	1	0	M	O
$S < T$	S	1	0	M	O
$G > O$ and $\frac{G}{T} mod 1 = 0$	T	1	$1 \cdot \frac{G}{T}$	$\left(1 + \frac{G}{T}\right) \cdot M$	$O - \frac{T}{2} + \frac{(N)}{2}$
$G > O$ and $\frac{T}{G} mod 1 = 0$	G	$1 \cdot \frac{T}{G}$	1	$\left(1 + \frac{T}{G}\right) \cdot M$	$O - \frac{T}{2} + \frac{(G)}{2}$
$G > O$ and $\left(2 \cdot \frac{G}{T}\right) mod 1 = 0$	$\frac{T}{2}$	2	$2 \cdot \frac{G}{T}$	$\left(2 + 2 \cdot \frac{G}{T}\right) \cdot M$	$O - \frac{T}{2} + \frac{\left(\frac{N}{2}\right)}{2}$
$G > O$ and $\left(2 \cdot \frac{T}{G}\right) mod 1 = 0$	$\frac{G}{2}$	$2 \cdot \frac{T}{G}$	2	$\left(2 + 2 \cdot \frac{T}{G}\right) \cdot M$	$O - \frac{T}{2} + \frac{\left(\frac{G}{2}\right)}{2}$
$G > O$ and $\left(3 \cdot \frac{G}{T}\right) mod 1 = 0$	$\frac{T}{3}$	3	$3 \cdot \frac{G}{T}$	$\left(3 + 3 \cdot \frac{G}{T}\right) \cdot M$	$O - \frac{T}{2} + \frac{\left(\frac{N}{3}\right)}{2}$
$G > O$ and $\left(3 \cdot \frac{T}{G}\right) mod 1 = 0$	$\frac{G}{3}$	$3 \cdot \frac{T}{G}$	3	$\left(3 + 3 \cdot \frac{T}{G}\right) \cdot M$	$O - \frac{T}{2} + \frac{\left(\frac{G}{3}\right)}{2}$
else	don't handle, give message				

Legend:

T = Original slice thickness

S = Original space between slices

G = Gap = *S* - *T*

M = Number of original images

O = Original image set origin

N = New slice thickness

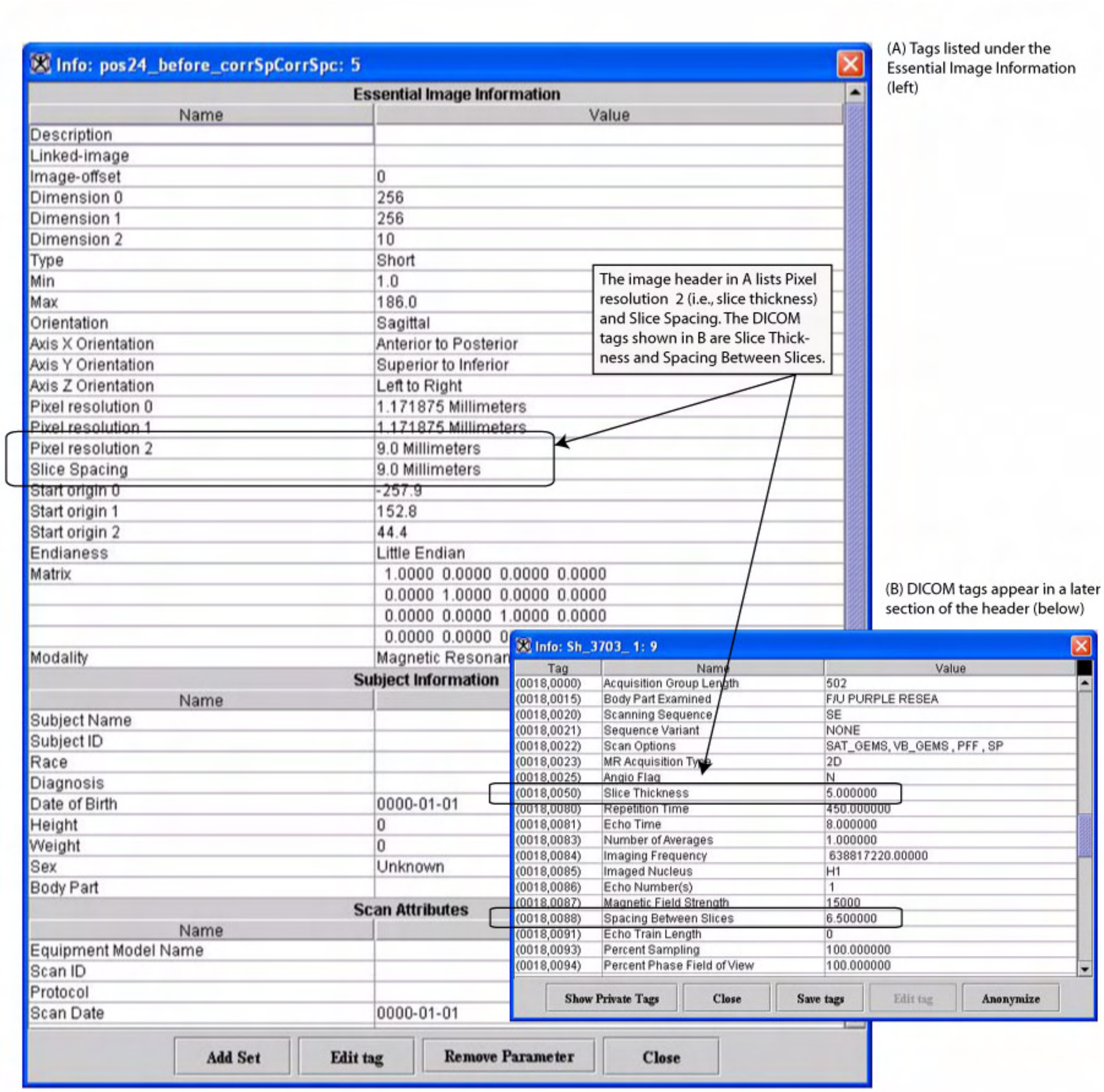


Figure 233. Views of the Header dialog box showing the tags listed under (A) Essential Image Information and (B) the DICOM tags in a later section of the header

To save DICOM images as XML files

The Correct Image Spacing utility works only on XML files.

1. Open the DICOM image.
2. Select File > Save Image as in the MIPAV window. The Open dialog box appears.
3. Type the name for the file including the XML extension in File Name.
4. The Select DICOM tags to convert to XML dialog box opens.
5. In the left dialog box window, select tags which you would like to convert to XML. Use the CTRL+right mouse button combination of keys to select multiple tags. Then, use the arrows to move selected tags to the right window. See Figure 234 for more information.
6. Press Save to save the file.

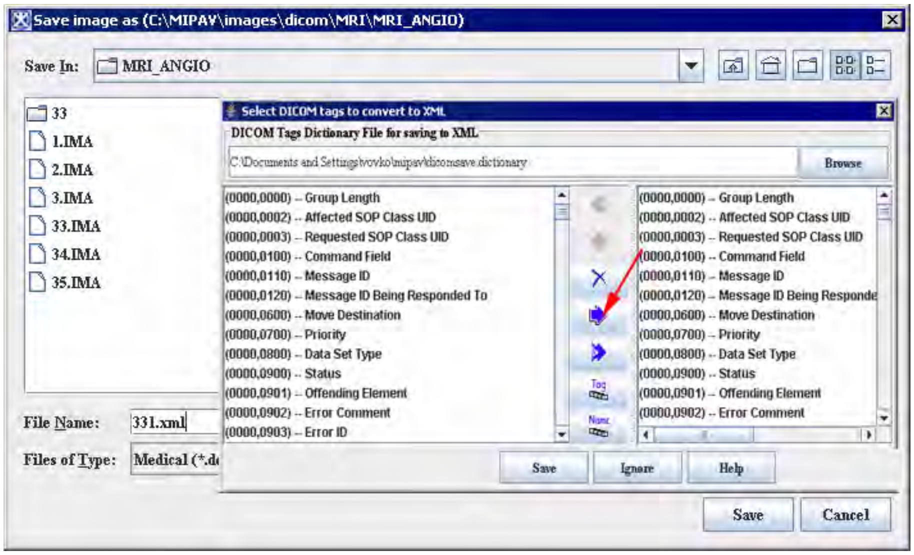


Figure 234. Saving a DICOM image as an XML file

To save other images as XML files

The Correct Image Spacing utility works only on XML files.

1. Open the image that has spacing problems if you have not already done so.
2. Select File > Save Image as in the MIPAV window. The Open dialog box appears.
3. Type the name for the file including the XML extension in File Name.
4. Select Medical, which includes XML files, in Files of Type.
5. Click OK. The Save Range of Slices dialog box appears. Complete the dialog box and press OK. See Figure 235.
6. The program saves the image as an XML file.

General Options	
Save image slices to separate files	If checked, allows to save the image slices as separate files. The slices will be saved as files with the file name contains the name of the original image followed by the slice number specified by a user, e.g. `patientX001`. Here `patientX` is the name of the original image and `001` is the number of the slice. Refer to "Understanding contiguous planes' effect on image scanning" on page 423.
First File Starting Number **File Name Number of Digits**	Set the start counter number and how many digits will appear in the counter number.
Choose Range of Slices to Save	
First Slice	Specify the number of the first slice.
Last Slice	Specify the number of the last slice.
Choose Range of Time Periods to Save	
First Time period	Specify the first time period.

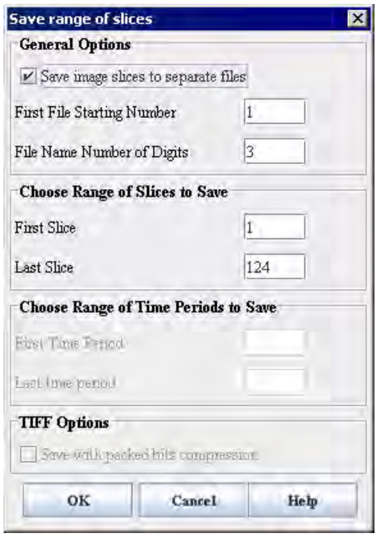


Figure 235. The Save Range of Slices dialog box options

Last Time Period	Specify the last time period.
TIFF options	
Save with packed bit compression	Packed bit compression uses a single control byte to indicate how the next byte or series of bytes will be used. The high bit in the control byte indicates if the following byte or bytes should be a used as a series or should be used as individual bytes. The remaining 7 bits indicate the number of bytes that are indicated in the control byte.
OK	Applies the changes you made in this dialog box, saves the image as a range of slices, and closes the dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not save the image as a range of slices.
Help	Displays online help for this dialog box.

Figure 235. The Save Range of Slices dialog box options (continued)

To correct image spacing

1 Open the image that contains spacing problems.

Tip: To determine whether images contain spacing problems, remember to view them using the triplanar view or to view the header file.

2 Save the image as an XML file.

3 Select Utility > Correct Image Spacing.

If the image file does have spacing problems, then the program runs the utility.

If the image spacing is correct, the program displays a message indicating that the spacing is correct.

Cropping images

You can use the Crop command on the Utilities menu to trim an image so that only the selected portion of the image remains.

To crop an image using a VOI

1 Open an image.

2 Draw one or more VOI contours on the image on the area that you want to remain in the image. When the utility is run, the area inside the contours remains and the areas outside the contour are discarded from the new dataset.

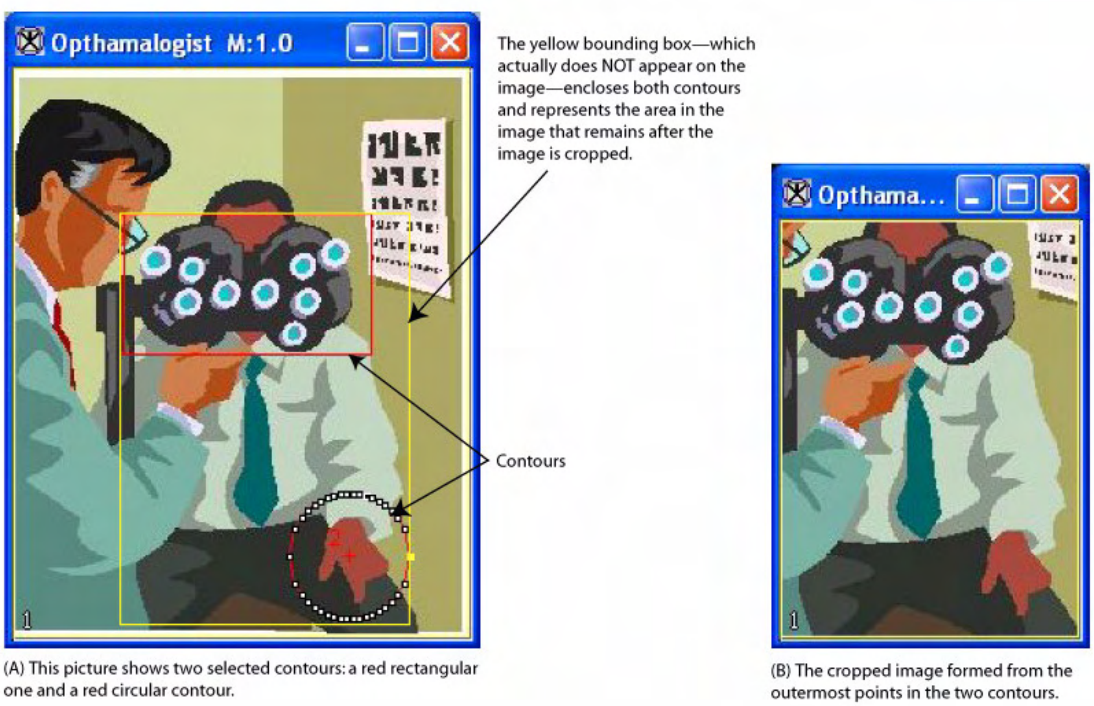


Figure 236. Contouring the area that should remain in the cropped image

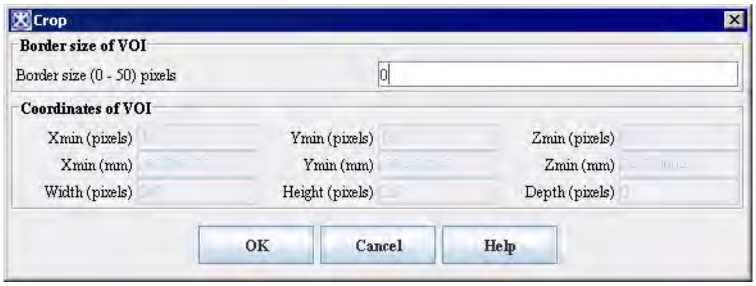
3 Select the VOI or one or more of the contours.

4 Select Utilities > Crop. The Crop dialog box (Figure 237) appears.

5 Type the number of pixels that you want to use for the border size in Border size box. Specify a number from 0 to 50.

6 Select either the New image or the Replace image check box to indicate whether the program should display the cropped image in a new image window or replace the original image with the cropped image.

7 Click OK. After a moment, the new image appears.



Border size of VOI: Border size (0 - 50)	Indicates the width of a strip of voxels that surround (border) the contours. When you crop an image, the area inside the contours and the border remain. All other voxels are trimmed from the image.
Coordinates of VOI	Displays the coordinates for the X, Y, and Z axes of the VOI.
New image	Shows the cropped image in a new image window.
Replace image	Replaces the current active image with the cropped image.
OK	Applies the parameters that you specified to crop the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not crop the image.
Help	Displays online help for this dialog box.

Figure 237. The Crop dialog box options

To crop an image using parameters

1 Open an image.

2 Call Utilities>Crop>Using parameters.

3 The Crop Boundary Pixels dialog box appears, see Figure .

4 Select either the New image or the Replace image check box to indicate whether the program should display the cropped image in a new image window or replace the original image with the cropped image.

5 Complete the dialog box.

6 Click OK. After a moment, the new image appears.

Pixels around image	Specify the number of pixels which should be cropped. **For 3D images** also specify the number of slices at the front and at the back of the image.
Select destination	**New Image** – shows the cropped image in a new image window. **Replace Image** – replaces the current active image with the cropped image.
OK	Applies the parameters that you specified to crop the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not crop the image.
Help	Displays online help for this dialog box.

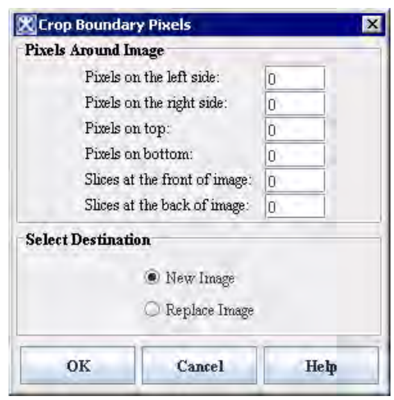


Figure 238. The Crop Boundary Pixels dialog box options

Masking (filling) images

Masks allow you to remove portions of images and display only those portions as separate image files. For example, you may only be interested in a section of an image and only want to display and work with that section.

To create a mask, you first, need to create one or more VOIs on the image. Depending on the type of mask you want to create, the VOIs should do either of the following:

- Enclose the image areas you want to preserve
- Enclose the image areas you do *not* want to preserve

If the VOIs indicate the areas of the image that should be deleted from the image, you need to create an *interior mask* and specify the number of pixels the program should use to hide the image inside the VOIs, or interior (Figure 239-B).

To preserve the image inside the VOIs, however, you need to create an *exterior mask*. An exterior mask fills the image outside of the VOIs with the number of pixels you specify. This filling hides the portions of the image outside the VOIs, or exterior (Figure 239-C).

A B

Figure 239. An original image (A), interior mask (B) and exterior mask (C)

To create an interior mask

1 Open an image.

2 Create one or more VOIs that enclose the portions of the image that interest you.

3 Select Utilities > Fill. The Fill dialog box (Figure 240) opens.

4 Type the number of pixels to fill the interior of the VOIs.

5 Select Interior fill.

6 Select either New Image or Replace Image depending on which you prefer.

7 Click OK. The program either displays a new image or replaces the current image with an image that hides the portions of the image that are inside the VOIs (Figure -B).

To create an exterior mask

1 Open an image.

2 Create one or more VOIs that enclose the portions of the image that interest you.

3 Select Utilities > Fill. The Mask dialog box (Figure 240) opens.

4 Type the number of pixels to fill the interior of the VOIs.

5 Select Exterior fill.

6 Select either New Image or Replace Image depending on which you prefer.

7 Click OK. The program either displays a new image or replaces the current image with an image that hides the exterior of the image outside the VOIs (Figure -C).

Parameters	Specifies the value in pixels to fill the VOI.
Interior fill	Applies the value to the interior of the VOI.
Exterior fill	Applies the value to the exterior of the VOI.
New image	Shows the cropped image in a new image window.
Replace image	Replaces the current active image with the cropped image.
OK	Applies the parameters that you specified to mask the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not mask the image.
Help	Displays online help for this dialog box.

Figure 240. The Fill Image dialog box

Flipping images

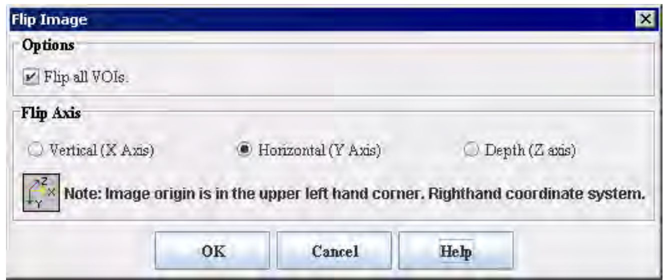
MIPAV allows you to flip images either horizontally, or vertically, or invert the order of slices in the image dataset. The last option is available via the Utilities>Flip Image> In Depth menu (refer to page 437 for more information).

To flip an image,

1 Call Utilities > Flip > Horizontal, Vertical of In Depth.

2 Then, complete the dialog box that appears, and press OK.

3 After a few moments, the flopped image replaces the original image in the same image window. For dialog box options, refer to Figure 241.



Options	**Flip all VOIs** – flips an image along with all VOIs.
Flip Axis	Vertical (X Axis) – flips the image vertically; Horizontal (Y Axis) – flips the image horizontally; Depth (Z Axis) – flips the image about the Z axis.
OK	Applies the parameters that you specified to flip the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not flip the image.
Help	Displays online help for this dialog box.

Figure 241. The Flip Image dialog box options

A B C

Figure 242. An original image (A), image flipped horizontally (B), and then vertically (C)

To invert the order of images in datasets

1. Open an image that contains slices.
2. Select Utilities > Flip> Flip in Depth.

   A status message appears and indicates that the software is reordering the slices in the dataset. In a moment or two the MIPAV window refreshes, and the image window in which the original dataset appeared displays the reordered slices.

Image Calculator

This algorithm adds, subtracts, multiplies, and divides, etc. the pixel values of one image by the pixel values of another image. In addition, two images can be ANDed, ORed or XORed together. Also, more advanced operator expressions can be entered in the dialog text field.

Background

Image Calculator performs arithmetic and logical operations between two images selected from popup menus.

The commands in the main Image Calculator dialog box take the pixel values of Image A, and then add, subtract, multiply, etc. them to the pixel values of Image B. For 2D and 3D images the specified operation is performed on all image slices. Refer to Figure 255.

The advanced dialog options (refer to page 449) allow performing advanced math operations on both images and also performing math operations on the selected image using the specified constant, e.g. *Image A*C*, where *C* is a constant.

CLIP MODE

When clip mode is selected and the result value is greater than the maximum value of Image A's data type, the value is clamped at the maximum value of Image A's data type. When clip mode is selected and the result value is less than the minimum value of Image A's data type, then the value is clamped at the minimum value of Image A's data type. See Figure 255.

PROMOTION MODE

When the Promote Destination Image Type mode is selected and the result value is not within the legal range of Image A data type, then the result image type is promoted from the default of the Image A data type to a new data type with a data type range capable of expressing the minimum and maximum values.

Promotion sequence:

- Boolean is promoted to byte, byte is promoted to unsigned byte, unsigned byte is promoted to short, short is promoted to unsigned short, unsigned short is promoted to integer, integer is promoted to unsigned integer, unsigned integer is promoted to long, long is promoted to float, and float is promoted to double. Double is not promoted.
- Color with 3 bytes is promoted to color with 3 unsigned shorts, and color with 3 unsigned shorts is promoted to color with 3 floats. Color with 3 floats is not promoted.
- Complex is promoted to double complex.
- Double complex is not promoted.

IMAGE TYPES

You can apply Image Calculator to all 2D and 3D color and grayscale images. Image A and Image B must have the same number of dimensions and the length of every dimension in Image A must equal the length of the same dimension in Image B. Also, both Image A and Image B must be color images, or both Image A and Image B must be black and white images. The Image Calculator will not accept one black and white Image And one color image. There are no other restrictions on data type.

Image calculator options

ADD

Adds the pixel values of Image A to the pixel values of Image B. Basically, this operation adds the colors of the overlay to the background causing the two images ether overflow, or saturate. However, if the added colors excess the color limits, the color will be capped (or clamped) and the result will not necessarily be as you expect.

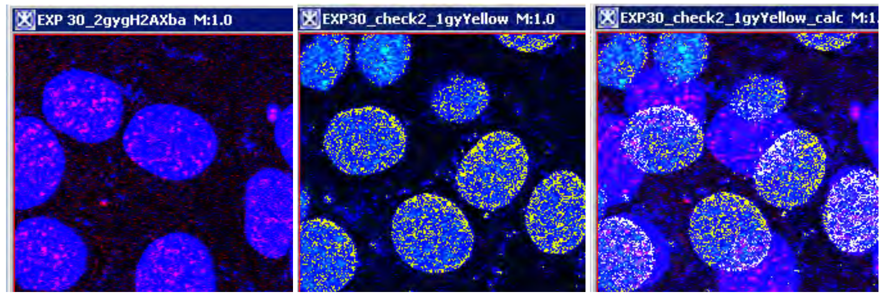


Image A sample point (255.0,0.0,255.0)

Image B (0.0,12.0,255.0)

Result= Image A + Image B (255.0,12.0,255.0)

Figure 243. Adding Image A and Image B

AND

Does AND of the Image A and the specified Image B. For each pixel in Image A, the Image A pixel value is set to zero, if either the Image A pixel value equals zero, or corresponding Image B pixel value equals zero. Otherwise, the Image A pixel value is left unchanged.

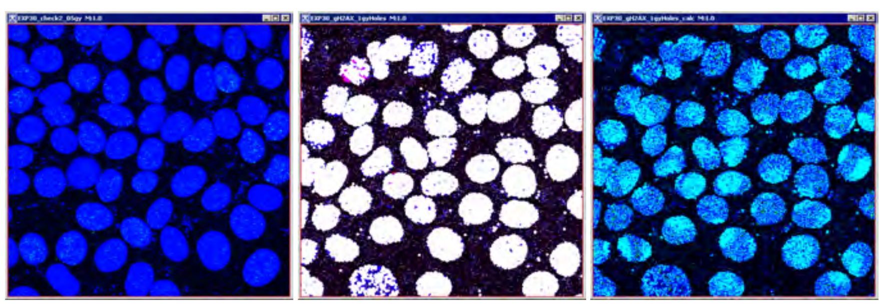


Image A

Image B

Result=(Image A AND Image B)

Figure 244. ANDing images

AVERAGE

Average of two images is calculated as (Image A + Image B)/2.

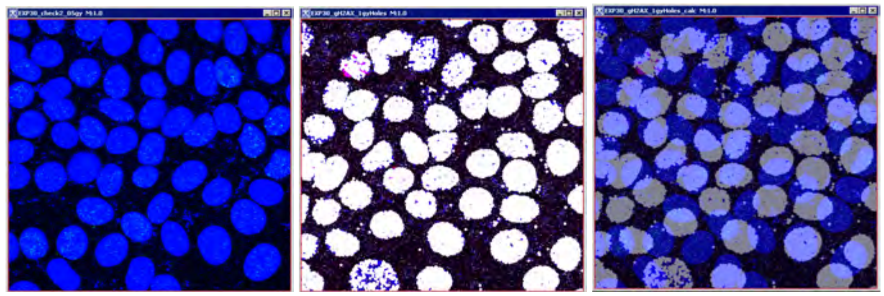


Image A Image B (Image A+Image B)/2

Figure 245. Averaging images

MAXIMUM

RGB values for pixels in the Image A and the corresponding RGB values for the Image B are compared, and then replaced with the greater RGB values. As shown in Figure 246 below, for a chosen point, it took MAX R=12.0 from Image B, MAX G=30.0 from Image A, and MAX B=243.0 from Image A.

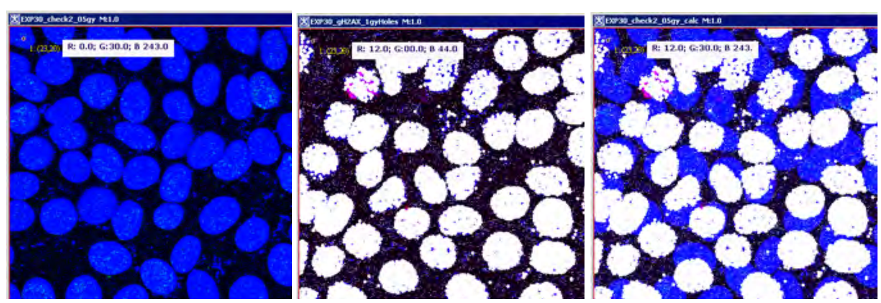


Image A, for the selected point RGB is (0.0, **30.0**, **243.0**)

Image B, RGB (**12.0**, 0.0, 44.0)

Result=MAX (Image A, Image B), RGB (12.0, 30.0, 243.0)

Figure 246. Maximum (Image A; Image B)

MINIMUM

RGB values for pixels in the Image A and the corresponding RGB values for the Image B are compared, and then replaced with the minimum RGB values. As shown in Figure 247 below, for a chosen point, it took MIN

R=0.0 from Image A, MIN G=00.0 from Image B, and MIN B=44.0 from Image B.

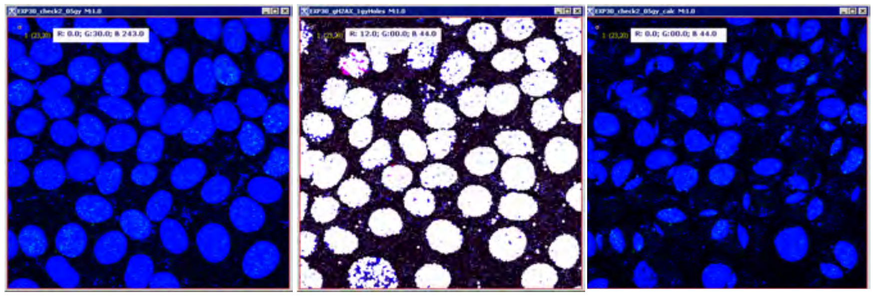


Image A, for the selected point RGB is (**0.0**, 30.0, 243.0)

Image B, RGB (12.0, **0.0**, **44.0**)

Result=MIN (Image A; Image B), RGB (0.0, 0.0, 44.0)

Figure 247. Minimum(Image A; Image B)

MULTIPLY

Multiplies the Image A pixel values by the Image B pixel values. In clip mode, for RGB images with byte values, results greater than 255 are set to 255. This method works very well if one image (either A, or B) is basically black or gray, or just has a light background, see Figure 248. If both images are color images, then you may get strange results. This technique is also perfect for overlaying line drawings, diagrams or images on an image with very light white or colored background. See also "clip mode" on page 438.

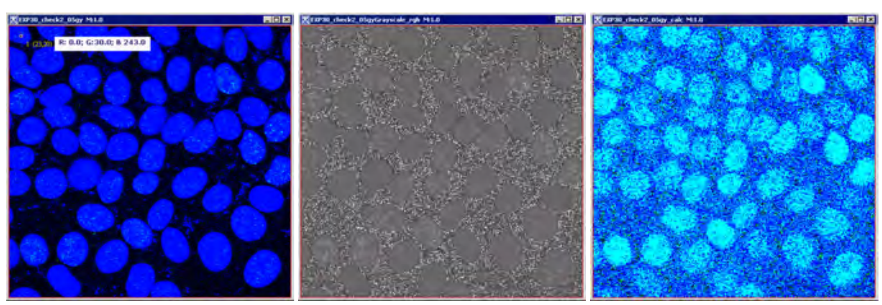


Image A

Image B

Image A*Image B

Figure 248. Multiply(Image A; Image B)

DIFFERENCE

The difference operation is a sequence of two operations. First, a subtraction operation is performed. Then, the absolute value of the result of the subtraction operation is taken. The variation between the colors black and white difference will produce a maximum result of white. However, any colors which are the same will produce black. You can use this function to see where images are similar and where they differ.

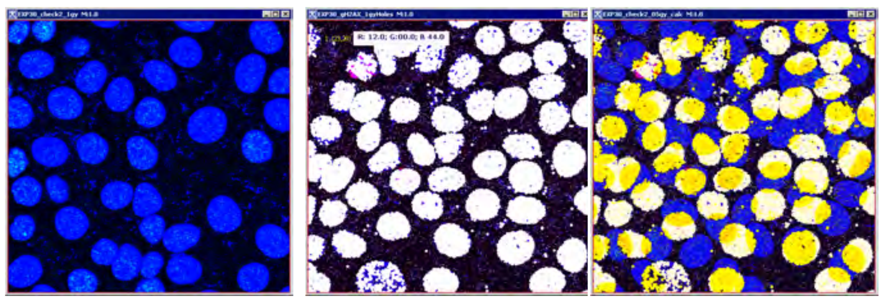


Image A Image B |Image A-Image B|

Figure 249. Difference (Image A; Image B)

DIVIDE

Divides the pixel values of Image A by the specified Image B. A positive number divided by zero could yield either the maximum positive value, or positive infinity. A negative number divided by zero could yield either the maximum negative value or negative infinity. Zero divided by zero could yield either zero or not a number.

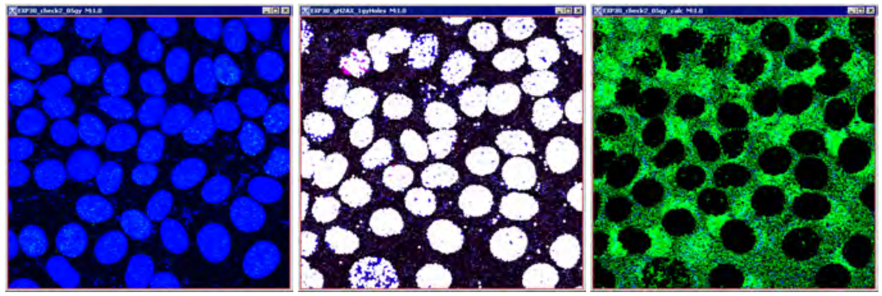


Image A Image B Image A/Image B

Figure 250. Dividing images

OR

If the pixel value from a chosen pixel in Image A equals zero, then the result pixel value is set equal to the corresponding Image B pixel value. Otherwise, the result pixel value is the same as for the Image A (i.e. stays unchanged).

- Result=Image B, if Image A=0;
- otherwise, Result=Image A.

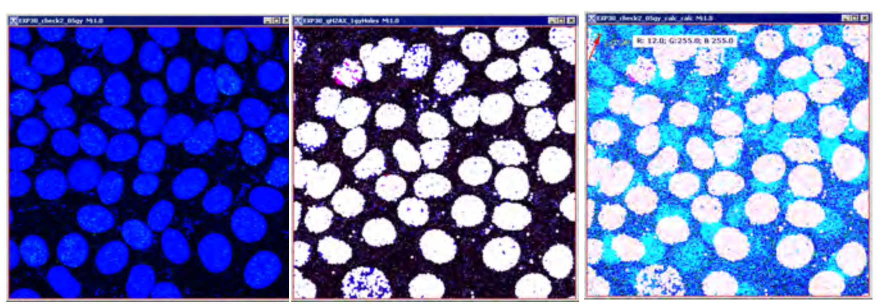


Image A Image B (Image A OR Image B)

Figure 251. OR(Image A; Image B)

XOR

It overlays two images together, but then clears the area that is overlapped back to transparency. If both Image A and Image B pixel values are nonzero, then the result image is set to zero. If both Image A and Image B pixel values are zero, then the result image remains zero. If only one of Image A and Image B is nonzero, then the result retains or assumes the nonzero value.

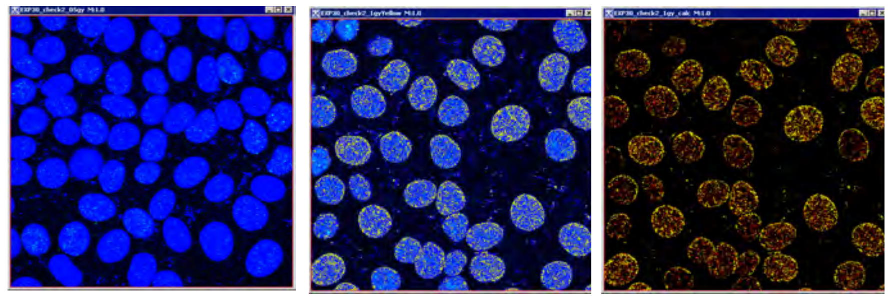


Image A Image B (Image A XOR Image B)

Figure 252. XOR(Image A; Image B)

SUBTRACT

Subtracts the pixel values of Image B from the corresponding pixel values of Image A.

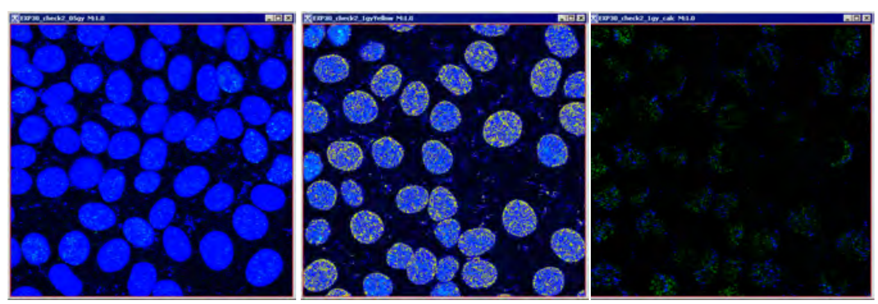


Image A Image B Image A-Image B

Figure 253. Subtract(Image A; Image B)

Advanced math options

Abs

Generates the absolute value of the active image or selection.

Exp

Applies the function *f(p) = exp(p)* to each pixel (p) in the image or selection. For RGB images, this function is applied to all three color channels.

Ln

Applies the function *f(p) = ln(p)* to each pixel (p) of the image or selection. For RGB images, this function is applied to all three color channels.

Log

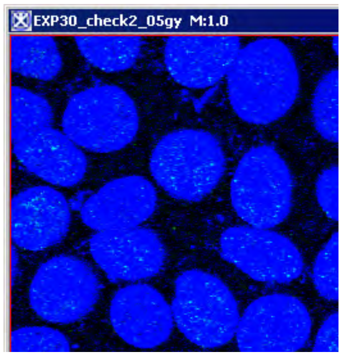
Applies the function *f(p) = log(p)* to each pixel (p) in the image or selection. For RGB images, this function is applied to all three color channels.

Pow

The function *pow (x,y)* produces a value of x raised to the power of y, e.g., *pow(2,3)* gives 8. For RGB images, this function is applied to all three color channels. In clip mode, for RGB images with byte values, results greater than 255 are set to 255. In promote destination image mode, the image will be promoted to ARGB_USHORT and capable of storing color values of up to 65535.

SIN, COS, TAN

Apply the functions *sin(p)*, *cos(p)*, and *tan(p)* to each pixel (p) of the image or selection. For RGB images, the chosen function is applied to all three color channels.



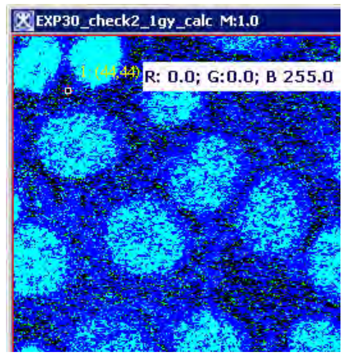


Image A f(p)=pow(Image A;3)

Figure 254. Image A after applying f(p)=pow(Image A;3). The function was applied for all three color channels. The result values greater than 255, as for the B channel, were set to 255

Applying the Image Calculator to images

To run this algorithm, complete the following steps:

1. Open at least two images. The images must have the same dimensions.
2. Select Utilities >Image Calculator. The dialog box opens. See Figure 255.
3. Use the Operator list box to select the math operator (Add, AND, Average, etc.).
4. Use the Image B list box to select Image B.
5. Check the Clip option if you want to clamp result data to the bounds of the input image type. Check the Promote option to promote image type so that the range of the result fits into the new image type. Refer to “clip mode” on page 438 and “promotion mode” on page 438.
6. Click OK. The algorithm begins to run.

7 Depending on whether you selected New Image or Replace Image A, the result appears in a new window or replaces the image to which the algorithm was applied.

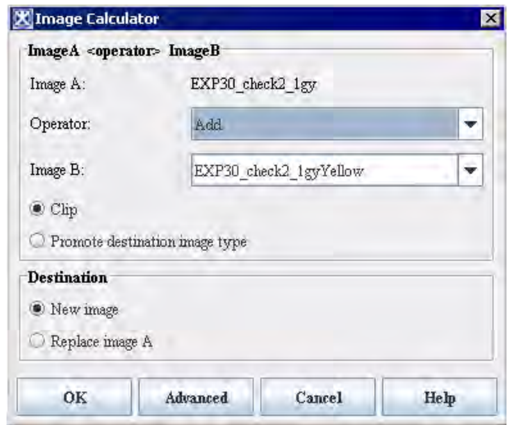


Image A	Image A.
Operator	Select the math operator from the list.
Image B	Select Image B from the list.
Clip	Select this option if you want the result image to be clamped at the maximum (minimum) value of Image A data type. See also "clip mode" on page 438.
Promote destination image type	When this mode is selected and the result value is out of the legal range of Image A data type, then the result image type is promoted to a new data type with a data type range capable of expressing the minimum and maximum values. See also "promotion mode" on page 438.
Destination	
New image	The result image will open in a new image frame.
Replace image	The result image will replace Image A.
OK	Applies the algorithm according to the specifications in this dialog box.
Advanced	Calls the Enter Advanced Operator dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes it.
Help	Displays online help for this dialog box.

Figure 255. The Image Calculator dialog box options

Advanced Image Calculator Options

To use advanced options provided by the Image Calculator dialog box, complete the following steps:

1. Open at least two images. The images must have the same dimensions.
2. Select Utilities >Image Calculator. The dialog box opens.
3. Use the Image B list box to select Image B.
4. Check the Clip option if you want to clamp result data to the bounds of the input image type. Check the Promote option to promote image type so that the range of the result fits into the new image type.
5. Use the Advanced button to call the advanced options. The Enter Advanced Operator dialog box appears.
6. Use the dialog box options to perform any of the standard operations for which you would normally use a handheld calculator. Advanced operator performs on the images basic arithmetic, such as addition and subtraction, as well as functions found on a scientific calculator, such as logarithms and exponent.

Note: that in this dialog box you can combine images and constants in one math operation.

7. Click OK. The algorithm begins to run.
8. Depending on whether you selected New image or Replace Image A, the result appears in a new window or replaces the Image A to which the algorithm was applied.

Enter Advanced Operator dialog box

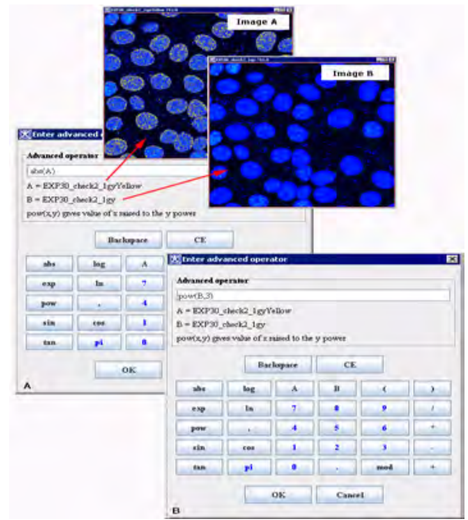
A= [file name]	Image A.	
B=[file name]	Image B.	
All available math functions can be entered in the Advanced operator text box manually or using the corresponding buttons.		
Numbers also can be entered in the Advanced Operator text box manually or using the calculator buttons.		
Backspace Uses the same function as the Backspace button in the handlend calculator.		
CE Uses the same function as the CE button in the handlend calculator. **OK** Applies the algorithm according to the specifications in this dialog box.		
Cancel	Disregards any changes that you made in this dialog box and closes it.	
Help	Displays online help for this dialog box.	

Figure 256. The Enter Advanced Operator dialog box options

REFERENCES

The parsing of the advanced operator expression is largely based on the file *Func.java* by Leen Ammeraal. The Func.java file appears at *ftp://ftp.expa.fnt.hvu.nl/pub/ammeraal* and *http://home.wxs.nl/~ammeraal*

It is also contained in Chapter 8 of *Computer Graphics for Java Programmers* by Leen Ammeraal, Copyright 1998 by John Wiley & Sons Ltd.

Image Calculator–Bulk Images

As for now, the Image Calculator–Bulk Images utility adds the pixel values of selected images. Basically, this operation adds the colors of the overlay to the background causing the two images ether overflow, or saturate. However, if the added colors excess the color limits, the color might be capped (or clamped) and the result might not necessarily be as you expect. See also Image Calculator, "clip mode" and "promotion mode" on page 438.

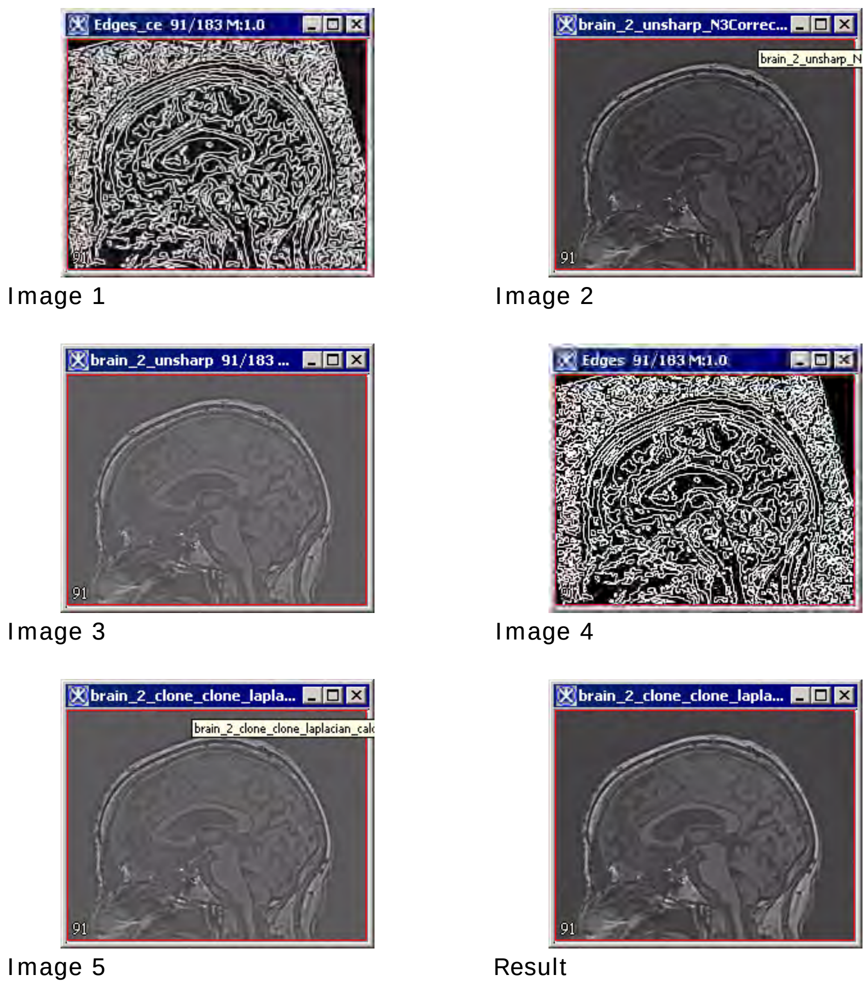


Figure 257. Adding images using the Image Calculator–Bulk Images utility

IMAGE TYPES

You can apply Image Calculator–Bulk Images to all 2D and 3D color and grayscale images. Images must have the same number of dimensions and the length of every dimension in each image must be the same. Also, all images must be color images, or grayscale images. There are no other restrictions on data type.

Applying the Image Calculator – Bulk Images

To run the utility, complete the following steps:

1. Open images which you would like to add. The images must have the same dimensions.
2. Select Utilities >Image Calculator–Bulk Images. The dialog box opens. See Figure 258.
3. The list of images appears in the Images window.
4. If you want to remove a certain image(s), select it, and then press Remove.
5. If you want to add images to the list press Browse, and then select the image(s) from your disk.
6. Check the Clip option, if you want to clamp result data to the bounds of the input image type. Check the Promote option to promote image type so that the range of the result fits into the new image type. Refer to “clip mode” on page 438 and “promotion mode” on page 438.
7. Click OK. The algorithm begins to run. And the result image appears in a new image frame.

Figure 258. Image Calculator–Bulk Images dialog box

Image Math

Algorithm that adds, subtracts, multiplies, or divides an image by some user specified value. In addition, the square root, absolute value, or log of an image can be calculated. If the new image exceeds the range that can be stored in an image of that type, the data is either clipped and stored in the original image or a new image of a type (int, float...) that can store the range of new data is generated.

See also: "Image Calculator", Sections "clip mode" and "promotion mode" on page 438.

IMAGE TYPES

You can apply Image Calculator to all 2D and 3D grayscale images. And here is the difference between Image Math and Image Calculator (refer to "Image Calculator" on page 438). The last one you can ally to all types of images 2D, 3D, 4D color and grayscale.

Applying the Image Math to images

To run this algorithm, complete the following steps:

1. Open an image of interest.
2. Select Utilities >Image Math. The dialog box opens. See Figure 259.
3. Use the Operator list box to select the math operator (Absolute value, Add, Average, etc.).
4. Use the Value text box to enter a numerical value.
5. Select the Clip option if you want to clamp result data to the bounds of the input image type. Select the Promote option to promote image type so that the range of the result fits into the new image type. Select the Convert to Float option to convert the result into float image type.
6. Click OK. The algorithm begins to run.
7. Depending on whether you selected New Image or Replace Image, the result appears in a new window or replaces the image to which the algorithm was applied. See also Figure 259.

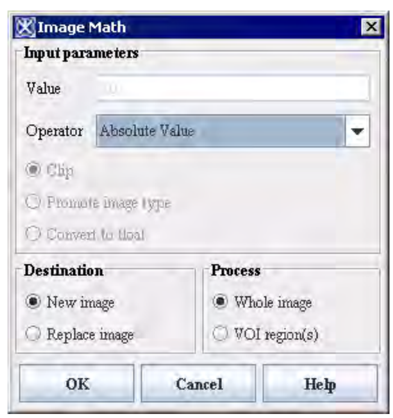
Value	Enter the numerical value here. Note that the text box become available after you select the math operator.	
Operator	Select the math operator from the list.	
Clip	Select this option if you want the result image to be clamped at the maximum (minimum) value of the original image data type.	
	See also "clip mode" on page 438.	

Figure 259. The Image Math dialog box options

Promote image type	When this mode is selected and the result value is out of the legal range of the original image data type, then the result image type is promoted to a new data type with a data type range capable of expressing the minimum and maximum values. See also "promotion mode" on page 438.
Convert to float	When this mode is selected and the result value is out of the legal range of the original image data type, then the result image type is converted to float data type.
Destination	
New image	The result image will open in a new image frame.
Replace image	The result image will replace the original image.
Process	
Whole image	The algorithm applies to the selected VOI(s) region.
VOI region	The algorithm applies to the whole image.
OK	Applies the algorithm according to the specifications in this dialog box.
Advanced	Calls the Enter Advanced Operator dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes it.
Help	Displays online help for this dialog box.

Figure 259. The Image Math dialog box options (continued)

EXAMPLES OF USING IMAGE MATH

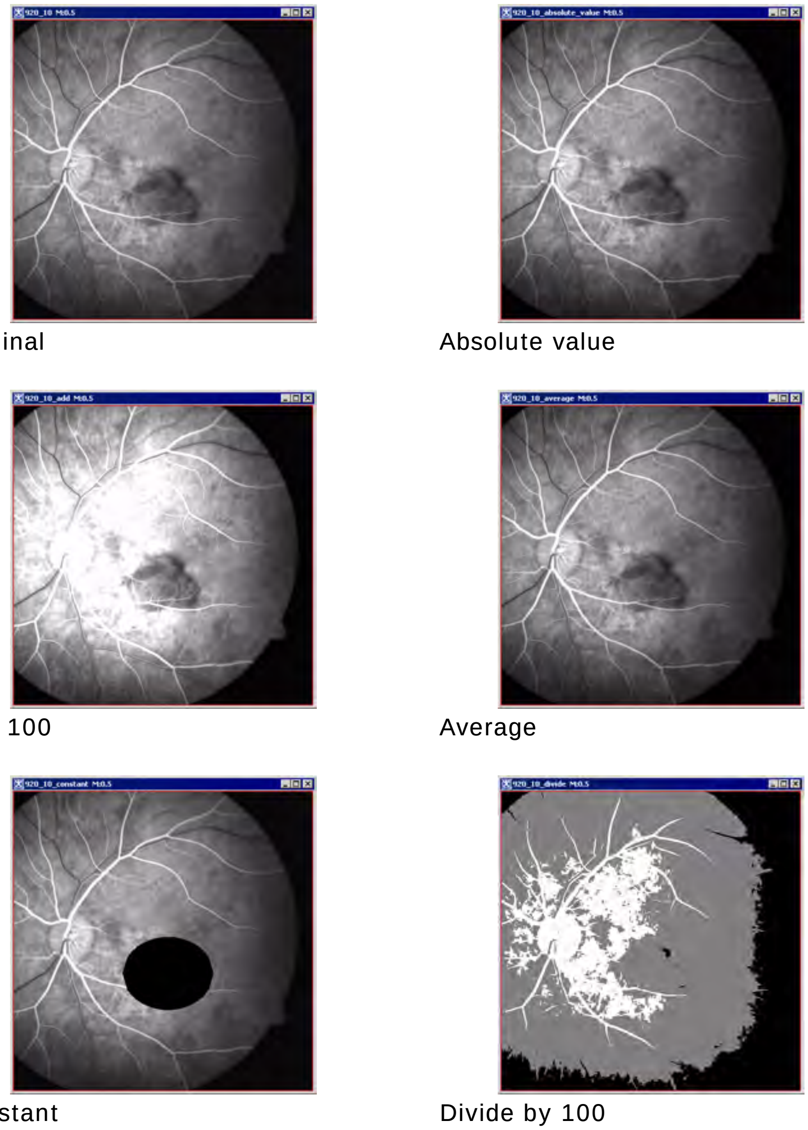


Figure 260. Applying Image Math to images


Log


Multiply by 5


Square


Square root


Subtract 5


Sum

Figure 260. Applying Image Math to images (continued)

Inverting the image

The Utilities>Invert command inverts the colors of the image, creating a photo-negative effect. There are no options for this effect. Simply select New Image to send the inverted image to a new image frame or Replace Image to overwrite the file with your changes, or click Cancel to discard the changes and leave the image as-is.

Figure 261. The Invert Image dialog box

Matching images

The Match Images command on the Utilities menu assists in any comparison of two images. To use this utility, you need to first open two images: image A, which has the desired characteristics, and image B, whose characteristics need to match those of Image A. The Match Images dialog box presents the following four ways in which you can match the images:

- **Match image orientation**—Rotates Image B so that its orientation is the same as the orientation of image A.
- **Match resolutions**—Subsamples the image for each dimension with lower resolution (i.e., larger pixel size) so that both images have the same resolution. To ensure that the fields of view remain the same, the program adds extra pixels to the image.
- **Match origins**—Adds margins to one of the images for each dimension if the origins (i.e., the starting location) of the two images are not the same. The program also adds pixels to the left, top, or front of the image.
- **Match image dimensions**—Adds pixels to the right, bottom, or back of either image so that they have the same dimensions.

You can select one or more of these choices in the dialog box. Before actually performing any selected comparison, MIPAV first checks to see whether the images do not already match in that regard. If they do, the given match is not performed.

To match image orientation

1 Open two images.

2 Select Utilities > Match images. The Match Images dialog box (Figure 262) opens.

3 Select one of the open images as image A, and select the other image as image B.

4 Select Match image orientations (based on imaging orientation).

5 Select any or all of the other check boxes to also, if desired, match image resolutions, origins, or dimensions.

6 Click OK. If the orientations of the images differ, the program changes the orientation of Image B to match that of Image A.

To match image resolutions

1 Open two images.

2 Select Utilities > Match images. The Match Images dialog box (Figure 262) opens.

3 Select one of the open images as image A, and select the other image as image B.

4 Select Match image orientations (based on imaging orientation).

5 Select any or all of the other check boxes to also, if desired, match image resolutions, origins, or dimensions.

6 Click OK. If the resolutions of the images differ, the program changes the resolution of Image B to match that of image A.

To match image origins

1 Open two images.

2 Select Utilities > Match images. The Match Images dialog box (Figure 262) opens.

3 Select one of the open images as image A, and select the other image as image B.

4 Select Match image origins (by adding margins where necessary).

5 Select any or all of the other check boxes to also, if desired, match image orientations, resolutions, or dimensions.

6 Click OK. If the origins of the images differ, the program changes the origin of Image B to match that of image A.

To match image dimensions

1 Open two images.

2 Select Utilities > Match images. The Match Images dialog box (Figure 262) opens.

3 Select one of the open images as image A, and select the other image as image B.

4 Select Match image dimensions.

5 Select any or all of the other check boxes to also, if desired, match image orientations, resolutions, or origins.

6 Click OK. If the dimensions of the images differ, the program changes the dimension of Image B to match that of image A.

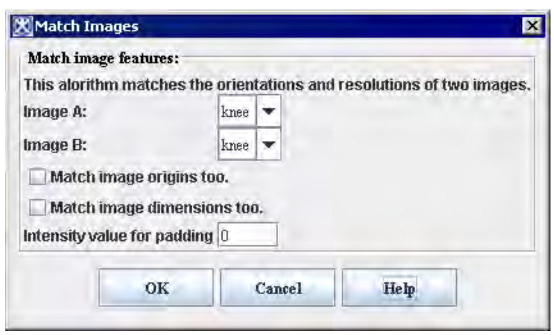


Image A	Specifies the image that contains the orientation, resolution, origins, or dimensions with which Image B needs to match.
Image B	Specifies the image on which to perform the image matching.
Match image orientations (based on imaging orientation)	Applies the image orientations (based on image orientation) used in Image A to Image B.
Match image resolutions, axis by axis (subsampling image with lower resolution)	Matches the image resolutions in Image A to Image B. Note that this subsamples Image A with a lower resolution if necessary.

Figure 262. Match Images dialog box

Match image origins (by adding margins where necessary)	Matches the image origins used in Image A to Image B. Note that, if necessary, this adds margins to Image B.
Match image dimensions	Applies the image dimensions used in Image A to Image B.
OK	Applies the parameters that you specified to add margins to this image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not perform image matching.
Help	Displays online help for this dialog box.

Figure 262. Match Images dialog box (continued)

Maximum Intensity Projection

Maximum Intensity Projection (MIP) is a volume rendering technique for 3D images that projects in the visualization plane the voxels with maximum intensity that fall in the way of parallel rays traced from the viewpoint to the plane of projection. At each pixel the highest data value encountered along the corresponding viewing ray is determined.

MIP exploits the fact, that within MRI data sets the intensity values of vascular structures are higher than the intensity values of the surrounding tissue. By depicting the maximum intensity value seen through each pixel, the structure of the vessels contained in the image can be captured.

The method provides a very good understanding of the structures defined by high signal intensities. It also helps to avoid the problem with occluding structures, which can block visualization of thin inner parts.

Background

A MIP algorithm accepts a single grayscale 3D image and generates three 2D images representing the maximum intensities in x, y, and z directions.

Let **I** be an input grayscale 3D image of size ($m*n*l$). Let **X**, **Y,** and **Z** be the output 2D images representing the maximum intensities in x, y, and z directions.

X is a 2D image of size ($y*z$) formed by viewing along the x-axis and selecting the highest intensities in the $y-z$ plane. **Y** image is of size ($x*z$) formed by viewing along the *y*-axis and selecting highest intensities in the $x-z$ plane. Similarly, **Z** image is formed by viewing along the Z axis and selecting highest intensities in the $x-y$ plane and is of size $(x * y)$.

Note: The MIP algorithm allows to set the minimum and maximum thresholds on the image before computation which aids in enhancing MIP visualization in certain cases.

Image Types

This algorithm works with 3D grayscale images (all image types except complex). By default, the result images are of type float.

Special Notes

The origin of the result images is at the top left corner and the original resolutions of the 3D image in all directions are preserved.

Applying the Maximum Intensity Projection

To run the method,

1. Open an image of interest.
2. Select Utilities > Maximum intensity projection (MIP).
3. The Maximum Intensity Projection dialog box appears.
4. Complete the dialog box. You might choose to fill out two input fields – Threshold Minimum and Threshold Maximum; or just use the default values. These fields are initially populated with the minimum and maximum intensities of the image.
5. Click OK.

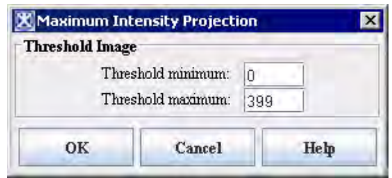


Figure 263. The Maximum Intensity Projection dialog box. The input fields are initially populated with the maximum and minimum intensities of the image

The algorithm begins to run, and a pop-up window appears with the status. The following message appears: *Computing Maximum Intensity Projection*. When the algorithm has finished to run, the pop-up window closes and the three 2D images appear in three different windows. See Figure 264.

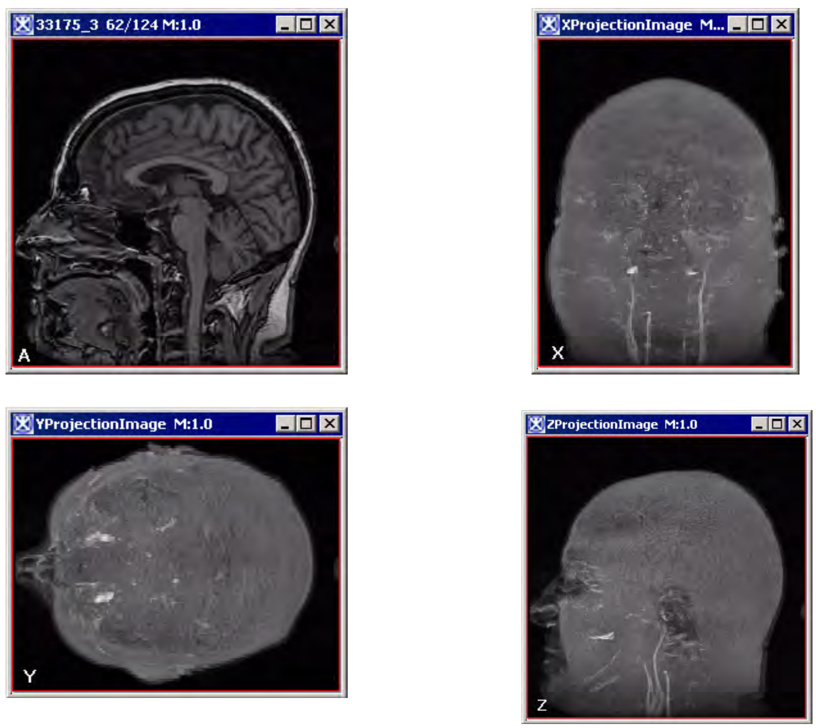


Figure 264. The original image (A) and three 2D result images; x–projection (X), y–projection (Y), and z–projection (Z)

Adding noise to images

Adding noise to images allows you to test the robustness and performance of an algorithm in the presence of known amounts of noise. When you select Utilities > Noise, the program clamps either Gaussian or Uniform noise to the lowest or highest value in the source image type.

For example, for a byte image, if the intensity of the source pixel is 120 and noise is 15, then intensity + noise = 135. This value (135) would be clamped to the maximum pixel value for a byte image (127).

This class relies heavily on the Java Random class and is used to generate a stream of pseudorandom numbers. The class uses a 48-bit seed, which is modified using a linear congruency formula (refer to Donald Knuth, *The Art of Computer Programming*, Volume 2, Section 3.2.1.).

To add noise to images

1. Open an image on which you want to test the effectiveness of an algorithm.
2. Select Utilities > Noise in the MIPAV window. The program displays the Additive Noise dialog box (Figure 265).
3. Type the level of noise that you want to add to the image in the Noise level box. You can specify a value from 10 to 32,768.

Noise level (0-32768)	Specifies the level of noise to add to the image. The default value is 10.
Noise Type	
Gaussian	Adds Gaussian noise to the image.
Uniform	Adds Uniform noise to the image.
Destination	
New image	Shows the cropped image in a new image window.
Replace image	Replaces the current active image with the cropped image.

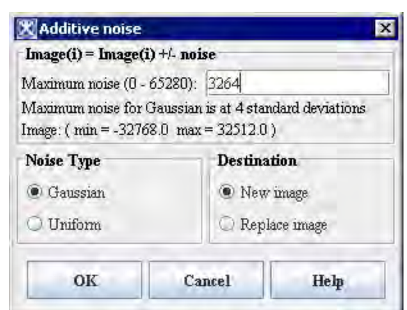


Figure 265. Additive Noise dialog box

OK	Applies the parameters that you specified for adding noise to this image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not add noise to this image.
Help	Displays online help for this dialog box.

Figure 265. Additive Noise dialog box (continued)

4 Type the level of noise that you want to add to the image in the Noise level box. You can specify a value from 10 to 32,768.

5 Select either of the following:

- *Gaussian*—To add Gaussian noise to the image
- *Uniform*—To add Uniform noise to the image

6 Select either of the following:

- *New image*—To generate the resulting image in a new image window
- *Replace image*—To replace the current image with the resulting image

7 Click OK. The program adds the Gaussian or Uniform noise to the image and displays either a new image or replaces the current image with the one to which noise was added.

8 Now, select the algorithm in the Algorithms menu to test its effectiveness on the image.

Example

In the following example (Figure 266), the original image appears at the top of the figure. The next row shows the image on the left that results from adding Uniform noise of 1000. To its right is the image after running the Median algorithm. The image on the left on the last row results from adding Gaussian noise of 1000 to the original image. Running the Median algorithm on that image produces the image on its right. In this example, the Median algorithm performs better on an image containing Gaussian noise.

A

A is an original image

B is the image after applying Uniform noise (1000)

C is the image B after applying the Median filter

D is the image after adding Gaussian noise (1000)

F is the image D after running the Median filter on it

Figure 266. Examples of adding a noise to an image, and then using it to test an algorithm's effectiveness in removing the noise

Pad

The Pad command on the Utilities menu allows to pad images with certain values specified by the user.

To use the Pad utility

1 Open an image of interest.

2 Call Utilities>Pad.

3 The Pad Image dialog box appears. Enter the number of pixels to be padded on the left side, right side, top and bottom. If the image is a 3D image, you can also add a number slices to the front or back of the image.

4 Specify the intensity value for padded pixels.

5 Click OK. Depending on the selected option – New Image or Replace Image – the padded image appears either in a new image frame or replaces the original image.

Pixels Around Image	
Pixels on the left side	Specify the number of pixels to add to the left side of the image.
Pixels on the right side	Specify the number of pixels to add to the right side of the image.
Pixels on top	Specify the number of pixels to add to the top of the image.
Pixels on bottom	Specify the number of pixels to add to the bottom of the image.
Slides at the front of image	Specify the number of slices to add to the front of the image.
Slides at the back of image	Specify the number of slices to add to the back of the image.
Pad value	Specify the intensity value for the added pixels.

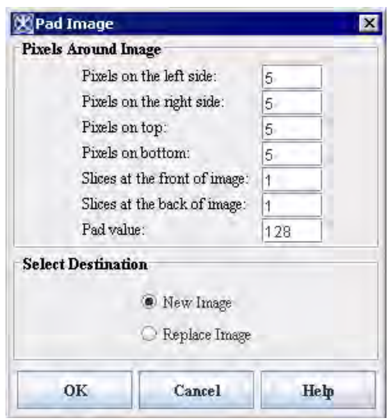


Figure 267. The Pad Image dialog box

Destination	
New image	Shows the padded image in a new image window.
Replace image	Replaces the current active image with the padded image.
OK	Applies the padding values that you specified to the image.
Cancel	Disregards any changes you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

Figure 267. The Pad Image dialog box (continued)

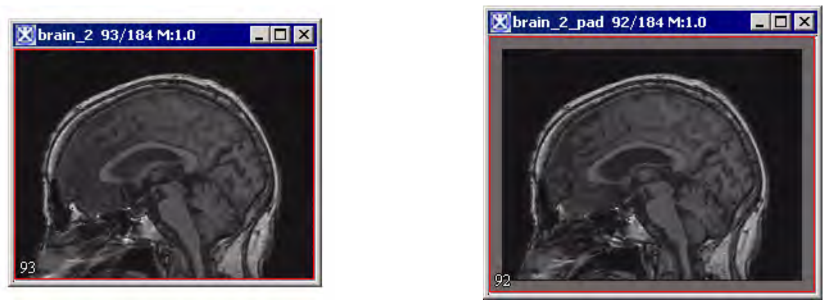


Figure 268. The original image (on your left) and the image padded to (10;10;10;10). The intensity value for padded pixels is set to 200

Quantify Mask

There are two algorithms Quantify Mask(s) and Quantify Using Mask that calculate Center of Mass, area (in resolutions), and number of pixels for a selected mask(s). The algorithms work with Boolean, byte, unsigned byte, and short masks.

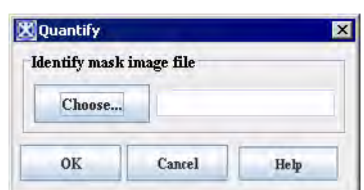
Identify Image Mask file		
Choose	Press to select an image mask from your hard drive.	
OK	Applies the Quantify masks(s) or Quantify Using Mask algorithms to the image(s).	

Figure 269. The Quantify dialog box

Cancel	Disregards any changes you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

Figure 269. The Quantify dialog box

To apply the Quantify Mask(s) algorithm:

1. Open an image of interest.
2. Create a mask as described in Chapter 8, “Generating masks”.
3. Save the mask as Boolean, byte, unsigned byte, or short.
4. Alternatively, you can open the mask file from the disk using the Quantify dialog box options.

Note: the algorithm doesn't work for paint masks.

5. Call Utilities>Quantify> Quantify Mask(s).
6. The algorithm begins to run and the data appear in the Output window. See Figure 270.
7. Later, you can save the data using the File>Save Images option provided by the Output window.

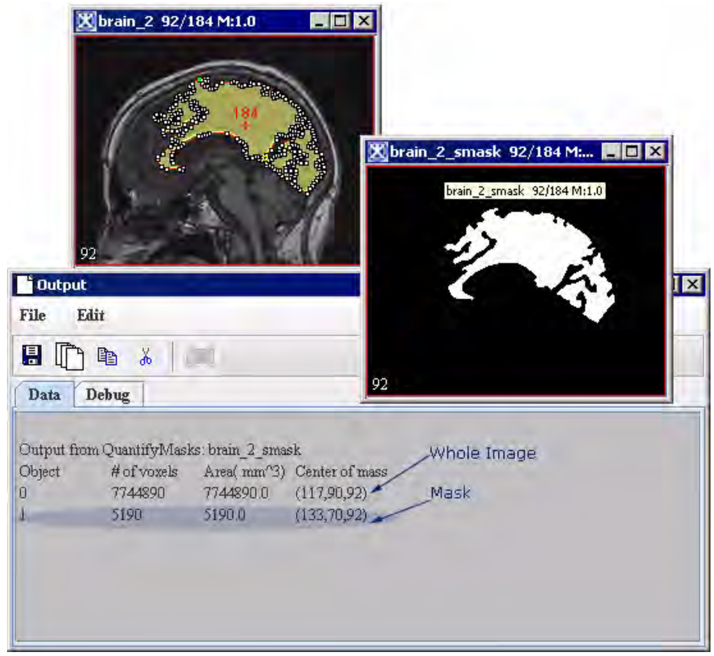


Figure 270. The Quantify Mask(s) algorithm

To apply the Quantify Using Mask algorithm:

1. Open an image of interest.
2. Create the image mask as described in Chapter 8, "Generating masks".
3. Alternatively, you can open the mask file from the disk.
4. Call Utilities>Quantify> Quantify Using Mask.
5. The algorithm begins to run and the data appear in the Output window. See Figure 271.
6. Later, you can save the data using the File>Save Images option provided by the Output window.

Note: the algorithm works with Boolean, byte, unsigned byte, and short masks, but it doesn't work with paint masks.

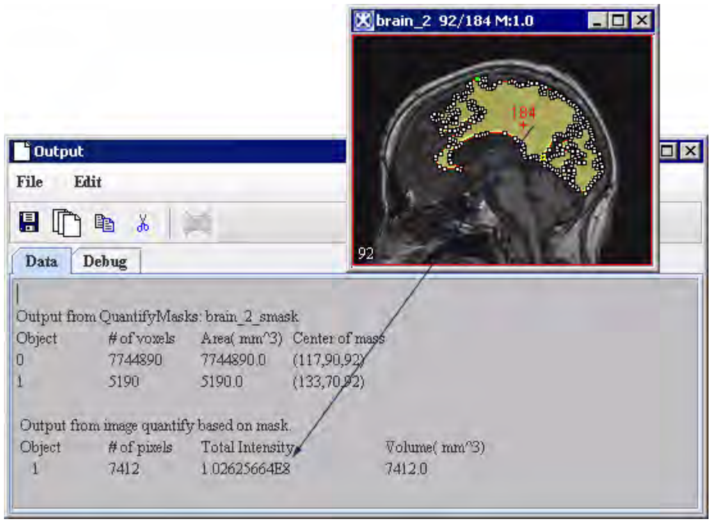


Figure 271. The Quantify Using Mask algorithm

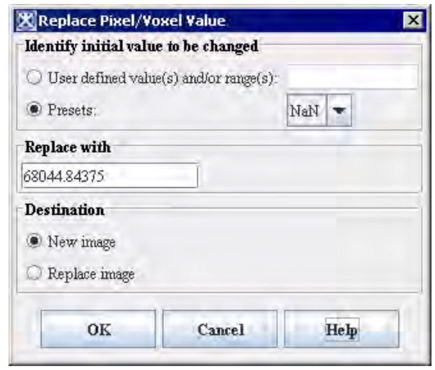
Replacing pixel/voxel value in images

You can replace the values of pixels or voxels in images by using the Replace pixel/voxel values command on the Utilities menu. Suppose, for example, that you have a set of images that contain *not a number* (NaN) values. To be able to use MIPAV algorithms on those images, you first need to replace those not a number values. The Replace pixel/voxel value command provides an easy way to replace those values or any pixel or voxel values.

To replace pixel or voxel value in images

1. Open the image whose pixels or voxels you want to change.
2. Select Utilities > Replace pixel/voxel values.

   The Replace Pixel/Voxel Value dialog box (Figure 272) opens.



User-defined value	Specifies the value that you want to replace in the image.
Presets	*NaN*—Not a number. —Positive infinity. —Negative infinity.
Replace with	Accept the default value or type another value.
New image	Shows in a new image window the image in which the value was replaced.
Replace image	Replaces the current image with the image in which the value was replaced.
OK	Applies the parameters that you specified to subsample the image.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not subsample the image.
Help	Displays online help for this dialog box.

Figure 272. Replace Pixel/Voxel Value dialog box

3 Do one of the following:

a Select User defined value and type the value that you want to replace in the text box

b Select Presets and then select one of the following in the list box:

- *NaN*—Not a number
- —Positive infinity
- —Negative infinity

4 Do one of the following:

- Accept the default value in Replace with.
- Type another value in the text box.

5 Select one of the following:

- *New image*—To create a new image in its own image window using the replaced value

- *Replace image*—To overwrite the original image in the original image window with an image using the replaced value

6 Click OK. The image with the replaced value appears in either a new window or in the original image window.

Rotating images

You can rotate images about the *x*, *y*, and *z* axes. For example, suppose that you opened the following image:

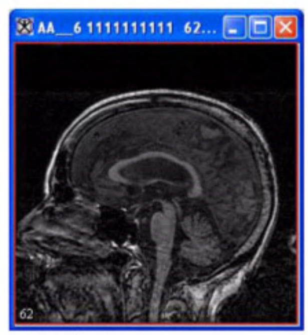


Figure 273. Original image to be rotated

To rotate images

1 Select Utilities > Rotate in the MIPAV window.

2 Select one of the following:

- *X Axis +90:* To rotate the image about the *x* axis by +90 degrees.
- *X Axis -90:* To rotate the image about the *x* axis by -90 degrees.
- *Y Axis +90:* To rotate the image about the *y* axis by +90 degrees.
- *Y Axis -90:* To rotate the image about the *y* axis by -90 degrees.
- *Z Axis +90:* To rotate the image about the *z* axis by +90 degrees.
- *Z Axis -90:* To rotate the image about the z axis by -90 degrees.

A status message appears. When rotation is complete, the status window closes, and MIPAV replaces the original image in the image window with the rotated image.

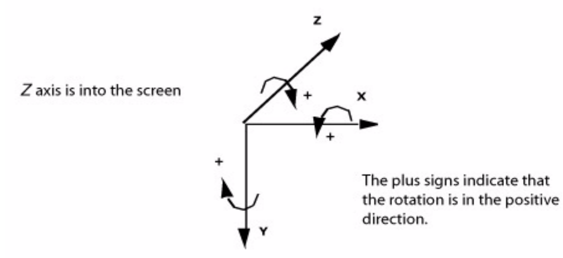


Figure 274. Rotation

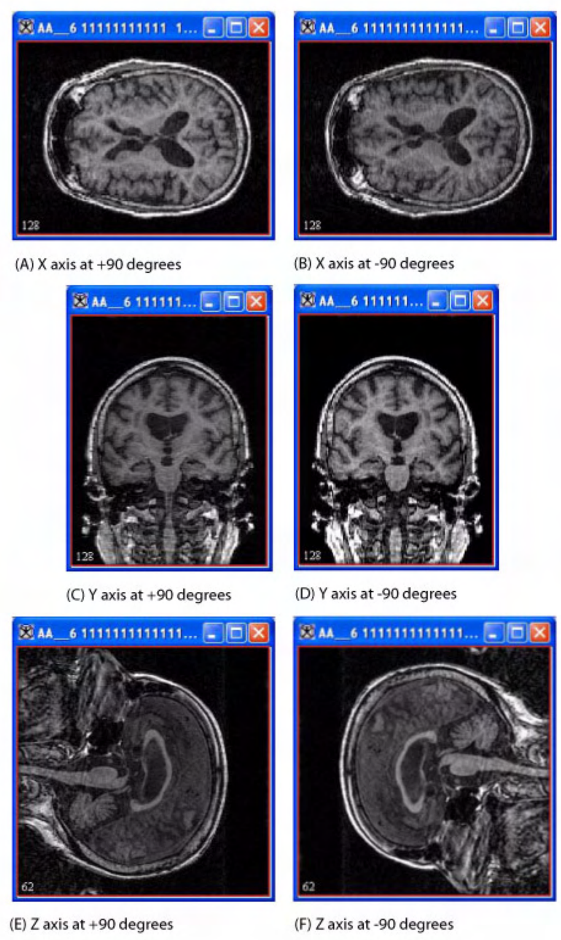


Figure 275. Examples of rotating images

This figure shows the image in Figure 273 at the six different types of x, y, and z rotation.

Slice tools

Concatenating images

The Concatenate command on the Utilities menu provides you with a way to append, or add, two datasets together.

To concatenate image datasets

1. Open the two datasets that you want to join.
2. Select the dataset to which you want to append the other dataset.
3. Select Utilities > Concatenate. The Concatenate Images dialog box (Figure 276) opens.
4. Note that the dataset you selected appears in read-only form in the box labeled Image A.
5. Select Image B, which is the name of the dataset that you want to add to the end of Image A.
6. Select the dimensionality that you want for the resulting dataset by selecting either 3D or 4D.

Note: If both datasets contain the same number of slices, you can select either 3D or 4D. If the datasets contain a different number of slices, you can only select 3D.

7. Click OK. The program adds the Image B dataset to the end of the Image A dataset and displays the resulting dataset in a new image window.

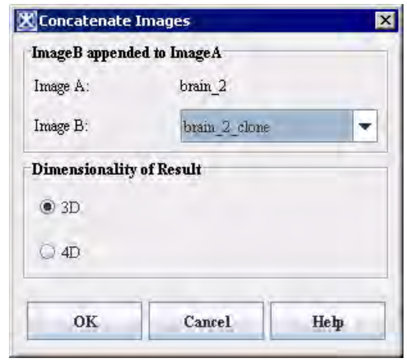


Image A	Specifies the name of the dataset to which to append Image B. This dataset is the one you selected before using the Concatenate command.
Image B	Specifies the name of the dataset that you want to append to the Image A dataset.
3D	Indicates that the dataset that results from appending Image B to Image A is three dimensional.
4D	Indicates that the dataset that results from appending Image B to Image A is four dimensional.
OK	Applies the changes you made in this dialog box and closes the dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not append Image B to Image A.
Help	Displays online help for this dialog box.

Figure 276. Concatenate Images dialog box

To verify that the datasets are joined

1. Select the new dataset that resulted from concatenating the Image A and Image B datasets.
2. Notice the following on the MIPAV window:
   - If you selected 3D as the dimensionality of the resulting dataset in the Concatenate Images dialog box, the image slice slider (Figure 277 upper part) lists twice as many slices as either the Image A and Image B datasets.
   - If you selected 4D as the dimensionality of the resulting dataset, a time slice slider (Figure 277 lover part) appears beneath the image slice slider and the image slice slider lists the same number of slices as the Image A and Image B datasets.

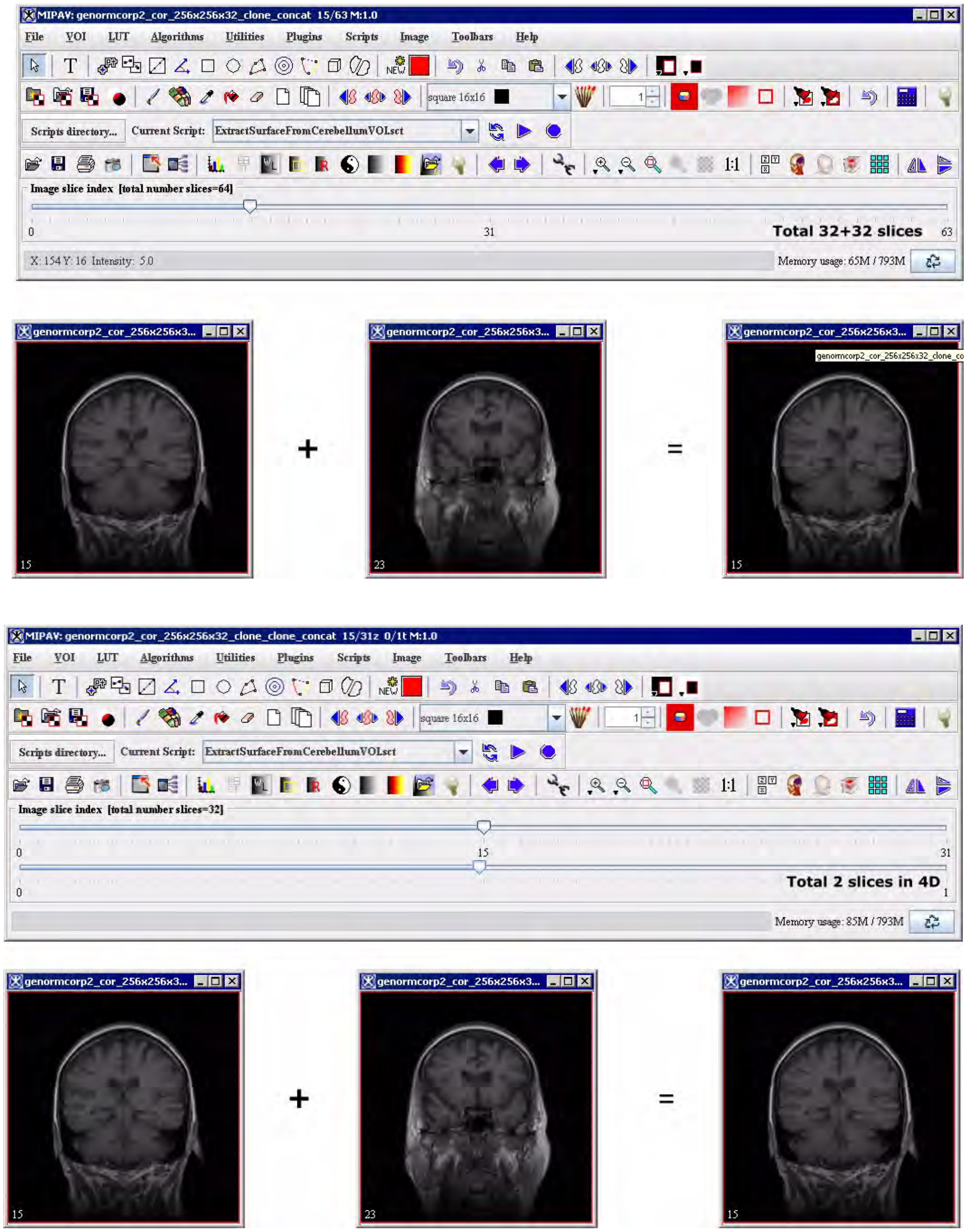


Figure 277. The MIPAV window after concatenating datasets to generate 3D (upper) or 4D (lover) datasets

Extracting slices/volumes

The Extract Slices/Volume utility makes copies of the slices or volumes you select and displays them in separate image windows. Unlike the Remove Slices utility, the original image dataset is untouched.

There are several ways to select slices or volumes. You can select all slices; only specific slices; only the even-numbered slices; only the odd-numbered slices; or one or more ranges of slices.

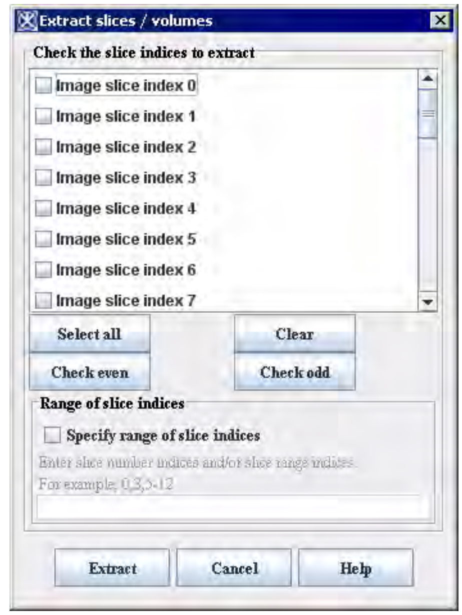
To extract slices or volumes from image datasets

1. Select Utilities > Slice Tools>Extract Slices/Volumes. The Extract Slices/Volumes dialog box opens.
2. Use the image slider in the MIPAV window to locate slices that you want to extract.
3. Decide whether to remove:

   - *All of the slices*—Click Select all. Check marks appear in all of the image slice check boxes.
   - *One or more slices*—Select the specific check boxes for the image slices. Check marks appear in only the check boxes you selected.
   - *One or more ranges of slices*—Select Specify range of slices, and then type the slice numbers or ranges of slice numbers in Enter slice numbers and/or slice ranges box.
   - *All even-numbered slices*—Click Check even. Check marks appear in only those check boxes for even-numbered slices, such as 2, 4, 6, and so on.
   - *All odd-numbered slices*—Click Check odd. Check marks appear in only those check boxes for odd-numbered slices, such as 1, 3, 5, and so on.

Tip: Use the image slider on the MIPAV window to look at specific slices to decide which slices to remove.

4. Click Extract. A status message appears. Then the MIPAV window refreshes and the slices you selected appear in new image windows.



Check the slices to extract	Indicates the slices that you want to extract from the dataset. You can scroll through the list to select one or more slices.
Select all	Selects all of the slices in the dataset to extract.
Clear	Clears all of the slices that are currently selected.
Check even	Selects all of the even-numbered slices to extract.
Check odd	Selects all of the odd-numbered slices to extract.
Specify range of slices	Indicates that you want to extract one or more specific slices or a range of slices from the dataset.
	If you select this check box, you must specify the slices or ranges of slices you want to extract in the Slice number and/or range of slices box below.
Enter slice numbers and/or slice ranges	Indicates the slices and range of slices that you want to extract. This box only becomes available after you select Specify range of slices.
Extract	Makes copies of the slices/volumes you selected and displays them in separate image windows.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

Figure 278. Extract Slices/Volumes dialog box

Insert Missing Slices

The Insert Missing Slices utility insert slices into the image dataset. There are several ways insert a slice into the image dataset. You can select Average, or Blank, or Original.

- If Average is selected, the inserted slice is set equal to the mean of the two surrounding slices, unless it is a new first of last slice. A new first or last slice preserves the usual slice spacing.
- If Blank is selected, the slice with all pixel zero is inserted.
- If the Original option is used, a 2D image of the same dimensions is inserted.

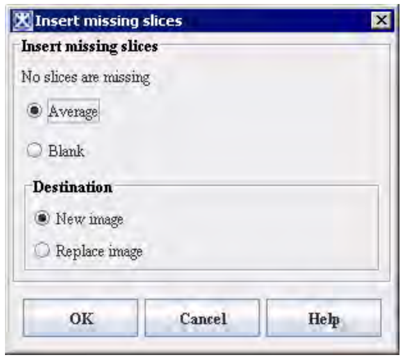
Image types

The algorithm works with 3D and 4D, grayscale, black and white, and color images. It preserves the image type when inserts slices, e.g. grayscale into grayscale images, color into color.

Applying Insert Missing Slices

To insert a slice to an image dataset

1. Select Utilities > Slice tools>Insert Missing Slices. The Insert Missing Slices dialog box opens.
2. The dialog box appears displaying a note at the top of window that indicates how many slices are missing.
3. Now, you have to choose the method for slice insertion: Average, Blank, or Original.
4. Select the destination. Choose New image if you want the image with inserted slices appear in a new image frame; choose Replace image to replace the original image.
5. Click OK to proceed.



Insert before slice (1-n) or enter n+1 for new last slice	Specifies the position of the new slice.
Average	Inserts a slice that is the average the adjacent slices.
Copy previous adjacent	Insert a slice that is a copy of the preceding slice.
Copy next adjacent	Inserts a slice that is a copy of the following slice.
Blank	Inserts a blank slice.
Original	Inserts an image from the original dataset.
OK	Applies the parameters that you specified to insert the slice in the image dataset.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not insert the slice into the image dataset.
Help	Displays online help for this dialog box.

Figure 279. The Insert Missing Slice dialog box options

Inserting slices into image datasets

When you change an image dataset, you are inserting or removing slices in the dataset or changing their order.

You can use the Insert Slice command on the Utilities menu to insert a slice into an image dataset. Although you cannot insert a slice from another file, you can insert either a blank slice or a new slice that is an average of two adjacent slices in the dataset.

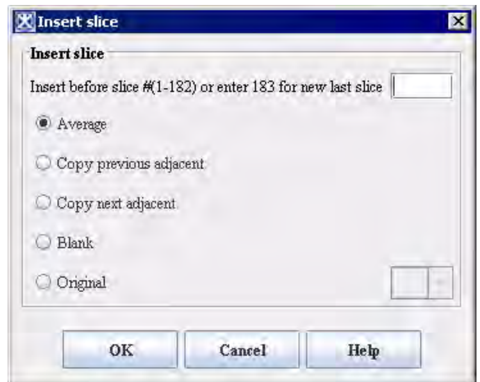
To insert slices into image datasets

1 Select Utilities Slice Tools> Insert Slice. The Insert Slice dialog box appears.

2 Enter the number of the new slice position in the Insert before slice (1-124) or enter 125 for new last slice box.

3 Select either the Average or Blank radio button. If you select Average, MIPAV analyzes the slices immediately before and after the new slice in the dataset. The software then generates an intermediary composite by morphing the two slices. This composite becomes the new slice. If you select Blank, MIPAV inserts a blank slice in the dataset.

4 Click OK. MIPAV inserts the new slice into the dataset and opens the new dataset in a new image window.



Insert before slice (1-124) or enter 125 for new last slice	Specifies the position of the new slice.
Average	Inserts a slice that is the average the adjacent slices.
Copy previous adjacent	Insert a slice that is a copy of the preceding slice.
Copy next adjacent	Inserts a slice that is a copy of the following slice.
Blank	Inserts a blank slice.
Original	Inserts an image from the original dataset.
OK	Applies the parameters that you specified to insert the slice in the image dataset.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not insert the slice into the image dataset.
Help	Displays online help for this dialog box.

Figure 280. The Insert Slice dialog box options

Pad slices to power of 2

The utility pads an image with blank slices to an even power of 2 (e.g. 2^{2k}). Depending on the selected option (Pad to Front, Pad to Back, or Pad to Half Front and Back), all slices can be inserted in front, in back, or half the slices in front and half the slices in back.

Image types

The algorithm works with 3D and 4D, grayscale, black and white, and color images. It preserves the image type when inserts slices.

Applying the pad to power of 2 utility

To pad the image dataset to power of 2

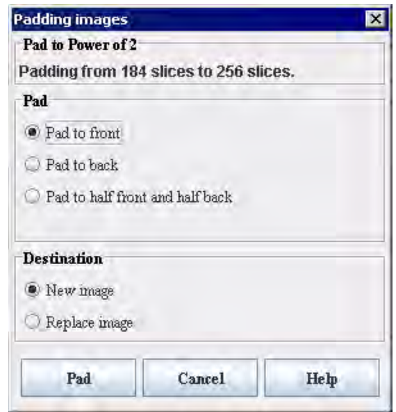
1 Select Utilities > Slice tools>Pad to Power of 2. The Padding Images dialog box opens.

2 The dialog box window appears indicating how many slices should be added to the image.

3 Choose the method for slice insertion: Pad to Front, Pad to Back, or Pad to Half Front and Half Back.

4 Select the destination. Choose New Image if you want the image with inserted slices appear in a new image frame; choose Replace Image to replace the original image.

5 Click Pad to proceed.



Pad to Front	Inserts slices before the first slice of the image.
Pad to Back	Inserts slices after the last slice of the image.
Copy previous adjacent	Insert a slice that is a copy of the preceding slice.
Destination	
Pad to half front and half back	Inserts half of the slices before the first slice of the image and half after the last slice.
New Image	Opens the modified image in a new image frame.
Replace image	Replaces the existing image.
Pad	Applies the parameters that you specified to insert the slices in the image dataset.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not insert the slice into the image dataset.
Help	Displays online help for this dialog box.

Figure 281. The Padding Images dialog box options

Randomizing image (slice) order

To randomize the order of slices in the image dataset, do the following:

1 Open an image that contains slices.

2 Write the slice numbers on a piece of paper.

3 Select Utilities > Slice Tools>Randomize Slice Order.

4 The MIPAV window refreshes and, depending on your selection, the new dataset appears in the same image window as the original dataset.

The new randomized slice ordering appears as a table in the Output window.

Removing images (slices) from datasets

MIPAV allows you to remove one, several, or all slices from an image dataset. In addition, you can specify one or more ranges of slices to remove, or slice removal can include only the even-numbered or only the odd-numbered slices.

To remove slices from datasets

1 Select Utilities > Remove slices. The Remove Slices dialog box opens.

2 Use the image slider in the MIPAV window to locate slices that you want to remove.

3 Decide whether to remove:

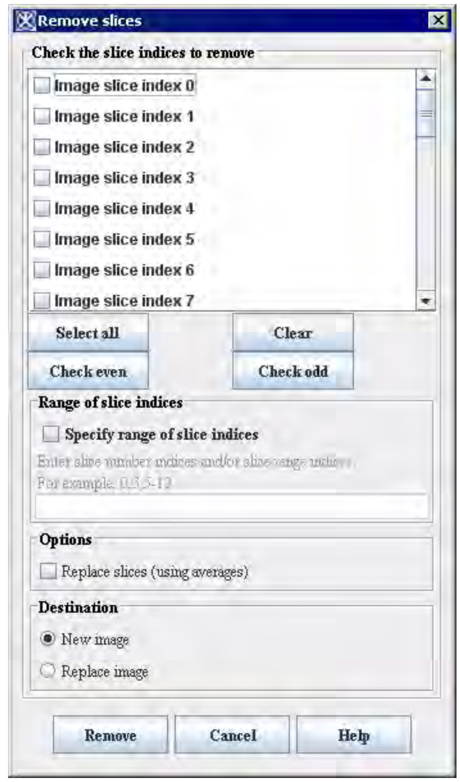
- One or more slices
- Most of the slices
- One or more ranges of slices
- All even-numbered slices
- All odd-numbered slices

Tip: Use the image slider on the MIPAV window to look at specific slices to decide which slices to remove.

4 Select one of the following:

- *New image*—If the new image dataset should appear in its own image window
- *Replace image*—If the dataset should overwrite the original dataset and appear in the original image window

5 Select Remove. A status message appears. Then the MIPAV window refreshes and, depending on your selection, the new dataset appears in either a new image window or in the same image window as the original dataset.



Slices	Indicates the slices that you want to remove from the dataset. You can scroll through the list to select one or more slices.
Select all	Selects all of the slices in the dataset.
Clear	Clears all of the slices that are currently selected.
Check even	Selects all of the even-numbered slices.
Check odd	Selects all of the odd-numbered slices.
Specify range of slices	Indicates that you want to remove one or more slices or a range of slices from the dataset. If you select this check box, you must specify the slices or ranges of slices you want to remove. in the Slice number or range of slices box below.
Replace slices	If you select that option, the removed slices would be replaced using the average method.
New image	Shows the results of the slice removal in a new image window.
Replace image	Replaces the current active dataset with the dataset resulting from the slice removal in the same image window.
Remove	Removes the slices that you indicated in this dialog box.
Cancel	Disregards any changes that you made in this dialog box and closes the dialog box.
Help	Displays online help for this dialog box.

Figure 282. Remove Slices dialog box

Replace blanks with averages

The algorithm replaces blank slices with slices that carry average pixel values calculated from the neighbor slices. The algorithm, first, counts the number of consecutive blank slices, and then does the following:

1. If the consecutive slices are at the beginning of the image, the all replaced with the first nonblank slice.
2. If the consecutive slices are at the end of the image, they all replaced with the last nonblank slice.
3. If the consecutive slices are somewhere in the middle, they replaced by the weighted sum of the surrounded nonblank slices.

Example: for a single blank slice, the pixel values are replaced by the averages of two surrounding slices. For two consecutive blank slices, the pixel values for the first slice are calculated as (2/3 * bottomNonBlank + 1/3 * topNonBlank); and for the second blank slice, the pixel values are calculated as (1/3 * bottomNonBlank + 2/3 * topNonBlank).

To run the algorithm, call Utilities >Slice Tools> Replace Blanks with Averages.

Image types

The algorithm is only applicable for 3D, grayscale images.

Replace Slices

For 3D and 4D images, the algorithm allows to replace image slices with some other slices (2D) chosen by a user.

To apply the Replace Slices algorithm,

1. Open an image of interest (3D or 4D).
2. Move the image slider so that it shows the slice that you wish to replace.
3. Open the 2D image that you would like to put in place of the chosen slice.
4. Call Utilities>Slice Tools>Replace Slices.

5 In the dialog box that appears, indicate the slice number that you would like to replace.

6 Press OK. The algorithm replaces the selected slice with the chosen 2D image. See Figure 283.

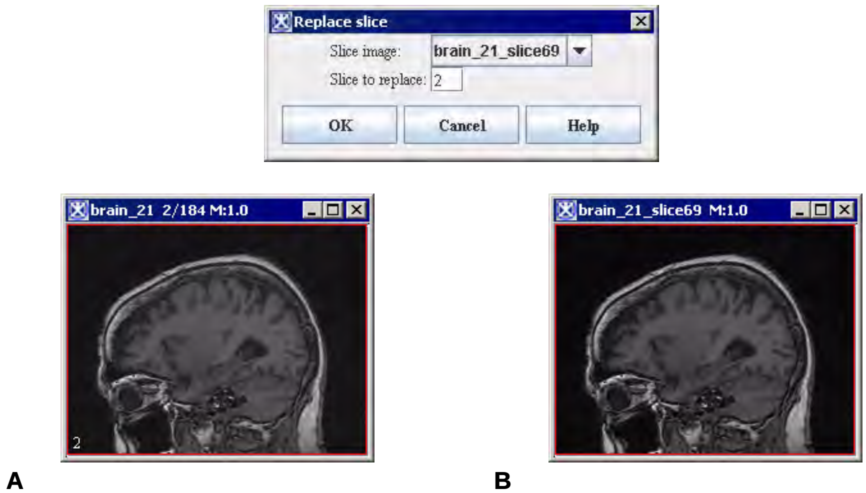


Figure 283. The Replace Slice dialog box (at the top), the original image (A), and the image (B) that was used to replace the slice 2 of the original image

IMAGE TYPES

The algorithm is applicable for both grayscale and color 3D images. The image that you use to replace the slice must be 2D.Note that the type of the original image and the image that you use to replace the slice must be the same. E.g. you cannot replace the grayscale slice with the color image.

Subtract VOI Background

The Subtract VOI Background algorithm subtracts the mean or median value of a VOI from the image. If only one contour is present, that contour is used whether or not it is selected. If multiple contours are present, at least one contour must be selected, and only selected contours are used.

Note: that under MIPAV all selected contours must belong to the same VOI.

IMAGE TYPES

The algorithm is applicable for any 2D, 3D, 4D, color and grayscale images. If the new image exceeds the range that can be stored in an image of that type, the data is either clipped and stored in the original image (Clip mode). Or a new image of a type (int, float...) that can store the range of new data is generated (Promote Image Type mode).

For color images, the dialog box currently disables promotion for color because MIPAV cannot handle negative color values.

Recommendation: To learn more about Clip and Promotion modes, see also "clip mode" on page 438 and "promotion mode" on page 438.

Applying the algorithm

To apply the Subtract VOI Background algorithm do the following:

1. Call Utilities>Subtract VOI Background.
2. In the dialog box that appears, select either Median or Mean.
3. Specify the clip mode.And specify where you want the new image to appear – in the new image frame or in the same frame, replacing the old image.
4. Click OK. The algorithm begins to run and the new image appears in the chosen destination.

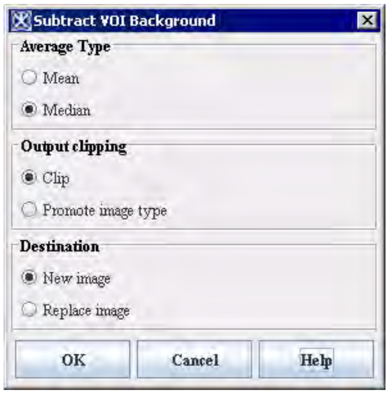
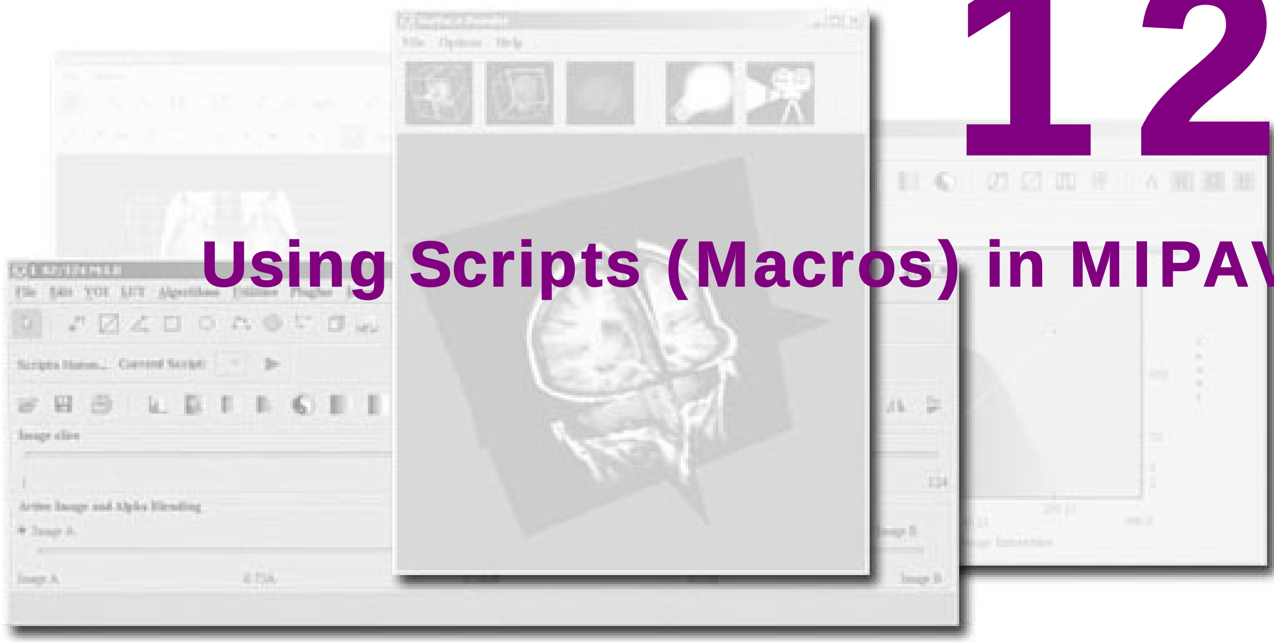
Average type		
Mean	Uses mean values to calculate average VOI background.	
Median	Uses median values to calculate average VOI background.	
Clip	Select this option if you want the result image to be clamped at the maximum (minimum) value of Image A data type. See also "clip mode" on page 438.	
Promote	When this mode is selected and the result value is out of the legal range of Image A data type, then the result image type is promoted to a new data type with a data type range capable of expressing the minimum and maximum values. See also "promotion mode" on page 438.	
New Image	Opens the modified image in a new image frame.	
Replace image	Replaces the existing image.	
OK	Runs the algorithm.	
Cancel	Disregards any changes you made in this dialog box and closes the dialog box.	
Help	Displays online help for this dialog box.	

Figure 284. The Subtract VOI Background dialog box options



12 Using Scripts (Macros) in MIPAV

In this chapter...

MIPAV provides three different methods of customizing the program.

The first method involves developing scripts, which you use directly within the program (see ‘‘Developing and using scripts’’).

In the second method, you also develop scripts but you initiate, or call, them from another program. Developing scripts does not require programming skills or learning a new script language, and calling them from another program may require, depending on the program, only minimal programming knowledge (see “Combining scripts and other programs- TBD” on page 527). This chapter discusses both of these methods of developing and using scripts.

The third method of customizing MIPAV—developing plug-in programs—does require Java programming skills. For more information about this method, refer to Chapter 11, “Developing Plug-in Programs.”

Developing and using scripts

Scripts, sometimes referred to as *macros* in other programs, record a series of commands or actions on specific images or groups of images that you can run with a single command. Using scripts can increase productivity and improve efficiency in performing commonly repeated actions or series of actions.

Tip: Most important! You can use scripts to process a large set of user-defined images.

This section explains the following tasks:

- Setting up scripting (refer to "Setting up scripting" on page 492
- Planning scripts (refer to "Planning scripts" on page 495)
- Creating scripts (refer to "Recording scripts" on page 498)
- Running scripts (refer to "Examples of working scripts" on page 525)
- Editing and deleting scripts (refer to "Editing and deleting scripts" on page 508)

Setting up scripting

Before you start create, edit, run, or delete scripts, you need to

1. Select the scripts home directory, refer to "Scripts Home" on page 493;
2. And display the scripting toolbar (optional, but very helpful).

To display the scripting toolbar

Like the paint toolbar, you can choose to show it on an as-needed basis or not to show it on an as-needed basis. To display the scripting toolbar on an as-needed basis, select Toolbars > Scripting Toolbar (Figure 285) MIPAV.

Scripting toolbar

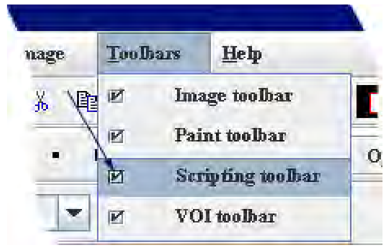


Figure 285. Scripting Toolbar command on the Toolbar menu

The scripting toolbar immediately appears beneath the VOI toolbar or, if the Paint toolbar is also displayed, immediately below the Paint toolbar. See Figure 286.

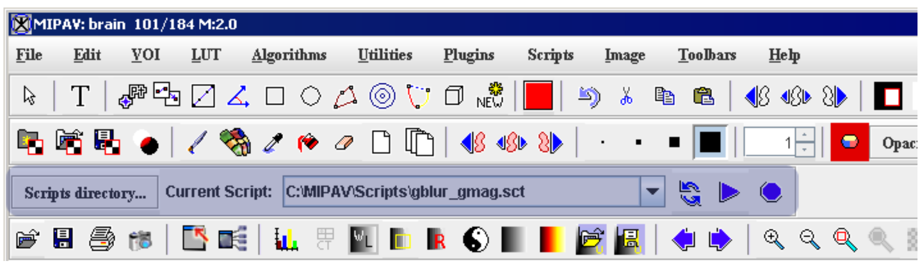


Figure 286. The Scripting toolbar is highlighted

Scripts Home

The *scripts home* is the directory where you store scripts, and from which you run them. After you select a scripts home, the name of one of the scripts (which are ordered alphabetically) in that directory appears after Current Script on the scripting toolbar (Figure 287). If there is more than one script in the directory, Current Script becomes a list box from which you can select the script to run.

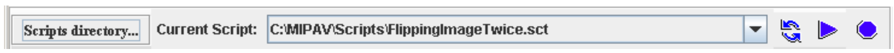
To set up Scripts Home

1 Start MIPAV. The initial MIPAV window opens.

2 Open an image. The expanded MIPAV window appears.

3 Select Toolbars > Scripting Toolbar. The scripting toolbar appears.

A dimmed rectangle appears immediately following the words *Current Script* in the scripting toolbar.




Refresh the script listing


Run the selected script


Start recording a script

Figure 287. The Scripting toolbar

4 Click Scripts Directory... The Choose Directory dialog box (Figure 288) appears.

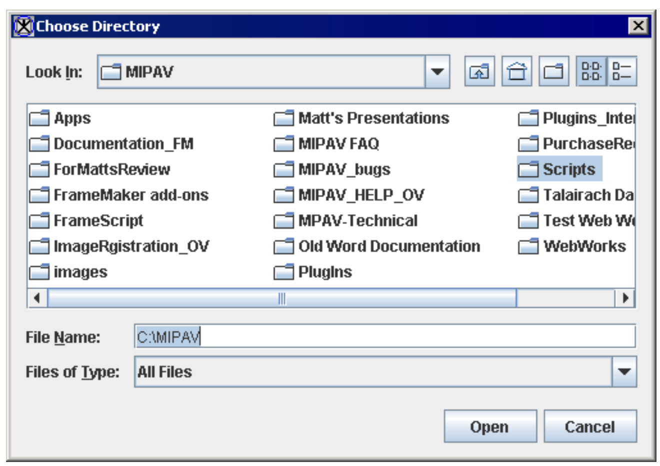


Figure 288. Choose Directory dialog box. Note that it shows only directories, not individual files

5 Select a directory in which to store your scripts.

6 Click Open. The directory opens and the Choose Directory dialog box closes.

After you've selected a *scripts home* and run a script for the first time, Current Script becomes a list box that shows the name of a script in the scripts home directory (Figure 287). After creating more scripts, you can select the one you want to run from Current Script.

Planning scripts

Although MIPAV allows you to easily create scripts, it is important to plan scripts before creating them, particularly if the scripts perform actions on VOIs. First, however, you need to understand:

- Methods for creating scripts (refer to page 495 below)
- Actions that can be performed in scripts, refer to page 496
- Use of VOIs in scripts, see page 496

Methods for creating scripts

There are two methods for creating scripts:

- To create a script using the Scripting toolbar, open an image of interest and simply click the Start Recording Script icon on the Scripting toolbar. See Figure 290.

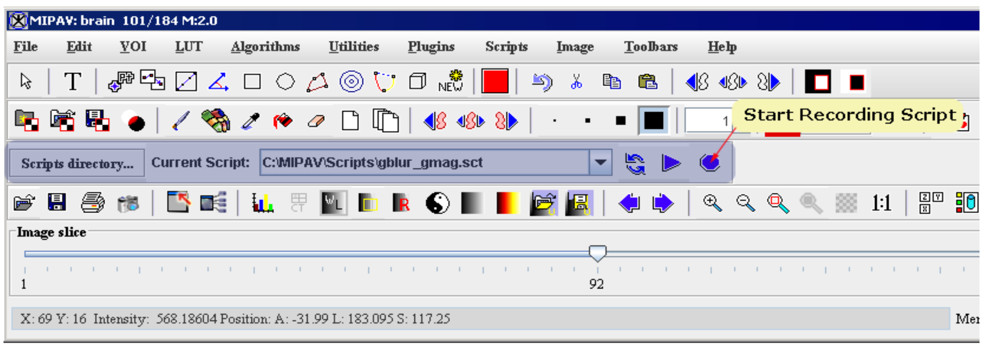


Figure 289. The Start Recording Script button

- To create a script using the main MIPAV menu, click Scripts > Record Script in the MIPAV window and simply start recording. See Figure 290.

For more information, refer to Section "Recording scripts" on page 498.

Figure 290. The Record Script command available via the Scripts menu

Actions permitted in scripts

It's also important to understand what actions or events you can record in scripts. For example, scripts neither record views, such as lightbox, triplanar, and volume renderer; nor do they record other visualization commands, such as those that adjust the lookup table.

Actions that you can record in scripts include:

- Saving and closing images
- Opening VOIs
- Saving VOIs including the Save all VOIs and Save all VOIs to commands
- Performing algorithms on images or on open VOIs on images[1]
- Applying utilities to images or to open VOIs on images
- Saving the global data page in the Output window
- Exiting from the MIPAV program[2]

Use of VOIs in scripts

The *only* VOI commands that can be recorded in scripts are the open VOI commands:

- VOI > Open VOI
- VOI > Open All VOIs

MIPAV scripts do not record and save VOIs. Therefore, to apply VOIs to images in scripts, you need to:

- Create and save all VOIs in *advance* of creating a script;
- Open and apply the VOIs when you are recording the script.

After you've created all necessary VOIs, you can save them through the Save

1. Not all MIPAV algorithms are scriptable.
2. `Exit( )` end is auto added when the script is running from the command line with `-hide` key.

VOI commands available via the VOI menu in the main MIPAV window:

- VOI > Save VOI
- VOI > Save VOI as
- VOI > Save All VOIs
- VOI > Save All VOIs to

Caution: Do **not** use the Exit command in MIPAV scripts unless you are calling the script from another program or scripts (Perl, C++, Windows batch).

Tip: You might consider to organize your workplace, for example, designate a specific folder(s) for VOIs using in scripts.

Saving VOIs: Some Reminders

You can save VOIs under names that MIPAV assigns or under names that you assign. If the VOIs have never been saved before, MIPAV assigns names to VOIs when you use the Save VOI or Save All VOIs commands. The name of the first VOI or set of VOIs that you save is *Area1.voi*; the second is *Area2.voi*; and so on.

If you want to specify a different name for the VOI or set of VOIs: select VOI > Save VOI as or VOI > Save All VOIs to. The Save VOI as dialog box opens. Type a name for the VOI in File name and the extension .voi or .xml, and click Save. The program saves the file under the specified name.

Recording scripts

To record a script using the Scripts toolbar:

1 Start MIPAV. The initial MIPAV window and the Output window open.

2 Open the images on which you want to run the script. The MIPAV window expands.

3 Click the Start Recording Script icon on the scripting toolbar. See Figure 291.

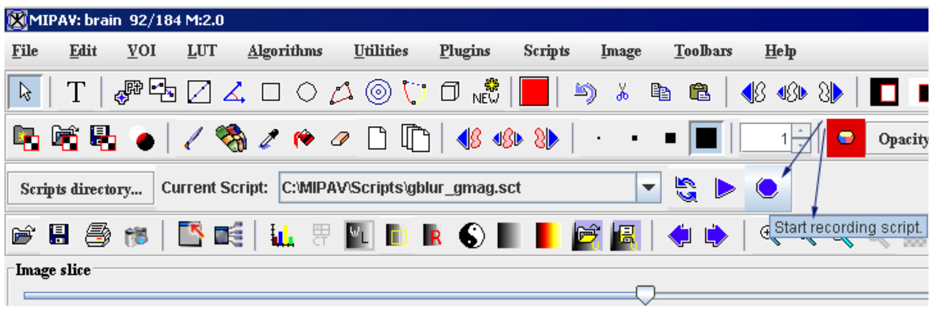


Figure 291. Click the Start Recording Script icon to record a script

Note: The Start Recording Script icon turns red to indicate that MIPAV is now recording a script.

4 The Record New Script dialog box (Figure 293) opens. The following placard message appears at the top of the dialog box: *The script is now recording. Your actions will appear below*.

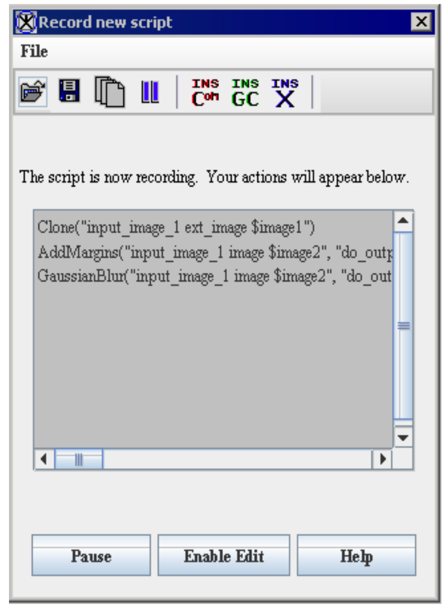
5 Open any previously created VOIs that you want to use with the images. Perform any algorithm or utility.

6 Repeat steps above as needed.

7 In the Record New Script dialog box, select File > Save. The Save dialog box opens. Type a name for the script and either the *.sct* or *.xml* extension in File name.

8 Click Save. The program saves the script in the scripts home directory.

9 Click File > Exit to close the Record New Script dialog box. The dialog box closes. You should now be able to select and run the script from Current Script.



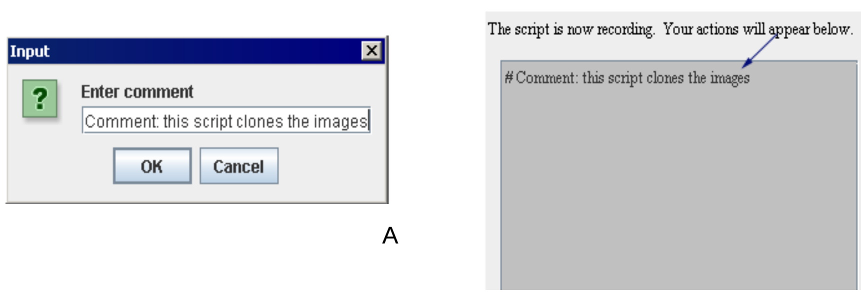
File	**Open**—Opens a previously saved script. When you select this command, the Open dialog box appears.
	Save—Saves the script under the specified name. When you select this command, the Save dialog box opens.
	Exit—Closes this dialog box without saving the script.
 **Open script**	**Open**—Opens a previously saved script. When you select this command, the Open dialog box appears.
 **Save script**	**Save**—Saves the script under the specified name. When you select this command, the Save dialog box opens.
 **Pause scripting**	Stops recording the script. When you select this icon, the icon changes to the Resume scripting icon.
 **Resume scripting**	Activates the recording process again. When you select this icon, the icon changes to the Pause scripting icon.
 **Insert comment**	Allows you to insert a comment into the script. When you select this icon, the Input dialog box opens. Type the comment in Enter comment and click OK. MIPAV adds your comment into the script after the word *Comment*.  A B **Figure 292. Input dialog box (A) showing a comment and (B) the comment as it appears in the scripting box**
 **Insert command to collect garbage (free memory)**	Frees memory by inserting the command to collect garbage, a Java method that clears all unnecessarily reserved memory. For more information, refer to "Saving a history of actions on images (TBD)" on page 126.

Figure 293. Record New Script dialog box

Insert command to end MIPAV	Inserts the command for exiting MIPAV (**Exit**) into the script, which allows MIPAV to close as a part of the script. **Caution**: Do **not** use the Exit command in MIPAV scripts unless you are calling the script from another program or scripts (Perl, C++, Windows batch, etc.).
Scripting box	Displays the action commands (algorithms and utilities) and any comments in the script.
Pause	Stops recording the script. When you select this button, the name of the button changes to Resume.
Enable Edit	Paused the script recording first, then it allows you to make changes or corrections to the script. When you select this button, the scripting box turns from gray to white to indicate that 1) the script is paused and 2) that you can now type, copy, paste, or delete information from the script, and the button name changes to Disable Edit.
Disable Edit	Prevents any changes or corrections being made to the script from typing, pasting, inserting, or deleting information. It does, however, allow MIPAV to record actions on images. When you select this button, the scripting box turns from white to gray, and the button name changes to Enable Edit. It also restarts the script recording.
Help	Displays online help for this dialog box.

Figure 293. Record New Script dialog box (continued)

Recording a script using the Scripts > Record Script menu

1 Start MIPAV.

2 Select Scripts > Record Script. The Record New Script dialog box opens. The following appears at the top of the dialog box: *The script is now recording. Your actions will appear below.*

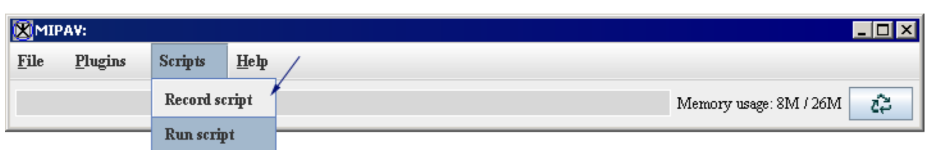


Figure 294. Recording a script via Scripts > Record Script menu

3 Open an image on which you want to run the script. The MIPAV window expands.

4 Perform any algorithms or utilities that you want recorded in the script, as if you run recording from the Scripting toolbar.

For example,

flip the image twice applying step by step the Flip Vertically and then Flip Horizontally option. See Figure 295.

5 The following appears in the Record New Script window:

```
Flip("input_image_1 ext_image $image1", "flip_axis string X")
Flip("input_image_1 image $image1", "flip_axis string Y")
```

6 Then, save the image in MINC format using the Save As option and specifying an `*.mnc` extension in the dialog box.[1] The following string appears in the Record New Script window:

```
SaveImageAs("input_image_1 ext_image $image1", "file_type string
.mnc")
```

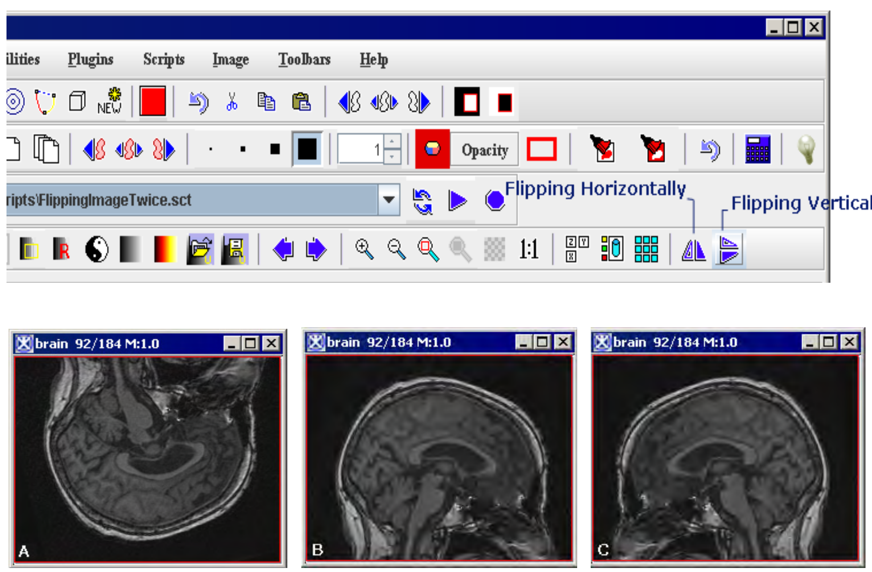


Figure 295. At the top: the flipping options available via the Image Toolbar. Below: the original image (A), the same image flipped vertically (B) and then horizontally (C)

7 In the Record New Script dialog box, select File > Save. The Save dialog box opens.

1. http://www.bic.mni.mcgill.ca/software/minc/

8 Enter a name for the script, for example, *FlippingImagesTwice.sct*, and then press Save. By default, the script will be saved in the Scripts Home directory. See "Scripts Home" on page 493 for details.

9 Click File > Exit to close the Record New Script dialog box. The dialog box closes.

10 Use the Refresh button on the Scripts toolbar to refresh the Scripts Home directory.

11 You should now be able to select and run the script from the Current Script box. See Figure 295.

To pause recording a script

	Pause
	Resume Recording

If you are interrupted or need to take a break while you are recording a script, use the Pause icon located on the Scripting toolbar. This allows you to stop recording the script. When you click Pause, the following message appears at the top of the Record New Script dialog box: *The script is now paused. Press Resume to resume recording*; and the Pause icon changes to Resume Recording. When you return to recording the script, click Resume Recording to resume.

Running scripts

You can run a script either using the Scripts toolbar or via the Scripts>Run Script menu.

If you are running the script via the Scripting toolbar and have at least one image open and selected, the script will automatically apply to that active image. Here, the *active image* is the image that is currently selected, refer to Figure 296.

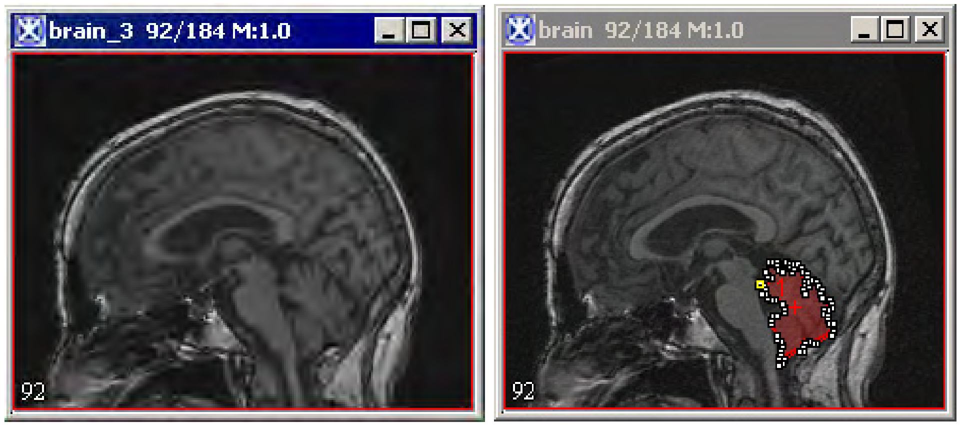


Figure 296. An active image, which bar is highlighted (on your left) and inactive image, which bar is dimmed and grey (on your right)

To run scripts from the Scripts toolbar

1 Open an image of interest and VOI, if needed.

2 Select a script from Current Script in the scripting toolbar, then click the Run the Selected Script icon. See Figure 297.

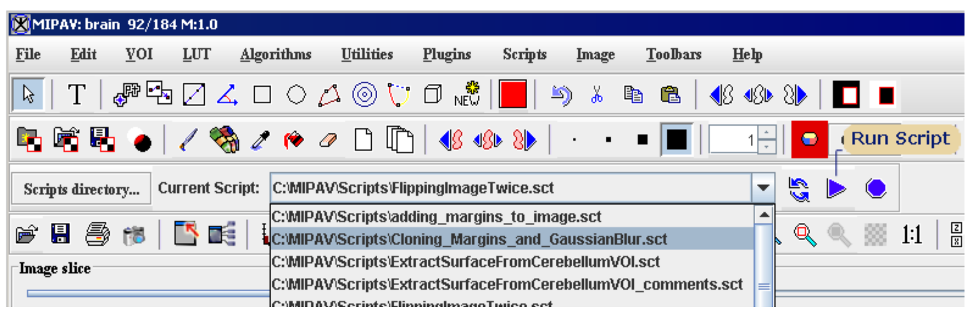


Figure 297. Current Script lists all of the scripts in the scripts home directory alphabetically. To reorder the list of scripts in Current Script, click the Refresh icon

3 MIPAV automatically performs all of the actions in the script on the active image.

If the script requires more than one image or VOI, it will pop up the Script Execution Setup dialog box, where you can add more images and VOIs to the script procedure. Refer to "To run scripts from the Run Script menu" on page 504 and Figure 298.

To run scripts from the Run Script menu

1 Call Scripts > Run Script from the main MIPAV menu.

2 Select a script you wish to run and press OK. The MIPAV Script Tool dialog box opens. Refer to Figure 298.

3 Add images to the Script Execution Setup, refer to "Adding images to the Script Execution Setup" on page 504.

4 Add VOIs to the Script Execution Setup, if needed. Refer to "Adding VOIs to the Script Execution Setup" on page 505.

5 Repeat step 3 and step 4 for as many images and VOIs on which you want to run the script.

6 Press Run Script.

7 MIPAV automatically performs all of the actions in the script on the images indicated in the Script Execution Setup dialog.

Adding images to the Script Execution Setup

1 Use the Add Image from File option to choose the images and add them to the dialog box. Selected images appear in the Images window. See Figure 298. Images which have already been opened in MIPAV also appear in the Images window.

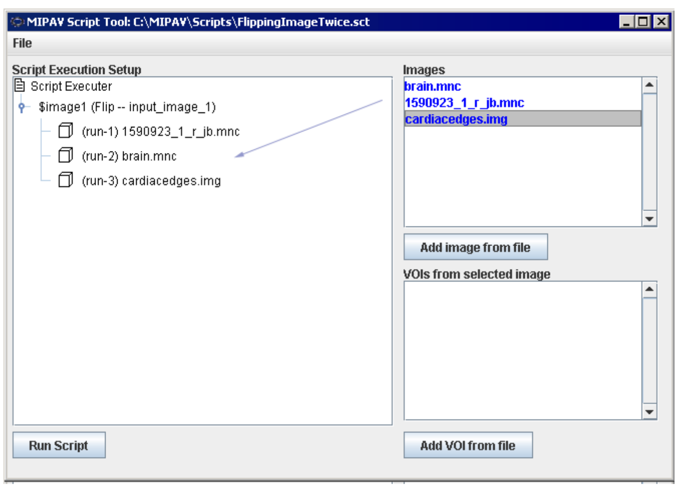


Figure 298. The MIPAV Script Tool dialog box

2 Use the mouse to drag selected images from the Images window to the Script Execution Setup. Arrange the order of the images.

3 If each script execution requires more than one image (e.g. you are applying algorithm(s) to Image 1, and then to Image 2), make sure that you add the same number of images under $image 1 and $image 2 in the Script Execution Setup. The same is also applied to VOIs. Refer to Figure 299.

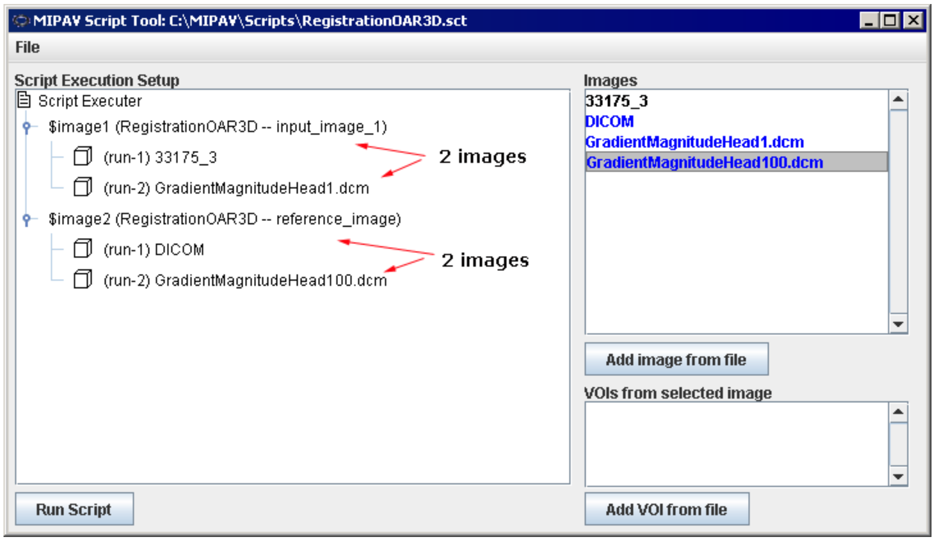


Figure 299. Adding images to the Script Execution Setup

4 Press Run Script.

5 Multiple images assigned to the same script will be used for separate execution of the chosen script.

Adding VOIs to the Script Execution Setup

1 Add images to the Images window and then to the Script Execution Setup window.

2 Add one or more VOIs to the VOIs from Selected Image window. In order to do that, click the Add VOI from File button and then select a VOI.

You don't need to manually associate VOIs for those images that already have associated VOIs. Those VOIs appear in the Script Execution Setup window automatically under the corresponding image.

3 For those images that don't have linked VOIs, add VOIs manually by dragging files from the VOIs window to the designated image in the Script Execution Setup. Linked VOI is a result of `OpenVOI()` action in the script. Refer to Figure 300 for details.

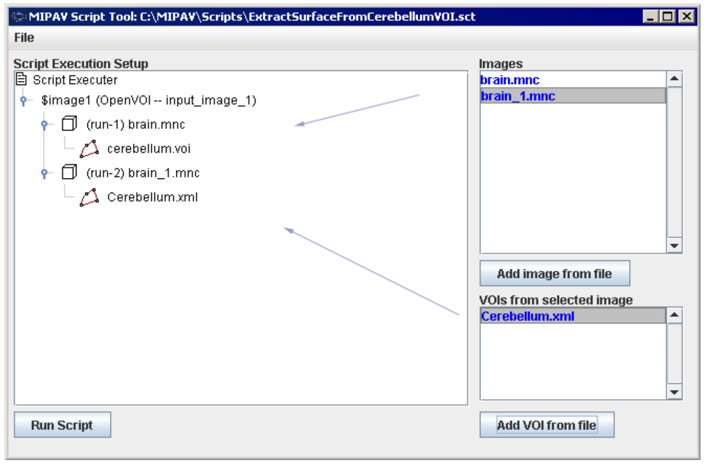


Figure 300. Adding VOIs to images

4 Press Run Script.

5 Multiple images and VOIs assigned to the same script will be used for separate execution of the chosen script.

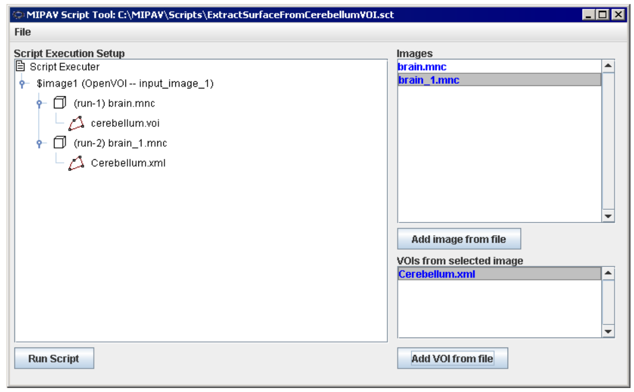
Note: After assigning an image or VOI to a script in the Script Execution Setup, your can delete the assignment using the Delete context menu option.

Notes about running scripts

Use the Scripting toolbar to run simple scripts which involve one image and one VOI. Use the Run Script menu option to run more complicated scripts which apply to multiple images and VOIs. You can also use the Scripting menu to execute the same script multiple times.

For example, you can consider to use the toolbar option to execute the script while you are setting the script parameters and/or testing it; and after the script has been tested and debugged, you can run it over the set of images via the Scripting menu.

THE MIPAV SCRIPT TOOL DIALOG BOX OPTIONS



Script Execution Setup window	Contains a list of images and associated VOIs.
Images	Lists the images that appear in the script
Add image from file	Allows you to add one or more images on which to run the script.
VOIs from selected file	Lists any saved VOIs that are necessary when you run the script.
Add VOI from file	Allows you to add one or more VOIs to the script.
File menu	
Open saved image and VOI selections	Opens the corresponding dialog box, which allows you to browse through the folders and select an image file or VOI with any specified extension.
Save current image and VOI selections	Allows you to save the selected image or VOI.
View current script contents	Opens a window that shows the script contents. Here, you can only select the script contents and copy it to the Clipboard using the CTRL+C combination of keys.
Close	Closes the Script Execution Setup.
Run Script	Begins running the script.

Figure 301. The MIPAV Script Tool dialog box

Tip: To speed up the process of associating VOIs with images, you can open all of the images at once and all of their associated VOIs at once. The Images box shows all of the images selected and the VOI box lists all of the VOIs selected. Select an image in the Images box and then select the VOI that applies to that image. Drag the image and VOI to the Script Execution Setup pane. Repeat these steps for as many images and VOIs as you need.

Editing and deleting scripts

You can either edit a script while you are creating it, or you can edit any previously saved script. MIPAV saves the script in text format so that any text editor, such as Microsoft Wordpad or Notepad, can modify the file.

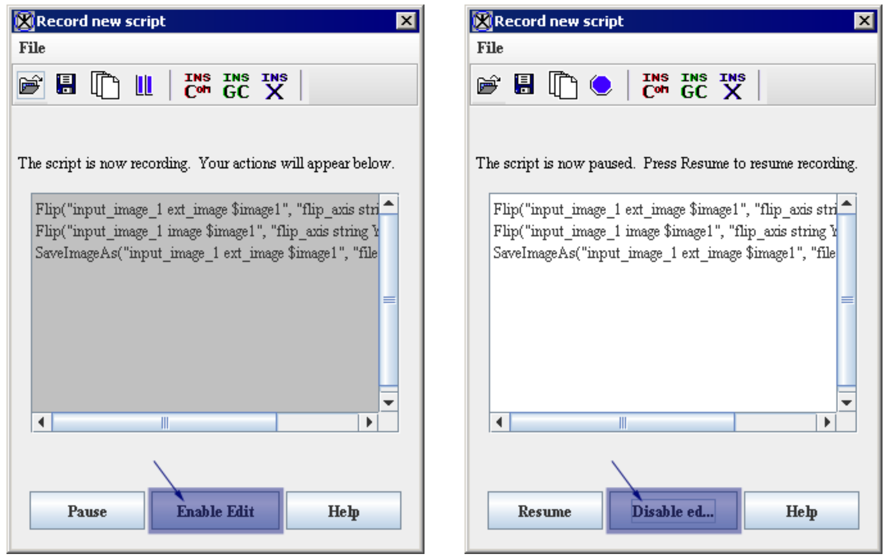
Caution: Editing scripts may cause serious problems if the script commands are not entered correctly. This task is recommended for more advanced users.

To edit scripts during creation

Suppose you make an error while you are recording a script, or perhaps you've changed your mind about performing a specific action. It can be more practical to correct the problem right away.

To correct errors or make changes to scripts while you're creating and running them, simply click the Enable Edit button, in the Record New Script dialog box (Figure 302). Three things occur:

1. It pauses the script execution.
2. The scripting box in the dialog box turns from gray to white to indicate that you can now type, copy, or paste information into it and delete information from it.
3. The Enable Edit button changes to the Disable Edit button. If the scripting box contains actions that you want to erase, select the actions and delete them. To add new actions, you can type them into the scripting box directly. You can also rearrange actions by copying and pasting them.



A B

Figure 302. Record New Script dialog box (A) before and (B) after clicking Enable Edit

When you've finished editing the script, click Disable Edit. The record of script actions resumes. The scripting box turns gray and the Disable Edit button becomes the Enable Edit button.

To edit previously created scripts

1. Click the Start Recording Script icon on the scripting toolbar. The icon turns red, and the Record New Script dialog box opens.
2. Select File > Open on the Record New Script dialog box. The Open dialog box opens.
3. Select the script that you want to edit. The script appears in the scripting box on the Record New Script dialog box.
4. Click Enable Edit. The actions box turns from gray to white.
5. Select any of the actions and copy, rearrange, or delete them, or type new actions directly into the scripting box.
6. Click Disable Edit when you are satisfied with the script. The scripting box returns to gray.
7. Continue recording actions if needed until you are satisfied the script.

8 Save the script by clicking File > Save. The Save dialog box opens.

9 Type the name of the script followed by the *.sct* extension in the File Name box.

10 Click Save. MIPAV saves the script under the specified name.

To delete scripts (Windows specific)

If you decide at some point that you want to delete a script, go to Windows Explorer and navigate to the scripts home directory.

1. Open Windows explorer.
2. Navigate to the scripts home directory.
3. Select the script or scripts that you want to delete.
4. Right-click on the selected script(s).
5. Click Delete. Windows deletes the selected scripts and places them in the Recycle Bin where they stay until you empty the bin.

Examples:

Applying Face De-Identification, BSE, and Fuzzy C-Means

In this example, we will to show you how to record a script, and then run it in via the Scripting menu. The script includes:

1. Applying the Face De-Identification algorithm;
2. Applying the Extract Brain Surface (BET) algorithm;
3. And finally, performing brain segmentation using the Fuzzy C-Means > Single Channel algorithm.

To start recording the script:

Open an image of interest and then, press Start Recording Script button on the Scripts toolbar. The Record New Script dialog box appears indicating that recording a script is started.

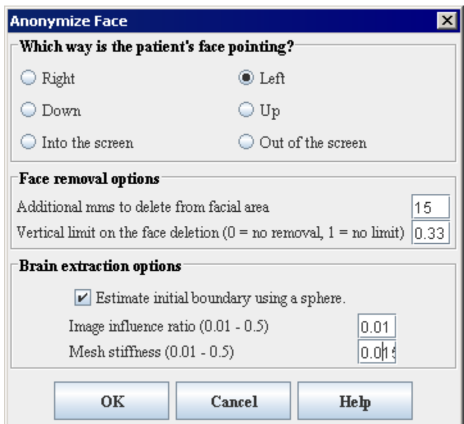
Recording the Face De-Identification algorithm execution:

1 Call Algorithms > Brain Tools > Face De-Identification;

2 The Anonymize Face dialog box opens;

3 Fill out the dialog box as shown in Figure 303. The right column displays the script text corresponding to the dialog box options;

4 Press OK to run the algorithm.



FaceAnonymizerBET("input_image_1 ext_image $image1", "face_orientation int 2", "mms_to_delete_from_face int 15", "vertical_deletion_limit_ratio float 0.33", "bet_do_estimate_with_sphere boolean true", "bet_image_influence float 0.01", "bet_stiffness float 0.015")

Figure 303. The Anonymize Face dialog box options and the corresponding script text. Here, "face_orientation int 2" **is an image specific and corresponds to the Left option chosen in the "Which way is the patient's face pointing?" dialog**

After the algorithm finishes running and the result image appears on the screen, make the result image active and call the next Extract Brain Surface (BET) algorithm.

Recording the BET algorithm execution:

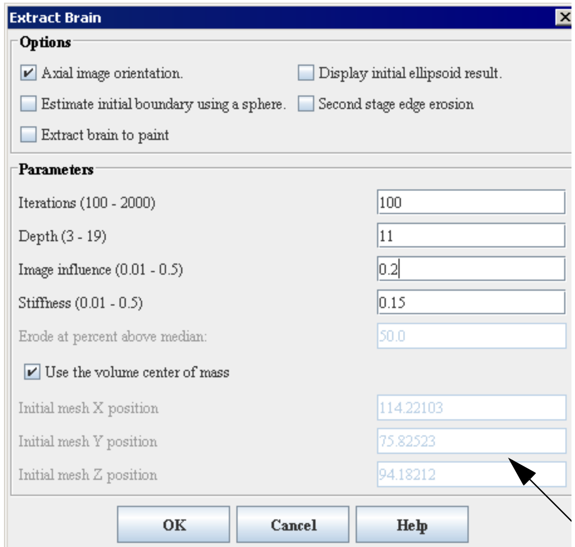
1 Call Algorithms > Brain Tools > Extract Brain Surface (BET);

2 The Extract Brain dialog box opens;

3 Fill out the dialog box as shown in Figure 304. The right column displays the part of the script corresponding to the dialog box options;

4 Press OK to start the algorithm.

After the algorithm finishes running and the result image appears on the



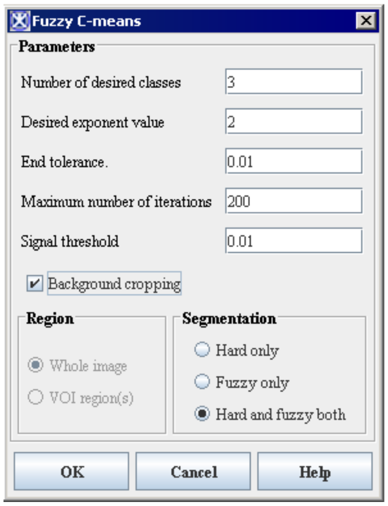
ExtractBrain("input_image_1 image $image1", "orientation_type int 0", "do_use_sphere_estimation boolean false", "do_show_just_init_ellipse boolean false", "num_iterations int 100", "depth int 11", "image_ratio float 0.2", "stiffness float 0.15", "do_second_stage_erosion boolean false", "factor_above_median_to_erode float 0.0", "do_extract_paint boolean false", "do_init_with_center_of_mass boolean true", "init_center_point list_float 114.22103,75.82523,94.18212")

Figure 304. The Extract Brain dialog box options and the corresponding script text. Here, "init_center_point list_float 114.22103,75.82523,94.18212" **is an image specific**

screen, make the result image active and call the last algorithm – Fuzzy C-Means > Single Channel.

Recording the Fuzzy C-Means > Single Channel algorithm execution:

1. Call Algorithms > Segmentation > Fuzzy C-Means > Single Channel;
2. The Fuzzy C-Means dialog box opens;
3. Fill out the dialog box as shown in Figure 305. The right column displays the part of the script corresponding to the dialog box options;
4. Press OK to start the algorithm.



FuzzyCMeans("input_image_1 image $image1", "number_of_result_images int 4", "do_process_whole_image boolean true", "number_of_classes int 3", "exponent_q float 2.0", "do_crop_background boolean true", "threshold float 0.01", "end_tolerance float 0.01", "max_iterations int 200", "segmentation_type int 0", "centroids list_float 8318.767,20926.652,33534.54")

Figure 305. The FUzzy C-Means dialog box options and the corresponding script text. Here, "centroids list_float 8318.767,20926.652,33534.54" **is an image specific**

After the algorithm finishes running and the result image appears on the screen, close all intermediate images, and then save the final image.

Now, save the script. You might consider to add comments to the script text. In order to do that,

1 Click the Insert Comment icon located on the Record New Script dialog box toolbar;

2 The Enter Comment window appears;

3 Type in the comment and press OK;

4 The comment text appears in the Record New Script dialog box;

5 Save the script;

6 Close the dialog box.

To run the script via the Scripts menu

1 Call Scripts > Run Script from the main MIPAV menu;

2 Select the script you've just recorded, and press OK;

3 The MIPAV Script Tool dialog box opens;

4 In the dialog box, use the Add Images from File button to select and add images, on which you wish to run the script, to the Images window;

5 Then, use the mouse to drug images from the Images window to the Script Execution Setup window. Refer to Figure 306 and "Adding images to the Script Execution Setup" on page 504;

6 Press Run Script.

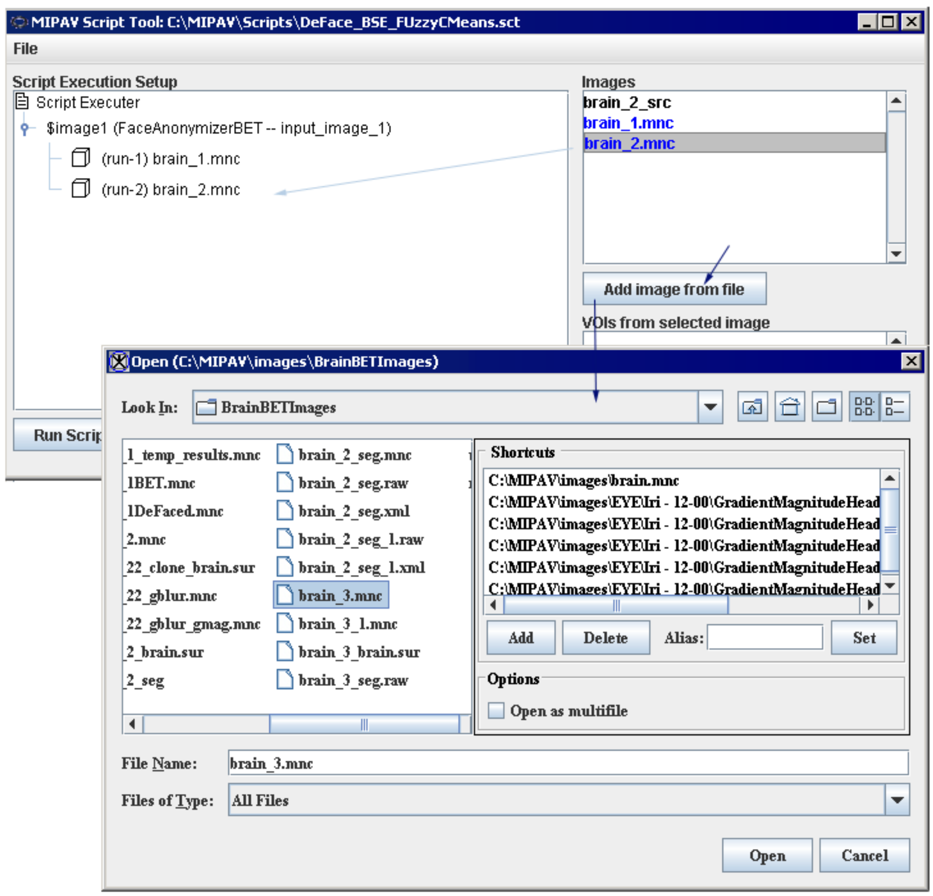


Figure 306. Adding images to the Images window, first, and then moving the images to the main Script execution Setup window

The script starts running and the intermediate images and progress bars appear on the screen. See Figure 307.

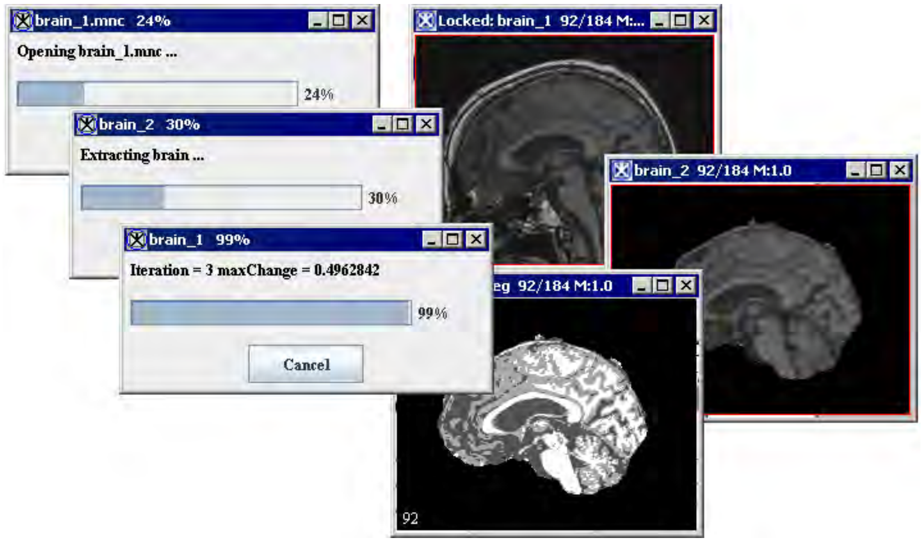


Figure 307. The script is running in group mode and the intermediate images and progress bars appear on the screen

When the algorithm finishes running, the progress bars disappear, and the results replace the original and intermediate images.

Script

The script text is shown below. Note that the comment strings included in the script start with "#".

```
# First Face De-Identification

FaceAnonymizerBET("input_image_1 ext_image $image1", "face_orientation int 2", "mms_to_delete_from_face int 15", "vertical_deletion_limit_ratio float 0.33", "bet_do_estimate_with_sphere boolean true", "bet_image_influence float 0.01", "bet_stiffness float 0.015")

#Here, "face_orientation int 2" is an image specific.

# Second BET

ExtractBrain("input_image_1 image $image1", "orientation_type int 0", "do_use_sphere_estimation boolean false", "do_show_just_init_ellipse boolean false", "num_iterations int 100", "depth int 11", "image_ratio float 0.2", "stiffness float 0.15", "do_second_stage_erosion boolean false", "factor_above_median_to_erode float 0.0", "do_extract_paint boolean false", "do_init_with_center_of_mass boolean true", "init_center_point list_float 114.22103,75.82523,94.18212")

#Here, "init_center_point list_float 114.22103,75.82523,94.18212" is an image specific.

# Third Fuzzy C-Means
```

```
FuzzyCMeans("input_image_1 image $image1", "number_of_result_images int 4", "do_process_whole_image boolean true", "number_of_classes int 3", "exponent_q float 2.0", "do_crop_background boolean true", "threshold float 0.01", "end_tolerance float 0.01", "max_iterations int 200", "segmentation_type int 0", "centroids list_float 8318.767,20926.652,33534.54")

#Here, "centroids list_float 8318.767,20926.652,33534.54" is an image specific.

#Closing intermediate images

CloseFrame("input_image_1 image $image4")

CloseFrame("input_image_1 image $image3")

CloseFrame("input_image_1 image $image2")

#Saving results

SaveImageAs("input_image_1 image $image5", "file_type string .xml")
```

Cloning images, adding margins, and applying Gaussian blur

In this example, we will to show you how to record the script that does the following:

1. Clones an active image;
2. Adds margins to the clone;
3. Applies the Gaussian Blur algorithm to the clone, and produces the second clone;
4. Closes the original image and saves the clones.

To get started,

open an image of interest and then, press Start Recording Script button. The Record New Script dialog box appears indicating that recording a script is started, see Figure 308.

1. Call the Utilities > Clone (Copy). This clones the active image and the following string appears in the Recording Script dialog:

```
Clone("input_image_1 ext_image $image1")
```

Here, ext_ indicates that this image is not produced within the script, but must be selected by a user.

Here, the command in the script is Clone. This command clones an image which is labeled as $image1 in the script, and then passes it as input_image_1 parameter to the Clone operator.

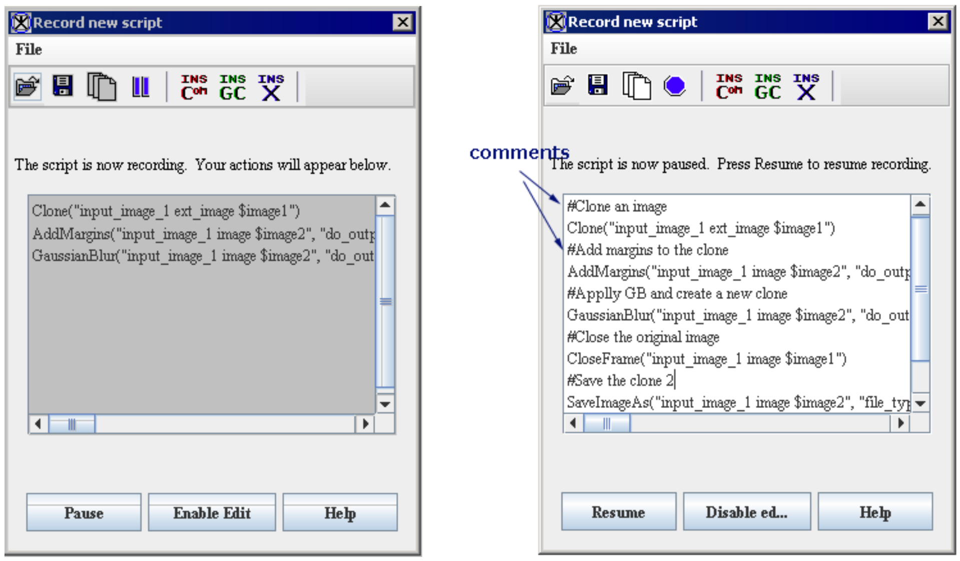
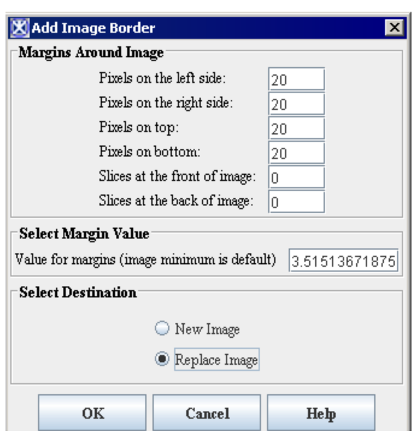


Figure 308. The Record New Script dialog box appears indicating that recording a script is started. The initial script (A) and the script with comments (B)

2 Now, select the clone image and call Utilities > Add Image Margins. In the Add Image Border dialog box, specify the margins as shown in Figure 309 and press OK.



In the dialog box:

20 points margins are added to the top, bottom, left and right sides of the image

The Replace Image option is checked indicating that the modified image will appear in the same frame

Figure 309. The Add Image Border dialog box options

This adds margins to the image clone1, and the following string appears in the Recording Script dialog:

```
AddMargins("input_image_1 image $image2", "do_output_new_image
boolean false", "left_side int 20", "right_side int 20", "top_side int 20",
"bottom_side int 20", "front int 0", "back int 0", "margin_value double -
5213.51513671875", "margin_value_rgb list_double 0.0,0.0,0.0")
```

Here, the command in the script is AddMargins(parameter*1*,...parameter*N*). Script parameters are as follows:

"input_image_1 image $image2" shows that the script applies to the active image clone

"do_output_new_image boolean false" indicates that the new image (with margins) appears in the same image frame replacing the old one[1]

"left_side int 20", "right_side int 20", "top_side int 20", "bottom_side int 20" are the image margins, refer to Figure 309

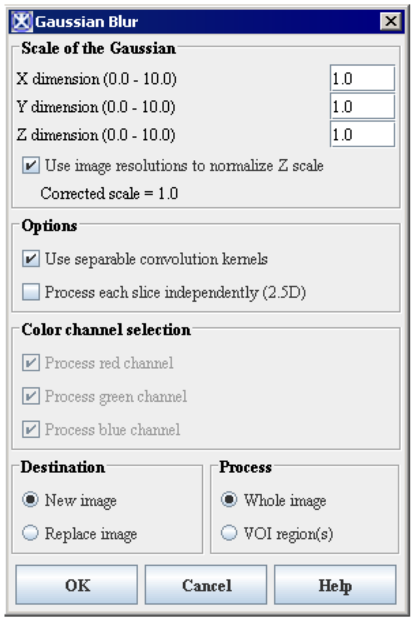
"front int 0", "back int 0" indicates that there are no slices added to the front and back of the image, refer to Figure 309

"margin_value double -5213.51513671875" – the margin fill value for grayscale images

"margin_value_rgb list_double 0.0,0.0,0.0" – the margin fill value for RGB images

3. Select the clone image with margins and call Algorithms > Filters (spatial) > Gaussian Blur. In the Gaussian Blur dialog box that appears, specify the parameters as shown in Figure 310 and press OK.

1. "do_output_new_image boolean true" will indicate that the result image appears in a new image frame



For the first time, use the default parameters for Scale of Gaussian, Options, and Color Channel Selection

Check the Destination > New Image option

Check the Process > Whole Image option

Press OK

Figure 310. The Gaussian Blur dialog box

This applies the Gaussian Blur algorithm to the image clone, which now has margins added, and the following string appears in the Recording Script dialog:

```
GaussianBlur("input_image_1 image $image2", "do_output_new_image
boolean true", "do_process_whole_image boolean true", "do_process_in_2.5D
boolean false", "do_separable_convolution boolean true", "gauss_std_dev
list_float 1.0,1.0,1.0", "gauss_do_z_resolution_correction boolean true",
"do_process_r_g_b_channel list_boolean false,false,false")
```

Here, the command in the script is GaussianBlur(parameter*1*,...parameter*N*). The parameters are as follows:

"input_image_1 image $image2" shows that the script applies to the image clone

"do_output_new_image boolean true" indicates that the new image appears in a image frame

"do_process_whole_image boolean true" indicates that the algorithm is applied to the whole image (not only to VOI regions) "do_separable_convolution boolean true" corresponds to checking the Use Separable Convolution Kernels option in the dialog box. For more information, refer to Volume 2 Algorithms, Section "Applying the Gaussian Blur algorithm"

4 As a result, the second image clone appears on the screen. See Figure 311.

A B C

Figure 311. The original image (A), the first clone after applying margins (B), and the second clone (C) after applying Gaussian Blur

5 Now, you can close the original image. And the following string appears in the script: CloseFrame("input_image_1 image $image1") indicating that the image frame has been closed.

6 You might save last two images. In order to do that, select the image frame, and then call File > Save Image As from the main MIPAV menu. The image will be saved in specified format in the chosen location. The following strings appears in the script: SaveImageAs("input_image_1 image $image2", "file_type string .mnc") indicating that the image has been saved as a MINC file.[1]

7 To save the script, click File > Save. The program saves the script in the scripts home directory.

8 You might add the comments in the script. In order to do that, press the Insert Comment button, and enter the comment in the dialog box that appears. Add as many comments as needed. Then, save the script.

9 Click File > Exit to close the Record New Script dialog box. The dialog box closes. You should now be able to select and run the script from the Current Script box.

1. This saves only the image with margins, not the post-blurred image.

APPLYING GAUSSIAN BLUR AND GRADIENT MAGNITUDE

In this example, we want to show you the script that does the following:

1 Clones an image;

2 Apples Gaussian Blur to the clone;

3 Saves the clone;

4 Then, applies Gradient Magnitude to the saved clone;

5 Saves the result.

To get started,

open an image of interest and then, press the Start Recording Script button. The Record New Script dialog box appears indicating that recording a script is started. Refer to Figure 308.

1 Call the Utilities > Clone (Copy). This clones the active image and the following string appears in the Recording Script dialog box:

```
Clone("input_image_1 ext_image $image1")
```

2 Apply the Gaussian Blur algorithm to the clone. The following string appears in the Recording Script dialog box:

```
GaussianBlur("input_image_1 image $image2", "do_output_new_image
boolean true", "do_process_whole_image boolean true", "do_process_in_2.5D
boolean false", "do_separable_convolution boolean true", "gauss_std_dev
list_float 1.1,1.1,1.1", "gauss_do_z_resolution_correction boolean true",
"do_process_r_g_b_channel list_boolean false,false,false")
```

See also Figure 312 (Gaussian Blur).

3 Now, save the image which appears after applying Gaussian Blur. Use either File > Save Image or File > Save Image As option. The following string appears in the Recording Script dialog box:

```
SaveImageAs("input_image_1 image $image3", "file_type string .mnc")
```

4 Apply the Gradent Magnitude algorithm to the same image. The following string appears in the Recording Script dialog box:

```
GradientMagnitude("input_image_1 image $image3", "do_output_new_image
boolean true", "do_process_whole_image boolean true", "do_process_in_2.5D
```

boolean false", "do_separable_convolution boolean true", "gauss_std_dev list_float 1.0,1.0,1.0", "gauss_do_z_resolution_correction boolean true", "do_process_r_g_b_channel list_boolean false,false,false")

See also Figure 312 (Gradient Magnitude).

5 Save the image using the Save Image As option. The following string appears in the Recording Script dialog box:

SaveImageAs("input_image_1 image $image4", "file_type string .mnc")

6 Save the script, and then close the Recording Script dialog box. You should now be able to select and run the script from the Current Script box. If the script doesn't appear under Current Script, click the Refresh Script Listing icon and try again.

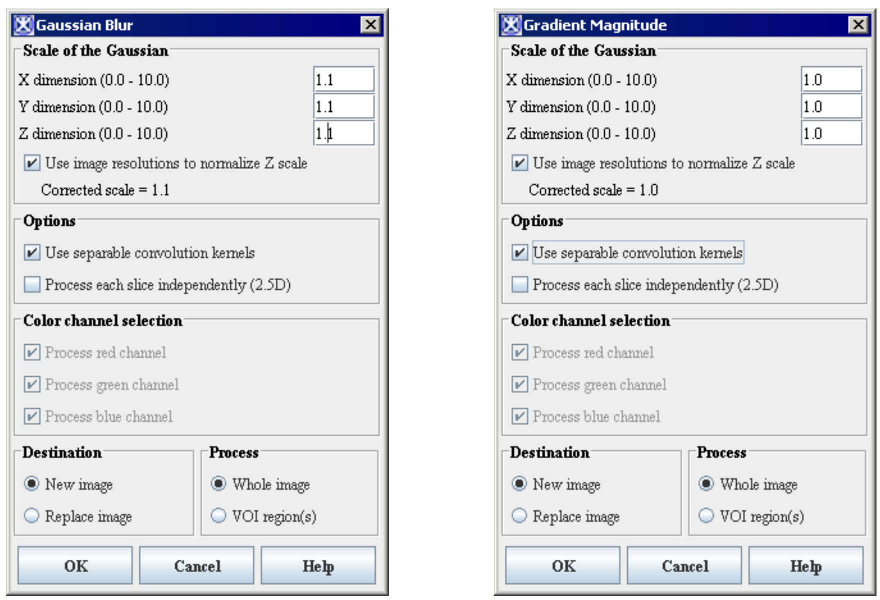


Figure 312. The Gaussian Blur and Gradient Magnitude parameters which were used to record the script

Figure 312 shows the options selected for the Gaussian Blur and Gradient Magnitude algorithms. You might consider to compare the parametes chosen in the dialog boxes with the corresponding text of the script as it shown in step 2 and step 4.

Tip: If the script doesn't appear under Current Script, click the Refresh Script Listing icon and try again.

EXTRACTING A SURFACE FROM A VOI

In this example, we want to show you how to record the script which includes:

1 Delineating a VOI on the opened image
2 Extracting a surface from the VOI

Please, remember that to apply VOIs to images in scripts, you need to create and save the VOIs in *advance* of creating a script, and then, open and apply the VOIs while you are recording the script. In this example, we will use the VOI which helps extract the cerebellum from the brain image. See Figure 313.

To delineate the VOI on the image, use the Draw Polygon/Polyline VOI tool from the MIPAV toolbar. After you've created the VOI, save it through the VOI >Save VOI command.

Figure 313. The Cerebellum VOI is delineated on the image

To record the script

Open an image of interest and then, press Start Recording Script button. The Record New Script dialog box appears indicating that recording a script is started.

1 Call the VOI > Open VOI menu, and then select the VOI file. The following string appears in the Record New Script dialog box:

```
OpenVOI("input_image_1 ext_image $image1")
```

Here, the command in the script is `OpenVOI(parameters)`; in other words, apply a VOI to `$image1`, which is listed in the parameters list with a label `input_image_1`.

2 Now, select the VOI and then call Algorithms > Extract Surface (marching cubes).

3 In the Extract Surface dialog box that appears, specify the algorithm parameters as shown in Figure 314, then press OK.

4 The following string appears in the Record New Script dialog box:

ExtractSurfaceCubes("input_image_1 image $image1", "extraction_type string VOI", "do_decimate boolean true", "file_name string brain.sur", "do_blur_before_extraction boolean false", "blur_std_dev float 0.5")

Here, the command in the script is ExtractSurfaceCubes(parameter*1*,..parameter*N*). The parameters are as follows:

"input_image_1 image $image1" indicates that the script is applying to the image

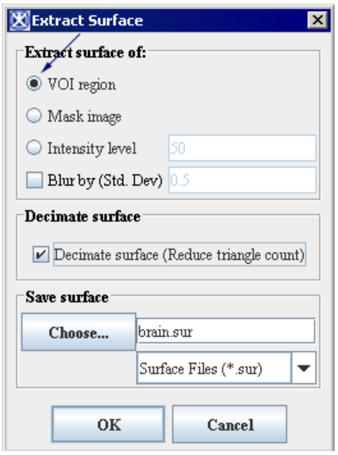
"extraction_type string VOI" corresponds to selecting the VOI region option in the dialog box and indicates that the algorithm is applied only to the image region bounded by the VOI

"do_decimate boolean true" corresponds to checking the Decimate Surface parameter in the dialog box

"file_name string brain.sur" indicates that the extracted surface will be saved as the brain.sur file

"do_blur_before_extraction boolean false" – shows that there is no blurring will be applying to the image before extraction. This corresponds to *not checking* the Blur by (Std.Dev.) checkbox

"blur_std_dev float 0.5" – the default value for the Blur by (Std.Dev.) parameter. It is only used if the above parameter is set to true



Check the VOI Region checkbox;

Check the Decimate Surface parameter;

Do not check the Blur by checkbox;

Enter the surface file name;

Press OK.

Figure 314. The Extract Surface dialog box

5 Now, click File > Save. The program saves the script in the scripts home directory.

6 Click File > Exit to close the Record New Script dialog box. The dialog box closes. You should now be able to select and run the script from the Current Script box. If the script name doesn't appear in Current Script, press Refresh to reload the scripts home directory.

Tip: The file extension for the script file is ".sct." Make sure that you type this extension at the end of the file name.

EXAMPLES OF WORKING SCRIPTS

Below are the summary of the scripts that were used as examples in Section "Examples:" on page 510. Comments using in the scripts are shown as *#comment*.

Script: Cloning an image, then adding margins and applying Gaussian Blur to the clone.

```
#Clone an image
Clone("input_image_1 ext_image $image1")
#Add margins to the clone – $image1
AddMargins("input_image_1 image $image2", "do_output_new_image boolean false", "left_side int 20", "right_side int 20", "top_side int 20", "bottom_side int 20", "front int 0", "back int 0", "margin_value double -5213.51513671875", "margin_value_rgb list_double 0.0,0.0,0.0")
#Apply GB and create a new clone – $image2
GaussianBlur("input_image_1 image $image2", "do_output_new_image boolean true", "do_process_whole_image boolean true", "do_process_in_2.5D boolean false", "do_separable_convolution boolean true", "gauss_std_dev list_float 1.0,1.0,1.0", "gauss_do_z_resolution_correction boolean true", "do_process_r_g_b_channel list_boolean false,false,false")
#Close the original image
CloseFrame("input_image_1 image $image1")
#Save the clone 2 – $image2
SaveImageAs("input_image_1 image $image2", "file_type string .mnc")
```

Script: Flipping an image twice.

```
#Flip the image vertically
```

```
Flip("input_image_1 ext_image $image1", "flip_axis string X")
#Flip the result image horizontally
Flip("input_image_1 image $image1", "flip_axis string Y")
#Save the result
SaveImageAs("input_image_1 ext_image $image1", "file_type string .mnc")
```

Script: Applying Gaussian Blur and then Gradient Magnitude.

```
#Applying Gaussian Blur and sending output to as a new image frame –
$image2
GaussianBlur("input_image_1 ext_image $image1", "do_output_new_image
boolean true", "do_process_whole_image boolean true", "do_process_in_2.5D
boolean false", "do_separable_convolution boolean true", "gauss_std_dev
list_float 1.0,1.0,1.0", "gauss_do_z_resolution_correction boolean true",
"do_process_r_g_b_channel list_boolean false,false,false")
#Applying Gradient Magnitude to $image2 and sending output to as a new
image frame –$image3
GradientMagnitude("input_image_1 image $image2", "do_output_new_image
boolean true", "do_process_whole_image boolean true", "do_process_in_2.5D
boolean false", "do_separable_convolution boolean true", "gauss_std_dev
list_float 1.0,1.0,1.0", "gauss_do_z_resolution_correction boolean true",
"do_process_r_g_b_channel list_boolean false,false,false")
```

Combining scripts and other programs- TBD

To increase productivity and efficiency, you can integrate MIPAV functions into your normal workflow by calling the scripts you've created. A simple example is creating a DOS batch file that opens and runs MIPAV scripts.

Caution: this functionality has changed and the text below needs revision!

Using the mipav command

To call scripts from other programs, you use the **mipav** command in the Command Prompt dialog box. The correct syntax of this command follows.

Syntax of the mipav command

mipav [-hH] [-iI] imageFileName [-sS] ScriptFileName [-vV] voiFileName [-hideHide]

Parameters	Purpose
[-h][-H]	Displays help for the **mipav** command in a Command Prompt window
[-hide][-HIDE]	Hides application frame
[-i][-I]	Image file name
[-s][-S]	Script file name
[-x][-X]	XML script file name
[-v][-V]	VOI file name
[-o][-O]	Specifies the output file name when "Save Image As" script command is used
[-d] [-D]	Set a value of a variable used in the script
[-inputdir] [-INPUTDIR]	Specifies the input directory with images
[-outpudir] [-OUTPUTDIR]	Specifies the output directory with images
Exit	Exits the MIPAV program

Figure 315 shows examples of the **mipav** command.

Note: When calling MIPAV scripts from other programs, be sure to enter an Exit command at the end of the MIPAV script.

Example 1: Starts MIPAV
`> mipav`
Example 2: Starts MIPAV and opening an image
`> mipav imageFileName`
Example 3: Starts MIPAV but does not display frame, opens an image, and runs a script on the image.
`> mipav -i imageFileName -s scriptFileName -hide`
Example 4: Starts MIPAV, runs a script, opens the first image, opens two VOIs associated with that image, opens a second image, and associates another VOI with that image
`> mipav -s scriptFileName -i imageFileName1 -v voiName1 -v voiName2 -i imageFileName2 -v voiName3`
Example 5: Exiting the MIPAV program
> mipav Exit

Figure 315. Examples of using the mipav command

To display help for using the mipav command

1. Navigate to the mipav directory on your computer.
2. Select Start > All Programs > Accessories > Command Prompt. The Command Prompt dialog box opens.
3. Type **mipav -H** (refer to Figure 316).

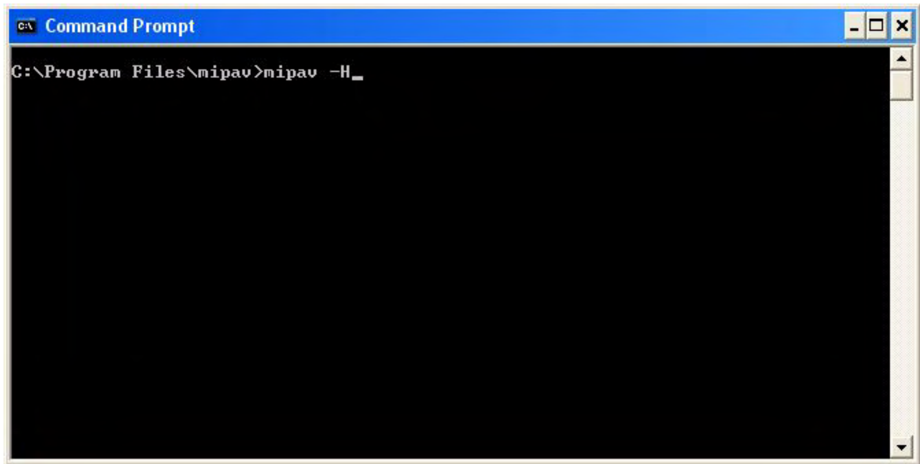


Figure 316. Command Prompt dialog box showing command to open Command Line Help dialog box

4 Press Enter. The Command Line Help dialog box (Figure 317) opens.

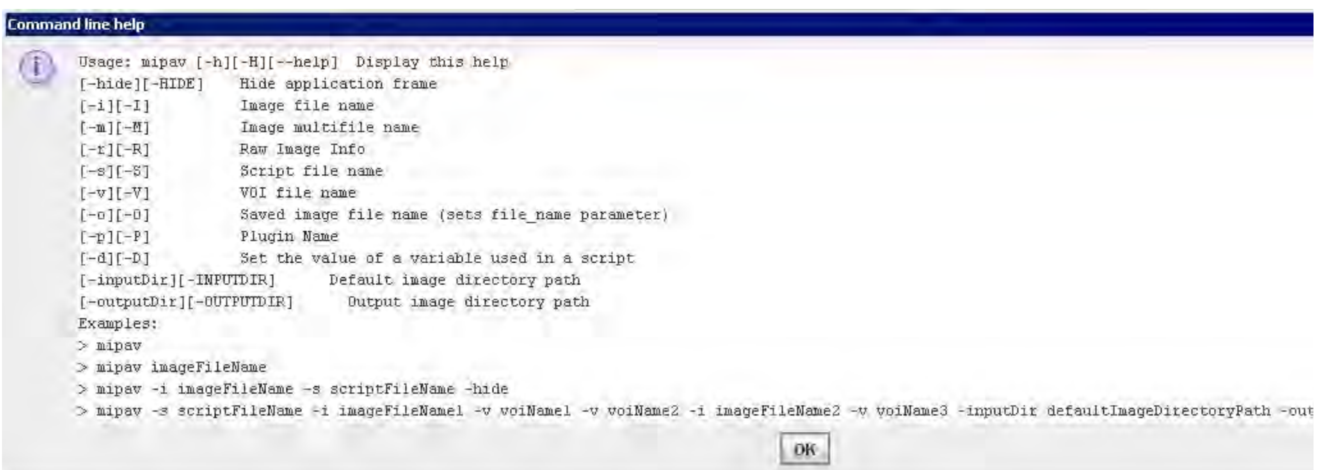


Figure 317. Command Line Help dialog box, which shows the syntax of the mipav command as well as examples

To open a DICOM image dataset

Suppose you want to open a single DICOM image from a collection of experiments made in 2004 named exp2004. You would type the following command in the Command Prompt dialog box in Microsoft Windows XP:

```
C:\ mipav -i i:\images\DICOM\exp2004\I04301.dcm
```

To open VOIs into that image

You can open VOIs as well as the image from the command line. In Windows XP, it would be the following:

```
C> mipav -i i:\images\DICOM\exp2004\I04301.dcm -v i:\VOIs\exp2004\levelset1.xml
```

In a UNIX BASH shell, this command looks like:

```
$ mipav -i ~/images/DICOM/exp2004/I043401.dcm -v ~/VOIs/exp2004/levelset1.xml
```

To open multiple images using compound commands

Suppose you know that there were multiple DICOM datasets in exp2004. To open every DICOM image on the Windows computer, you would type:

```
C> for %f in (i:\images\DICOM\exp2004\*01.dcm) do mipav -i %f
```

In this case, you must know something of the file structure of that dataset—you assumed that all image datasets had only one image ending in 01. However, the disadvantages of this format is the possibility of not opening all of the images at the same time.

A similar loop to open image sets on a UNIX BASH command line looks like:

```
$ for FI in `ls ~/images/DICOM/exp200?/*01.dcm`; do ./mipav -i $FI &; done
```

There are three significant differences between the BASH command and the Windows command (beside from how a directory is specified):

- **The use of the ls command when listing the directory**—The reason you must list the ls is due to the way a for loop works in BASH. The **for** requires a command and uses that command's return value as the boolean test to continue repeating the interior list of commands. By contrast, the Windows command shell expects a list of files. So long as the file listing has more results to list, BASH continues to repeat the **mipav** command.

- **The use of a wildcard when listing the directory**—BASH allows the directory list to use wildcard characters in more than one location, which permits searching for the images in any seven-character directory beginning with exp200 as well as all files ending in 01.dcm. This means that MIPAV starts with images from the exp2004 directory, as well as exp2003 or should it exist, exp200M, since the ? matches any character, not just a number. This is an example of a feature of the shell being used to expand the results. Windows command shell does not support this feature.
- **Sending the mipav command to operate in the background**—BASH is a shell that allows *job control*. Using this feature allows you to start MIPAV and continue it asynchronously, permitting BASH to retain control. BASH can then continue processing the loop and starting MIPAV with the next matching file. Each MIPAV runs concurrently and allows you to manipulate each image with MIPAV at will. Although this allows you to see the images at the same time, the disadvantage is that the various windows begin to clutter the screen causing operator confusion.

When there is more than one MIPAV application window running, it's possible to close the wrong image by closing the wrong MIPAV application. In addition, operations that can occur between windows when running a single MIPAV may not be transferable between images being run by separate MIPAV windows.

While starting more than one MIPAV to display a set of images may be fine in limited applications, it causes needless overhead within in the operating system wasting system resources.

Using Shell scripting to lessen typing

Using shell scripts to reduce the amount of repetitive work is a common reason for writing a script. When best used, several small scripts that work in concert can reduce the amount of typing required and the amount of time needed and can automate tasks.

The following example uses a Windows command shell to illustrate how you can shorten the number of keystrokes required. In this case, you would write a batch file to load a levelset VOI into an image.

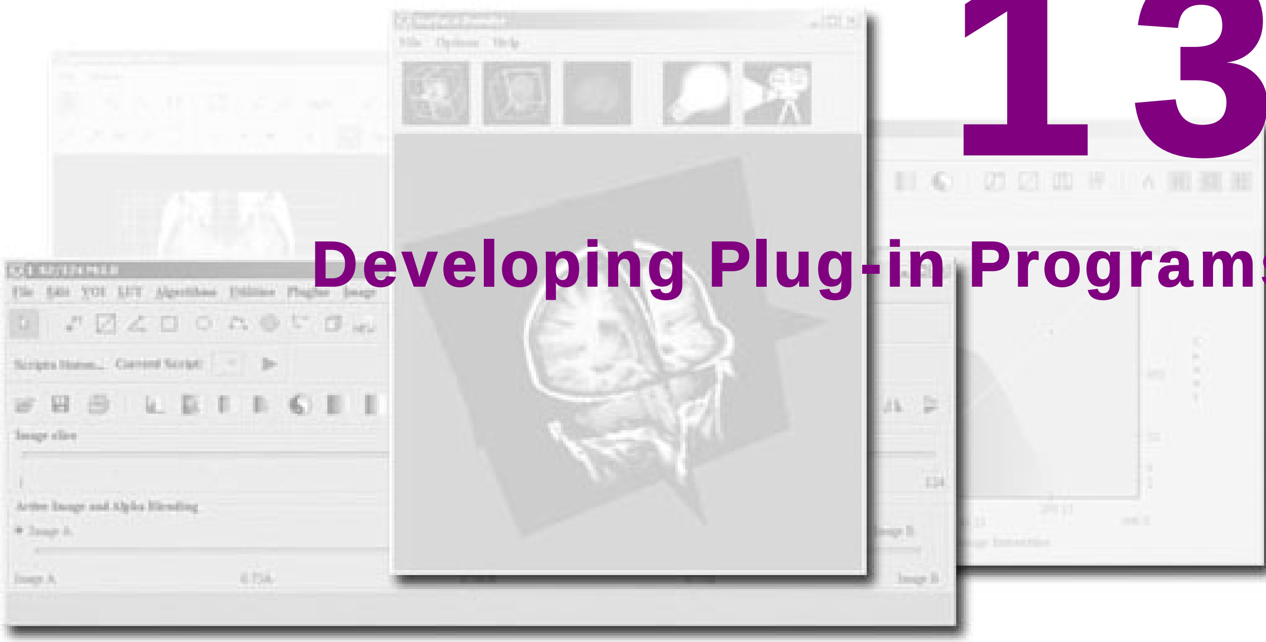
```
@echo off
rem -- %1 is the full path to the image file, though not
rem -- the file itself; we assume there to be a *01.dcm
rem -- file to exist in this directory.

rem -- VOI is assumed to be in the same directory with name
rem -- levelset1.xml

./mipav -i %1\*01.dcm -v %1\levelset1.xml
```

More efficient and more useful, starting MIPAV with multiple images is easily done in a simple script. Here is how it is done in BASH:

```
#! /bin/bash
# argument 1 is the file (with wild-cards) we want to open
# arg 1 must be escaped (with quotes) to allow the shell to send
# the wildcards unmolested to the script. Otherwise, the shell
# will try to expand the shorthand. This has a different effect.

LISTING=`ls $1`  # Generate the file listing
MIPARGS=
  # For each file in the listing, prepend it with '-i' and
  # the filename, then follow it with all the previous
  # files.
for FS in $LISTING;
 do
 MIPARGS=" -i $FS $MIPARGS";
done

 # start MIPAV:
./mipav $MIPARGS
```

Although this script doesn't include the line `./mipav $MIPARGS` with a “&” to run MIPAV in the background, it could have. This would have the effect of exiting the script with MIPAV in the background; as it is, the script does not exit—and return control to you at the command line—until MIPAV exits.

13



Developing Plug-in Programs

In this chapter. . .

Users who know how to program in Java can write a *plug-in program* that adds support for a new file format, creates a new view, or applies a new algorithm to an image. *This chapter does not intended to explain how to write a Java program; rather it presents information to help users who are writing plug-in programs to customize MIPAV.* You can find in this chapter how to:

- Gain access to and use the online MIPAV application programming interface (API) documentation
- Determine which version of Java to use
- Select one of the three plug-in types
- Include mandatory lines of code in plug-in programs so that they interface correctly with MIPAV
- Install plug-in programs

Understanding plug-in programs

Plug-in programs, also known simply as *plug-ins*, are utilities or sets of instructions that add functionality to a program without changing the program. In MIPAV, you use Java to write and compile plug-in programs to perform specific functions, such as automatically removing all odd-numbered images from the image dataset or adding support for a new file format. There are three types of plug-in programs that you may write for MIPAV:

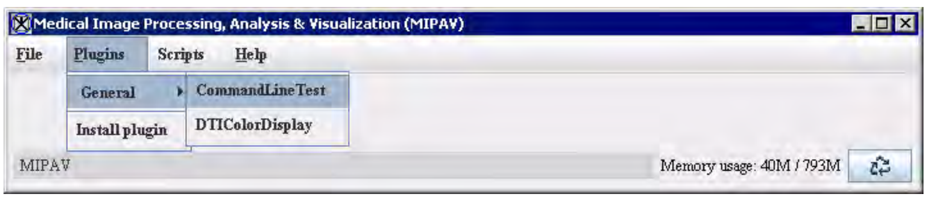
- **Algorithm**—An algorithm type of plug-in performs a function on an image. An example is a plug-in that applies a radial blur algorithm to an image. You can create plug-in algorithms through Java.
- **File**—A file type of plug-in allows MIPAV to support a new file format. An example is a plug-in that allows MIPAV to view Kodak Photo CD files (.pcd).
- **View**—A view type of plug-in introduces a new view, or the way in which the image is displayed. Examples include the lightbox, triplanar, and animate views.

Note: Because MIPAV already supports a large number of file formats and views and its development team makes it a practice to extend its capabilities in these areas, it is generally unnecessary to add file or view types of plug-ins. Most plug-in programs, therefore, are algorithms.

After developing a plug-in program, you can then install the plug-in program into the MIPAV application and access it from the PlugIns menu in the MIPAV window. The MIPAV window labeled "(A)" in Figure 318 shows the PlugIns menu as it appears before any plug-in programs are installed. The picture labeled "(B)" in Figure 318 shows the PlugIns menu as it appears after two plug-in programs—in this case, the Fantasm plug-in program and the Talairach Transform plug-in program—are installed. Because the Fantasm and Talairach Transform plug-in programs are algorithms, they appear under the PlugIns > Algorithm menu.

Note: If a plug-in program is a file type of plug-in, it would appear under a PlugIns > File menu. If it is a view type, it would appear under a PlugIns > View menu.

a



b

Figure 318. Plug-ins menu in the MIPAV window: (a) Before a plug-in was installed and (b) after two plug-ins were installed

Using the API documentation

Documentation for the application programming interface (API) is located on the MIPAV web site <http://mipav.cit.nih.gov/>. You can use the documentation directly on the web site. However, if your internet access is limited or slow, you can download, install, and use either a zipped version of the documentation on a Windows workstation or a tar version on a UNIX workstation.

To access the API documentation via the internet

1. Go to the MIPAV web site:< http://mipav.cit.nih.gov/>.
2. Click Development in the links on the left side of the page. The Development page appears. See Figure 319.
3. Here, use the following links: MIPAV API and MIPAV XML based Formats.

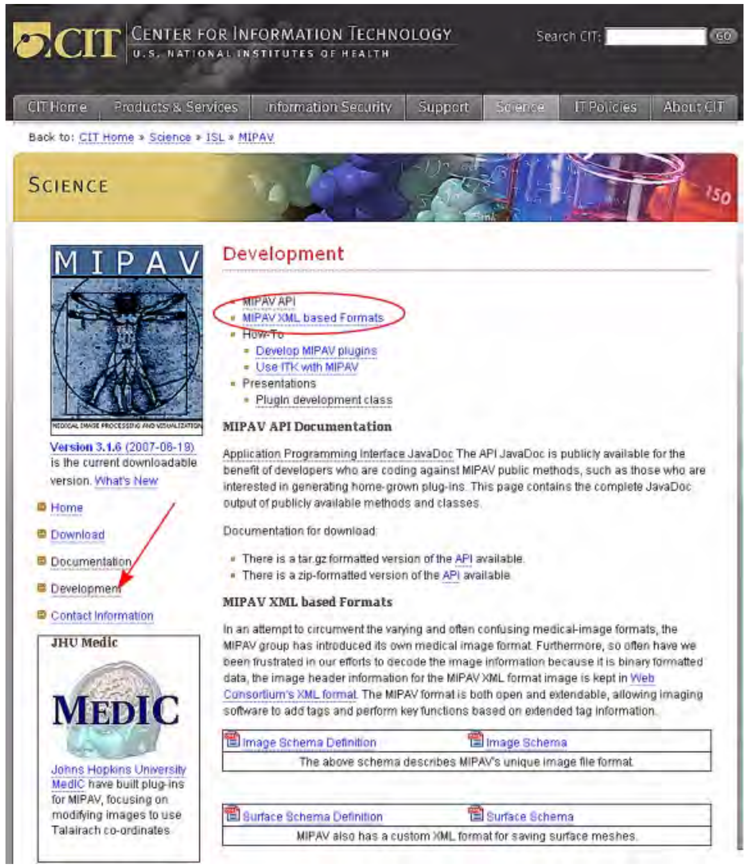


Figure 319. The Development page on the MIPAV web site offers a lot of helpful links

To download and install the API documentation on a Windows workstation,

1. Under **Documentation for download**, select a zip-formatted version. Save the file to a directory of your choice.

2. Go to the directory, double-click `api.zip`, and extract the files. Extraction creates a directory named "api" under the directory you chose to place the files.

3. Open the api directory, and double-click `index.html`. The API documentation appears in your browser.

To download and install the API documentation on a UNIX workstation,

1 Under **Documentation for download**, select a tar.gz-formatted version. Save the file to a directory of your choice.

2 Go to the directory, double-click `api.tar.gz`, and extract the files. Extraction creates a catalogue named "api" under the directory you chose to place the files.

3 Open the `api` directory, and double-click `index.html`. The API documentation appears in your browser.

Viewing MIPAV API documentation online

On the Development page, click the Application Programming Interface JavaDoc link <http://mipav.cit.nih.gov/documentation/api/index.html>. The API documentation page appears displaying the following three frames:

- **Top left frame**—Shows all of the Java packages for the MIPAV application. When you select the All Classes link at the top of this frame, all of the classes in MIPAV appear in alphabetical order in the bottom left frame. If you select a particular package, the bottom left frame displays only the classes that pertain to the selected package.
- **Bottom left frame**—Lists either all of the classes in the MIPAV application or all of the classes in a selected package.
- **Right frame**—Displays information based on the command that you select in the menu at the top of the frame:
- **Overview**—Lists all of the packages in the MIPAV application
- **Package**—Lists and summarizes all of the classes and interfaces in the package
- **Class or Interface**—Lists descriptions, summary tables, and detailed member descriptions
- **Tree**—Displays a hierarchy of the class or package
- **Deprecated**—Lists deprecated APIs

- **Index**—Provides an alphabetical list of all classes, interfaces, constructors, methods, and fields
- **Help**—Provides help for the API documentation

Several links appear beneath the menu.

- **Prev and Next**—These links take you to the next or previous class, interface, package, or related page.
- **Frames and No Frames links**—These links show and hide the HTML frames. All pages are available with or without frames. See Figure 320.

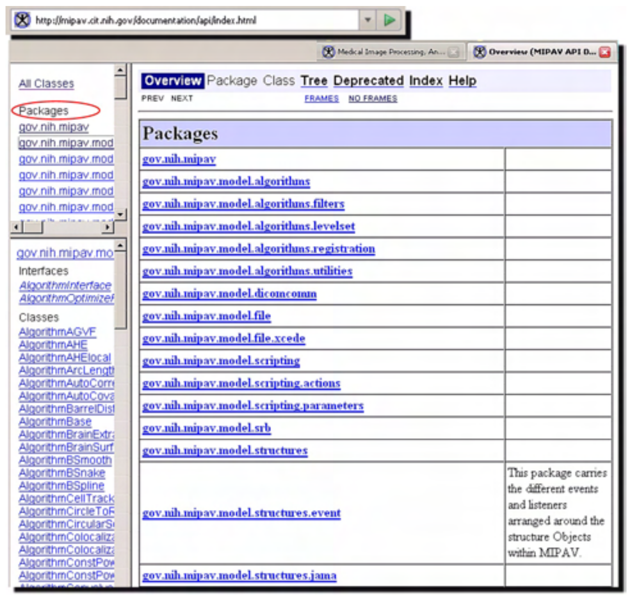


Figure 320. The Overview page

OVERVIEW PAGE

The Overview page is the page that initially appears when you gain access to the API documentation. This page displays a list of all of the packages in MIPAV. The Overview menu becomes available after you move to another page. To return to the Overview page from the any other page, click

Overview. The Overview page appears and displays a list of all of the packages in MIPAV.

PACKAGE PAGE

When you select one of the packages listed on the Overview page, the Package page appears. This page provides a summary of each interface (if any), class, and exception (if any) in the package. When you click an interface or class, the Interface page or the Class page appears. Clicking an exception displays the Exception page. See Figure 321.

INTERFACE OR CLASS PAGES

When you select an interface or class on the Package page, either the Interface page or the Class page appears. Each interface, nested interface, class, and nested class has its own separate page. Each of these pages has three sections consisting of an interface or class description, summary tables, and detailed member descriptions:

- Class inheritance diagram
- Direct known subclasses
- All known subinterfaces or subclasses
- All known implementing classes
- Interface or class declaration
- Interface or class description
- Nested class summary
- Field summary
- Constructor summary
- Method summary
- Field detail
- Constructor detail
- Method detail

Each summary entry contains the first sentence from the detailed description for that item. The summary entries are alphabetical, while the detailed descriptions are in the order they appear in the source code. This preserves the logical groupings established by the programmer. See also Figure 321.

Note: Each serialized or externalized class has a description of its serialization fields and methods. This information is of interest to re-implementors, not to developers using the API. To access this information, go to any serialized class and clicking Serialized Form in the See also section of the class description.

Exception page

The Exception page appears when an exception on the Package page is selected. This page includes a constructor summary and constructor detail.

Tree (Class Hierarchy) page

When you click Tree on the menu, a Tree, or class hierarchy, page appears. This page displays either the class hierarchy for a particular package, or, if you select All Packages, the class hierarchy for all packages. See Figure 321.

- If you were viewing the Overview page and then clicked Tree, the class hierarchy for all packages appears on the Tree page.
- If you were viewing a Package, Interface, Class, or Exception page and then clicked Tree, the hierarchy for only that package, which includes the class, interface, and exception hierarchies, appears on the Tree page.

Each hierarchy page contains a list of classes, interfaces, and exceptions (if any). The classes are organized by inheritance structure starting with `java.lang.Object`. The interfaces do not inherit from `java.lang.Object`.

Deprecated API page

The Deprecated API page appears when you click Deprecated on the menu. This page lists all of the methods in the API that have been deprecated. A deprecated method is **not recommended** for use, generally due to improvements, and a replacement API is usually given.

Warning: Deprecated APIs may be removed in future implementations.

Index

The Index page provides an alphabetic list of all classes, interfaces, constructors, methods, and fields with definitions of each. Clicking an entry displays the usage in the product.

Help page

The Help page provides help for using the API documentation.

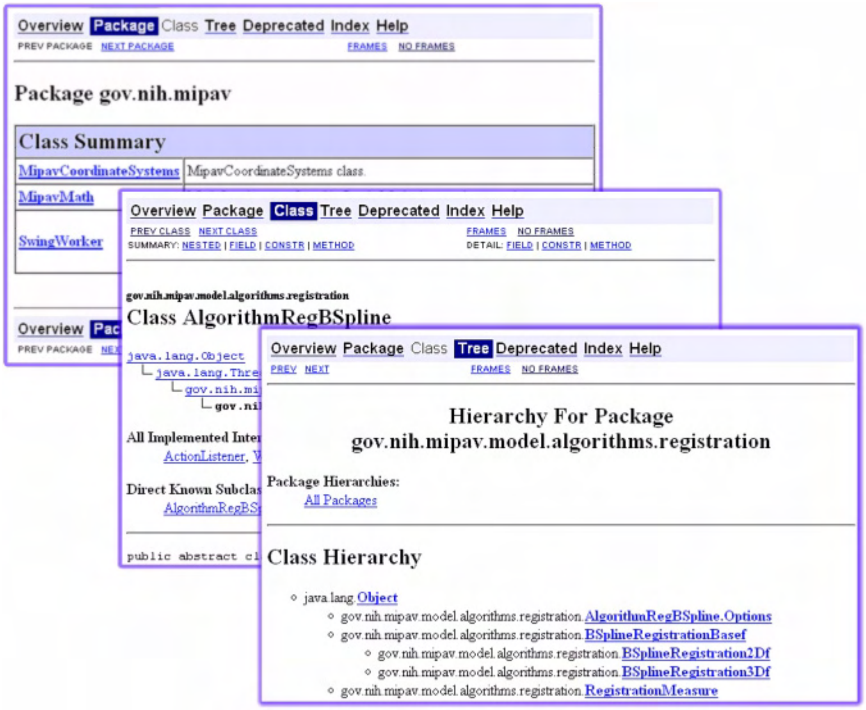


Figure 321. The Package, Case and Tree pages of MIPAV API

TO DISPLAY,

All interfaces, classes, and exceptions in a package

1 Go to <http://mipav.cit.nih.gov/documentation/api/>. The Overview page appears.

2 Click one of the packages listed in the:

- **Frame on the right**—When you click one of the packages listed on this page, the Package page appears in the frame. The Package page displays a list of all interfaces, classes, and exceptions (if any) in the package.
- **Top frame on the left**—The top frame on the left also lists all of the packages. When you select a package, the bottom frame on the left displays a list of interfaces, classes, and exceptions (if any) in the package.

The methods associated with an interface or with a class

1 Go to <http://mipav.cit.nih.gov/documentation/api/>. The Overview page appears.

2 Do either of the following:

- Click one of the packages listed in the frame on the right or in the top frame on the left. The Package page appears in the right frame.
- Click one of the packages in the top frame on the left. A list of interfaces, classes, and exceptions appear in the bottom frame on the left.

3 Do one of the following:

- Click an interface. The Interface page appears in the right frame.
- Click a class. The Class page appears in the right frame.

4 Scroll down the page, or click METHODS beneath the menu. The Method Summary table appears.

5 Click a method. The Method Detail section of the page, which lists a description of the method and its parameters, throws, and returns, appears.

Developing plug-in programs

MIPAV provides the following classes for developing plug-in programs:

- PlugInAlgorithm.class
- PlugInFile.class
- PlugInView.class

Plug-in programs are developed in the same way as the other Java programs are. The high-level steps of creating plug-ins follow.

1 **Determining the type of plug-in program**—Before you begin to write the code for the plug-in, determine the plug-in type: algorithm, file, or view. Refer to "Determining the type of plug-in program" .

2 **Determining which version of Java to use**—Detailed instructions appear in "Determining which version of Java to use" and Figure 322.

3 **Writing the source code**—Some lines of code must appear in the source code so that the plug-in program interfaces correctly with MIPAV. Refer to "Writing the source code" .

4 **Building and compiling plug-in programs**—You should keep back-up copies of the source and compiled files in case you need to update or change plug-in programs. See "Building and compiling plug-in programs" .

5 **Creating a self-contained plug-in frame**—A self-contained plug-in is a Java application that does not rely on the default MIPAV user-interface, but, instead, hides MIPAV and display its own image(s) with action/algorithm handling specific to its frame. See "Creating a self-contained plug-in frame" .

6 **Installing plug-in programs**—This section explains how to install plug-in programs. Refer to "Installing plug-in programs" .

7 **Sample plug-in programs**—This section provides a couple of examples of MIPAV plug-ins. Refer to "Examples of MIPAV plug-ins" .

Note: This section does not explain how to write a Java program; however, it explains what must be incorporated in the plug-in program so that it correctly interfaces with the MIPAV application.

Determining the type of plug-in program

The first step of creating a plug-in program is to determine the type you want to create, which depends on its purpose. As mentioned earlier, MIPAV plug-in programs can be of the algorithm, file, or view type. However, most users want MIPAV to perform very specific additional functions on images. Since these functions may not be currently available in MIPAV, users choose to add the functions by developing the algorithm type of plug-in program.

Determining which version of Java to use

To avoid compatibility problems when you create a plug-in program, use the same version of Java that was used to create MIPAV. To determine which version of Java the latest version of MIPAV uses, select Help > JVNM

Information in the MIPAV window. The About System dialog box opens. See Figure 322.

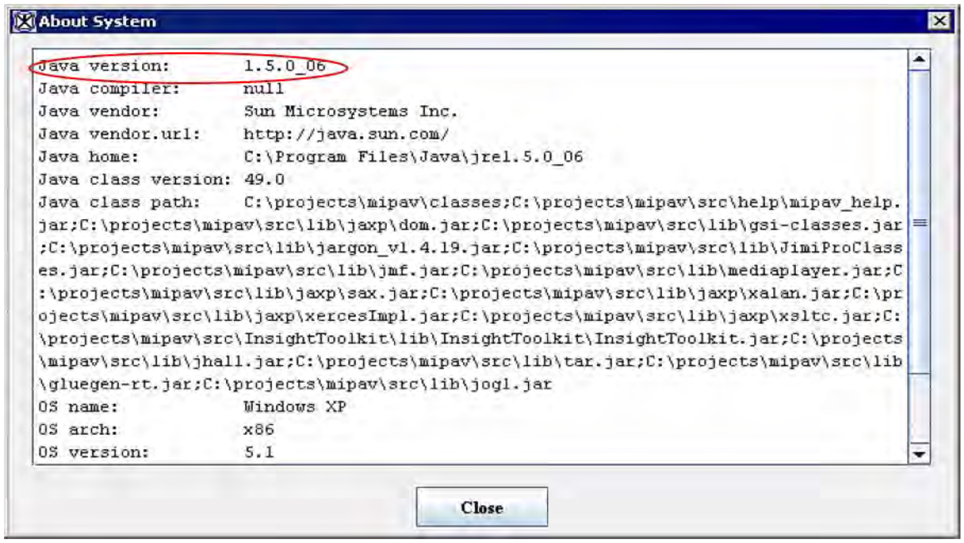


Figure 322. About System dialog box

The first line in the About System dialog box indicates the version of Java that was used to develop MIPAV. To obtain the correct version of Java, go to the following web site: <http://www.java.sun.com>

Writing the source code

Note: In this section, \$MIPAV is used to represent the MIPAV user directory, which is the directory where MIPAV is installed. The user directory is indicated in the About System dialog box. In the MIPAV main window, select Help > JVM Information to view the About System dialog box.

When you develop a plug-in for MIPAV, several lines must be present in the code so that it executes properly. Some mandatory code should be included in **all** plug-in files. Other code might change depending on the plug-in type.

Including mandatory code

The next three figures (Figure 323—Figure 325) show the mandatory source code needed for creating a file type of plug-in, a view type of plug-in, and an algorithm type of plug-in. The plug-ins directory of MIPAV includes these three files (e.g. C:\[$MIPAV]\mipav\plugins):

- **PlugInFile.java**—Mandatory source code for a file type of plug-in. See Figure 323;
- **PlugInView.java**—Mandatory source code for a view type of plug-in. See Figure 324;
- **PlugInAlgorithm.java**—Mandatory source code for an algorithm type of plug-in. See Figure 325.

```
package gov.nih.mipav.plugins;

import gov.nih.mipav.view.*;

import java.awt.*;

public interface PlugInFile extends PlugIn {

    /**
     *   run
     *   @param UI           MIPAV main user interface.
     */
    public void run(ViewUserInterface UI);
}
```

Figure 323. Mandatory code for a file type of plug-in (PlugInFile.java). For readability purposes, keywords in all code reproduced in this chapter appear in bold, and comments appear in green type

```
package gov.nih.mipav.plugins;

import gov.nih.mipav.model.structures.*;
import gov.nih.mipav.view.*;

import java.awt.*;

public interface PlugInView extends PlugIn {

    /**
     *   run
     *   @param UI           MIPAV main user interface.
     *   @param parentFrame frame that displays the MIPAV image.
     *                       Can be used as a parent frame when building
     *                       dialogs.
     *   @param image        model of the MIPAV image.
     *   @see   ModelImage
     *   @see   ViewJFrameImage
     *
     */
    public void run(ViewUserInterface UI, Frame parentFrame, ModelImage image);
}
```

Figure 324. Mandatory code for a view type of plug-in (PlugInView.java). For readability purposes, keywords in all code reproduced in this chapter appear in bold, and comments appear in green type

```
package gov.nih.mipav.plugins;

import gov.nih.mipav.model.structures.*;
import gov.nih.mipav.view.*;

import java.awt.*;


public interface PlugInAlgorithm extends PlugIn {

    /**
     *   run
     *   @param UI           MIPAV main user interface.
     *   @param parentFrame frame that displays the MIPAV image.
     *                       Can be used as a parent frame when building
     *                       dialogs.
     *   @param image        model of the MIPAV image.
     *   @see   ModelImage
     *   @see   ViewJFrameImage
     *
     */
    public void run(ViewUserInterface UI, Frame parentFrame, ModelImage image);


}

```

Figure 325. Mandatory code for an algorithm type of plug-in (PlugInAlgorithm.java). For readability purposes, keywords in all code reproduced in this chapter appear in bold, and comments appear in green type

REFERENCING FILES

To reference a class, you must specify it using the Import keyword. For example, line 2 in PlugInFile.java imports the view functions (Figure 326).

```
import gov.nih.mipav.view.*;
```

Figure 326. Importing the view functions in PlugInFile.java

Lines 3, 4, and 6 in the PlugInView.java and PlugInAlgorithm.java files import the model structures, view functions, and the basic Java package that has GUI functions (Figure 327).

```
import gov.nih.mipav.model.structures.*; // MIPAV package where main
     // MIPAV structures are located (e.g., model image)
import gov.nih.mipav.view.*;

import java.awt.*
```

Figure 327. Importing model structures, view functions, and [java.awt]

If you reference a class, you must include it in the plug-in package so that it can be called from the main file. After you write and compile, you must now install files in the user or home directory:

Windows

`c:\Documents and Settings\<user ID>\mipav\plugins`

UNIX

`/user/<user ID>/mipav/plugins`

An example of this appears in the first line of Figure 328.

```
package plugins; // added to plugins pkg. so PlugInSampleStub may
                 // call it.
```

Figure 328. Example of placing referenced files in the \$MIPAV\plugins directory

Lines of code that are dependent on plug-in type

Two lines of code depend on the type of plug-in program being developed:

- Declaration
- Parameters for the run method

Declaration

The declaration used in a plug-in depends on the type of plug-in being developed. For instance, in line 9 in PlugInAlgorithm.java (Figure 325), the combination of words "**public interface** *PlugInAlgorithm*" indicates that the plug-in in an Algorithm. For File or View types of plug-ins, simply replace *PlugInAlgorithm* with *PlugInFile* (line 7 in PlugInFile.java, see Figure 323) or *PlugInView* (line 8 in PlugInView.java, see Figure 324), respectively.

Table 4. Declarations dependent on type of plug-in

Type of plug-in	Declaration
File	`public interface PlugInFile extends PlugIn (`
View	`public interface PlugInView extends PlugIn (`
Algorithm	`public interface PlugInAlgorithm extends PlugIn (`

Parameters for the run method

The parameters for the run method also depend on the plug-in type. Compare the run methods used in PlugInFile.java (Figure 323), PlugInView.java (Figure 324), and PlugInAlgorithm.java (Figure 325).

Table 5. Parameters for run methods dependent on type of plug-in

Type of plug-in	Parameters for the run method
File	`public void run(ViewUserInterface UI);`
View	`public void run(ViewUserInterface UI, Frame parentFrame, ModelImage image);`
Algorithm	`public void run(ViewUserInterface UI, Frame parentFrame, ModelImage image);`

```
package gov.nih.mipav.plugins;

import gov.nih.mipav.model structures.*;
import gov.nih.mipav.view.*;

import java.awt *;

public interface PlugInAlgorithm extends PlugIn {

/**
* run
* @param UI           MIPAV main user interface.
* @param parentFrame Frame that displays the MIPAV image.
*                      Can be used as a parent frame when building dialogs.
* @param image        Model of the MIPAV image.
* @see ModelImage
* @see ViewJFrameImage
*/
public void run(ViewUserInterface UI, Frame parentFrame, ModelImage image;)

}
```

Figure 329. PlugInAlgorithm.java. For readability purposes, keywords in all code reproduced in this chapter appear in bold, and comments appear in green type

Building and compiling plug-in programs

To build a new plug-in program for MIPAV, you must first install a build environment, alter the path environment variable, and compile the plug-in files.

INSTALLING A BUILD ENVIRONMENT

1. Download and install Java SE Development Kit (JDK), version 1.6 (JDK 6u2) **<http://java.sun.com/javase/downloads/index.jsp>**.
2. Download and install Apache Ant 1.7.0 <http://ant.apache.org/>.

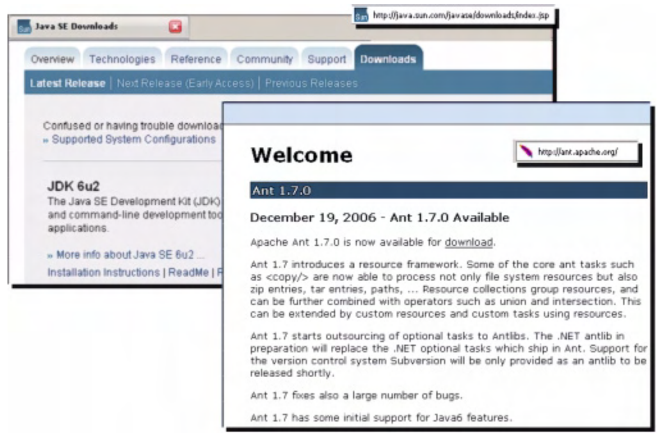


Figure 330. Download pages for Java SE Development Kit (JDK) and Apache Ant 1.7.0

CONFIGURING THE ENVIRONMENT

To configure your environment, you need to add two new variables—JAVA_HOME and ANT_HOME—and update the path variable in your system.

On Windows workstations

1 Click Start > Control Panel. The Control Panel window opens.

2 Double-click the System icon. The System Properties dialog box opens.

3 Click Advanced. The Advanced page of the System Properties dialog box appears.

4 Click Environment Variables. The Environment Variables dialog box opens.

5 Decide whether to add and edit variables in the User variables box or the System variables box based on which users should have access to the Java SDK and Ant.

6 Add the JAVA_HOME variable to your environment:

a Click New. The New User Variable dialog box or the New System Variable dialog box opens.

b Type `JAVA_HOME` in Variable name.

- **c** Type the path for the Java SDK on your computer (e.g., `C:\Program Files\Java\jdk1.6.0_02`) in Variable value.
- **d** Click OK. The JAVA_HOME variable appears in either the User variables box or System variables box as appropriate.

7 Add the **`ANT_HOME`** variable to your environment by doing the following:

- **a** Click New under either the User variables box or the System variables box. The New User Variable dialog box or the New System Variables dialog box opens as appropriate.
- **b** Type **`ANT_HOME`** in Variable name.
- **c** Type the path for the Ant on your computer (e.g., `C:\Program Files\Ant\apache-ant-1.7.0`) in Variable value.
- **d** Click OK. The ANT_HOME variable appears in either the User variables box or System variables box as appropriate.

8 Update either the PATH variable in the User variables box or the Path variable in the System variables box by doing the following:

- **a** Select the PATH variable in the User variables box, or select the Path variable in the System variables box.
- **b** Click Edit under the User variables box, or click Edit under the System variables box. Either the Edit User Variable dialog box or the Edit System Variable dialog box opens.
- **c** Type `;%JAVA_HOME%\bin;%ANT_HOME%\bin` to the end of the PATH variable or to the end of the Path variable.
- **d** Click OK. The edited variable appears either in the User variables box or the System variables box. See also Figure 331.

9 Open a new terminal for the change to take effect by doing the following:

- **a** Click Start > Run. The Run dialog box opens.
- **b** Type **`cmd`** in Open, and click OK. A terminal window opens.

10 Retrieve the sample Ant build file (build.xml) from the MIPAV web site and place it in the same directory as the plug-in `.java` files you want to compile.

11 Alter the *dir.mipav* and *dir.jdk* properties within the `build.xml` to point to the directory where MIPAV and the SDK are installed, respectively.

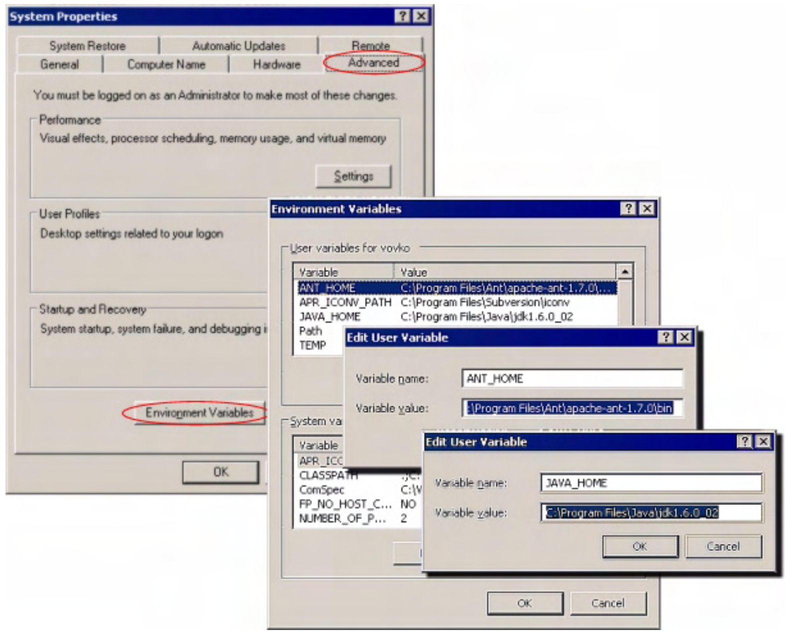


Figure 331. Configuring system variables for MS Windows

Note: Add and edit the variables in the User variables box if you want to limit the build environment to just yourself and no other users. Add and edit the variables in the Systems variables box to make the environment accessible to anyone who uses the workstation.

Recommendation: Although it is possible to update the path variable in either the User variables box or System variables box, you should add the statement to the same box in which you added the *JAVA_HOME* and *ANT_HOME* variables.

See also:

- "Installing Ant" on <http://ant.apache.org/manual/index.html>.
- "JavaTM SE 6 Release Notes—Microsoft Windows Installation (32-bit)" on <http://java.sun.com/javase/6/webnotes/install/jdk/install-windows.html>.

On Linux or UNIX workstations

Bash users should do the following:

1. Edit the file `$HOME/.bash_profile` and add lines similar to following:

```
ANT_HOME=/path/to/apache-ant-1.6.3
JAVA_HOME=/path/to/j2sdk1.4.2_08
PATH=$PATH:$JAVA_HOME/bin:$ANT_HOME/bin

export ANT_HOME
export JAVA_HOME
export PATH
```

   where ANT_HOME and JAVA_HOME are the paths where each application was installed.

2. Retrieve the sample Ant build file from the MIPAV web site, and place it in the same directory where the plug-in `.java` files you want to compile are located.

3. Alter the *dir.mipav* and *dir.jdk* properties within `build.xml` to point to the directory where MIPAV and the SDK are installed, respectively.

BUILD.XML

Figure 332 below displays the content of the **build.xml** file. build.xml is also available on the MIPAV web site <http://mipav.cit.nih.gov/documentation/presentations/plugins/build.xml>.

build.xml

```
<!-- build file for MIPAV plugin class -->
-
      <project basedir="." default="compile" name="mipav_plugin">
<property name="dir.mipav" value="c:\\Program Files\\mipav\\"/>
<property name="dir.jdk" value="c:\\Program Files\\Java\\jdk1.6.0_02"/>
-
      <target name="init">
<tstamp/>
-
      <path id="build.classpath">
<pathelement path="${dir.mipav}"/>
<pathelement location="${dir.mipav}/InsightToolkit/lib/InsightToolkit/InsightToolkit.jar"/>
-
      <fileset dir="${dir.mipav}">
<filename name="*.jar"/>
</fileset>
</path>
<property name="build.cp" refid="build.classpath"/>
</target>
-
      <target name="compile" depends="init">
<echo>classpath: ${build.cp}</echo>
-
      <javac debug="true" deprecation="true" description="Builds MIPAV" verbose="no"
listfiles="yes" nowarn="no" fork="true" memoryInitialSize="220M" memoryMaximumSize="1000M"
id="mipav build" source="1.4" target="1.4" destdir="." srcdir="." compiler="modern">
<classpath refid="build.classpath"/>
</javac>
</target>
-
      <target name="clean" depends="init">
-
      <delete>
-
      <fileset dir=".">
<include name="**/*.class"/>
</fileset>
</delete>
</target>
</project>
```

Figure 332. The contents of the build.xml file

COMPILING THE PLUG-IN FILES

Note: You should keep back-up copies of the source and compiled files in case you need to update or change the plug-in.

1 Type `ant compile` on your workstation (e.g., `cmd ant compile on` Windows or `xterm ant compile` on UNIX platforms). The BUILD SUCCESSFUL message should appear at the end of the Ant output.

2 Copy the `.class` files that Ant produced into MIPAV's plug-in directory.

- On Windows platforms:

  `C:\Documents and Settings\username\mipav\plugins`

- On UNIX platforms:

  `/home/username/mipav/plugins`

where `username` is the name of your account on the system.

3 Install the plug-in file. Select PlugIns > Install Plugin in the main MIPAV window. In the Install PlugIn dialog box, use the Browse buton to navigate to the `\plugins` directory . Select the plug-in and Press OK.

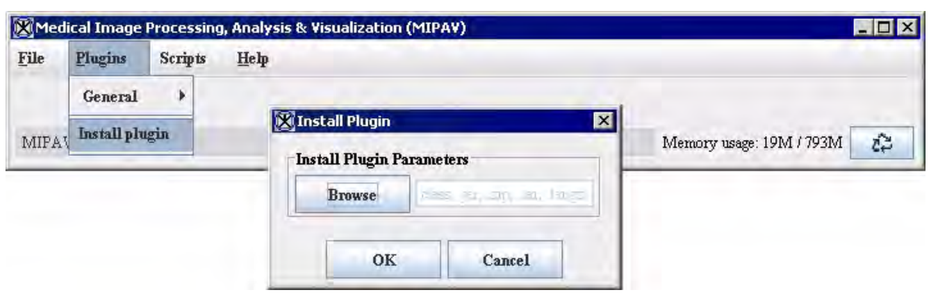


Figure 333. Installing a MIPAV plug-in.

Creating a self-contained plug-in frame

You can create a self-contained plug-in that does not rely on the default MIPAV user interface. When running, this type of plug-in hides MIPAV and displays its own image(s) with the action and algorithm handling specific to its frame.

To create a self-contained plug-in:

1. Extend `ViewJFrameImage`, as this will allow the plug-in to use a wide-range of `ViewJFrameImage` and `ViewJFrameBase` specific functions for storing and displaying `ModelImages`. These functions include, for example, the image and on-screen buffers, menu and toolbar builders, etc.
2. Override the `ViewOpenFrameInterface openFrame(ModelImage)` function. This handles the creation of a new `PlugIn` frame based on whether a result image is created within the dialog of an algorithm. For example, when the user runs an algorithm and selects the destination New Image rather than Replace Image, a new frame will be created with the result of the algorithm. To set all algorithms to work in place and disallow creating of new frames, call `ViewUserInterface setForceInPlace(true)` function, which tells the dialogs that all algorithms must work in place.
3. Create an `init()` function, where the `PlugIn` frame is layout and components will be initialized.
4. In the `init()` function, several methods should be called:
   - Call `initLUT()` for the `ModelImage` look-up table,
   - Call `initResolutions()` for the `ModelImage` resolutions,
   - Call `initZoom()` for the frame's zoom factor,
   - `initComponentImage()` creates a displayable `ComponentImage`,
   - `initExtentsVariables()` initializes z-slice and time-slice positions.
5. To add toolbars and menus to your plug-in, within `init()`, create a `ViewControlsImage` object, and then

- Call `buildToolBar()` to create pre-defined toolbars for image, VOI, paint, and scripting controls;
- Or call `buildSimpleToolBar` and pass `Vector<CustomUIBuilder.UIParams>` using `addCustomToolBar()` for each of the Vectors.

Pre-defined button and menu parameters are located in `CustomUIBuilder`. Pre-defined as well as the user-defined `UIParams` can be added and used in both toolbars and menus.

6 Create `ViewMenuBar`. This allows you to add either pre-defined or `UIParam` menus. The Vector from above (used on the custom toolbar) can be passed into the `ViewMenuBar makeCustomMenu()` function. `ViewMenuBar` also has pre-defined menus for a file, help, image, look-up tables, etc.

7 Finally, the `init()` function should handle the container for the `ViewJComponentEditImage` created from `initComponentImage()`. The component image should be added to a `JScrollPane` to accommodate the variable size of the display.

8 Override the `actionPerformed()` method to catch (handle) `ActionEvents`. If a custom toolbar and (or) menu bar was created using a Vector of `UIParams`, the `UIParam` contains the action event for each button and (or) menu item.

9 Override the `componentResized()` method to properly handle (or ignore) the resizing of the plug-in frame. Using the `ViewJFrameImage componentResized` function would likely create unwanted behavior as the layout of the plug-in is different from MIPAV's standard `ViewJFrameImage`.

10 Create a basic `PlugInGeneric` class that will be called as a command-line argument. This class should have the ability to choose/open a `ModelImage` using the `FileIO.readImage()` method. The self-contained plug-in frame should be instantiated within this class by passing in the `ModelImage`.

11 When running MIPAV, pass in the arguments `-hide -p [YourGenericPlugin]`. The `-hide` flag tells MIPAV not to bring up the User Interface and `MessageFrame`, while the `-p` flag tells which plug-in to run. See Figure 334.

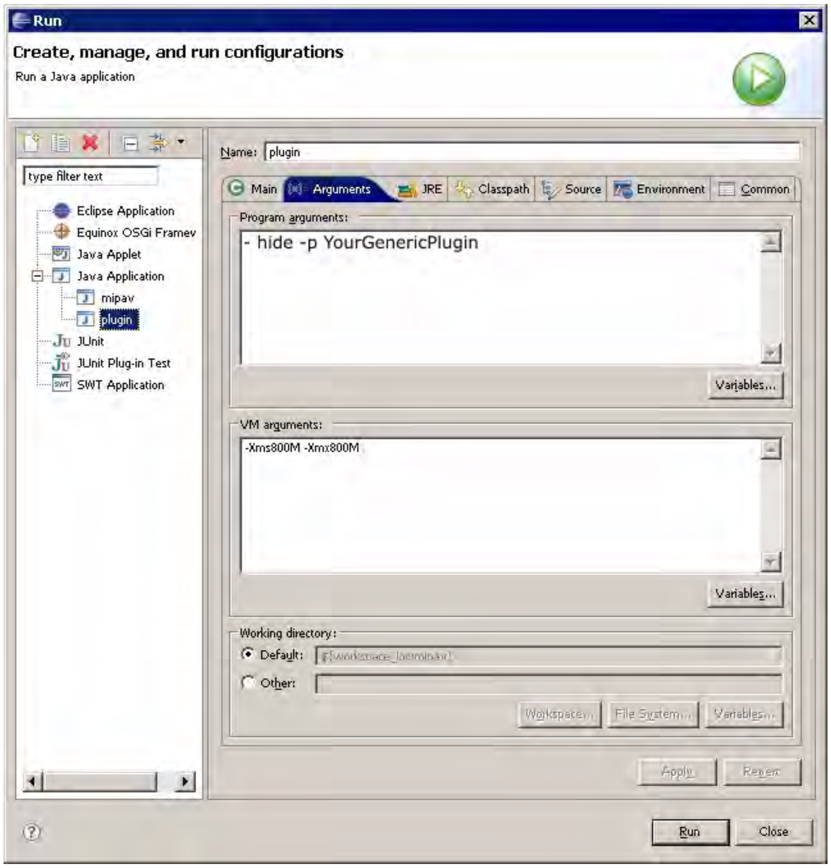


Figure 334. The arguments for running the self-contained plug-in frame.

Optional:

The `ViewJFrameMessage` Data and Debug tabs (as well as others) can be added to the plug-in frame by retrieving the `JTabbedPane` through `ViewUserInterface.getReference().getMessageFrame().getTabbedPane()`. This enables the Data and Debug message output to be displayed outside of the separate message frame that accompanies MIPAV.

See also: Figure 335 and Figure 336.

PlugInDialogImageVOIDisplay.java

```
import java.awt.*;
import java.awt.event.*;
import java.util.Vector;

import javax.swing.*;
import gov.nih.mipav.model.file.FileInfoBase;
import gov.nih.mipav.model.structures.ModelImage;
import gov.nih.mipav.model.structures.ModelLUT;
import gov.nih.mipav.model.structures.ModelRGB;
import gov.nih.mipav.model.structures.VOI;
import gov.nih.mipav.view.*;
import gov.nih.mipav.view.dialogs.*;


/**
 * Plugin example class for creating a simple, self-contained frame that extends ViewJFrameImage
 * Contains a subset of the VOI functions, as well as the message frame contained within the frame itself
 * @author linkb
 *
 */
public class PlugInDialogImageVOIDisplay extends ViewJFrameImage implements MouseListener, AdjustmentListener {



  //~ Constructors -------------------------------------------------------------------------------------------------

    /**
     * Default constructor
     */
    public PlugInDialogImageVOIDisplay(ModelImage image) {
      super(image, null, null, false, false);
        init();
    }


    /**
     * ViewOpenFrameInterface function for opening a model image (result) into a new frame
     */
    public PlugInDialogImageVOIDisplay openFrame(ModelImage image) {
      return new PlugInDialogImageVOIDisplay(image);
    }
```

Figure 335. A part of the code for PlugInDialogImageVOIDisplay.java. The full code can be found in "Examples of MIPAV plug-ins" , Figure 345.

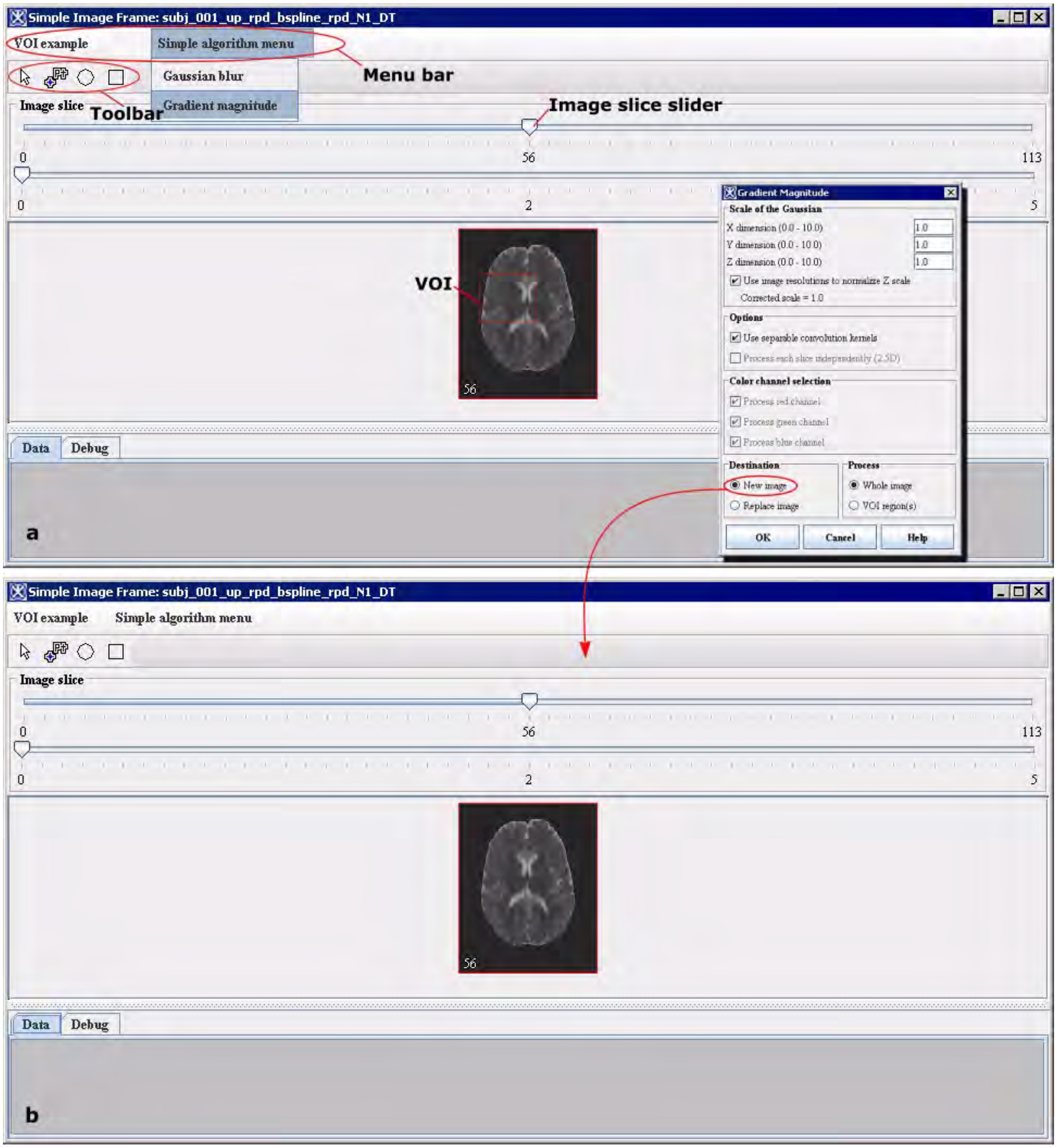


Figure 336. The Simple Image Frame plug-in opens the MIPAV independent image frame for a selected image. (a) The image frame contains the following elements of the interface – the VOI toolbar, image slice slider and menu bar. The Gradient Magnitude algorithm was called from the menu and applied to the image. (b) The result image opened in a new image frame.

Installing plug-in programs

Installing simple plug-in programs merely copies files into the user's home directory.

Windows

`c:\Documents and Settings\<user ID>\mipav\plugins`

UNIX

`/user/<user ID>/mipav/plugins`

You can choose one of two methods for copying the files:

- Use MIPAV's plug-in installation tool, e.g. in the MIPAV window, select PlugIns > Install PlugIns.
- Use the operating system's tool for copying the files. This method requires the user to restart MIPAV so that the new plug-in appears in the PlugIns menu. When MIPAV starts, it parses the user's home directory and builds the PlugIns menu.

Warning: The MIPAV installation tool does *not* work for more complex plug-ins that consist of more complicated package class hierarchy, such as the Medic Talairach plug-in program. To learn more about Medic Talairach plug-in program, refer to MIPAV Technical Guide 1.

Examples of MIPAV plug-ins

To build plug-in programs, three files are typically required:

- **Plugin*Foo*.java**—Provides an interface to MIPAV and the plugin.
- **PluginDialog*Foo*.java**—Invokes the dialog to get user-supplied parameters; it can be hidden when no parameters are required.
- **PluginAlgorithm*Foo*.java**—Provides the actual algorithm to be implemented. It can be a mixture of calls to MIPAV's API, C programs, Perl, ITK, etc.

Where *Foo* is the name that you supply for the program. The following sample plug-in program(s) are included in MIPAV documentation:

- PlugInSample—a sample plug-in, see "Sample plug-in program" below.

- PlugInCT_MD—a typical plug-in. (Refer to the MIPAV Users Guide, PDF version.)
- PlugInAlgorithm.Median—a very complicated plug-in. Refer to MIPAV Volume 1 Users Guide, Appendix D.
- PlugInDialogImageVOIDisplay.java – a self contained plug-in.

SAMPLE PLUG-IN PROGRAM

The source code for the plug-in program, PlugInSample.java is an example of a simple algorithm type of plug-in. See Figure 337.

PlugInSample.java

```
import gov.nih.mipav.plugins.*;  // needed to load PluginAlgorithm / PluginView /
                                 // PlugInFile interface
import gov.nih.mipav.view.*;
import gov.nih.mipav.model.structures.*;
import java.awt.*;

/***  This is a simple plugin to display a image in a new frame @see PlugInAlgorithm */

/** This is an Algorithm type of PlugIn and therefore must implement PlugInAlgorithm
** Implementing the PlugInAlgorithm requires this class to implement the run method
** with the correct parameters */

public class PlugInSample implements PlugInAlgorithm {
      /**
       * Defines body of run method, which was declared in the interface.
       * @param UI User Interface
       * @param parentFrame ParentFrame
       * @param image Current ModelImage--this is an image already loaded into
       * MIPAV. Can be null.
       */

      public void run (ViewUserInterface UI, Frame parentFrame, ModelImage image){
            if (parentFrame instanceof ViewJFrameImage) {
             new PlugInDialogSample(parentFrame,image);
            } else {
             MipavUtil.displayError("PlugInSample only runs on an image frame.");
          }
      }
}
```

Figure 337. PlugInSample.java

PlugInSample.java opens an image in a new image frame using its own dialog box. It requires three files:

- **PlugInSample.java**—Provides an interface to MIPAV and the plug-in program. See Figure 337 on page 563.
- **PlugInDialogSample.java**—Invokes the dialog to get user-supplied parameters. Refer to Figure 338 on page 564.
- **PlugInAlgorithmSample.java**—Implements the algorithm. See Figure 339.

PlugInDialogSample.java

```java
import gov.nih.mipav.view.*;
import gov.nih.mipav.view.dialogs.*;
import gov.nih.mipav.model.structures.*;
import gov.nih.mipav.model.algorithms.*;

import java.awt.event.*;
import java.awt.*;
import java.util.*;
import javax.swing.*;


public class PlugInDialogSample extends JDialogBase implements AlgorithmInterface {

/** Source image reference. */
    private ModelImage image; // source image
         private ViewUserInterface userInterface;

/** Sample algorithm reference. */
      private PlugInAlgorithmSample sampleAlgo = null;

      public PlugInDialogSample(Frame theParentFrame, ModelImage im) {
             super(theParentFrame, false);

              if ((im.getType() == ModelImage.BOOLEAN) || im.isColorImage()) {
                  MipavUtil.displayError("Source Image must NOT be Boolean or Color");
                  dispose();

                  return;
              }

              image = im;
              userInterface = ViewUserInterface.getReference();
              init();
      }

```

Figure 338. PlugInDialogSample.java

```
    // ************************************************************************
    // ************************** Event Processing ****************************
    // ************************************************************************

    /**
     * Closes dialog box when the OK button is pressed and calls the algorithm.
     * @param  event  Event that triggers function.
     */

   public void actionPerformed(ActionEvent event) {
        String command = event.getActionCommand();

        if (command.equals("OK")) {
        callAlgorithm();
        } else if (command.equals("Cancel")) {
            dispose();
        }
    }

    /**
     * Sets up the GUI (panels, buttons, etc) and displays it on the screen.
    */
    private void init() {

        // Build the Panel that holds the OK and CANCEL Buttons
        JPanel OKCancelPanel = new JPanel();

        JLabel questionLabel = new JLabel("Display Images?");

        // size and place the OK button
        buildOKButton();
        OKCancelPanel.add(OKButton, BorderLayout.WEST);

        // size and place the CANCEL button
        buildCancelButton();
        OKCancelPanel.add(cancelButton, BorderLayout.EAST);
        getContentPane().add(questionLabel, BorderLayout.NORTH);
        getContentPane().add(OKCancelPanel, BorderLayout.SOUTH);

        pack();
        setVisible(true);
        setResizable(false);
        System.gc();
    }
```

Figure 338. PlugInDialogSample.java (continued)

```
/*** This method is required if the AlgorithmPerformed interface is implemented. It is called by the algorithm when it has completed or failed to to complete, so that the dialog can be display the result image and/or clean up. */

/** @param  algorithm  Algorithm that caused the event. */

    public void algorithmPerformed(AlgorithmBase algorithm) {
     if (algorithm instanceof PlugInAlgorithmCT_MD) {
       if ( sampleAlgo.isCompleted() ) {
             dispose();
       }
       }
    }


/*** Once all the necessary variables are set, call the Gaussian Blur algorithm based on what type of image this is and whether or not there is a separate destination image. */

    protected void callAlgorithm() {
        sampleAlgo = new PlugInAlgorithmSample(null, image);
        sampleAlgo.addListener(this);
        setVisible(false); // Hide dialog

        if (isRunInSeparateThread()) {

//*** Start the thread as a low priority because we wish to still have user interface work fast.*/
            if (sampleAlgo.startMethod(Thread.MIN_PRIORITY) == false) {
                MipavUtil.displayError("A thread is already running on this object");
            }
        } else {
        sampleAlgo.run();
        }
    }

}
```

Figure 338. PlugInDialogSample.java (continued)

PlugInAlgorithmSample.java

```java
import gov.nih.mipav.model.algorithms.AlgorithmBase;
import gov.nih.mipav.model.structures.*;

import gov.nih.mipav.view.*;


public class PlugInAlgorithmSample extends AlgorithmBase {

      private ViewJFrameImage frame;

/*** Constructor for 3D images in which changes are placed in a predetermined destination
image.
*/

/**
* @param  destImg  Image model where result image is to stored.
* @param  srcImg   Source image model.
*/
    public PlugInAlgorithmSample(ModelImage destImg, ModelImage srcImg) {
        super(destImg, srcImg);
    }

    //~ Methods --------------------------------------------------------------------------------/

/**
* Prepares this class for destruction.
*/
    public void finalize() {
        destImage = null;
        srcImage = null;
        super.finalize();
    }


/**
* Starts the algorithm.
*/
    public void runAlgorithm() {
      frame = new ViewJFrameImage((ModelImage)srcImage.clone());
      setCompleted(true);
    }

}
```

Figure 339. PlugInAlgorithmSample.java

PLUGINCT_MD, A TYPICAL PLUGIN PROGRAM

PlugInCT_MD is a typical example of a plug-in program. It consists of three files:

- **PlugInCT_MD.java**—Provides an interface to MIPAV and the plug-in program.
- **PlugInDialogCT_MD.java**—Invokes the dialog to get user-supplied parameters.
- **PlugInAlgorithmCT_MD.java**—Implements the algorithm.

PlugInCT_MD.java

The file in Figure 340 provides an interface between MIPAV and PlugInCT_MD.

PlugInDialogCT_MD.java

The PlugInDialogCT_MD.java file invokes a dialog box to obtain user-supplied data. Refer to Figure 341 on page 570.

PlugInAlgorithmCT_MD.java

Figure 343 on page 580 shows the content of PlugInAlgorithmCT_MD.java.

PlugInDialogImageVOIDisplay.java

Figure 345 on page 586 shows a sample code for a self-contained frame plug-in.

Note: For readability purposes, keywords in all code reproduced in this chapter appear in bold, and comments appear in green type

PlugInCT_MD.java

```
import plugins.PlugInDialogCT_MT;     //associated class file
import gov.nih.mipav.plugins.*;       //needed to load PlugInAlgorithm / PlugInView /
                                      //PlugInFile interface
import gov.nih.mipav.view.*;
import gov.nih.mipav.model.structures.*;

import java.awt.*;

/**
*  This is a simple plugin for the University of Maryland to simple segment an
*  imagebased on CT Hounsfield units.
*
* @see PlugInAlgorithm
*/

//This is an Algorithm type of PlugIn, and therefore must implement PlugInAlgorithm
//Implementing the PlugInAlgorithm requires this class to implement the run method
//with the correct parameters
public class PlugInCT_MD implements PlugInAlgorithm {

   /**
     * Defines body of run method, which was declared in the interface.
     * @param UI            User Interface
     * @param parentFrame   ParentFrame
     * @param image         Current ModelImage--this is an image already loaded into
     *                      MIPAV. Can be null.
     */
     public void run (ViewUserInterface UI, Frame parentFrame, ModelImage image){

         if (parentFrame instanceof ViewJFrameImage)
           new PlugInDialogCT_MD (parentFrame,image);

         else
           MipavUtil.displayError ("PlugIn CT_MD only runs on an image frame.");
         }
     }
}
```

Figure 340. PlugInCT_MD.java

PlugInDialogCT_MD.java

```
import gov.nih.mipav.view.*;
import gov.nih.mipav.view.dialogs.*;
import gov.nih.mipav.model.structures.*;
import gov.nih.mipav.model.algorithms.*;

import java.awt.event.*;
import java.awt.*;
import java.util.*;

import javax.swing.*;


/**
 *
 *   JDialogBase class.
 *
 *   Note:
 *
 *   @version    July 12, 2002
 *   @author
 *   @see        JDialogBase
 *   @see        JDialogMedian
 *   @see        AlgorithmInterface
 *
 *   $Logfile: /mipav/src/plugins/PlugInDialogCT_MD.java $
 *   $Revision: 6 $
 *   $Date: 8/05/04 5:44p $
 *
 */
public class PlugInDialogCT_MD extends JDialogBase implements AlgorithmInterface {

    private     PlugInAlgorithmCT_MD ctSegAlgo = null;
    private     ModelImage  image;                 // source image
    private     ModelImage  resultImage = null;    // result image
     private ViewUserInterface userInterface;

    private     String       titles[];

    private     float        correctionVal;
    private     JTextField   fatLValTF;
    private     JTextField   fatHValTF;
    private     JTextField   ldmLValTF;
    private     JTextField   ldmHValTF;
    private     JTextField   hdmLValTF;
    private     JTextField   hdmHValTF;

    private     int         fatLVal;
    private     int         fatHVal;
    private     int         ldmLVal;
    private     int         ldmHVal;
    private     int         hdmLVal;
    private     int         hdmHVal;

```

Figure 341. PlugInDialogCT_MD.java

```
      /**
       *  Creates new dialog for Median filtering using a plugin.
       *  @param parent          Parent frame.
       *  @param im              Source image.
       */

         public PlugInDialogCT_MD(Frame theParentFrame, ModelImage im) {
           super(theParentFrame, true);
           if (im.getType() == ModelImage.BOOLEAN || im.isColorImage()) {
               MipavUtil.displayError("Source Image must NOT be Boolean or Color");
               dispose();
               return;
           }
           image = im;
           userInterface = ((ViewJFrameBase)(parentFrame)).getUserInterface();
           init();
         }

       /**
       * Used primarily for the script to store variables and run the algorithm.  No
       * actual dialog will appear but the set up info and result image will be stored
       *  here.
       * @param UI   The user interface, needed to create the image frame.
       * @param imSource image.
       */
       public PlugInDialogCT_MD(ViewUserInterface UI, ModelImage im) {
           super();
           userInterface = UI;
           if (im.getType() == ModelImage.BOOLEAN || im.isColorImage()) {
               MipavUtil.displayError("Source Image must NOT be Boolean or Color");
               dispose();
               return;
           }

           image = im;
       }

       /**
       * Sets up the GUI (panels, buttons, etc) and displays it on the screen.
       */
         private void init(){

     setForeground(Color.black);
           setTitle("CT_segmentation");

                JPanel inputPanel = new JPanel(new GridLayout(3, 3));
                inputPanel.setForeground(Color.black);
                 inputPanel.setBorder(buildTitledBorder("Input parameters"));

               JLabel labelFat = new JLabel("Fat thresholds: ");
               labelFat.setForeground(Color.black);
                labelFat.setFont(serif12);
                inputPanel.add(labelFat);

```

Figure 341. PlugInDialogCT_MD.java (continued)

```
                fatLValTF = new JTextField();
                fatLValTF.setText("-190");
                fatLValTF.setFont(serif12);
                inputPanel.add(fatLValTF);

                fatHValTF = new JTextField();
                fatHValTF.setText("-30");
                fatHValTF.setFont(serif12);
                inputPanel.add(fatHValTF);

                JLabel labelLDM = new JLabel("Low density muscle thresholds: ");
                labelLDM.setForeground(Color.black);
                labelLDM.setFont(serif12);
                inputPanel.add(labelLDM);

                ldmLValTF = new JTextField();
                ldmLValTF.setText("0");
                ldmLValTF.setFont(serif12);
                inputPanel.add(ldmLValTF);

                ldmHValTF = new JTextField();
                ldmHValTF.setText("30");
                ldmHValTF.setFont(serif12);
                inputPanel.add(ldmHValTF);

                JLabel labelHDM = new JLabel("High density muscle thresholds: ");
                  labelHDM.setForeground(Color.black);
                labelHDM.setFont(serif12);
                inputPanel.add(labelHDM);

                hdmLValTF = new JTextField();
                hdmLValTF.setText("31");
                hdmLValTF.setFont(serif12);
                inputPanel.add(hdmLValTF);

                hdmHValTF = new JTextField();
                hdmHValTF.setText("100");
                hdmHValTF.setFont(serif12);
                inputPanel.add(hdmHValTF);

            getContentPane().add(inputPanel, BorderLayout.CENTER);

        // Build the Panel that holds the OK and CANCEL Buttons
            JPanel OKCancelPanel = new JPanel();

            // size and place the OK button
            buildOKButton();
              OKCancelPanel.add(OKButton, BorderLayout.WEST);
            // size and place the CANCEL button
            buildCancelButton();
              OKCancelPanel.add(cancelButton, BorderLayout.EAST);
              getContentPane().add(OKCancelPanel, BorderLayout.SOUTH);
```

Figure 341. PlugInDialogCT_MD.java (continued)

```
                pack();
                setVisible(true);
                setResizable(false);
          System.gc();

          } // end init()

         /**
        *  Accessor that returns the image.
        *  @return          The result image.
        */
        public ModelImage   getResultImage(){return resultImage;}



          /**
          *      Accessor that sets the correction value
          *      @param num   Value to set iterations to (should be between 1 and 20).
          */
          public void  setCorrectionValue(float num){correctionVal = num;}

        //************************************************************************
        //************************** Event Processing ****************************
        //************************************************************************

          /**
          *  Closes dialog box when the OK button is pressed and calls the algorithm.
          *  @param event       Event that triggers function.
          */
          public void actionPerformed(ActionEvent event) {
                String command = event.getActionCommand();

                if (command.equals("OK")) {
                      if (setVariables()) {
                            callAlgorithm();
                      }
                }
                else if (command.equals("Script")) {
                      callAlgorithm();
                }
                else if (command.equals("Cancel")) {
                      dispose();
                }
       }

        //************************************************************************
        //************************** Algorithm Events ****************************
        //************************************************************************

        /**
        * This method is required if the AlgorithmPerformed interface is implemented.
        *   It is called by the algorithm when it has completed or failed to to complete,
        *   so that the dialog can be display the result image and/or clean up.
        *   @param algorithm   Algorithm that caused the event.
        */
        public void algorithmPerformed(AlgorithmBase algorithm) {
```

Figure 341. PlugInDialogCT_MD.java (continued)

```
    ViewJFrameImage imageFrame = null;
          if ( algorithm instanceof PlugInAlgorithmCT_MD) {
              image.clearMask();
              if(ctSegAlgo.isCompleted() == true && resultImage != null) {
                  //The algorithm has completed and produced a new image to be displayed.

                  updateFileInfo(image, resultImage);
                  resultImage.clearMask();
                  try {
                      //resultImage.setImageName("Median: "+image.getImageName());

                      int dimExtentsLUT[] = new int[2];
                      dimExtentsLUT[0]    = 4;
                      dimExtentsLUT[1]    = 256;
                      ModelLUT LUTa   = new ModelLUT(ModelLUT.COOLHOT, 256, dimExtentsLUT);
                     imageFrame = new ViewJFrameImage(resultImage, LUTa, new Dimension(610,200),
                                   userInterface);
                  }
                  catch (OutOfMemoryError error){
                      System.gc();
                      MipavUtil.displayError("Out of memory: unable to open new frame");
                  }
              }
              else if (resultImage == null) {
                  // These next lines set the titles in all frames where the source image
               // is displayed to image name so as to indicate that the image is now
               // unlocked! The image frames are enabled and then registered to the
               // userinterface.
                  Vector imageFrames = image.getImageFrameVector();
                  for (int i = 0; i < imageFrames.size(); i++) {
                         ((Frame)(imageFrames.elementAt(i))).setTitle(titles[i]);
                         ((Frame)(imageFrames.elementAt(i))).setEnabled(true);
                         if  ( ((Frame)(imageFrames.elementAt(i))) != parentFrame) {
                            userInterface.registerFrame((Frame)(imageFrames.elementAt(i)));
                         }
                  }
                  if (parentFrame != null) userInterface.registerFrame(parentFrame);
                  image.notifyImageDisplayListeners(null, true);
              }
              else if (resultImage != null){
                      //algorithm failed but result image still has garbage
                      resultImage.disposeLocal();  // clean up memory
                      resultImage = null;
                      System.gc();
              }
          }
          if (ctSegAlgo.isCompleted() == true) {
                 if (userInterface.isScriptRecording()) {
                    userInterface.getScriptDialog().append("Flow " +
                    userInterface.getScriptDialog().getVar(image.getImageName()) + " "
                    + correctionVal + "\n");
          }
          }
          dispose();

```

Figure 341. PlugInDialogCT_MD.java (continued)

```
    } // end AlgorithmPerformed()


      /**
      * Use the GUI results to set up the variables needed to run the algorithm.
      * @return      <code>true</code> if parameters set successfully, <code>false
     *  </code> otherwise.
      */
      private boolean setVariables() {
        String tmpStr;


         // verify iteration is within bounds
         tmpStr = fatLValTF.getText();
          if ( testParameter(tmpStr, -4000, 4000) ){
              fatLVal = Integer.valueOf(tmpStr).intValue();
          }
          else{
              fatLValTF.requestFocus();
              fatLValTF.selectAll();
              return false;
          }

          tmpStr = fatHValTF.getText();
          if ( testParameter(tmpStr, -4000, 4000) ){
              fatHVal = Integer.valueOf(tmpStr).intValue();
          }
          else{
              fatHValTF.requestFocus();
              fatHValTF.selectAll();
              return false;
          }

          tmpStr = ldmLValTF.getText();
          if ( testParameter(tmpStr, -4000, 4000) ){
              ldmLVal = Integer.valueOf(tmpStr).intValue();
          }
          else{
              ldmLValTF.requestFocus();
              ldmLValTF.selectAll();
              return false;
          }

          tmpStr = ldmHValTF.getText();
          if ( testParameter(tmpStr, -4000, 4000) ){
              ldmHVal = Integer.valueOf(tmpStr).intValue();
          }
              else{
              ldmHValTF.requestFocus();
              ldmHValTF.selectAll();
              return false;
          }

```

Figure 341. PlugInDialogCT_MD.java (continued)

```
tmpStr = hdmLValTF.getText();
        if ( testParameter(tmpStr, -4000, 4000) ){
            hdmLVal = Integer.valueOf(tmpStr).intValue();
        }
        else{
            hdmLValTF.requestFocus();
            hdmLValTF.selectAll();
            return false;
        }

        tmpStr = hdmHValTF.getText();
        if ( testParameter(tmpStr, -4000, 4000) ){
            hdmHVal = Integer.valueOf(tmpStr).intValue();
        }
        else{
            hdmHValTF.requestFocus();
            hdmHValTF.selectAll();
            return false;
        }

      return true;
    }   // end setVariables()

      /**
      *      Once all the necessary variables are set, call the Gaussian Blur
      *      algorithm based on what type of image this is and whether or not there
      *      is a separate destination image.
      */
      private void callAlgorithm() {
             String name = makeImageName(image.getImageName(), "_CTseg");

        // stuff to do when working on 2-D images.
        if (image.getNDims() == 2 ) {                  // source image is 2D
            int destExtents[] = new int[2];
            destExtents[0] = image.getExtents()[0];    // X dim
            destExtents[1] = image.getExtents()[1];    // Y dim

            try{
                // Make result image of Ubyte type
                resultImage     = new ModelImage(ModelStorageBase.UBYTE, destExtents, name,
                                  userInterface);

                // Make algorithm
                boolean entireFlag = true;

 //ctSegAlgo = new PlugInAlgorithmFlowWrapFix(resultImage, image, iters,
// kernelSize, kernelShape, stdDev, regionFlag);
                ctSegAlgo = new PlugInAlgorithmCT_MD(resultImage, image);

                System.out.println("Dialog fatL = " + fatLVal + " fatH = " + fatHVal);
                ctSegAlgo.fatL = fatLVal;
                ctSegAlgo.fatH = fatHVal;
                ctSegAlgo.ldmL = ldmLVal;
                ctSegAlgo.ldmH = ldmHVal;
```

Figure 341. PlugInDialogCT_MD.java (continued)

```
                    ctSegAlgo.hdmL = hdmLVal;
                    ctSegAlgo.hdmH = hdmHVal;



                    // This is very important. Adding this object as a listener allows the
                 // algorithm to notify this object when it has completed or failed. See
                  // algorithm performed event.
                    // This is made possible by implementing AlgorithmedPerformed interface
                    ctSegAlgo.addListener(this);
                    setVisible(false);  // Hide dialog

                    if (runInSeparateThread) {
                        // Start the thread as a low priority because we wish to still have
                    // user interface work fast.
                        if (ctSegAlgo.startMethod(Thread.MIN_PRIORITY) == false){
                            MipavUtil.displayError("A thread is already running on this object");
                        }
                    }
                    else {
                        ctSegAlgo.run();
                    }
                }
                catch (OutOfMemoryError x){
                    MipavUtil.displayError("Dialog median: unable to allocate enough memory");
                    if (resultImage != null){
                        resultImage.disposeLocal();  // Clean up memory of result image
                        resultImage = null;
                    }
                    return;
                }
            }
            else if (image.getNDims() == 3 ) {
                int destExtents[] = new int[3];
                destExtents[0] = image.getExtents()[0];
                destExtents[1] = image.getExtents()[1];
                destExtents[2] = image.getExtents()[2];

                try{
                    // Make result image of float type
                    resultImage     = new ModelImage(ModelStorageBase.UBYTE, destExtents, name,
                                      userInterface);
                    boolean entireFlag = true;

                    ctSegAlgo = new PlugInAlgorithmCT_MD(resultImage, image);
                    ctSegAlgo.fatL = fatLVal;
                    ctSegAlgo.fatH = fatHVal;
                    ctSegAlgo.ldmL = ldmLVal;
                    ctSegAlgo.ldmH = ldmHVal;
                    ctSegAlgo.hdmL = hdmLVal;
                    ctSegAlgo.hdmH = hdmHVal;

```

Figure 341. PlugInDialogCT_MD.java (continued)

```
                       // This is very important. Adding this object as a listener allows the
                       // algorithm to notify this object when it has completed or failed.
                    // See algorithm performed event. This is made possible by implementing
                       //  AlgorithmedPerformed interface
                      ctSegAlgo.addListener(this);
                      setVisible(false);        // Hide dialog

                      if (runInSeparateThread) {
                          // Start the thread as a low priority because we wish to still have
                      // user interface work fast.
                          if (ctSegAlgo.startMethod(Thread.MIN_PRIORITY) == false){
                              MipavUtil.displayError("A thread is already running on this object");
                          }
                      }
                      else {
                          ctSegAlgo.run();
                      }
                  }
                  catch (OutOfMemoryError x){
                      MipavUtil.displayError("Dialog median: unable to allocate enough memory");
                      if (resultImage != null){
                          resultImage.disposeLocal();      // Clean up image memory
                          resultImage = null;
                      }
                      return;
                  }
              }
          } // end callAlgorithm()

}
```

Figure 341. PlugInDialogCT_MD.java (continued)

PlugInAlgorithmCT_MD.java

```
import gov.nih.mipav.model.algorithms.*;
import gov.nih.mipav.model.structures.*;
import gov.nih.mipav.view.*;

import java.io.*;
import java.util.*;


/**
*
*   This shows how to extend the AlgorithmBase class.
*
*     Supports the segmentation
*     CT scans:
```

Figure 342. PlugInAlgorithmCT_MD.java

```
 *      Fat:                   -190 to -30
 *      Low density muscle:       0 to  30
 *      High density muscle:     31 to 100
 *      If you have any questions, please drop me a line.
 * =====
 * Matthew J. Delmonico, MS, MPH
 * Graduate Research Assistant, Exercise Physiology
 * 2132 HHP Building
 * University of Maryland
 * College Park, MD  20742
 * (301) 405-2569
 * (301) 793-0567 (cell)
 *
 *    @version    July 12, 2002
 *    @author
 *    @see        AlgorithmBase
 *
 *    $Logfile: /mipav/src/plugins/PlugInAlgorithmCT_MD.java $
 *    $Revision: 10 $
 *    $Date: 10/13/04 1:09p $
 *
 */
public class PlugInAlgorithmCT_MD extends AlgorithmBase {


    private boolean     entireImage = true;

    public int          fatL    = -190;
    public int          fatH    = -30;

    public int          ldmL    = 0;
    public int          ldmH    = 30;

    public int          hdmL    = 31;
    public int          hdmH    = 100;


    /**
     * Constructor for 3D images in which changes are placed in a predetermined
    * destination image.
     *   @param destImg      Image model where result image is to stored.
     *   @param srcImg       Source image model.
     */
      public PlugInAlgorithmCT_MD(ModelImage destImg, ModelImage srcImg) {
       super(destImg, srcImg);
      }

/**
     * Prepares this class for destruction.
*/
      public void finalize(){
          destImage   = null;
          srcImage    = null;
          super.finalize();
      }

```

Figure 342. PlugInAlgorithmCT_MD.java (continued)

```
       /**
       *   Starts the algorithm.
       */
         public void run() {

           if (srcImage  == null) {
               displayError("Source Image is null");
               notifyListeners(this);
               return;
           }
           if (destImage  == null) {
               displayError("Source Image is null");
               notifyListeners(this);
               return;
           }


           // start the timer to compute the elapsed time
           setStartTime();

           if (destImage != null){     // if there exists a destination image
               if (srcImage.getNDims() == 2){
                    calcStoreInDest2D();
                 }
                 else if (srcImage.getNDims() > 2) {
                     calcStoreInDest3D();
                 }
           }

           // compute the elapsed time
           computeElapsedTime();
           notifyListeners(this);
       }

       /**
       * This function produces a new image that has been median filtered and places
      *  filtered image in the destination image.
       */
       private void calcStoreInDest2D(){


           int length;               // total number of data-elements (pixels) in image
           float buffer[];           // data-buffer (for pixel data) which is the "heart"
                              // of the image
```

Figure 343. PlugInAlgorithmCT_MD.java

```
        try {
            // image length is length in 2 dims
            length = srcImage.getExtents()[0]  * srcImage.getExtents()[1];
            buffer       = new float[length];
            srcImage.exportData(0,length, buffer); // locks and releases lock
        }
        catch (IOException error) {
            buffer = null;
            errorCleanUp("Algorithm CT_MD reports: source image locked", true);
            return;
        }
        catch (OutOfMemoryError e){
            buffer = null;
            errorCleanUp("Algorithm CT_MD reports: out of memory", true);
            return;
        }

        int mod = length/100; // mod is 1 percent of length
        initProgressBar();

        // Fat:  -190 to -30
        // Low density muscle:  0 to 30
        // High density muscle:  31 to 100
        BitSet mask = null;
        if (srcImage.getVOIs().size() > 0 ) {
            mask = srcImage.generateVOIMask();
            entireImage = false;
        }

        int fat      = 0;
        int ldMuscle = 0;
        int hdMuscle = 0;
        for (int i = 0; i < length && !threadStopped; i++){
            if (isProgressBarVisible() && (i)%mod==0)
                progressBar.setValue(Math.round((float)(i)/(length-1) * 100));

            if (entireImage == true || mask.get(i) ) {
                if( buffer[i] >= fatL && buffer[i] <= fatH  ) {
                    destImage.set(i, 20);
                    fat++;
                }
                else if( buffer[i] >= ldmL && buffer[i] <= ldmH  ) {
                    destImage.set(i, 40);
                    ldMuscle++;
                }
                else if( buffer[i] >= hdmL && buffer[i] <= hdmH  ) {
                    destImage.set(i, 60);
                    hdMuscle++;
                }
                else {
                    destImage.set(i, 0);
                    //buffer[i] = (float)srcImage.getMin();

            }
            }
        }


```

Figure 343. PlugInAlgorithmCT_MD.java

```
        //destImage.releaseLock();

        if (threadStopped) {
        finalize();
        return;
        }

        float area = srcImage.getFileInfo()[0].getResolutions()[0] *
                     srcImage.getFileInfo()[0].getResolutions()[1];

        destImage.getUserInterface().getMessageFrame().append("Number of Fat pixels = " +
           fat , ViewJFrameMessage.DATA );
        destImage.getUserInterface().getMessageFrame().append("  Area = " + (fat*area) +
           " mm^2\n", ViewJFrameMessage.DATA );

        destImage.getUserInterface().getMessageFrame().append("Number of LDM pixels = " +
           ldMuscle , ViewJFrameMessage.DATA );
        destImage.getUserInterface().getMessageFrame().append("  Area = " + (ldMuscle*area) +
           " mm^2\n", ViewJFrameMessage.DATA );

        destImage.getUserInterface().getMessageFrame().append("Number of HDM pixels = " +
           hdMuscle , ViewJFrameMessage.DATA );
        destImage.getUserInterface().getMessageFrame().append("  Area = " + (hdMuscle*area) +
           " mm^2\n", ViewJFrameMessage.DATA );

        destImage.calcMinMax();
        setCompleted(true);
    }

    /**
    * This function produces a new volume image that has been median filtered.
    * Image can be filtered by filtering each slice individually, or by filtering
   *  using a kernel-volume.
    */
    private void calcStoreInDest3D(){

        int totLength, imgLength;
        float buffer[];

        float vol = srcImage.getFileInfo()[0].getResolutions()[0] *
                    srcImage.getFileInfo()[0].getResolutions()[1] *
                    srcImage.getFileInfo()[0].getResolutions()[2];

        try {
            // image totLength is totLength in 3 dims
            imgLength = srcImage.getSliceSize();
            totLength = srcImage.getSliceSize() * srcImage.getExtents()[2];
            buffer = new float[totLength];
            srcImage.exportData(0,totLength, buffer); // locks and releases lock
            buildProgressBar(srcImage.getImageName(), "Processing image ...", 0, 100);
        }

catch (IOException error) {
            buffer = null;
            errorCleanUp("Algorithm CT_MD: source image locked", true);
            return;
        }
```

Figure 343. PlugInAlgorithmCT_MD.java

```
        catch (OutOfMemoryError e){
            buffer = null;
            errorCleanUp("Algorithm CT_MD: Out of memory creating process buffer", true);
            return;
        }

        int totFat      = 0;
        int totLdMuscle = 0;
        int totHdMuscle = 0;
        initProgressBar();

        for (int i = 0; i < srcImage.getExtents()[2] && !threadStopped; i++){
            int fat      = 0;
            int ldMuscle = 0;
            int hdMuscle = 0;

            if ( isProgressBarVisible() )
                    progressBar.setValue(Math.round((float)(i)/(srcImage.getExtents()[2]-1) *
                    100));

            for (int j = 0; j < imgLength && !threadStopped; j++){
                //System.out.println(" j = " + j);
                int index = i*imgLength+j;
                if( buffer[index] >= fatL && buffer[index] <= fatH  ) {
                    destImage.set(index, 60);
                    totFat++;
                    fat++;
                }
                else if( buffer[index] >= ldmL && buffer[index] <= ldmH  ) {
                    destImage.set(index, 120);
                    totLdMuscle++;
                    ldMuscle++;
                }
                else if( buffer[index] >= hdmL && buffer[index] <= hdmH  ) {
                    destImage.set(index, 200);
                    totHdMuscle++;
                    hdMuscle++;
                }
                else {
                    destImage.set(index, 0);
                    //buffer[i] = -1024;
                }
            }
           destImage.getUserInterface().getMessageFrame().append("\n\n ***************** Slice
                " + i + " totals ***************\n",
            ViewJFrameMessage.DATA);
            destImage.getUserInterface().getMessageFrame().append("Number of fat pixels = " +
                fat , ViewJFrameMessage.DATA );
            destImage.getUserInterface().getMessageFrame().append("  Volume = " + (fat*vol) +
                " mm^3\n", ViewJFrameMessage.DATA );

             destImage.getUserInterface().getMessageFrame().append("Number of LDM pixels = " +
                ldMuscle , ViewJFrameMessage.DATA );
            destImage.getUserInterface().getMessageFrame().append("  Volume = " +
                (ldMuscle*vol) + " mm^3\n", ViewJFrameMessage.DATA );

```

Figure 343. PlugInAlgorithmCT_MD.java

```
                        destImage.getUserInterface().getMessageFrame().append("Number of HDM pixels
                              = " + hdMuscle , ViewJFrameMessage.DATA );
                    destImage.getUserInterface().getMessageFrame().append("  Volume = " +
                            (hdMuscle*vol) + " mm^3\n", ViewJFrameMessage.DATA );
                }

                destImage.releaseLock();

                if (threadStopped) {
                finalize();
                return;
                }

                destImage.getUserInterface().getMessageFrame().append("\n ************************
                    Totals ********************\n",
                ViewJFrameMessage.DATA);
                destImage.getUserInterface().getMessageFrame().append("Number of totFat pixels = " +
                    totFat , ViewJFrameMessage.DATA );
                destImage.getUserInterface().getMessageFrame().append("  Volume = " + (totFat*vol) +
                    " mm^3\n", ViewJFrameMessage.DATA );

                destImage.getUserInterface().getMessageFrame().append("Number of LDM pixels = " +
                     totLdMuscle , ViewJFrameMessage.DATA );
               destImage.getUserInterface().getMessageFrame().append("  Volume = " + (totLdMuscle*vol)
                     + " mm^3\n", ViewJFrameMessage.DATA );

                destImage.getUserInterface().getMessageFrame().append("Number of HDM pixels = " +
                     totHdMuscle , ViewJFrameMessage.DATA );
               destImage.getUserInterface().getMessageFrame().append("  Volume = " + (totHdMuscle*vol)
                     + " mm^3\n", ViewJFrameMessage.DATA );

                destImage.calcMinMax();
                progressBar.dispose();
                setCompleted(true);
            }
        }
```

Figure 343. PlugInAlgorithmCT_MD.java

PlugInCT_MD.java

```java
import plugins.PlugInDialogCT_MT;     //associated class file
import gov.nih.mipav.plugins.*;       //needed to load PlugInAlgorithm / PlugInView /
                                      //PlugInFile interface
import gov.nih.mipav.view.*;
import gov.nih.mipav.model.structures.*;

import java.awt.*;

/**
*  This is a simple plugin for the University of Maryland to simple segment an
*  imagebased on CT Hounsfield units.
*
* @see PlugInAlgorithm
*/

//This is an Algorithm type of PlugIn, and therefore must implement PlugInAlgorithm
//Implementing the PlugInAlgorithm requires this class to implement the run method
//with the correct parameters
public class PlugInCT_MD implements PlugInAlgorithm {

   /**
     * Defines body of run method, which was declared in the interface.
     * @param UI             User Interface
     * @param parentFrame   ParentFrame
     * @param image         Current ModelImage--this is an image already loaded into
     *                      MIPAV. Can be null.
     */
     public void run (ViewUserInterface UI, Frame parentFrame, ModelImage image){

         if (parentFrame instanceof ViewJFrameImage)
           new PlugInDialogCT_MD (parentFrame,image);

         else
           MipavUtil.displayError (“PlugIn CT_MD only runs on an image frame.”);
         }
     }
}
```

Figure 344. PlugInCT_MD.java

PlugInDialogImageVOIDisplay.java

```
import java.awt.*;
import java.awt.event.*;
import java.util.Vector;

import javax.swing.*;
import gov.nih.mipav.model.file.FileInfoBase;
import gov.nih.mipav.model.structures.ModelImage;
import gov.nih.mipav.model.structures.ModelLUT;
import gov.nih.mipav.model.structures.ModelRGB;
import gov.nih.mipav.model.structures.VOI;
import gov.nih.mipav.view.*;
import gov.nih.mipav.view.dialogs.*;


/**
 * Plugin example class for creating a simple, self-contained frame that extends ViewJFrameImage
 * Contains a subset of the VOI functions, as well as the message frame contained within the frame itself
 * @author linkb
 *
 */
public class PlugInDialogImageVOIDisplay extends ViewJFrameImage implements MouseListener, AdjustmentListener {}
 //~ Constructors -------------------------------------------------------------------------------------------------

   /**
    * Default constructor
    */
    public PlugInDialogImageVOIDisplay(ModelImage image) {
      super(image, null, null, false, false);
        init();
    }


    /**
     * ViewOpenFrameInterface function for opening a model image (result) into a new frame
     */
    public PlugInDialogImageVOIDisplay openFrame(ModelImage image) {
      return new PlugInDialogImageVOIDisplay(image);
//~ Methods -------------------------------------------------------------------------------------------------

    // ************************************************************************
    // ************************** Event Processing ****************************
    // ************************************************************************
 public void adjustmentValueChanged(AdjustmentEvent e) {
      updateImages(true);
    }

```

Figure 345. PlugInDialogImageVOIDisplay.java

```
    /**
         * Closes dialog box when the OK button is pressed and calls the algorithm.
         *
         * @param  event  Event that triggers function.
         */
        public void actionPerformed(ActionEvent event) {
            String command = event.getActionCommand();
            System.err.println("command: " + command);

            //run through toggle buttons to see if a menu selected one (updates the button status)
            getControls().getTools().setToggleButtonSelected(command);

            if (command.equals("Gaussian blur")) {
                new JDialogGaussianBlur(this, getActiveImage());
            } else if (command.equals("Gradient magnitude")) {
                new JDialogGradientMagnitude(this, getActiveImage());
            } else if (command.equals("Open")) {
            //ViewUserInterface.getReference().openImageFrame();
            } else if (command.equals(CustomUIBuilder.PARAM_VOI_DEFAULT_POINTER)) {
                componentImage.setCursorMode(ViewJComponentEditImage.DEFAULT);
            } else if (command.equals(CustomUIBuilder.PARAM_VOI_POINT.getActionCommand())) {


                if 
    (!componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(), 
    VOI.POINT, getControls())) {
                    componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);
                }

                componentImage.setCursorMode(ViewJComponentEditImage.POINT_VOI);
            } else if (command.equals(CustomUIBuilder.PARAM_VOI_LINE.getActionCommand())) {

                if 
    (!componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(), 
    VOI.LINE, getControls())) {
                    componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);
                }

                componentImage.setCursorMode(ViewJComponentEditImage.LINE);
            } else if (command.equals("SplitVOI")) {
                componentImage.setCursorMode(ViewJComponentEditImage.SPLIT_VOI);
            } else if (command.equals(CustomUIBuilder.PARAM_VOI_POLY_SLICE.getActionCommand())) {

             if 
    (!componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(), 
    VOI.POLYLINE_SLICE, getControls())) {
                     componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);
                 }

                componentImage.setCursorMode(ViewJComponentEditImage.POLYLINE_SLICE_VOI);
            } else if (command.equals("protractor")) {

             if 
    (!componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(), 
    VOI.PROTRACTOR, getControls())) {
                     componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);
                 }
```

Figure 345. PlugInDialogImageVOIDisplay.java (continued)

```
componentImage.setCursorMode(ViewJComponentEditImage.PROTRACTOR);
        } else if (command.equals("Polyline")) {

        if
(!componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(),
VOI.POLYLINE, getControls())) {
                componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);
            }

            componentImage.setCursorMode(ViewJComponentEditImage.POLYLINE);
        } else if (command.equals(CustomUIBuilder.PARAM_VOI_TEXT.getActionCommand())) {

        componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);


            componentImage.setCursorMode(ViewJComponentEditImage.ANNOTATION);
        } else if (command.equals("RectVOI")) {

        if
(!componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(),
VOI.CONTOUR, getControls())) {
                componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);
            }

            componentImage.setCursorMode(ViewJComponentEditImage.RECTANGLE);
        } else if (command.equals("EllipseVOI")) {
        if
(!componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(),
VOI.CONTOUR, getControls())) {
                componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);
            }

            componentImage.setCursorMode(ViewJComponentEditImage.ELLIPSE);
        } else if (command.equals("LevelSetVOI")) {
        componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(),
VOI.CONTOUR, getControls());
            componentImage.setCursorMode(ViewJComponentEditImage.LEVELSET);
        } else if (command.equals("Rect3DVOI")) {
        componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(),
VOI.CONTOUR, getControls());
            componentImage.setCursorMode(ViewJComponentEditImage.RECTANGLE3D);
        } else if (command.equals("LiveWireVOI")) {
        componentImage.getVOIHandler().checkForVOICompatibility(getActiveImage().getVOIs(),
VOI.CONTOUR, getControls());

            if (componentImage.getVOIHandler().isLivewireNull()) {
                JDialogLivewire dialog = new JDialogLivewire(this);

                if (!dialog.isCancelled()) {
                    componentImage.getVOIHandler().setModeLivewire(dialog.getSelection());
                    componentImage.setCursorMode(ViewJComponentEditImage.LIVEWIRE);
                }
            } else {
                componentImage.setCursorMode(ViewJComponentEditImage.LIVEWIRE);
            }
        } else if (command.equals("NewVOI")) {
            componentImage.setCursorMode(ViewJComponentEditImage.NEW_VOI);
```

Figure 345. PlugInDialogImageVOIDisplay.java (continued)

```
int id = (getActiveImage().getVOIs().size() > 0)
                    ? (((VOI) (getActiveImage().getVOIs().lastElement())).getID() + 1) : -1;

            getControls().setVOIColor(id);
        } else if (command.equals("cutVOI")) {

            if (componentImage.getVOIHandler().copyVOItoClipBrd()) {
                componentImage.getVOIHandler().deleteSelectedVOI(true);
            }
        } else if (command.equals("copyVOI")) {
            componentImage.getVOIHandler().copyVOItoClipBrd();
        } else if (command.equals("pasteVOI")) {
            componentImage.getVOIHandler().pasteVOI();
        } else if (command.equals("selectAllVOIs")) {
            componentImage.getVOIHandler().selectAllVOIs(true);
        } else if (event.getActionCommand().equals("voiSelectNone")) {
            componentImage.getVOIHandler().selectAllVOIs(false);
        } else if (command.equals("deleteVOI")) {
            componentImage.getVOIHandler().deleteSelectedVOI(true);
        } else if (command.equals("BringToFront")) {
            componentImage.getVOIHandler().changeVOIOrder(false, VOIHandler.FRONT);
        } else if (command.equals("SendToBack")) {
            componentImage.getVOIHandler().changeVOIOrder(false, VOIHandler.BACK);
        } else if (command.equals("BringContourToFront")) {
            componentImage.getVOIHandler().changeVOIOrder(true, VOIHandler.FRONT);
        } else if (command.equals("SendContourToBack")) {
            componentImage.getVOIHandler().changeVOIOrder(false, VOIHandler.BACK);
        } else if (command.equals("PropVOIUp")) {

            // It appears JButtons don't pass key modifiers
            // if((event.getModifiers() & ActionEvent.SHIFT_MASK) != 0) {}
            if (componentImage.getVOIHandler().propVOI(1, false) == true) {
                incSlice();
            }
        } else if (command.equals("PropVOIDown")) {

            if (componentImage.getVOIHandler().propVOI(-1, false) == true) {
                decSlice();
            }
        } else if (command.equals("PropVOIActiveUp")) {

            // It appears JButtons don't pass key modifiers
            // if((event.getModifiers() & ActionEvent.SHIFT_MASK) != 0) {}
            if (componentImage.getVOIHandler().propVOI(1, true) == true) {
                incSlice();
            }
} else if (command.equals("PropVOIActiveDown")) {

            if (componentImage.getVOIHandler().propVOI(-1, true) == true) {
                decSlice();
            }
} else if (command.equals("PropVOIAll")) {
            componentImage.getVOIHandler().propVOIAll();
        } else if (command.equals("BringForward")) {
            componentImage.getVOIHandler().changeVOIOrder(false, VOIHandler.FORWARD);
        } else if (command.equals("SendBackward")) {
```

Figure 345. PlugInDialogImageVOIDisplay.java (continued)

```
componentImage.getVOIHandler().changeVOIOrder(false, VOIHandler.BACKWARD);
        } else if (command.equals("SendContourForward")) {
            componentImage.getVOIHandler().changeVOIOrder(true, VOIHandler.FORWARD);
        } else if (command.equals("SendContourBackward")) {
            componentImage.getVOIHandler().changeVOIOrder(true, VOIHandler.BACKWARD);
        } else if (command.equals("VOIProperties")) {

            componentImage.getVOIHandler().showVOIProperties(false);

        } else if (command.equals("VOIPropertiesColor")) {

            if (getActiveImage().getVOIs().size() > 0) {

                ViewVOIVector VOIs = getActiveImage().getVOIs();

                int i;
                int nVOI = VOIs.size();

                for (i = 0; i < nVOI; i++) {

                    if ((VOIs.VOIAt(i).isActive() == true) &&
                            ((VOIs.VOIAt(i).getCurveType() == VOI.CONTOUR) ||
                                 (VOIs.VOIAt(i).getCurveType() == VOI.POLYLINE) ||
                                 (VOIs.VOIAt(i).getCurveType() == VOI.POINT) ||
                                 (VOIs.VOIAt(i).getCurveType() == VOI.LINE) ||
                                 (VOIs.VOIAt(i).getCurveType() == VOI.PROTRACTOR))) {
                        break;
                    } else if ((VOIs.VOIAt(i).isActive() == true) &&
(VOIs.VOIAt(i).getCurveType() == VOI.ANNOTATION)) {
                        MipavUtil.displayInfo("Double-click annotation to change properties");
                        i = -1;

                        break;
                    }
                }

                if (i == nVOI) {
                    MipavUtil.displayError("Please select VOI");
                } else if (i == -1) { // there was an annotation selected, do nothing
                } else {
                    componentImage.getVOIHandler().showVOIProperties(true);
                }
            } else {
                MipavUtil.displayWarning("Image has no VOIs!");
            }

        }
    }

    /**
     * Can handle actions for the resizing of the frame
     */
    public synchronized void componentResized(ComponentEvent event) {

    }

```

Figure 345. PlugInDialogImageVOIDisplay.java (continued)

```
    /**
     * Override MouseListener functions to prevent MouseEvent catching in ViewJFrameImage
     */
     public void mousePressed(MouseEvent e) {}
     public void mouseReleased(MouseEvent e) {}
     public void mouseEntered(MouseEvent e) {}
     public void mouseExited(MouseEvent e) {}
     public void mouseClicked(MouseEvent e) {}


     /**
      * Initialize the frame using a lut (can be null)
      * @param LUTa the ModelLUT
      * @throws OutOfMemoryError
      */
     private void init() throws OutOfMemoryError {

         try {
             setIconImage(MipavUtil.getIconImage("davinci_32x32.gif"));
         } catch (Exception error) {
             Preferences.debug("Exception ocurred while getting <" + error.getMessage() +
                                  ">.  Check that this file is available.\n");
         }

         setResizable(true);

         // initialize logMagDisplay
         this.LUTa = initLUT(imageA);

         initResolutions();
         initZoom();

         int[] extents = createBuffers();

         initComponentImage(extents);
         initExtentsVariables(imageA);

         // create and build the menus and controls
         controls = new ViewControlsImage(this); // Build controls used in this frame
         menuBuilder = new ViewMenuBuilder(this);

         // build the menuBar based on the number of dimensions for imageA
         menuBarMaker = new ViewMenuBar(menuBuilder);

         //create a custom menu bar using Vectors of UIParams
         JMenuBar menuBar = new JMenuBar();

         //add pre-defined UIParams to the vector (will be added to both menu and toolbar)
         Vector<CustomUIBuilder.UIParams> voiParams = new Vector<CustomUIBuilder.UIParams>();
         voiParams.addElement(CustomUIBuilder.PARAM_VOI_DEFAULT_POINTER);
         voiParams.addElement(CustomUIBuilder.PARAM_VOI_POINT);
         voiParams.addElement(CustomUIBuilder.PARAM_VOI_ELLIPSE);
         voiParams.addElement(CustomUIBuilder.PARAM_VOI_RECTANGLE);
         Vector<CustomUIBuilder.UIParams> algoParams = new Vector<CustomUIBuilder.UIParams>();
         algoParams.add(new CustomUIBuilder.UIParams("Gaussian blur", null, null));
         algoParams.add(new CustomUIBuilder.UIParams("Gradient magnitude", null, null));
```

Figure 345. PlugInDialogImageVOIDisplay.java (continued)

```
menuBar.add(menuBarMaker.makeCustomMenu("VOI example", voiParams));
         menuBar.add(menuBarMaker.makeCustomMenu("Simple algorithm menu", algoParams));

         //create a simple toolbar (rather than the default ViewJFrameImage specific toolbar)
         //buttons will be added to the toolbar with the function call .addCustomToolBar()
         controls.buildSimpleToolBar();

         controls.addCustomToolBar(voiParams);

         setTitle();

         JPanel centerPanel = new JPanel();
         centerPanel.add(componentImage, BorderLayout.CENTER);

         // The component image will be displayed in a scrollpane.
       scrollPane = new JScrollPane(centerPanel, JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                                        JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);

       JSplitPane splitPane = new JSplitPane(JSplitPane.VERTICAL_SPLIT, scrollPane,
               ViewUserInterface.getReference().getMessageFrame().getTabbedPane());
       splitPane.setDividerLocation(350);

       getContentPane().add(splitPane);
       scrollPane.setBackground(Color.black);

       setBackground(Color.black);



         // MUST register frame to image models
         imageA.addImageDisplayListener(this);

         if (imageB != null) {
             imageB.addImageDisplayListener(this);
         }

         windowLevel = new JDialogWinLevel[2];

         this.setLocation(100, 50);

         setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
 pack();

         scrollPane.setPreferredSize(new Dimension(800,800));
         scrollPane.getVerticalScrollBar().addAdjustmentListener(this);
         scrollPane.getHorizontalScrollBar().addAdjustmentListener(this);
         scrollPane.addComponentListener(this);

         setSize(1000,750);
         // User interface will have list of frames
         userInterface.registerFrame(this);
this.updateImages(true);
         addComponentListener(this);

         this.setJMenuBar(menuBar);
         getContentPane().add(controls, BorderLayout.NORTH);
```

Figure 345. PlugInDialogImageVOIDisplay.java (continued)

```
this.addWindowListener(new WindowAdapter() {
             public void windowClosing(WindowEvent we) {
                 System.exit(0);
             }
         });

         setVisible(true);
     } // end init()


      /**
       * Sets the title of the frame
       */
      public void setTitle() {
            this.setTitle("Simple Image Frame: " + imageA.getImageName());
      }



}
```

Figure 345. PlugInDialogImageVOIDisplay.java (continued)

14 Technical support

In this chapter . . .

Troubleshooting

If a problem occurs while you are using MIPAV, please contact us so that it can be addressed in a future release of the software. Please perform the following steps so that we have the correct information needed to resolve your problem:

1. Place MIPAV in debug mode, refer to page 595.
2. Gather software information, refer to page 597.
3. Contact technical support, refer to page 598.

Placing MIPAV in debug mode

When you place MIPAV in debug mode, MIPAV creates a log of the software error messages that occur when MIPAV is being used. You can view these messages to pinpoint problems or you can save a copy to use as a reference when you contact technical support.

To turn debug mode on

1. Select Help > Program Options. The MIPAV Options dialog box opens.
2. Open the Other tab. See Figure 346.
3. Check the **Show Data/Debugging Output Window** check box. Then select the debugging level. The five debugging levels – from Minor to Scripting –correlate to the amount of debugging information provided on the Debug page and the amount of computer memory used. The **Minor** level provides the least amount of debugging information, and the **Scripting** level provides the highest level of the debugging information. See Figure 346.
4. Click Apply, then Click Close. From this point on, MIPAV places debugging data on the Debug page in the Output window. If software errors have occurred, messages appear in the Debug window. See Figure 346
5. Save the debug information:
   - To save the debug information to a file, select File > Save Messages. The Save dialog box opens. Type a name for the file in File Name, and then click Save.
   - To copy the information to the clipboard, select Edit > Copy. You can then paste the information in any text editing, word processing, desktop publishing, or e-mail program of your choice.

Note: If you are running MIPAV on a Windows platform, before you place MIPAV in debug mode, it is helpful to exit from the program and then restart it again by holding the Crtl key while you select Start > MIPAV. A DOS window opens on your desktop behind the MIPAV application windows.

Note: If you've contacted MIPAV developers about a problem with the program, they may ask you to save the debugging data in a text file. To do this, you would mark Redirect errors to file and type the name of the file in the box beside the check box. You can then send the file to them via e-mail.

Removing MIPAV from debug mode

1. Select Help > Program Options. The MIPAV Options dialog box opens.
2. Uncheck the **Show Data/Debugging Output Window** check box.
3. Click Apply, then click Close.

When the debug mode is off, no error messages display on the Debug page in the Output window.

MIPAV Options dialog box – the Other tab

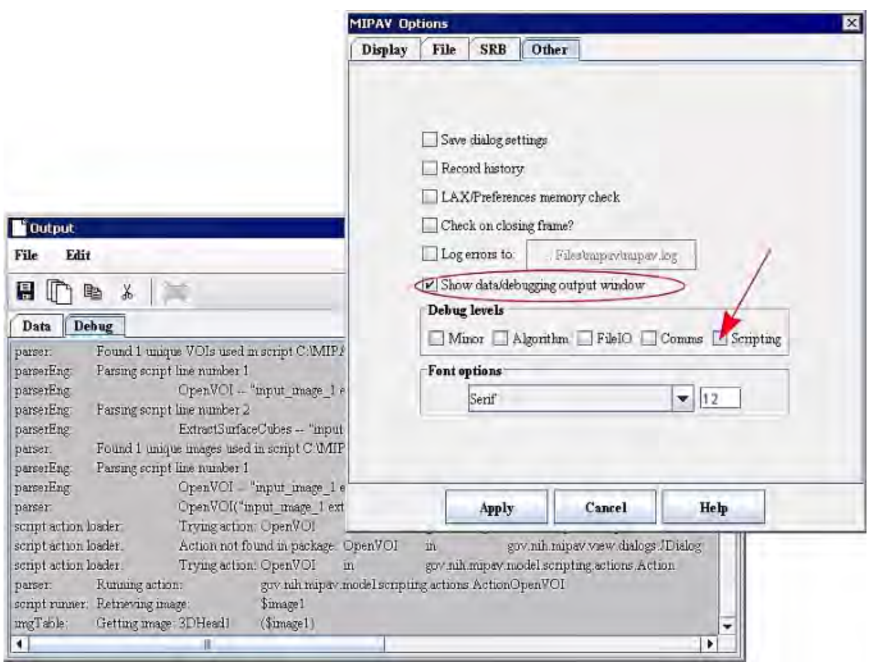


Figure 346. The Other tab of the MIPAV Options dialog box and the Debug tab of the Output window

Save dialog settngs	Saves the active dialog box settings in a log file.
Record history	Keeps a record of all of the actions—algorithms and utilities—performed on images. The history may be viewed on the History page in the Image Attributes dialog box while the actions are performed or in the XML file when an image is saved to an XML file.By default, this check box is clear.
LAX / Preferences memory check	Enables MIPAV developers to debug the program. If you experience problems with the program, you may be asked to select this check box. **Recommendation:** Do not select this check box unless otherwise instructed by MIPAV development. By default, this check box is clear.
Check on closing frame?	Adds messages to confirm deletions of images. By default, this check box is clear.
Log errors to:	Allows you to select the file in which MIPAV records any errors that occur. By default MIPAV logs errors to C:\Program Files\mipav\mipav.log unless the name and path of the log file was changed during installation.
Debug levels:	**Minor** records only minor error messages in mipav.log. **Algorithm** records only error messages with algorithms in mipav.log. **FIleIO** records only FileIO (file input and output) error messages in mipav.log. **Comms** records only error messages involving communications in mipav.log. **Scripts** records all error messages in mipav.log.
Apply	Saves and immediately applies all of the selected parameters in this dialog box.
Cancel	Disregards any changes you made in this dialog box, closes the dialog box, and does not save the specified options.
Help	Displays online help for this dialog box.

Figure 346. The Other tab of the MIPAV Options dialog box and the Debug tab of the Output window (continued)

Gather software information

Please gather the following information:

- **Software version number**

  If you don't know the software version number, select Help > About MIPAV in the MIPAV window. The About MIPAV dialog box opens. The software version number appears in that dialog box.

- **Steps taken to remedy the problem**

  Indicate what steps you took to solve the problem (if any).

Contact MIPAV technical support

To contact technical support, send an e-mail message to the following address:

mcmatt@exchange.cit.nih.gov

This address is also listed on the MIPAV web site:

http://mipav.cit.nih.gov/bugs.htm

Appendix A
References

The following references were used during the preparation of the *MIPAV User's Guide*, Volume 1, *Basics*.

Born, G. *The File Formats Handbook.* International Thomson Computer Press, 1995.

Digital Imaging and Communications in Medicine (DICOM): Part 1, Introduction and Overview. PS 3.1-1998. National Electrical Manufacturers Association (NEMA), 1998.

Digital Imaging and Communications in Medicine (DICOM): Part 10, Media Storage and File Format for Media Interchange. PS 3.1-1998. National Electrical Manufacturers Association (NEMA), 1998.

Murray, James D., vanRyper, W. *Encyclopedia of Graphics Formats: Second Edition*. O'Reilly and Associates, Inc., 1996.

Russ, John C. *The Image Processing Handbook*, 3rd Edition. IEEE Press, 1999.

APPENDIX B

DICOM Conformance Statement

In this appendix . . .

The purpose of this conformance statement is to facilitate communications and interoperations with the National Institutes of Health (NIH) Medical Image Processing, Analysis, and Visualization program (MIPAV).

This introduction describes the MIPAV application and briefly summarizes the Digital Imaging and Communications in Medicine (DICOM) standard services employed by MIPAV.

MIPAV

MIPAV is an *n*-dimensional, general purpose image-processing program designed to assist the NIH research and clinical communities in extracting quantitative information from various medical imaging modalities to better understand, diagnose, monitor, and treat medical disorders.

MIPAV, which is written in Java, takes advantage of the programming language's intrinsic object-oriented capabilities to improve code reuse, functionality, and portability. MIPAV is available on any Java-capable operating system, such as Windows, Macintosh, Irix, and Solaris.

Although this is a general purpose image-processing platform, Dr. McAuliffe presently uses this platform to develop specific and unique image processing techniques to meet the requirements of his collaborators.

MIPAV DICOM communications interface

The MIPAV DICOM communications interface automates the process of querying and retrieving DICOM standard formatted files. The transfer of images can be clumsy and time consuming when studies are manually transferred to removable media or even to File Transfer Protocol (FTP) access, which does not ensure file format compatibility. MIPAV allows you to transfer DICOM standard formatted images over the network using the DICOM communications protocol that runs over the Transmission Control Protocol/Internet Protocol (TCP/IP) stack.

The MIPAVapplication starts a DICOM image receiver that runs in the background and listens on a given port for incoming DICOM-store requests. When a store request is received by the DICOM receiver, the DICOM-formatted images are saved to the local system disk in the user-designated images directory. Once stored, you can use MIPAV to access, visualize, and analyze images.

MIPAV can also send selected images that are on the local disk to a remote destination by implementing the composite storage (C-STORE) service class as a service class user (SCU).

Finally, and most important, MIPAV has a query and retrieve capability that allows you to query a remote DICOM query server for patient, study, series, and image information. You can select from the responses to the query the set of images to be retrieved (moved) to the local disk for visualization and analysis.

Implementation model

This section describes the application entities (AEs) in the MIPAV DICOM communications interface and how they relate to both local and remote real-world activities. The implementation model consists of an application data flow diagram and functional definitions of all DICOM processes handled by the MIPAV AE.

The MIPAV DICOM receiver conforms to the DICOM standard as a service class provider (SCP) of most C-STORE service object pair (SOP) classes. The MIPAV query/retrieve process conforms to the DICOM standard as a SCU for DICOM C-FIND and C-MOVE services. In addition, the MIPAV DICOM sender conforms to the DICOM standard as a SCU for most DICOM C-STORE SOP classes.

Application data flow diagram

Figure 1 shows the relationship between the MIPAV AE and its DICOM processes and the remote AE and its DICOM processes. The remote AE could be any DICOM-compliant system that acts as a query/retrieve server and a DICOM image file receiver and sender, such as a DICOM image archive.

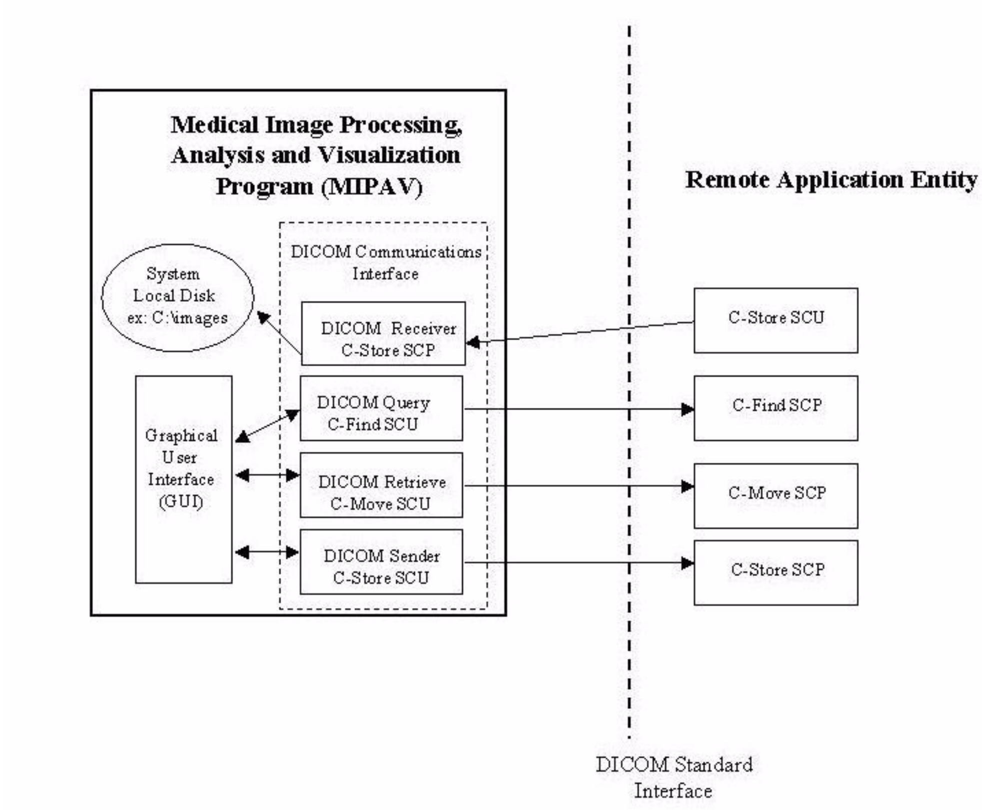


Figure 1. MIPAV dataflow diagram

Functional definitions of AEs

This section describes the image verification, query, and transfer functions to be performed by the MIPAV AE and the DICOM services used to accomplish these functions.

Verification

The MIPAV DICOM communications interface verifies application-level communication with a remote DICOM AE with the C-ECHO (Verification) SOP class in the SCP role. A remote DICOM AE supporting the Verification SOP class SCU role shall send a C-ECHO request to the MIPAV application. The MIPAV AE then sends a response of SUCCESS to confirm DICOM communications between the two AEs.

DICOM RECEIVER (C-STORE SCP)

The DICOM image receiver is initialized as a standalone resident program when the MIPAV application is started. The DICOM receiver waits for a remote AE to request a connection at the presentation address configured for its AE Title. The presentation address of the DICOM receiver consists of the system IP address, AE Title, and communications port. The AE Title and communications port for the DICOM receiver are user configurable in the preferences file mipav.preferences.

The DICOM receiver accepts associations with Presentation Contexts for the SOP Classes of the Storage Service Class. Thus, the DICOM receiver accepts storage requests for Computed Tomography (CT), Magnetic Resonance (MR), Ultrasound (US), Nuclear Medicine (NM), Computed Radiography (CR), and most other modalities. It receives the images and writes them to files in the format specified in *Digital Imaging and Communications In Medicine (DICOM) Part 10: Media Storage and File Format for Media Interchange* (see Appendix E for a full citation.)

DICOM QUERY (C-FIND SCU)

The MIPAV application supports the DICOM C-FIND query class as a SCU by allowing you to query a remote DICOM query server (C-FIND SCP) for patient, study, series, and image information. MIPAV employs the Study Root Query/Retrieve Information Model based on the three-level hierarchy:

- *Study*—Study is the top level. It contains attributes associated with the study and patient information entity's (IE).
- *Series*—The series level, which is below the study level, contains attributes associated with the series, frame of reference, and equipment IEs.
- *Image*—Image is the lowest level. It contains attributes associated with the Image IE.

You can use a Study Root Study Level C-FIND request message, with search key attributes of patient name or patient ID and study date range, to query the SCP for a patient list or for demographic information about a given patient. A Study Root Study Level query, with a known search key attribute of patient ID, can be sent to the Query SCP for the study list corresponding to the given patient ID. Once the desired study is queried, then MIPAV can send a Study Root Series Level query with the known Unique Key Attributes of Patient ID and Study Instance UID to the query server for the list of series corresponding to the given study. Finally, once the desired series is located, you can query at the Study Root Image Level with known Key Attributes of Patient ID, Study Instance UID, and Series Instance UID to get the list of images corresponding to the selected series.

The MIPAV query routine interprets all PENDING status responses from the C-FIND SCP as matches to the key attributes in the query request. A status equal to SUCCESS, FAILED or REFUSED conveys the end of query request.

To cancel the C-FIND service, the MIPAV AE issues a C-FIND-CANCEL request at any time during the processing of the C-FIND query. The MIPAV query routine that issued the C-FIND request recognizes a status of CANCELED to indicate that the C-FIND-CANCEL was successful.

DICOM RETRIEVE (C-MOVE SCU)

MIPAV supports the DICOM C-MOVE SOP class as a SCU. You can request the transfer of images from a remote AE to the local system or to a desired remote destination with a DICOM C-MOVE service request. The destination for the move, whether it is the local disk or a remote system, may be configured and selected from a host table from within the MIPAV DICOM Communication Panel window. The Move Destination is specified by the parameters of AE Title, IP Address, and Communications Port number in the Hosts table. To review or modify the configuration of the Hosts table, the user selects the Hosts tab from the DICOM Communication Panel window.

The DICOM C-MOVE class employs, like the DICOM Query, the Study Root Query/Retrieve Information Model based on the three-level hierarchy:

- **Study**—A C-MOVE request at the study level transfers all images related to a study to the designated move destination.
- **Series**—A C-MOVE request at the series level transfers all images related to a series.
- **Image**—A C-MOVE request at the image level transfers all selected individual images.

The MIPAV retrieve routine supplies unique key values to identify an entity at the level of retrieval to the C-MOVE SCP. The SCP executes C-STORE suboperations for the corresponding storage SOP instances identified by the unique key values in the C-MOVE request. The MIPAV retrieve routine interprets all PENDING status responses from the C-MOVE SCP as matches to the key attributes in the retrieve request. A status equal to SUCCESS, FAILED, or REFUSED conveys the end of the retrieve request.

The MIPAV AE may cancel the C-MOVE service request by issuing a C-MOVE-CANCEL request at any time during the processing of the C-MOVE request. The MIPAV retrieve routine that issued the C-MOVE request recognizes a status of CANCELED to indicate that the C-MOVE-CANCEL was successful.

DICOM sender (C-STORE SCU)

MIPAV provides the DICOM C-STORE SOP class as a SCU.

To access the DICOM send option

1 Select File > DICOM Database Access in the MIPAV window. The DICOM Communication Panel window opens.

2 Select Send to view the Send page.

3 Select a patient, study, series, or image to send to a designated destination. The store destination is specified by the parameters of AE Title, IP Address, and Communications Port number in the hosts table.

To review or modify the configuration of the hosts table

1 Select Hosts in the DICOM Communication Panel window. The Hosts page opens.

2 Select the desired image data and the storage destination.

3 Click OK. The MIPAV sender routine establishes an association with the selected destination and transfers the image data.

Sequencing of real-world activities

Not applicable.

AE specifications

This section provides detailed specifications of the MIPAV DICOM communications interface. It lists the SOP classes supported and outlines the policies with which MIPAV initiates or accepts associations. A description of proposed (for association initiation) and acceptable (for association acceptance) Presentation contexts is also provided.

Note that a Presentation Context consists of an Abstract Syntax and a list of acceptable Transfer Syntaxes. The Abstract Syntax identifies one SOP Class or Meta SOP Class. By listing the AEs with their proposed and accepted

Presentation Contexts, this Conformance Statement identifies the set of Information Objects and Service classes recognized by MIPAV.

For each SOP Class related to an Abstract Syntax, a list is given of any supported SOP options.

MIPAV AE SPECIFICATION

This section summarizes the SOP classes that are supported by the MIPAV DICOM Communications interface. The supported SOP classes are listed in two categories:

- SOP classes supported by MIPAV as a SCU
- SOP classes supported by MIPAV as a SCP

MIPAV provides Standard Conformance to the DICOM V3.0 SOP Classes shown in Table 1 as a SCU.

Table 1. DICOM query, retrieve, and sender classes supported by MIPAV

SOP class name	SOP class UID
DICOM query	
Study Root Query/Retrieve Information Model, C-FIND	1.2.840.10008.5.1.4.1.2.2.1
DICOM retrieve	
Study Root Query/Retrieve Information Model, C-MOVE	1.2.840.10008.5.1.4.1.2.2.2
DICOM sender	
CR Image Storage	1.2.840.10008.5.1.4.1.1.1
CR Image Storage	1.2.840.10008.5.1.4.1.1.2
MR Image Storage	1.2.840.10008.5.1.4.1.1.4
NM Image Storage	1.2.840.10008.5.1.4.1.1.20
US Image Storage	1.2.840.10008.5.1.4.1.1.6
Secondary Capture Image Storage	1.2.840.10008.5.1.4.1.1.7

MIPAV provides Standard Conformance to the DICOM version 3.0 SOP Classes shown in Table 2 as a SCP.

Table 2. Verification and DICOM receiver classes supported by MIPAV

SOP class name	SOP class UID
Verification	
Verification SOP Class	1.2.840.10008.1.1
DICOM receiver	
CR Image Storage	1.2.840.10008.5.1.4.1.1.1
CT Image Storage	1.2.840.10008.5.1.4.1.1.2
MR Image Storage	1.2.840.10008.5.1.4.1.1.4
NM Image Storage	1.2.840.10008.5.1.4.1.1.20
US Image Storage	1.2.840.10008.5.1.4.1.1.6

DICOM QUERY (C-FIND SCU) AE SPECIFICATION

MIPAV provides standard conformance to the DICOM 3.0 Query/Retrieve SOP class listed in Table 3.

Table 3. Supported C-FIND SOP class

SOP class name	SOP class UID
Study Root Query/Retrieve Information Model C-FIND	1.2.840.10008.5.1.4.1.2.2.1

Association establishment policies

General

The MIPAV query routine initiates an association with a remote DICOM query server. Extended negotiation is not provided.

The maximum Protocol Data Unit (PDU) size in an association request defaults to 16 kilobytes.

Number of associations

Each query request within MIPAV initiates an association with a remote DICOM query server. Thus, multiple associations can be opened and processed by MIPAV in one working session.

Asynchronous nature

The DICOM Query routine only allows a single outstanding operation on an Association. Thus, there is no asynchronous activity in this implementation.

Implementation identifying information

(TBD. Need information on the Implementation Class Unique Identifier (UID) for the MIPAV query routine. Note that this may be the same for all applications, one implementation UID for a DICOM application. For information, contact Richard Eaton at NEMA at 703/841-3248 or e-mail at ric_eaton@nema.org).

Association initiation by real-world activity

To initiate an association from MIPAV to a remote query server (C-FIND SCU), select the Send Query button in the DICOM Communication Panel window.

Query request

After you insert the search keys and set the study date range in the DICOM Communication Panel window, you can then select the Send Query button to transfer the C-FIND request to the remote DICOM query server. Each query opens an association with the query server. Select Cancel at the bottom of the DICOM Communication Panel window to cancel the C-FIND request. The C-FIND-CANCEL request is sent over the same association as the originating C-FIND request.

Associated real-world activity. The initiation of a C-FIND request is the associated real-world activity.

Proposed Presentation Contexts. When MIPAV initiates a C-FIND request, a presentation context is proposed for the Study Root Query/Retrieve C-FIND supported SOP Class. No extended negotiation is supported.

Table 4. Presentation context proposed by MIPAV as a result of real-world activity query request to an external query server

Presentation context table					
Abstract syntax		Transfer syntax		Role	Extended negotiation
Name	UID	Name list	UID list		
Study Root Q/R C-FIND	1.2.840.10008.5.1.4.1.2.2.1	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None

SOP specific conformance statement for SOP class study root query/retrieve information model C-FIND

The attributes listed in Table 5 comprise the Study Root Query/Retrieve C-FIND identifier that is sent in the DICOM query message. The level column indicates the query level at which the attributes can be included.

Table 5. DICOM data elements supported for SOP class study root query/retrieve information model C-FIND SCU"

Level	Description	Tag	Type
Study			
Study	Study Date	(0008,0020)	R
Study	Study Time	(0008,0030)	R
Study	Study ID	(0020,0010)	R
Study	Patient's Name	(0010,0010)	R
Study	Patient ID	(0010,0020)	R
Study	Study Instance UID	(0020,000D)	U
Study	Referring Physician's Name	(0008,0090)	O
Series			
Series	Modality	(0008,0060)	R
Series	Series Number	(0020,0011)	R
Series	Series Instance UID	(0020,000E)	U
Series	Series Date	(0008,0021)	O
Series	Series Description	(0008,103E)	O
Series	Body Part Examined	(0018,0015)	O
Image			
Image	Image Number	(0020,0013)	R
Image	SOP Instance UID	(0008,0018)	U

Table 5. DICOM data elements supported for SOP class study root query/retrieve information model C-FIND SCU"

Level	Description	Tag	Type
Image	Image Date	(0008,0023)	O
Image	Image Time	(0008,0033)	O

Association acceptance policy

Not applicable.

DICOM RETRIEVE (C-MOVE SCU) AE SPECIFICATION

MIPAV provides standard conformance to the DICOM 3.0 Query/Retrieve SOP class.

Table 6. Supported image storage service

SOP class name	SOP class UID
Study Root Query/Retrieve Information Model C-MOVE	1.2.840.10008.5.1.4.1.2.2.2

Association establishment policies

General

The MIPAV retrieve routine initiates an association with a remote DICOM query/retrieve server. Extended negotiation is not provided.

The maximum PDU size in an association request defaults to 16 kilobytes.

Number of associations

Each retrieve (C-MOVE) request within the MIPAV application initiates an association with a remote DICOM query/retrieve server. Thus, multiple associations for the C-MOVE SOP class can be opened and processed by MIPAV in one working session.

Asynchronous nature. The DICOM Receiver only allows a single outstanding operation on an association. Thus, there is no asynchronous activity in this implementation.

Implementation identifying information. (TBD. Need the Implementation Class Unique Identifier (UID) for the MIPAV move request routine. Note that this may be the same for all applications—one implementation UID for a DICOM application. For information, contact Richard Eaton at NEMA at 703/841-3248 or e-mail at ric_eaton@nema.org.

Association initiation by real-world activity

To initiate an association from MIPAV to a remote query/retrieve server (C-MOVE SCP), select Q/R Client in the DICOM Communication Panel window. Then complete the appropriate parameters in the Q/R Client page. Initially run a query from the Q/R Client page. You can then select an entry from the query responses that indicates the desired patient, study, series, or image to be moved to the set destination. The destination to be moved to with the C-MOVE request is user configurable in the Hosts page, which is in the DICOM Query Panel window.

Retrieve request

After you successfully query the remote query/retrieve server and locate the set of images to move to the local system, you can then send a C-MOVE request to move the desired images to the local system. To initiate the C-MOVE request, click Move Image in the DICOM Communication Panel window, which then opens an association to the remote query/retrieve server. The remote server then responds to the C-MOVE request by initiating a C-STORE request on a new association to the C-STORE SCU process in the MIPAV application.

Associated real-world activity. The initiation of a C-MOVE request is the associated real-world activity.

Proposed presentation contexts. When MIPAV initiates a C-MOVE request, a presentation context is proposed for the Study Root Query/Retrieve C-MOVE supported SOP Class. No extended negotiation is supported.

Table 7. Presentation context proposed by MIPAV as a result of real-world activity "MOVE Request to an External Query Server"

Presentation context table					
Abstract syntax		Transfer syntax		Role	Extended negotiation
Name	UID	Name list	UID list		

Table 7. Presentation context proposed by MIPAV as a result of real-world activity “MOVE Request to an External Query Server”

Presentation context table					
Study Root Q/R C-MOVE	1.2.840.10008.5.1.4.1.2.2.2	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	1.2.840.10008.1.2

SOP specific conformance statement for SOP Class Study Root Query/Retrieve Information Model C-MOVE. The attributes listed in Table 8 comprise the Study Root Query/Retrieve C-MOVE identifier that is sent in the DICOM retrieve message. The level column indicates the query level at which the attributes can be included. Note that the table of attributes below is identical to those for the Study Root Query/Retrieve C-FIND identifier in Table 8

Table 8. DICOM data elements supported for SOP Class Study Root Query/Retrieve Information Model C-MOVE SCU

Level	Description	Tag	Type
Study			
Study	Study Date	(0008,0020)	R
Study	Study Time	(0008,0030)	R
Study	Study ID	(0020,0010)	R
Study	Patient's Name	(0010,0010)	R
Study	Patient ID	(0010,0020)	R
Study	Study Instance UID	(0020,000D)	U
Study	Referring Physician's Name	(0008,0090)	O
Series			
Series	Modality	(0008,0060)	R
Series	Series Number	(0020,0011)	R
Series	Series Instance UID	(0020,000E)	U
Series	Series Date	(0008,0021)	O
Series	Series Description	(0008,103E)	O
Series	Body Part Examined	(0018,0015)	O
Image			
Image	Image Number	(0020,0013)	R
Image	SOP Instance UID	(0008,0018)	U
Image	Image Date	(0008,0023)	O
Image	Image Time	(0008,0033)	O

Association acceptance policy

Not applicable.

DICOM SENDER (STORAGE SCU) AE SPECIFICATION

MIPAV provides standard conformance to the DICOM 3.0 Storage SOP classes listed in Table 9

Table 9. Supported C-STORE SOP classes

SOP class name	SOP class UID
Computed Radiography Image Storage	1.2.840.10008.5.1.4.1.1.1
CT Image Storage	1.2.840.10008.5.1.4.1.1.2
MR Image Storage	1.2.840.10008.5.1.4.1.1.4
Nuclear Medicine Image Storage	1.2.840.10008.5.1.4.1.1.20
Ultrasound Image Storage	1.2.840.10008.5.1.4.1.1.6
Secondary Capture Image Storage	1.2.840.10008.5.1.4.1.1.7

Association establishment policies

General

The MIPAV DICOM image sender initiates an association with a remote DICOM image receiver. Extended negotiation is not supported.

The maximum PDU size in an association request defaults to 16 kilobytes.

Number of associations

Each send (C-STORE) request within the MIPAV application initiates an association with a remote DICOM image receiver. Thus, multiple associations for the C-STORE SOP class can be opened and processed by MIPAV in one working session.

Asynchronous nature. The DICOM Sender only allows a single outstanding operation on an Association. Thus, there is no asynchronous activity in this implementation.

Implementation identifying information

(TBD. Need the Implementation Class Unique Identifier (UID) for the MIPAV DICOM Sender. Note that this may be the same for all applications-one implementation UID for a DICOM application. For information, contact Richard Eaton at NEMA at 703/841-3248 or through e-mail at ric_eaton@nema.org.

Association initiation by real-world activity

To initiate an association from MIPAV to a remote image receiver (C-STORE SCP), select Send in the DICOM Query Panel window. When you select File > DICOM Database Access in the MIPAV window, the DICOM Communication Panel window opens.

DICOM send request

Associated real-world activity. The initiation of a Send image request is the associated real-world activity.

Proposed presentation contexts. When MIPAV initiates a C-STORE request, a different presentation context is proposed for each of the different supported C-STORE SOP Classes. No extended negotiation is supported.

Table 10. Presentation contexts proposed by MIPAV as a result of real-world activity “store request to an external query server

Presentation context table

Abstract syntax		Transfer syntax			
Name	**UID**	**Name list**	**UID list**	**Role**	**Extended negotiation**
CR Image Storage	1.2.840.10008.5.1.4.1.1.1	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
CT Image Storage	1.2.840.10008.5.1.4.1.1.2	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
MR Image Storage	1.2.840.10008.5.1.4.1.1.4	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
US Image Storage	1.2.840.10008.5.1.4.1.1.6	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
Secondary Capture Image Storage	1.2.840.10008.5.1.4.1.1.7	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None

SOP specific conformance statement for supported storage SOP classes. The following overview summarizes the behavior of the MIPAV DICOM Sender routine depending on the responses to the C-STORE request:

- **Successful C-STORE response:** In the case of a response of SUCCESS for the C-STORE request, the MIPAV Send status panel in the Send page in the DICOM Communication Panel window displays a status of SUCCESS. This indicates that the images were properly received by the image receiver (C-STORE SCU).
- **Unsuccessful C-STORE response:** In the case of a response of REFUSED, CANCEL, or FAILED, for the C-STORE request, the DICOM sender routine aborts the association. The MIPAV Send status panel in the Send page in the DICOM Communication Panel window displays a status of FAILED. The software makes no further attempts to retry the transfer of the aborted images.
- **Warning status in C-STORE response:** In the case of a response of Warning for the C-STORE request, the MIPAV DICOM sender routine behaves the same as if a response of SUCCESS was received.

The DICOM Sender does not attempt any extended negotiation.

The DICOM Sender supports all type 1, type 2, and type 3 attributes defined in the Information Object Definition (IOD) associated with the SOP class. No attributes are discarded or coerced by the DICOM Sender. The originally saved DICOM file is read from disk and forwarded to the desired remote system. Note that the DICOM Sender does not validate that the attributes of the SOP instance for the outgoing C-STORE message request meet the requirements of the IOD. It is assumed that the saved DICOM image file is stored in a valid DICOM file format.

The SOP Instance UID (group 0x0008, element 0x0018), Study Instance UID (group 0x0020, element 0x000D), and Series Instance UID (group 0x0020, element 0x000E) consist of a root and suffix. The root consists of the date and time of transaction. The suffix is conforms to the DICOM standard.

Association acceptance policy

Not applicable.

VERIFICATION (C-ECHO SCP) AE SPECIFICATION

MIPAV provides standard conformance to the DICOM 3.0 Verification SOP class listed in Table 11.

Table 11. Supported verification SOP class

SOP class name	SOP class UID
Verification SOP Class	1.2.840.10008.1.1

Association establishment policies

General
The MIPAV DICOM Verification routine responds to a verification of communication request from a remote DICOM AE by sending a C-Echo response of a status of SUCCESS.

The maximum PDU size in an association request defaults to 16 kilobytes.

Number of associations
Each verification (C-Echo) request sent to the MIPAV application is responded to on an association opened by the remote AE. Multiple associations for the C-ECHO SOP class can be accepted and processed by MIPAV in one working session.

Asynchronous nature
The DICOM verification routine only allows a single outstanding operation on an Association. Thus, there is no asynchronous activity in this implementation.

Implementation Identifying Information
(TBD. Need the Implementation Class Unique Identifier (UID) for the MIPAV DICOM verification routine. Note that this may be the same for all applications- one implementation UID for a DICOM application. For information contact Richard Eaton at NEMA- (703) 841-3248, email- ric_eaton@nema.org)

Association initiation by real-world activity

Not applicable.

Association Acceptance Policy

MIPAV accepts all associations for C-ECHO requests initiated by remote systems.

DICOM Verification Request

Associated Real-World Activity. The arrival of a verification, or C-Echo, request is the associated real-world activity.

Proposed Presentation Contexts. When the MIPAV Message Receiver gets a verification request, the presentation context accepted for the C-ECHO SOP class is listed in Table 12. No extended negotiation is supported.

Table 12. Presentation contexts accepted by MIPAV as a result of real-world activity "verification" request

Presentation context table					
Abstract syntax		**Transfer syntax**		**Role**	**Extended negotiation**
Name	**UID**	**Name list**	**UID list**		
Verification	1.2.840.10008.5.1.4.1.1.1	Implicit VR Little Endian	1.2.840.10008.1.1	SCP	None

SOP Specific Conformance Statement for Verification SOP Class. The Verification AE follows the DICOM 3.0 standard for handling of C-ECHO requests. A status of SUCCESS is returned to a valid C-ECHO verification request.

DICOM IMAGE RECEIVER (STORAGE SCP) AE SPECIFICATION

MIPAV provides standard conformance to the DICOM 3.0 Storage SOP classes listed in Table 13

Table 13. Supported C-STORE SOP classes

SOP class name	SOP class UID
CR Image Storage	1.2.840.10008.5.1.4.1.1.1
CT Image Storage	1.2.840.10008.5.1.4.1.1.2
MR Image Storage	1.2.840.10008.5.1.4.1.1.4

Table 13. Supported C-STORE SOP classes

SOP class name	SOP class UID
NM Image Storage	1.2.840.10008.5.1.4.1.1.20
US Image Storage	1.2.840.10008.5.1.4.1.1.6
Secondary Capture Image Storage	1.2.840.10008.5.1.4.1.1.7

Association Establishment Policies

General

The MIPAV DICOM image receiver opens a node at the port specified in the mipav.preferences file and waits for an association from a DICOM application at the specified port. MIPAV accepts an association from a remote DICOM image sender. Extended negotiation is not supported.

The maximum PDU size in an association request defaults to 16 kilobytes.

Number of Associations

The DICOM Image Receiver initiates a new process for each connection request it receives. Thus, the image receiver can have multiple simultaneous connections and there are no inherent limitations on the total number of simultaneous associations that the image receiver can maintain.

Asynchronous Nature

The DICOM Image Receiver only allows a single outstanding operation on an Association. Thus, there is no asynchronous activity in this implementation.

Implementation Identifying Information

(TBD. Need the Implementation Class Unique Identifier (UID) for the MIPAV DICOM Receiver. Note that this may be the same for all applications- one implementation UID for a DICOM application. For information contact Richard Eaton at NEMA- (703) 841-3248, email- ric_eaton@nema.org)

Association Initiation by Real-World Activity

Not applicable. The DICOM Image Receiver never initiates an association.

Association Acceptance Policy

When the MIPAV Image Receiver accepts an association, it receives any images transferred on that association and stores the images on the local disk in the native machine file system, in the format specified in *Digital*

Imaging and Communications In Medicine (DICOM) Part 10: Media Storage and File Format for Media Interchange (see Appendix E for a full citation.) The Image Receiver places no limitation on who may connect to it, or the number of simultaneous connections it can support.

DICOM image receiver

Associated real-world activity. The storage of an image on the disk of the local system is the associated real-world activity for the Image Receiver.

Proposed presentation contexts. The presentation contexts accepted by the MIPAV Image Receiver are listed in Table 14. No extended negotiation is supported.

Table 14. Presentation contexts proposed by MIPAV as a result of real-world activity "receive and store images"

Presentation context table					
Abstract syntax		**Transfer syntax**		**Role**	**Extended negotiation**
Name	**UID**	**Name list**	**UID list**		
CR Image Storage	1.2.840.10008.5.1.4.1.1.1		Implicit VR Little Endian		1.2.840.10008.1.2
CT Image Storage	1.2.840.10008.5.1.4.1.1.2		Implicit VR Little Endian		1.2.840.10008.1.2
MR Image Storage	1.2.840.10008.5.1.4.1.1.4		Implicit VR Little Endian		1.2.840.10008.1.2
NM Image Storage	1.2.840.10008.5.1.4.1.1.20		Implicit VR Little Endian		1.2.840.10008.1.2
US Image Storage	1.2.840.10008.5.1.4.1.1.6		Implicit VR Little Endian		1.2.840.10008.1.2
Secondary Capture Image Storage	1.2.840.10008.5.1.4.1.1.7		Implicit VR Little Endian		1.2.840.10008.1.2

SOP Specific Conformance Statement for Supported Storage SOP Classes. The MIPAV Image Receiver conforms to the Storage SOP Classes as a SCP at Level 2 (Full). No elements are discarded or coerced by the MIPAV Image Receiver. In the event of a successful C-STORE operation, the DICOM image file is successfully written to the local disk with a standard path and file name format. The default path for the image data is /images. The default path can be customized in the mipav.preferences file. The image receiver does not support any extended negotiation.

Communications profiles

OVERVIEW

This section lists all communication protocols supported by MIPAV.

SUPPORTED COMMUNICATIONS STACKS

TCP/IP

The MIPAV DICOM routines run over the TCP/IP stack.

Application Programmer's Interface (API)
The MIPAV DICOM routines are implemented using the Berkeley Sockets interface to TCP/IP services.

Physical Media Supported

The MIPAV DICOM routines run on any physical media supported by the TCP/IP stack that is run on the host machine.

Extensions, Specializations, and Privatizations

OVERVIEW

This section lists all DICOM Standard extended, specialized, or private SOP Class implementations.

MIPAV AE DICOM SERVICES

No extended, specialized, or private SOPs are specified.

DICOM Configuration Details

OVERVIEW

This section addresses the method of setting configurable parameters for the DICOM routines.

For the MIPAV DICOM Receiver, the port number and AE Title are configurable in the mipav.preferences file.

The MIPAV Query/Retrieve process also uses the AE Title, IP Address, and port number settings for the host machine and the remote query server. You can change the values for these settings on the Hosts page in the DICOM Communication Panel window. The settings can also be configured in the mipav.preferences file.

AE TITLE/PRESENTATION MAPPING

The MIPAV DICOM routines map the AE title to a presentation address (IP address and port number) by accessing information on the Hosts page in the DICOM Communication Panel window. This is then written and saved in the mipav.preferences file.

Support of Extended Character Sets

OVERVIEW

Any support for extended character sets, such as multibyte characters, are described in this section.

MIPAV ICOM AE

MIPAV does not support extended character sets.

Table 15. DICOM tags

Field	Description
Study Instance UID (0020,000D)	Unique identifier for the study. Only numeric characters with optional periods are allowed.
Modality (0008,0060)	Type of equipment that was used to acquire the data used to create the images in the dataset. Options are: Biomagnetic Imaging, Color Flow Doppler, Computed Tomography, Duplex Doppler, Computed Radiography, Diaphanography, Digital Radiography, Endoscopy, General Microscopy, Hard Copy, Intraoral Radiography, Laser Surface Scan, MR Angiography, Mammography, Magnetic Resonance, MR Spectroscopy, Nuclear Medicine, Other, PET, Panoramic XRay, Radio Fluoroscopy, Radiographic Imaging, Radiotherapy Dose, Radiotherapy Image, Radiotherapy Record, Radiotherapy Structure, Slide Microscopy, SPECT, Thermography, Ultrasound, XRay Angiography, and External Photography.
Series Instance UID (0020,000E)	Unique identifier for the study. Only numeric characters (with optional periods) are allowed.
Patient	
Patient's Name (0010,0010)	Patient's full name.
Patient's Birth Date (0010,0030)	Date of patient's birth.
Patient's Birth Time (0010,0032)	Time of patient's birth.
Other Patient Names (0010,1001)	Other names used to identify the patient.
Patient Comments (0010,4000)	User-defined comments about the patient.
Patient ID (0010,0020)	Primary hospital identification number or code used to identify the patient.
Patient's Sex (0010,0040)	Gender of the patient. Options are: Unknown, Male, Female, and Other.
Other Patient IDs (0010,1000)	Other IDs used to identify the patient.
Ethnic Group (0010,2160)	Ethnic group or race of the patient.
Patient Orientation (0020,0020)	Patient direction of the rows and columns of the image.
Study	
Study ID (0020,0010)	User- or equipment-generated study identifier.
Study Time (0008,0030)	Time the study started.

Table 15. DICOM tags (continued)

Field	Description
Study Description (0008,1030)	Institute-generated description or classification of the study (component) performed.
Physician(s) of Record (0008,1048)	Physician responsible for the overall patient care at the time of the study.
Admitting Diagnoses Description (0008,1080)	Description of the admitting diagnoses.
Patient's Size (0010,1020)	Length or size of the patient in meters.
Occupation (0010,2180)	Occupation of the patient.
Study Date (0008,0020)	Date the study started.
Accession Number (0008,0050)	An RIS-generated number which identifies the order for the study.
Referring Physician's Name (0008,0090)	Patient's referring physician.
Physician(s) Reading Study (0008,1060)	Physician(s) reading the study.
Patient's Age (0010,1010)	Age of the patient.
Patient's Weight (0010,1030)	Weight of the patient, in kilograms.
Additional Patient's History (0010,21B0)	Additional information about the patient's history.
Series	
Series Number (0020,0011)	A number that identifies this series.
Performing Physicians' Name (0008,1050)	Name(s) of the physician(s) administering the series.
Series Description (0008,103E)	User-provided description for the series.
Body Part Examined (00018,0015)	A text description of the body part that was examined. Options are: Unknown, Skull, CSpine, TSpine, LSpine, SSpine, Coccyx, Chest, Clavicle, Breast, Abdomen, Pelvis, Hip, Shoulder, Elbow, Knee, Ankle, Hand, Foot, Extremity, Head, Heart, Neck, Leg, Arm, and Jaw.
Smallest Pixel Value (0028,0108)	Minimum value of all images in this series.
Procedure Step ID (0040,0253)	Identification of that part of a procedures that was performed during this step.

Table 15. DICOM tags (continued)

Field	Description
Procedure Step Start Time (0040,0245)	Time when the procedure step started.
Laterality (0020,0080)	Options are: Unknown, Left, and Right.
Series Time (0008,0031)	Time series started.
Protocol Name (0018,1030)	User-defined description of the conditions under which the series was performed.
Operator's Name (0008,1070)	Name(s) of the technologist(s) supporting the series.
Patient Position (0018,5100)	Patient position relative to the imaging-equipment space. Options are: Unknown, Head-First Prone, Head-First Supine, Feet First-Prone, Feet First-Supine, HF-Decubitus Right, HF-Decubitus Left, FF-Decubitus Right, FF-Decubitus Left.
Largest Pixel Value (0028,0109)	Maximum value of all images in this series.
Procedure Step Start Date (0040,0244)	Date when the procedure step started.
Procedure Step Description (0040,0254)	Institute-generated description or classification of the procedure step that was performed.

APPENDIX C

Supported formats

This appendix provides information about graphical and file formats used and supported by MIPAV. Examples of MIPAV-related files can be found in the end of this appendix.

In this appendix:

Supported formats

MIPAV supports three categories of graphic file formats: vector (i.e., volume of interest, or VOI), bitmap (i.e., 2D to 5D grayscale or RGB images), and 3D images.

Graphic formats overview

Vector file formats are usually used to store line-based elements. These elements can be geometric shapes, such as polygons, curves, or splines. The data portion of the vector file contains a mathematical description of elements of an image with specific key points identified. A key point can be thought of as a *node* or *vertex;* it marks the place where the line changes direction. For example, a square may contain four key points while a circle may contain many more.

In addition to key points, the vector file usually indicates the line segment's starting point, direction, and length. In MIPAV you can create geometrically shaped contours to indicate VOIs. You can then store the VOIs in a vector formatted file. MIPAV can read and generate several vector-formatted file types including the MIPAV lookup table (LUT), MIPAV MTX, MIPAV Graphics Plot (PLT), and MIPAV VOI. See also "Understanding MIPAV-related files" on page 659.

Bitmap file formats are usually used to store real-world graphics data – photographs or medical images such as CT scans. The data portion of bit-map files contains numerical data that indicates the color of each pixel or voxel in the image. The bitmap file may also indicate the *bit depth*, which indicates the number of colors that a pixel or voxel can represent. Typically, the bit depth can be 1, 2, 4, 8, 15, 16, 24, or 32. The number of colors represented is the square of the bit depth. Thus, a 4-bit image can contain 16 colors; an 8-bit image can contain 64 colors; and so on. MIPAV can read and generate a number of bitmap formatted file types, such as Adobe Photoshop (PSD), Graphics Interchange File (GIF), Sun Raster (RS), Tagged Image File Format (TIFF), and Truevision Graphics Adapter (TGA).

MIPAV uses 3D graphic formatted files to store descriptions of the color and shape of 3D models of real-world and imaginary objects.

Supported file formats

To support the wide range of image-processing needs of the NIH intramural research community, MIPAV reads and writes the image file types indicated in Table 1 on page 628. This table shows the formal name of the file format, the file extensions associated with the format, the numerical format, and whether MIPAV supports the opening and saving of the files in that format.

Table 1. Image file formats supported by MIPAV

File format	Extencion	Supports	Open	Save	Does not support
AFNI	header HEAD, data BRIK	See "AFNI"	Y	Y[1]	

Table 1. Image file formats supported by MIPAV (continued)

File format	Extencion	Supports	Open	Save	Does not support
Analyze	header HDR, data IMG	See "Analyze files"	Y	Y	TBD
Apple Macintosh	PICT	TBD.	Y	Y	
ASCII text	TXT	MIPAV defined ascii text surface format. The text surface file records the triangle mesh vertices, normal and connectivity.	Y	Y	
Audio Video Interleave	AVI	Uncompressed RLE8 See "Audio Video Interleave (AVI) files"	Y	Y[2]	TBD
BFLOAT	BFLOAT	TBD.	Y	Y	
Bruker	d3pro, reco, acqp, 2dseq	See "Bruker data format"	Y	Y	TBD
Cheschire	IMC, IMG	TBD	Y	Y	TBD
Cheschire Overlay	OLY	TBD	Y	N	
DICOM 2.0	DCM, IMA	See "Digital Imaging and Communications in Medicine (DICOM) files"	Y	Y	TBD
DM3	DM3	TBD			
FITS	FTS	See "FITS"	Y	Y	TBD
FreeSurfer surface	ASCII, ASC	See "FreeSurfer image and surface files"	Y	Y	TBD
FreeSurfer image	COR	INFO for the header file and NNN for each slice where NN is the number of the slice	Y	Y	TBD
GE – Genesis 5X and LX	SIG	See "GE – Genesis 5X and LX"	Y	Y	TBD
GE Signa 4.x	GEDNO	TBD	Y	N	TBD
Image Cytometry Standard	ICS, IDS	See "ICS (Image Cytometry Standard)"	Y	Y	TBD

Table 1. Image file formats supported by MIPAV (continued)

File format	Extencion	Supports	Open	Save	Does not support
Interfile	HDR	TBD	Y	Y	TBD
Laser Scanning Microscope (Zeiss)	LSM	See "Laser Scanning Microscope (Zeiss)"	Y	N	TBD
LIFF	LIFF	See "LIFF"			TBD
Magnetron Vision	IMA	TBD	Y	Y	
MAP	MAP	TBD	Y	N	
Medical Image Network Common Data Form (including ROIs)	MNC	See "Medical Image NetCDF (MINC) files"	Y	Y	TBD
MEDIVISION	BIN	TBD	Y	Y	TBD
MetaMorph Stack	STK	TBD	Y	Y	TBD
Medical Research Council	MRC	See "Medical Research Council (MRC)"	Y	Y	TBD
MGH/MGZ volume format	MGH, MGZ	MGH for uncompressed storage; MGZ or.MGH.GZ for compressed storage	Y	Y	TBD
MICRO-CAT (LOG)	LOG, CT	See "MICRO-CAT"			TBD
MINC 1.0, 2.0	MNC	See "Medical Image NetCDF (MINC) files"	Y	Y	
MIPAV-specific file formats					
MIPAV LUT	LUT	See "MIPAV (LUT) files"			
MIPAV MTX	MTX	See "MIPAV (MTX) files"			
MIPAV PLT	PLT	See "MIPAV (PLT) files"			

Table 1. Image file formats supported by MIPAV (continued)

File format	Extencion	Supports	Open	Save	Does not support
MIPAV SUR	SUR	See "MIPAV (SUR) files"			
MIPAV TXT	TXT	See also "MIPAV text file format" on page 648			
MIPAV VOI	VOI, XML	See "MIPAV (VOI) files"			
MIPAV XML	XML	Full support, see "MIPAV XML"			N/A
End of MIPAV-specific file formats					
MRC	MRC	TBD	Y	Y	
Nearly Raw Raster Data (NRRD)	1 file with extension of.NRRD or 2 files with extension of.NHDR for header file and.RAW,.TXT,.HEX,.RAW.GZ, or.RAW.BZ2 for data file		Y	Y	
NIFTI	2 files with header HDR and data IMG; or a single file.NII	See "NIFTI"	Y	Y	
OSM	WU	TBD			
Optical Coherence Tomography	TMG	TBD	Y	Y	
Phillips	PAR and PARV2 for header files and REC and FREC for data files	PAR and PARV2 for header files and REC and FREC for data files See "Phillips PAR/REC"	Y	Y	TBD

Table 1. Image file formats supported by MIPAV (continued)

File format	Extencion	Supports	Open	Save	Does not support
PICT	PICT	Encoding of black and white images as bitmap mode PICTs. Encodes grayscale images from 2 to 8-bits as palette mode (PICT format does not have grayscale mode). Encodes palette images as palette mode PICTS, encodes other images as a 24-bit 3 component directed bits PICT. Decoding of black and white and up to (and including) 32-bit color PICT images. Also supports the decoding of compressed PICT images.	Y	Y	N/A
Polygon File Format or Stanford Triangle Format	PLY	See "Polygon File Format" on page 652	Y	Y	
QuickTime	QT	See "QuickTime–Apple"	Y	Y	TBD
RAW	RAW	See "Raw data files"	Y	Y	TBD
Siemens CT		Old Siemens Somatom CT format (DR3)	Y	Y	TBD
Siemens – Magnetom Vision	IMA	See "Siemens – Magnetom Vision"	Y	Y	TBD
Siemens MicroCAT	LOG, CT	See also "MICRO-CAT"	Y	Y	
SPM	SPM	TBD	Y	Y	
STL	STLA, STLB	See also "STL (ASCII and Binary)" on page 655	Y	Y	

Table 1. Image file formats supported by MIPAV (continued)

File format	Extencion	Supports	Open	Save	Does not support
Truevision Graphics Adapter	TGA, VST, VDA, ICB, TPIC	Encodes index color images in an uncompressed TGA palette format; encodes other images in an uncompressed 24-bit TGA format. Decodes black and white, palette, true color uncompressed images. Decodes 16-bit, 24-bit, and 32-bit true color variants (can include loading of alpha channel).	Y	Y	N/A
Washington University OSM dataset Structure	WU	TBD	Y	Y	TBD
Common raster file formats					
Adobe Photoshop	PSD	Encoding of bitmap, grayscale, RGB, and index mode images. Also supports the encoding of RLE compression for all image variants. Decoding of Adobe Photoshop v 2.5 images (and above) with the following modes: bitmap, grayscale, palette, and RGB.	Y	Y	Alpha saving
BIORAD	PIC	See “BIORAD”	Y	Y	TBD

Table 1. Image file formats supported by MIPAV (continued)

File format	Extencion	Supports	Open	Save	Does not support
Graphics Interchange File	GIF	Decoding of interlaced images and images with a transparency information set. See also "Graphics Interchange Format (GIF) files"	Y	N	Encoding of a GIF file Decoding of multiframe GIFs using the multiframe load mechanism
Joint Photographics Experts Group	JPEG, JPG, JFIF, JFI	See "Joint Photographic Experts Group (JPEG/JFIF) files"	Y	Y	TBD
Microsoft Windows Bitmap	BMP, DIB	Encoding of Index Color Model images as uncompressed BMP palette formats. Other Color Model images are encoded as uncompressed 24-bit BMP formatted files. Decoding of BMP version 2.x through 4.x files. Also supports the encoding of OS/2 variant files. Supports the encoding of true color and palette images, as well as Run-Length Encoding 8 (RLE8) and RLE4 images. See "Microsoft Windows Bitmap (BMP) files"	Y	Y	Decoding of old versions of BMP files (prior to version 2.x)
Microsoft Windows Cursor	CUR		Y	Y	
Microsoft Windows icon	ICO		Y	Y	
PC Paintbrush File Format	PCX, DCX, PCC	Encoding of PCX files including uncompressed and RLE-compressed support. Full support for decoding of PCX files. See "PC Paintbrush (PCX) files"	Y	Y	N/A

Table 1. Image file formats supported by MIPAV (continued)

File format	Extencion	Supports	Open	Save	Does not support
Portable Network Graphic Format	PNG	Encodes index images as palette PNG images. Also encodes grayscale formatted images (which include grayscale and direct color) with bit masks saved in grayscale file format. Direct color and RGB are encoded as RGB images with an alpha mask (if mask is available from the image). 16-bit depth images are truncated to 8-bit depth when the image is loaded. Decoding of palette, grayscale, and true color images. Also supports the transparency chunk for palette based images.	Y	Y	Some extension chunks (such as ALPHA)
Sun Raster	RS, RAS	Encodes 8-bit index color images with or without RLE compression or RGB format for other color models. Decoding of 4, 8, 16, 24, and 32-bit images. Supports the decoding of the old, standard, byte encoded, RGB, TIFF, and IFF format type. See "Sun Raster (RS) files"			Decoding of RAW color map or experimenta l type images
TARGA	TGA	See also "TARGA" on page 656	Y	Y	
Tag Image File Format	TIF, TIFF	See also "Tag Image File Format (TIFF) files"	Y	Y	Tiled
X BitMap	XBM	Encoding of 2-color XBM images. Fully supports the decoding of XBM formatted files.			N/A

Table 1. Image file formats supported by MIPAV (continued)

File format	Extencion	Supports	Open	Save	Does not support
X PixMap	XPM	Encoding of XPM palette images. Fully supports the decoding of XPM formatted files.	Y	Y	Decoding of image files with 3 or more characters per pixel/ voxel
End of common raster file formats					
Surface description file formats					
ASCII text	TXT		Y	Y	
ASCII Poly	POLY	See also “POLY”	Y	Y	
ASCII PLY	PLY	See also “Polygon File Format” on page 652			
ASCII STL	STL	See also “STL (ASCII and Binary)”	Y	Y	
Binary STL	STLB	See also “STL (ASCII and Binary)”	Y	Y	
Legacy VTK	VTK	See also “Legacy VTK”	Y	Y	
MIPAV TXT	TXT	See also “MIPAV text file format” on page 648	Y	Y	
MIPAV surface files	SUR	See also “MIPAV (SUR) files”	Y	Y	
VRML	WRL	See also “VRML”	Y	Y	
VTK XML	VTP	See also “VTK XML surface”	Y	Y	
XML surface	XML	See also “XML surface”	Y	Y	

1. MIPAV does not save extra information with the AFNI file. It only does the “bare minimum.”
2. AVIs are saved with encoding options: run-length, MPEG, MPEG-4, QuickTime. However, MIPAV requires the Java Media Framework, which is installed with MIPAV.

MIPAV XML

MIPAV XML is a medical image format used in MIPAV. MIPAV XML provides MIPAV users with a common XML framework that standardizes the creation, processing, and interchange of document metadata across different medical image analysis workflow.

MIPAV XML stores image data in two files - an XML file and `*.raw` file. The XML file contains a number of image attributes, including items such as file name, ID, number of dimensions, resolution, endianess, orientation, compression, and also the special attribute containing the image pixel data. The `*.raw` file contains the image data.

MIPAV Image Schema

A MIPAV XML file must be formed according to the MIPAV Image Schema. MIPAV Image Schema and Schema Definition (XSD) are available on the MIPAV web site. For more information, refer to Figure 1 and MIPAV web site http://mipav.cit.nih.gov/development.php

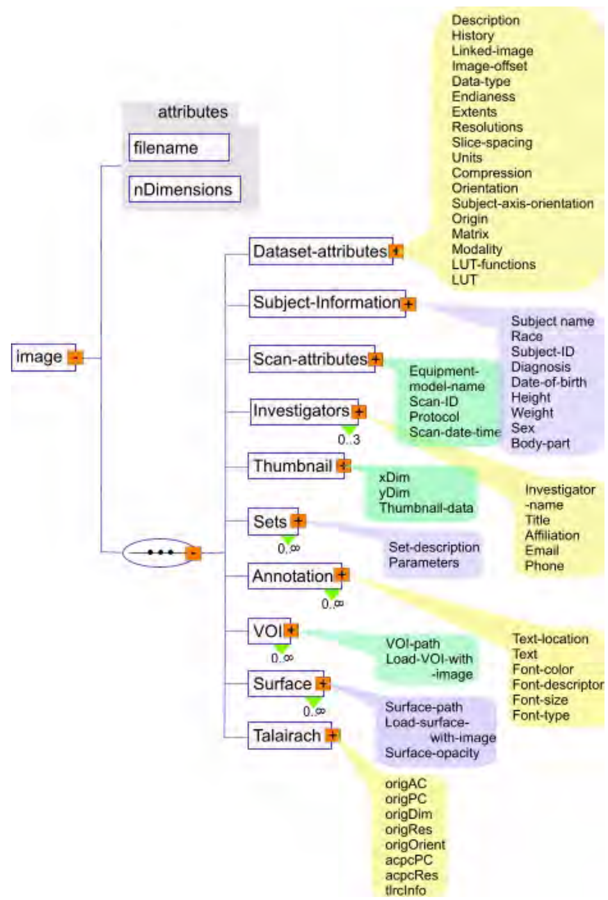


Figure 1. MIPAV Image Schema definition <http://mipav.cit.nih.gov/documentation/xml-format/image/>. MIPAV Image Schema can also be found here: <http://mipav.cit.nih.gov/documentation/xml-format/image/image.xsd>

ADVANTAGES OF MIPAV XML FORMAT

- MIPAV XML can not only be used to describe the image as a whole, e.g. file name, dimensionality, subject information, scan attributes, modality, resolution, etc., but it also stores the image pixel information, VOIs, LUTs, surfaces, and Talairach info associated with the image, as well as tags defining structural divisions of the image.
- This architecture makes it feasible to save as much as possible image information in a simple format that can be viewed and edited in any web browser or text editor (e.g. WordPad or Notepad).

- It also permits the image created/modified from an original image to retain the metadata associated with its source.
- MIPAV allows a user to choose what information should be saved when he/she saves an image to XML format. E.g., when a user saves a DICOM image as XML, the software provides he/she with the dialog box, where one can select which DICOM tags to save to an XML file. This can also be used to anonymize DICOM images.

Other formats supported by MIPAV

Adobe Photoshop (PSD) files

PSD is a bitmap file format created by Adobe. It can accommodate images with an unlimited amount of colors. PSD files are not multiple image files; the maximum image size is 30K x 30K pixels or voxels. PSD files are generally used to store images that were altered or manipulated by the Adobe Photoshop software. The header is embedded in the file and is 26 bytes in length. It contains information such as the height and width of the image and the color mode.

Analyze files

Analyze formatted files are generated by a UNIX-based, image-processing application developed at the Mayo Clinic. Analyze file format supports:

- 1 bit (packed binary)
- 8 bit (8 bits per pixel or voxel [unsigned byte])
- 16 bit (16 bits per pixel or voxel [signed short])
- 32 bit (32 bits per pixel or voxel [signed integers, or floating point])
- 64 bit (64 bits per pixel or voxel [doubles, or complex])
- 24 bit (RGB, 8 bits per channel red, green, blue)

Analyze images are formed from two files:

- *Header file*—This file describes the image type, size, and other important image attributes.

- *File Containing the Actual Image Data*—This file can be interpreted as a raw file since it does not have any header information within the file.

These files have the same name and are distinguished by the extensions `HDR` for the header file and `IMG` for image file. For example, `brain.hdr` is the header file that describes the image file and `brain.img` is the file that contains the image data. In support of some NIH intramural researchers who have used Analyze in the past and have generated a large number of legacy datasets, MIPAV reads and writes Analyze formatted images. See also "Analyze filesets" on page 671.

AFNI

The HEAD file for an AFNI dataset is in ASCII, so you can view it with a normal text editor (e.g., Notepad). The data within are organized into *attributes*, which are named arrays of floats, integers, or characters (strings). A sample float attribute is shown in Table 2. This array defines the voxel array dimensions. An example of a character array attribute is shown in Table 3.

```
type = integer-attribute
name = DATASET_DIMENSIONS
count = 5
256 256 124 0 0
```

Table 2. Voxel array dimensions

```
type = string-attribute
name = IDCODE_DATE
count = 25
'Mon Apr 15 13:08:36 1996~
```

Table 3. A character array attribute

Note that the data for a string attribute starts with a single apostrophe (') character, and that the number of bytes is given by the count parameter. The ASCII NUL character is replaced by the tilde ~ so that the HEAD file can be edited manually, if need be. On input, tildes will be replaced with NULs.

The dataset reading code requires a minimal set of attributes to be able to decipher a dataset; for example, the DATASET_DIMENSIONS attribute above is required. There are also some attributes that are not mandatory for a dataset to be successfully constructed from a HEAD file; for example, the IDCODE_DATE attribute above is not required. Attributes that AFNI doesn't know how to deal with are ignored. This makes it possible for a program to tuck extra information into the HEAD file and not cause a trouble.

A BRIK file contains nothing but voxel values. If a dataset has 1 sub-brick, which is 100x100x100 voxels, and the values stored are shorts (2 bytes each), then the BRIK file will have exactly 2,000,000 bytes. All the formatting information is stored in the HEAD file.

Note, this information is taken from the AFNI web page <http://afni.nimh.nih.gov/afni>. For more information, refer to the AFNI web site.

Audio Video Interleave (AVI) files

AVI is the Microsoft Video for Windows standard. AVI is a form of the Resource Interchange File Format (RIFF). In this file format, video and audio data are stored consecutively in an AVI file. The AVI file contains a 4-byte file header, followed by list information and then alternating video and audio streams.

BIORAD

Each Biorad confocal image file consists of three parts. They are 1) a 76 byte header which contains information such as the number of images in the file and how large each section is, 2) the images themselves and 3) notes after the images. Each set of BIORAD images from one set of raw data, is stored in a separate folder with the name provided. Within that folder is a sub-folder named `/Raw Data/` where the files are stored. A single image file is called `raw.pic`. In the multiple probe Z or T series, the files are named `raw01.pic`, `raw02.pic`, etc. There are may be also text files stored in the same folder, which contain information about the imaging parameters used.

BRUKER DATA FORMAT

Bruker format stores a single scanning session in its own directory. The directory is named according to the subject name or number, as typed in by the scanner operator. The directory name usually specifies the subject name/number and which session this is for that subject. For more information, refer to <http://imaging.mrc-cbu.cam.ac.uk/imaging/FormatBruker>.

CHESCHIRE (IMG OR IMC)

TBD.

DIGITAL IMAGING AND COMMUNICATIONS IN MEDICINE (DICOM) FILES

The DICOM format is a standard that defines a standard method of communication between two devices, such as computers, servers, or imaging devices. DICOM not only prescribes how to communicate with other imaging equipment and databases but also specifies how images are stored. Each image has accompanying header information describing the image format (i.e., height, width, etc.) as well as information that indirectly relates to the image (i.e., patient information, image equipment setup parameters).

See also "Working with DICOM Images" on page 148.

FITS

FITS stands for "Flexible Image Transport System" and is the standard astronomical data format endorsed by both NASA and the IAU. FITS is primarily designed to store scientific data sets consisting of multi-dimensional arrays (1-D spectra, 2-D images or 3-D data cubes) and 2-dimensional tables containing rows and columns of data. For more information about FITS refer to <http://heasarc.gsfc.nasa.gov/docs/heasarc/fits.html>.

FreeSurfer image and surface files

FreeSurfer uses two main kinds of data. The first one is a volume of voxels, as for example, from an MRI scanner. It provides a source of raw input. The other one is surface data which consist of lists of vertices as well as their positions in space and faces and also vertices associated with them.

GE – Genesis 5X and LX

Refer to <http://www.genesismedicalimaging.com/mobiles/mobilemri.html>.

Graphics Interchange Format (GIF) files

The Graphics Interchange Format (GIF) is a bitmap file format that was created by CompuServe, Inc. GIF is primarily an exchange and storage file format; GIFs can be used to store one or several bitmap images in one file. GIFs support pixel or voxel depths of 1 to 8 bits. The maximum image size is 64K x 64K pixel or voxels. There are two major revisions of the GIF format specification: GIF87a and GIF89a. Both formats begin with a 6-byte header that identifies the file format as GIF.

ICS (Image Cytometry Standard)

The Image Cytometry Standard (ICS) is a digital multidimensional image file format used in life sciences microscopy. It stores the image data and the microscopic parameters describing the optics during the acquisition. The newest ICS2 file format uses a single `ICS` file with both the header and the data together.

The ICS format is capable to store:

- multidimensional and multichannel data
- images in 8, 16 or 32 bit integer, 32 or 64 bit floating point and floating point complex data
- all microscopic parameters directly relevant to the image formation
- free-form comments

For more information, refer to <http://en.wikipedia.org/wiki/Image_Cytometry_Standard>.

Joint Photographic Experts Group (JPEG/JFIF) files

The JPEG File Interchange Format (JFIF) allows files containing JPEG-encoded data streams to be exchanged. Technically, JPEG refers to the compression type and the Joint Photographic Experts Group standards organization. However, the term *JPEG* is usually used to indicate the file format. JPEG file formats are bitmap files that are primarily used in image and graphics manipulation programs. Created by C-Cube Microsystems, the JPEG or JFIF file format does not accommodate multiple images per file and the maximum image size allowed is 64K by 64K pixel or voxels. JPEG files can accommodate 24-bit color images. Generally, JPEG header information appears between the Start of Image (SOI) and Application (APP0) markers.

Laser Scanning Microscope (Zeiss)

Refer to <http://en.wikipedia.org/wiki/LSM_(Zeiss)>

Legacy VTK

The legacy VTK file format is a surface format that consist of five basic parts. For more information about VTK file formats, refer to <http://

www.vtk.org/pdf/file-formats.pdf>.

1 The file version and identifier, it contains the single line: `# vtk DataFile Version x.x`. This line must be exactly as shown with the exception of the version number `x.x`, which will vary with different releases of VTK.

Note: the current version number is 3.0. Version 1.0 and 2.0 files are compatible with version 3.0 files.

2 The header, it consists of a character string terminated by end-of-line character `\n`. The header is 256 characters maximum. The header can be used to describe the data and include any other pertinent information.

3 The file format, which describes the type of file, either ASCII or binary. On this line the single word `ASCII` or `BINARY` must appear.

4 The dataset structure that can have the geometry section and actual data. The geometry section describes the geometry and topology of the dataset. This part begins with a line containing the keyword `DATASET` followed by a keyword describing the type of dataset. Then, depending upon the type of dataset, other keyword/data combinations define the actual data.

5 The dataset attributes. It begins with the keywords `POINT_DATA` or `CELL_DATA`, followed by an integer number specifying the number of points or cells, respectively. Other keyword/data combinations then define the actual dataset attribute values (i.e., scalars, vectors, tensors, normals, texture coordinates, or field data).

LIFF

Layered Image File Format is a file format used in the Openlab suite for microscope image processing. It is a proprietary format, but has an open, extensible form analogous to TIFF. It was specifically designed to contain a large number of high resolution images, and also all of the meta data generated by analysis of such images.

MACINTOSH PICT

The Macintosh PICT (Macintosh Picture) format is format for the Macintosh. It also known as PICT, Macintosh Picture, or QuickDraw Picture Format. All Macintosh PICT files start with a 512-byte header, which contains information that the Macintosh uses to keep track of the file. This is followed by three fields describing the image size (picSize), the image frame (picFrame), and a version number. For more information, refer to the Macintosh PICT File Format Summary page at <http://www.fileformat.info/format/macpict/egff.htm>.

MEDICAL IMAGE NETCDF (MINC) FILES

MINC is a medical-imaging file format that is based on the Network Common Data Form (NetCDF) file format. NetCDF is a platform-independent software interface that provides a means for storing named, multidimensional variables. Each multidimensional variable is defined by a name, dimensions, and attributes. For example, cardio (name) [256, 256] (dimensions), and "long_name" where this attribute is a string that describes the content of the image. MINC provides a standard for dimension, variable, and attribute names suitable for medical imaging. MINC also provides convenience functions to complement the NetCDF interface and convenience functions for using MINC files. See also the MINC web page <http://www.bic.mni.mcgill.ca/software/minc/>.

MEDICAL RESEARCH COUNCIL (MRC)

The MRC file format is a file format for electron density has become industry standard in Cryo-electron microscopy where the result of the technique is a three-dimensional grid of voxels each with a value corresponding to density of electrons. The MRC format is supported by almost every molecular graphics suite that supports volumetric data.

MICRO-CAT

See also <http://www.medical.siemens.com/siemens/en_US/rg_marcom_FBAs/files/Press_Releases/Siemens_Says/JuneSiemensSays.pdf>.

MICROSOFT WINDOWS BITMAP (BMP) FILES

BMP is a native bitmap format for the Microsoft Windows and OS/2 platforms. Developed by the Microsoft Corporation, it can be read on Intel machines running Microsoft Windows, Windows XP, Windows NT, Windows 95, OS/2, and MS-DOS. It is an uncompressed file format with a maximum image size of 32K x 32K or 2G x 2G pixels or voxels (depending on the version of BMP).

Depending on the version of BMP, for OS/2 the maximum image size is larger, 64K x 64K or 4G x 4G pixels or voxels. There are several versions of BMP for Microsoft Windows and OS/2. MIPAV supports Microsoft Window BMP versions 2.x and above and all OS/2 versions of the BMP file format. BMP version 2.x is designed for use with the Microsoft Windows 2.x platform. It has a 14-byte header (as does the OS/2 1.x bitmap header). It can accommodate images with 1-, 4-, 8-, or 24-bit colors. Version 3.x is designed for use with the Microsoft Windows 3.x and Windows NT platforms. Like version 2.x, it contains a 14-byte header. It also contains an additional bitmap header that is 40 bytes in size. The Microsoft Windows 3.x platform version accommodates images with 1-, 4-, 8- or 24-bit colors. Windows NT and Windows XP accommodate 16- and 32-bit images. Version 4.x was designed for use with Microsoft Windows 95. It contains the 14-byte header, and an additional 108-byte bitmap header. It can accommodate images of 1, 4, 8, 16, and 32 bits.

MIPAV (LUT) FILES

LUT is a vector file format that is used to store lookup table data. A sample file appears in "Understanding MIPAV-related files" , "LUT file" on page 662.

MIPAV (MTX) FILES

For a sample MTX file refer to “Matrix file” on page 664.

MIPAV (PLT) FILES

PLT is a vector file format that is used to store graphics data. For a sample PLT file refer to “Plot file” on page 666.

MIPAV TEXT FILE FORMAT

MIPAV defined ASCII text surface format. The text surface file records the triangle mesh vertices, normals and connectivity.

Vertices
x, y, z // vertex position
Normal
x, y, z // normal coordinate
Connectivity
x, y, z // index connection

MIPAV (SUR) FILES

Surface type (Triangle mesh or ClodMesh - Continuous level of details mesh)

Inverse Dicom matrix flag
3D mesh direction x, y, z
Start location x, y, z
Bounding box x, y, z
Inverse Dicom matrix
Vertex quantity
Vertex position x, y, z
Vertex normal x, y, z
Index quantity
Index connectivity x, y, z

Surface material property
Clodmesh collapse record

MIPAV (VOI) FILES

VOI is a vector file format that is used to store volume of interest contouring information. For a sample file refer to "VOI file" on page 668.

MIPAV XML

Refer to "MIPAV XML" on page 636.

MIPAV XML SURFACE FORMAT

See also "XML surface" on page 657.

NIFTI

NifTy-1 is a new Analyze-style data format, proposed by the NIfTI DFWG. NIfTI-1 was adapted from ANALYZE™ 7.5 file format. It uses the "empty space" in the ANALYZE 7.5 header to add several new features, which are listed below:

- Affine coordinate definitions relating voxel index (i,j,k) to spatial location (x,y,z);
- Codes to indicate spatio-temporal slice ordering for FMRI;
- "Complete" set of 8-128 bit data types;
- Standardized way to store vector-valued datasets over 1-4 dimensional domains;
- Codes to indicate data "meaning";
- A standardized way to add "extension" data to the header;
- Dual file (`hdr` & `img`) or single file (`nii`) storage;

For more information, refer to the NIfTI web site at <http://nifti.nimh.nih.gov/>.

PC Paintbrush (PCX) files

PC Paintbrush is a bitmap file format that is used primarily in Microsoft Windows and other Windows-based products. PCX is mainly used as an exchange and storage format. Created by ZSoft and packaged with Microsoft Windows, PCX can accommodate 1-, 4-, 8-, and 24-bit color images. It is uncompressed; the maximum image size is 64K x 64K pixel/voxels. Header information is embedded in the file; the first 128 pixel/voxels in the file contain information such as the PCX ID number, the bits per pixel/voxel, and the palette type.

PICT files

See also "Macintosh PICT" on page 646.

Phillips PAR/REC

See also <http://www.dclunie.com/medical-image-faq/html/part8.html>

and MIPAV User Guide, Volume 2, Algorithms "DTI Create List File".

POLY

The common *poly* file is represented by a printable ASCII file. It contains a number of features combined in three sections: a vertex list, a polygon list, and a surface list.

Each of lists is represented by a sequence of non-empty lines, called a section. The first line of a section is called its title.

The title of a section contains its name optionally followed by the attribute list enclosed in parentheses. Several attributes in the list are separated by +. Both the name and the attributes are built only of alphanumeric characters (letters, digits, underscore). By convention, data section names are written in upper case.

Below, is the description of POLY format originally written in 1993 by Pat

Flynn and revised 1/96 PJF. For more information refer to:

- http://sampl.ece.ohio-state.edu/data/3DDB/Models/polyformat.txt
- http://www.eg-models.de/formats/Format_Poly.html

Vertex list

Each vertex appears on its own line.

The format is `pnt %d: %f %f %f'.

Points are numbered starting at zero. There is no sentinel indicating the end of the points and the beginning of the polygons; your program logic can easily detect it.

Polygon list

Each polygon appears on its own line.

The format is `poly %d: [%d] %d %d %d .... ' The number on the left of the colon is the index of the polygon (numbered from zero). The number inside the [square brackets] is the number of vertices. The remaining numbers are the indices of the vertices. The vertices proceed clockwise around the polygon as you look at it from `outside' the object.

Surface list

The surface list allows to logically group polygons.

The format for a surface entry is:

`surf %d: [%d] %d'

The first number is the index of the surface (numbered from zero). The second number [inside square brackets] is the number of polygons belonging to that surface. The third number is the index of **the first polygon in the group**.

The polygons grouped into a surface must occupy a contiguous sub-list of the polygon list.

POLYGON FILE FORMAT

Also known as Stanford Triangle Format.

File Structure

- Header
- Vertex List
- Face List
- (lists of other elements)
-
-

File Format (ASCII description)

```
format ascii 1.0            { ascii/binary, format version number }
comment made by Greg Turk  { comments keyword specified, like all lines }
element vertex 8            { define "vertex" element, 8 of them in file }
property float x            { vertex contains float "x" coordinate }
property float y            { y coordinate is also a vertex property }
property float z            { z coordinate, too }
element face 6              { there are 6 "face" elements in the file }
property list uchar int vertex_index { "vertex_indices" is a list of ints }
end_header                  { delimits the end of the header }
0 0 0                       { start of vertex list }
0 0 1
0 1 1
0 1 0
1 0 0
1 0 1
1 1 1
1 1 0
3 1 2 3                     { start of face list }
3 6 5 4
3 4 5 1
3 5 6 2
3 6 7 3
3 7 4 0
```

PORTABLE NETWORK GRAPHIC FORMAT (PNG) FILES

PNG is a bitmap file format that is generally used to transmit and store network image data. The PNG format can store images with up to 16-bits (grayscale) or 48-bits (truecolor) per pixel/voxel. The maximum image size is 2G x 2G pixel/voxels. PNG is a compressed format. Multiple images cannot be stored in PNG. An 8-byte identification signature is followed by a header chunk, which contains basic information about the image data. The header chunk data area is 13 bytes in length.

QuickTime–Apple

The QuickTime file is a container file that contains one or more tracks, where track stores a particular type of data e.g., audio, video, effects, or text. Each track either contains a digitally-encoded media stream (using a specific codec) or a data reference to the media stream located in another file. Tracks are maintained in a hierarchal data structure consisting of objects called *atoms*. An atom can be a parent to other atoms or it can contain media or edit data, but it cannot do both.

See also: <http://en.wikipedia.org/wiki/QuickTime#QuickTime_file_format>

Raw data files

MIPAV supports reading and writing Raw image data of all the basic programming types (i.e., boolean, byte, short, etc.). Raw images have no header or a header of known length and unknown content at a fixed location at the beginning of the file. You must specify basic information about the Raw image before it is loaded in MIPAV. When loading a RAW data set, you must specify the image type, dimension and resolution, units, and header offset.

Siemens – Magnetom Vision

Refer to <http://www.dclunie.com/medical-image-faq/html/part4.html#MagnetomVision>.

STL (ASCII and Binary)

STL is a file format initially designed for the CAD software created by 3D Systems. The STL format specifies both ASCII and binary representations. An STL file describes an unstructured triangulated surface by the unit normal and vertices, which are ordered by the right-hand rule, using a three-dimensional Cartesian coordinate system. It describes only the surface geometry of a three dimensional object without any representation of color, texture or attributes.

See also <http://rpdrc.ic.polyu.edu.hk/old_files/stl_ascii_format.htm>.

Sun Raster (RS) files

RS files is the native bitmap format for Sun Microsystems UNIX platforms that run the SunOS operating system. This format stores bitmap data (color, grayscale, black and white) of any pixel/voxel depth. RS files can be of any size; however, multiple images per file are not supported. The header is embedded in the file; it is 32 bytes in the length and contains typical header information such as the width and height of the image and the type of color map used.

Tag Image File Format (TIFF) files

TIFF is a bitmap file format that is generally used to provide a portable, image-storage mechanism that describes image data. Multiple images can be stored in one file. TIFF files can accommodate a maximum image size of $2^{32}-1$ pixel/voxels. It is widely used and is a standard file format used in many desktop publishing, imaging, and paint programs. MIPAV supports most TIFF 6.0 formatted files used commonly in the image research community including 2D and 3D monochrome 8-bit, signed and unsigned 16-bit, and signed 32-bit images. Future support of 24-bit color and compressed images is planned. The TIFF Image File Header (IFH) is 8 bytes in length and contains 3 fields of information. If multiple images are in a file, an IFH is present for each image in the file.

TARGA

TGA or TARGA format is a format for describing bitmap images, it is capable of representing bitmaps ranging from black and white, indexed color, and RGB color. See also "Creating TGA Image files" written by Paul Burke.

Truevision Graphics Adapter (TGA) files

TGA is a bitmap file format that is often used in graphics, imaging, and paint applications that store up to 32-bit color images (8-, 16-, 24-, and 32-bit colors are supported). Created by Truevision, Incorporated, it is an uncompressed file format that does not support multiple images per file. The maximum image size however, is unlimited. TGA's header is 18 bytes in length and contains traditional header information including the depth of the color map entries.

VRML

The Virtual Reality Modeling Language (VRML) is a file format for describing interactive 3D objects and worlds. It is based upon the OpenInventor file format which was originally developed by Silicon Graphics, Inc. For more information and format specifications, refer to <http://www.web3d.org/x3d/specifications/vrml/vrml97/index.htm>

VTK XML surface

See VTK documentation available at <http://www.vtk.org/pdf/file-formats.pdf>.

X BITMAP (XBM) FILES

XBM is a monochrome bitmap file format that is used primarily for the storage of icon and cursor bitmaps in X Windows. It was developed by X Consortium. Because this format was developed for small amounts of data, the bitmap images are composed of collections of ASCII data rather than binary data. (XBM bitmap data is often found in C source header files.) Multiple images can be stored on a file; there is no limit to the image size. XBM has no header file, nor is a formal header embedded in the program; rather header information consists of four lines that begin with `#define`. The four lines of code indicate the height and width of the image and the coordinate of the hotspot (if any).

XML

Extensible Markup Language (XML) is a very flexible markup language format derived from SGML. However, XML was originally designed primarily for large-scale electronic publishing, now it is playing an significant role in the exchange of a wide variety of data including image processing data. For more information about XML, refer to <http://en.wikipedia.org/wiki/XML>.

MIPAV has it's own XML version – MIPAV XML, which is explained in Section "MIPAV XML" on page 636.

XML SURFACE

MIPAV defined XML surface format based on the MIPAV "surface.xsd" and "surfaceref.xsd" file. The XML surface file defines specific variables for reading and writing surface.xml files.

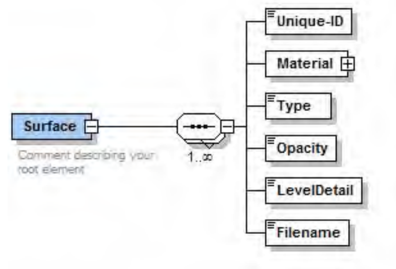


Figure 2. Surfaceref.xsd: surfaceref schema file set the reference to MIPAV .sur file

Surface.xsd: surface schema set the triangle mesh surface attributes in detail.

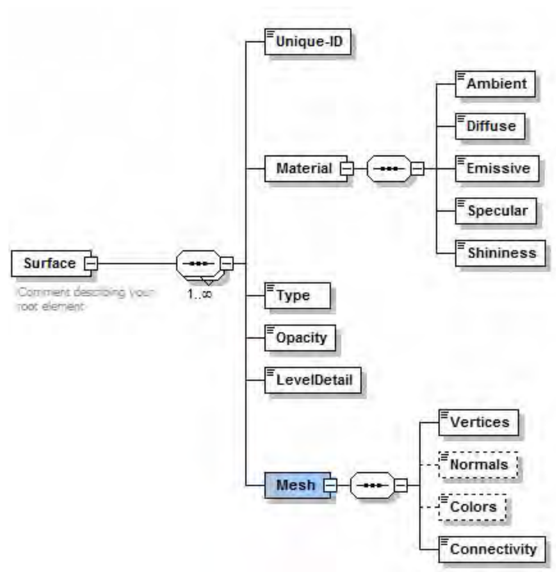


Figure 3. Surface.xsd: surface schema set the triangle mesh surface attributes in detail

X PIXMAP (XPM) FILES

XPM is a bitmap file format developed by Groupe Bull. It is used primarily to store X Windows pixmap information. XPM can store black and white, grayscale or color image data of unlimited colors. Like XBM, there is no limit for the image size and there can be multiple images per file. In addition, like XBM, XPM is written in ASCII. It can contain a section, `<Values>`, that contains header information, such as the height and width of the pixmap and the number of characters per pixel or voxel.

Understanding MIPAV-related files

This section includes examples of the following files: preference, LUT, IMG, matrix, plot, VOI, VOI XML, LAX, and Analyze.

Preference file

What is its purpose?

The preference file (mipav.preference) contains either the default or customized settings for running MIPAV. These settings determine whether some features appear or how certain features look.

You can customize the settings in the file by selecting Help > Program Options in the MIPAV window. For example, you can change whether the splash screen appears when you start the program, whether the scripting toolbar appears, the color of the crosshair cursor, whether the log mode is turn on, and so on.

When and how is it created?

MIPAV generates a preference file for all users when the program is run for the first time. Usually, the first settings recorded in the preference file are the memory setting, or maximum heap size, and the default image directory. When you first customize one or more of the settings in the Program Options dialog box, MIPAV adds those settings to the preference file as well as the default settings for those features you did not change. If you later make changes to those same preferences, the program adds those new settings to the preference file. In other words, the initial preference file is very small. It grows in size as you customize the program with the latest changes appearing last in the file.

WHERE IS IT STORED?

By default, MIPAV stores the preference file in your home directory.

Table 4. Default home directories

For . . .	The home directory is . . .
Window users	C:\Documents and Settings\userID
Unix users	
MacIntosh users	

However, if MIPAV is installed on a network and several users access it remotely, preference files are stored in the network home directory.

Note: The system administrator for the network may select another directory on the network in which to store proference files.

What does it look like? An example of the preference file appears in Figure 4. For ease of discussion, line numbers appear on the left of each line in the figure, although they don't appear in the actual preference file.

Note: Your preference file may not have the same ordering of information as in Figure 4; as you customize MIPAV, your preference file changes to reflect the additional information.

```
1  # MIPAV preference file
2  #Mon Mar 19 15:12:48 EST 2001
3  LightBoxGridRow=2
4  LightBoxBorderSize=3
5  EnableDICOMReceiver=true
6  TRIM=0.4
7  DEBUG=true
8  PlugInFile3=SpreadSheet
9  PlugInFile2=GraphPanel
10 PlugInFile1=Chart
11 Swing=true
12 LightBoxRowDependant=true
13 PlugInView1=DrawTest
14 PlugInAlgorithm1=GraphApplet
15 Server1=MRIPS;MARS;137.187.26.152;104;
16 Storage1=MIPAV;MIPAV;C\:\\images;3100
17 SplashGraphics=no
18 LightBoxGridColor=000000
19 LightBoxBorderColor=960000
20 LightBoxGridSize=5
21 LightBoxMagnification=45.0
22 LightBoxGridCol=2
23 ImageDirectory=D\:\\Java\\jdk1.3\\demo\\applets\\Sp
24 readSheet LightBoxLocation=813,23
```

1-2	Indicate the type of file and the date and time of the last update (comment lines).
3-4	Indicate default settings the light box image window.
5	Shows DICOM information and indicates whether the DICOM Receiver is on by default. If the DICOM Receiver is on, you can receive or retrieve DICOM image files.
6	Indicates the default trim parameter. The trim parameter is discussed in detail in Chapter 7.
7	Indicates whether MIPAV is placed in debug mode. If MIPAV is placed in debug mode and technical problems occur while you use the software, error messages are logged in the Output window.
11	Indicates whether the Swing-generated graphical user interface (GUI) is displayed. If the preference is set to false, the Advanced Windows Toolkit (AWT)-generated GUI appears instead.
12	Indicates default settings the light box image window.
13-14	Shows plug-in file information.

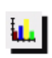
Figure 4. Example of MIPAV.preference file

15-16	Shows DICOM information and shows hosts table entries. (The hosts table is located in the DICOM Communication Panel window.) Line 15 shows the server (device) entry. Line 16 shows a storage destination entry, which indicates where image files are stored when they are received or retrieved. If you indicate more than one storage destination entry, more lines appear in the preference file. For more information on MIPAV's DICOM capabilities, refer to "Working with DICOM Images" on page 148.
16	Shows DICOM information.
17	Indicates whether an introductory splash screen appears when you start MIPAV.
18-22	Indicates default settings the lightbox image window.
23	Shows the image directory.

Figure 4. Example of MIPAV.preference file (continued)

LUT file

What is its purpose? The lookup table (LUT) file holds lookup table values, which you can modify, for the active image currently being displayed. **When and how is it created?** MIPAV generates the file when, after opening an image, you do the following:

1.  select the Lookup table icon, to call the Lookup Table window and display a lookup table for the image. See Figure 5.
2. Save the Lookup table by selecting LUT > Save or pressing Ctrl+S.

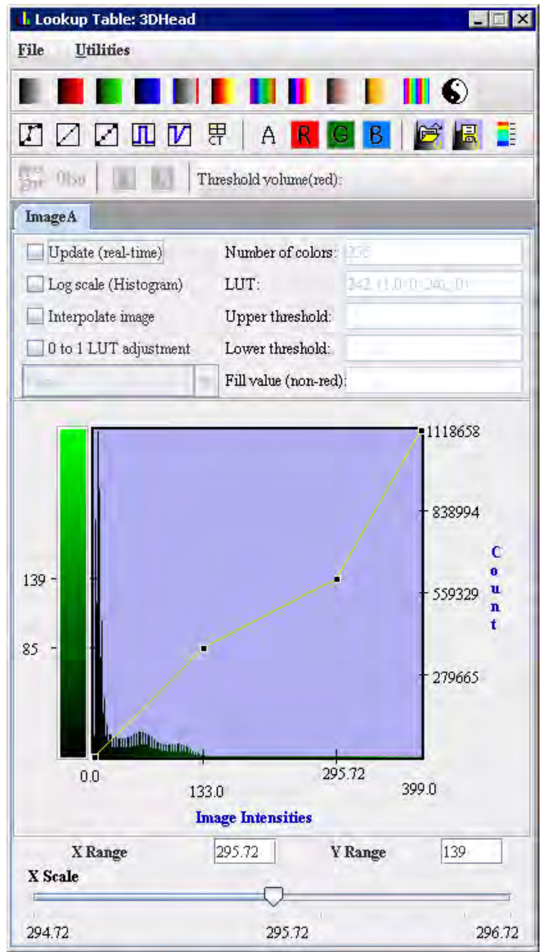


Figure 5. Lookup Table window

Where is it stored? By default, the LUT file is stored in the same directory that holds the image.

What does it look like? Figure 6 is an example of a LUT file. The middle section of the file was removed in this example because of space limitations, and the column headings—alpha, red, green, and blue—were added to make it is easier to interpret the values. The column headings do not appear in the actual LUT file.

```
                Alpha        Red          Green        Blue
<LUT>
256         # Size of LUT Arrays
0     1.0   0.0   0.0   0.0
1     1.0   0.996093750.996093750.99609375
2     1.0   1.99218751.99218751.9921875
3     1.0   2.98828122.98828122.9882812
4     1.0   3.9843753.9843753.984375
5     1.0   4.98046884.98046884.9804688
6     1.0   5.97656255.97656255.9765625
7     1.0   6.97265626.97265626.9726562
8     1.0   7.968757.968757.96875
9     1.0   8.9648448.9648448.964844
10    1.0   9.96093759.96093759.9609375
11    1.0   10.95703110.95703110.957031
12    1.0   11.95312511.95312511.953125
13    1.0   12.94921912.94921912.949219
14    1.0   13.945312513.945312513.9453125

       [This part of the LUT file was removed to save space.]

251   1.0   250.01953250.01953250.01953
252   1.0   251.01562251.01562251.01562
253   1.0   252.01172252.01172252.01172
254   1.0   253.00781253.00781253.00781
255   1.0   254.0039254.0039254.0039
```

Figure 6. Sample LUT file (bold column headings added to make it easier to interpret the values)

Matrix file

What is it and what is its purpose? The matrix file provides identity information for an image. It describes the image's orientation, translations, offset scales, rotations, and shears. When any of this information changes in the image, the identity matrix changes.

When and how is it created? Every image has an identity matrix. The matrix changes when you apply those algorithms or utilities to the image that change any of its identity information.

What does it look like? . If the image is currently open and active, you can view its matrix information by selecting Image > Attributes > View Header in the MIPAV window. The Information dialog box (Figure 7) opens.

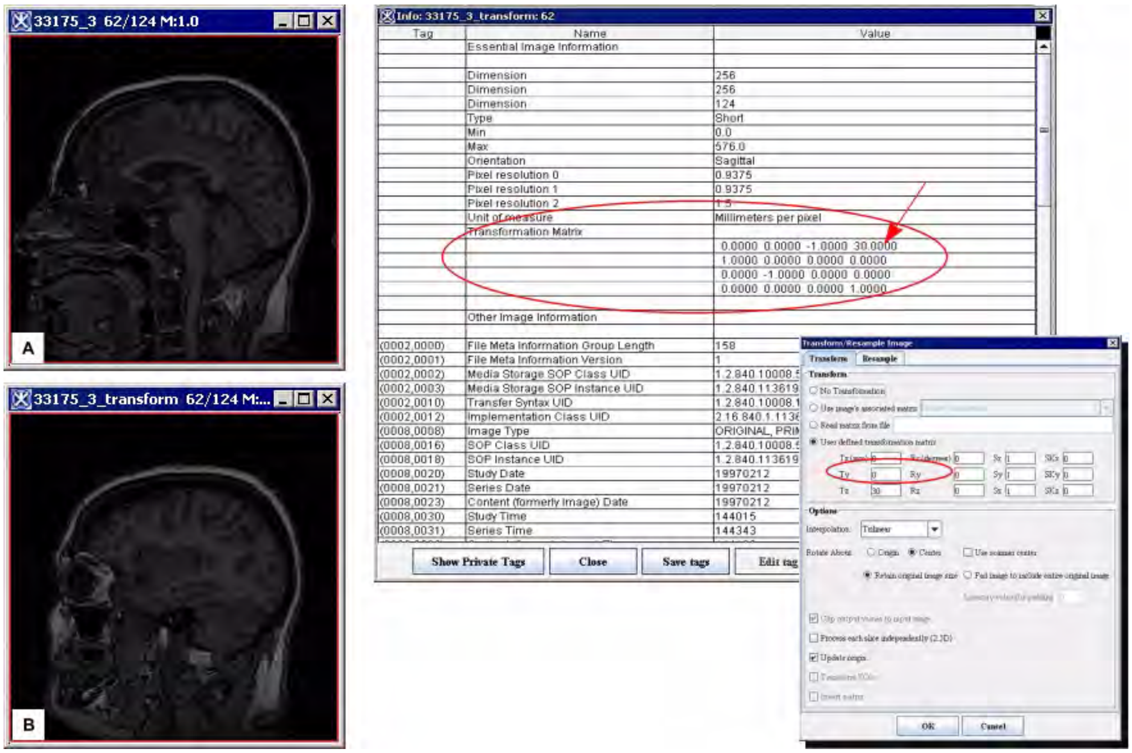


Figure 7. The original image (A) and transformed image (B). The Info dialog box displays the transformation matrix for the image when the Transform dialog box shows settings which determined that transformation

Where is it stored, TBD? MIPAV stores matrix files in the same directory as its associated image.

Plot file

What is it and what is its purpose? The plot (PLT) file contains plot data for an intensity profile, or graph, generated for an image.

When and how is it created? MIPAV generates the plot file when you delineate a VOI on an image and then do the following:

1. Select one of the following in the MIPAV window:
   - VOI > Graph > Boundary Intensity
   - VOI > Graph > 2.5D Total Intensity
   - VOI > Graph > 2.5D Average Intensity
2. Select File > Save Graph or press Ctrl+S in either the Contour VOI Graph window or Intensity Graph window.

What does it look like? An example of a plot file appears in Figure 8, and Figure 9 shows a graph for this plot file.

The graph has five functions. Each function has 32 points. This file is read vertically. The first two columns show the *x* and *y* values for the first function. The next two columns show the *x* and *y* values for the second function, and so on. The first point of the first function is (1, 8), the second is (2, 9), and the third is (3, 18).

The first point for the fifth function is (1, 22400), the second is (2, 21700), the third is (3, 21000), and so on. The graph for this plot file appears in Figure 9.

```
18  1       30729  1      100    1      1000   1      224002
9   2       27878  2      200    2      2000   2      217003
18  3       25209  3      300    3      3000   3      210004
41  4       22716  4      400    4      4000   4      20300
584 5       20393  5      500    5      5000   5      19600
61536       18234  6      600    6      6000   6      18900
72547       16233  7      700    7      7000   7      18200
83938       14384  8      800    8      8000   8      17500
95769       12681  9      900    9      9000   9      16800
1080910     11118  10     1000   10     10000  10     16100
11109811    9689   11     1100   11     11000  11     15400
12144912    8388   12     1200   12     12000  12     14700
13186813    7209   13     1300   13     13000  13     14000

     [This part of the LUT file was removed to save
space.]

271823427   153    27     2600   27     27000  27     4200
282039328   84     28     2700   28     28000  28     3500
292271629   41     29     2800   29     29000  29     2800
302520930   18     30     2900   30     30000  30     2100
312787831   8      31     3000   32     32000  31     1400
323072932   9      32     3100   33     33000  32     700
```

Figure 8. Example of a plot (.PLT) file

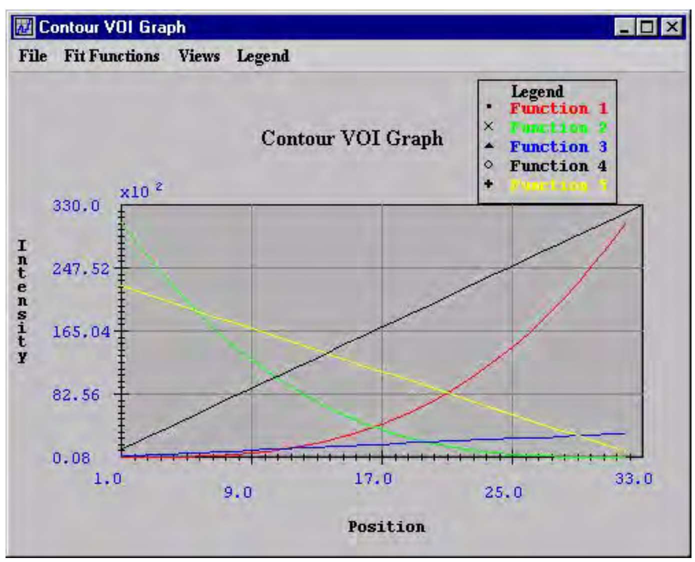


Figure 9. Intensity profile (graph) example

Where is it stored? By default, the PLT file is stored in the same directory that holds the image. However, you can store it in any directory of your choice.

VOI file

What is it and what is its purpose? The VOI file provides plots for contours that you have delineated on an image. Saving VOI information to a file means that it can be used again. In addition, if you create a script that includes algorithms that require VOIs, you need to save the VOIs so that the script can apply them to the images (refer to "Using Scripts (Macros) in MIPAV").

What does it look like? Figure 10 shows an example of a VOI file.

```
MIPAV VOI FILE
0                # curveType of the VOI
255              # color of VOI - red component
0                # color of VOI - green component
0                # color of VOI - blue component
255              # color of VOI - alpha component
1                # number of slices for the VOI
8                # slice number
1                # number of contours in slice
38               # number of pts in contour
167.0 124.0
161.0 147.0
156.0 156.0
150.0 162.0
144.0 166.0
139.0 170.0
133.0 172.0
128.0 174.0
122.0 175.0
116.0 176.0
111.0 176.0
105.0 175.0
139.0 78.0
144.0 82.0
150.0 86.0
156.0 92.0
161.0 101.0
24862902         # unique ID of the VOI
```

Figure 10. An example of a VOI file

When and how is it created? MIPAV creates a VOI file when you create one or more VOIs on an image and then select one of the following commands in the MIPAV window:

- VOI > Save VOI
- VOI > Save VOI as
- VOI > Save All VOIs
- VOI > Save All VOIs to

Because MIPAV VOI file format is a simple ASCII format, it can be easily adapted to other formats. The contours are given as simple closed polygons. One cross-section may contain several polygons. A contour can reside inside another contour. They are oriented so the inside of a contour (material) is on the right side. The *xy* coordinates range from 0 to 512.

Where is it stored? Although by default it is stored in same directory as the image, you can save it to another directory by selecting VOI > Save VOI as or VOI > Save All VOIs to, which open the Save window.

VOI XML FILE

What is it and what is its purpose? The VOI XML (`.xml`) file provides plots for contours in an XML format.

Where is it stored? Like VOI files, VOI XML files are stored, by default, in the same directory as the image. However, you can save them to another directory by selecting either VOI > Save VOI as or VOI > Save All VOIs to.

When and how is it created? You can save VOIs as VOI files or VOI XML files. To do so, you delineate one or more VOIs on an image and then select VOI > Save VOI as or VOI > Save All VOIs to. In the Save dialog box, you simply type the file name with a `.xml` extension and click Save.

What does it look like? An example of a VOI XML file appears in Figure 11. The middle part of the file was removed to save space.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <!--
 MIPAV VOI file
-->
- <VOI xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Unique-ID>24862902</Unique-ID>
<Curve-type>0</Curve-type>
<Color>255,255,0,0</Color>
- <Contour>
<Slice-number>8</Slice-number>
<Pt>167.0,124.0</Pt>
<Pt>161.0,147.0</Pt>
<Pt>144.0,166.0</Pt>
<Pt>139.0,170.0</Pt>
<Pt>133.0,172.0</Pt>
<Pt>128.0,174.0</Pt>

[This part of the LUT file was removed to save space.]

<Pt>144.0,82.0</Pt>
<Pt>150.0,86.0</Pt>
<Pt>156.0,92.0</Pt>
<Pt>161.0,101.0</Pt>
</Contour>
- <Contour>
<Slice-number>8</Slice-number>
<Pt>159.0,55.0</Pt>
<Pt>190.0,55.0</Pt>
<Pt>190.0,85.0</Pt>
<Pt>159.0,85.0</Pt>
</Contour>
</VOI>
```

Figure 11. An example of a VOI XML file

. MIPAV Surface file

See MIPAV API at <http://mipav.cit.nih.gov/documentation/api/gov/nih/mipav/view/renderer/J3D/model/file/package-summary.html>.

MIPAV LaunchAnywhere executable (LAX) file

What is it and what is its purpose? The LAX file is a LaunchAnywhere Executable (LAX) file that contains MIPAV's runtime environmental options. It may be helpful to access the file if you need to contact the software developers during the debugging process. Instructions on how to do this appear in Figure 8.

LaunchAnywhere, which is created by Macrovizion, Inc., is a commercial program that is used to install and start the MIPAV software on your system.

When and how is it created? The runtime environmental options in the LAX file are set during the development of the MIPAV software.

If you install MIPAV on any Windows, UNIX, or MAC OS operating system, the LaunchAnywhere program copies a LaunchAnywhere Executable (LAX) file to the same directory as your other MIPAV software files.

Analyze filesets

What are they and what are its purpose? Analyze image file sets consist of an image file and a header file. A color lookup file is sometimes included in the file set. Each type of file in the file set has a specific extension:

- **A .img extension—**signifies an image file. Example: `file.img`.
- **A .hdr extension—**Signifies a header file. Example: `file.hdr`.
- **A .lkup extension**—Indicates a lookup table file. Example: `file.ikup`.

When and how are they created? MIPAV could save images with `*.img` extension either in Analyze format or in NIFTI format. To save an image in Analyze format, call the File > Save Image as menu, and then type the file name with `*.img` extension. In the Choose Type of File to Write dialog box that appears, specify the file format (Analyze). Check the option **Always save .img files in Analyze format**, if you want MIPAV to save `.img` files only in that format.

What do they look like? An image file in Analyze format contains uncompressed voxel data for the images in one of several possible voxel formats:

- 1 bit packed binary (slices begin on byte boundaries)
- 8 bit unsigned char (grayscale unless `.lkup` file is present)
- 16 bit signed short

- 32 bit signed integers or float
- 24 bit RGB, 8 bits per channel

The header file is a C structure that describes the dimensions and properties of the voxel data. This structure appears in Figure 12 to Figure 14.

Where are they stored? MIPAV stores files in Analyze format in the catalogue which you specified in the Save Image as dialog box.

For more information about Analyze format refer to “Analyze files” on page 639.

```
/*
 *
 * (c) Copyright, 1986-1995
 * Biomedical Imaging Resource
 * Mayo Foundation
 *
 * dbh.h
 *
 *
 * database sub-definitions
 */

struct header_key/* header_key */
    {/* off + size*/
        int sizeof_hdr; /* 0 + 4 */
        char data_type[10];/* 4 + 10 */
        char db_name[18];/* 14 + 18 */
        int extents;/* 32 + 4 */
        short int session_error;/* 36 + 2 */
        char regular;/* 38 + 1 */
        char hkey_un0;/* 39 + 1 */
    };/* total=40  */

struct image_dimension/* image_dimension  */
    {/* off + size*/
        short int dim[8];/* 0 + 16    */
        char vox_units[4];/* 16 + 4    */
        char cal_units[8];/* 20 + 4    */
        short int unused1;/* 24 + 2    */
        short int datatype;/* 30 + 2    */
        short int bitpix;/* 32 + 2    */
        short int dim_un0;/* 34 + 2    */
        float pixdim[8];/* 36 + 32   */
                        /*
                                pixdim[] specifies the voxel
dimensions:
                                pixdim[1] - voxel width
                                pixdim[2] - voxel height
                               pixdim[3] - interslice distance
                                        ..etc
                        */
        float vox_offset;/* 68 + 4    */
        float funused1;/* 72 + 4    */
        float funused2;/* 76 + 4    */
        float funused3;/* 80 + 4    */
        float cal_max;/* 84 + 4    */
        float cal_min;/* 88 + 4    */
        int compressed;/* 92 + 4    */
        int verified;/* 96 + 4    */
        int glmax, glmin;/* 100 + 8   */
    };/* total=108 */
```

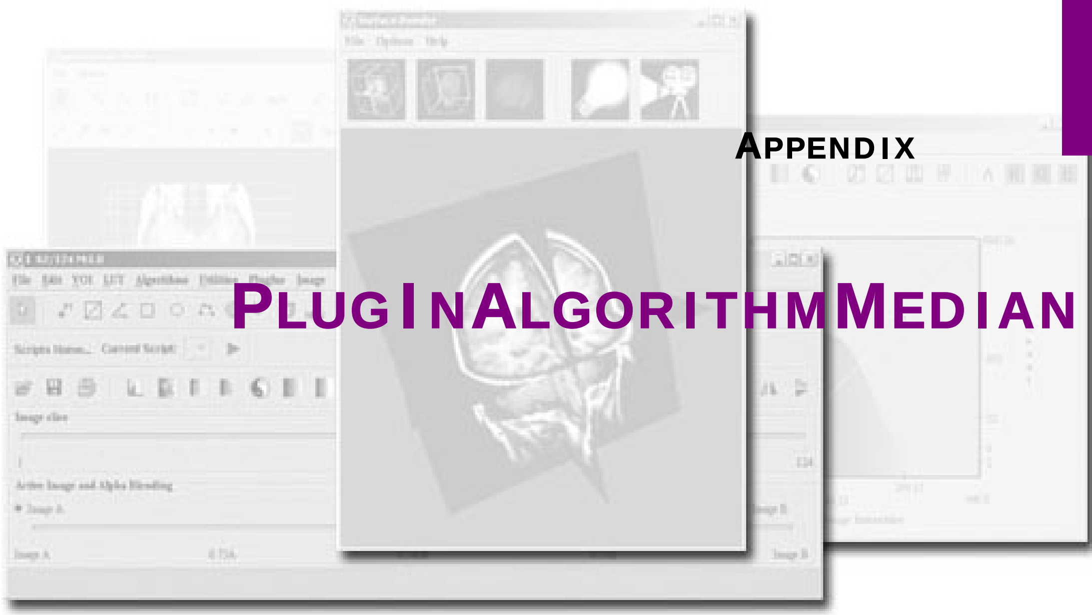
Figure 12. Analyze file, page 1

```
struct data_history/* data_history     */
    {/* off + size*/
        char descrip[80];/* 0 + 80    */
        char aux_file[24];/* 80 + 24   */
        char orient;/* 104 + 1   */
        char originator[10];/* 105 + 10  */
        char generated[10];/* 115 + 10  */
        char scannum[10];/* 125 + 10  */
        char patient_id[10];/* 135 + 10  */
        char exp_date[10];/* 145 + 10  */
        char exp_time[10];/* 155 + 10  */
        char hist_un0[3];/* 165 + 3   */
        int views;/* 168 + 4   */
        int vols_added;/* 172 + 4   */
        int start_field;/* 176 + 4   */
        int field_skip;/* 180 + 4   */
        int omax,omin;/* 184 + 8   */
        int smax,smin;/* 192 + 8   */
    };/* total=200 */
struct dsr/* dsr */
    {/* off + size*/
        struct header_key hk;/* 0 + 40    */
        struct image_dimension dime;/* 40 + 108  */
        struct data_history hist;/* 148 + 200 */
    };/* total=348 */


Comments:
        struct header_key
                int sizeof_header/* must indicate size of
header file */
                int extants;/* should be 16384 */
                char regular;/* 'r' */


        struct image_dimension struct decribes the
organization and
        side of images. These elements enable IO routines to
reference
        images by volume and slice number.

                short int dim[]/* array of image dimensions */
                        dim[0]/* number of dimensions;
usually 4 */
                        dim[1]/* image width */
                        dim[2]/* image height */
                        dim[3]/* volume depth */
                        dim[4]/* volumes in file */

                char vox_units[4]/* labels voxel spatial unit
*/
                char cal_units[4]/* labels voxel calibration
unit */
                short int datatype/* Acceptable values are */
```

Figure 13. Analyze file, page 2

```
#define DT_NONE0
#define DT_UNKNOWN0
#define DT_BINARY1
#define DT_UNSIGNED_CHAR2
#define DT_SIGNED_SHORT4
#define DT_SIGNED_INT8
#define DT_FLOAT16
#define DT_COMPLEX32
#define DT_DOUBLE64
#define DT_RGB128
#define DT_ALL255

short int bitpix/* bits per pixel */
float pixdim[] /* parallel array to dim giving voxel
dimensions in each dimension */
pixdim[1]/* voxel width */
pixdim[2]/* voxel height */
pixdim[3]/* voxel depth or slice thickness */

float vox_offset;/* byte offset in the .img file where
voxels start. A negative value specifies
that absolute value is applied to every
image in the file. */

float calibrated Max & Min /* spec. range of calibration
values */
int glmax, glmin    /* the max and min values for entire
 data set */


The data_history substructure is not required, but the
'orient' element
is used to indicate individual slice orientation and
determines whether
the ANALYZE 'Movie' program will attempt to flip the images
before
displaying a movie sequence.

orient:
                                      0 - transverse
unflipped
                                      1 - coronal unflipped
                                      2 - sagittal unflipped
                                      3 - transverse flipped
                                      4 - coronal flipped
                                      5 - sagittal flipped
```

Figure 14. Analyze file, page 3



APPENDIX D

PLUGINALGORITHMMEDIAN

PlugInAlgorithmMedian

The source code for the plugin program, PlugInAlgorithmMedian.java (Figure 1), is an example of an algorithm type of plugin. This plugin program runs a median filter on an image, using its own dialog box and implementation of the median filter.

Note: For ease of reading, comment lines in the source file appear in green type, and keywords appear in bold type.

```
// By leaving out the package keyword in this class, it is therefore in the default java
package
// for the application.

import gov.nih.mipav.model.algorithms.*;
import gov.nih.mipav.model.structures.*;
import gov.nih.mipav.view.*;

import java.io.*;
import java.util.*;
import java.awt.*;


/**
*   Example of a plugin implementation of the median filter.
*   This class creates the algorithm that runs on the image.
*   This shows how to extend the AlgorithmBase class.
*
*   Note:  The median algorithm is already implemented in
*           the MIPAV/IASO software.
*
*                  @version    July 12, 2002
*   @see        Algorithms
*   @see        AlgorithmMedian
*
*   $Logfile: /mipav/src/plugins/PlugInAlgorithmMedian.java $
*   $Revision: 7 $
*   $Date: 3/16/05 3:36p $
*
*/
public class PlugInAlgorithmMedian extends AlgorithmBase {

      private static final int       SQUARE_KERNEL      = 0;  //square kernel
      private static final int       CUBE_KERNEL        = 0;
      private static final int       CROSS_KERNEL       = 1;  // cross
      private static final int       AXIAL_KERNEL       = 1;  //
      private static final int       X_KERNEL           = 2;  // X-shaped kernel, from 1
                                                              // corner to opposite corner
      private static final int       HORZ_KERNEL        = 3;  // horizontal (2D only)
      private static final int       VERT_KERNEL        = 4;  // vertical (2D only)


      private BitSet                 mask = null;

      private int                    iterations;              // number of times to filter
                                                              // the image.
      private int                    kernelSize;              // dimension of the kernel
                                                              // (i.e., 5 = 5x5, 7 = 7x7,
                                                              // 9 = 9x9, etc.)
      private int                    kernelShape;             // user-selectable shape of
                                                              // the region for neighbor-
                                                               // selection
      private boolean                entireImage;             // true means apply to
                                                            // entire
                                                              // image, false only region
      private byte[]                 kernel;                  // mask to determine the
```

Figure 1. PlugInAlgorithmMedian.java

```
                                                    // region of pixels used in a
                                                    // median filter
        private float           stdDevLimit;        // compute median value of
                                                    // pixel if pixel magnitude is
                                                    // outside this fraction of
                                                    // the standard deviation
        private boolean         sliceFiltering;     // do all filtering slice-by-
                                                    // slice, rather than as a
                                                    // volume (applies only to
                                                    // 3D/volume images)
        private int             currentSlice = 0;//
        private int             numberOfSlices;
        private int             halfK;
        private int             kernelCenter;
        private float[]         kernelMask;
        private int             maskCenter;



        private boolean         isColorImage = false;   // indicates the image being
                                                    // messed with is a color
                                                    // image
        private int             valuesPerPixel=1;   // number of elements in a
                                                    // pixel. Monochrome = 1,
                                                    // Color = 4. (a, R, G, B)
        private boolean         rChannel = true;    // if T, filter the red
                                                    // channel
        private boolean         gChannel = true;    // the green channel
        private boolean         bChannel = true;    // the blue channel

    /**
     *          Constructor for 3D images in which changes are placed in a predetermined
     *          destination image.
     *   @param destImg             Image model where result image is to stored.
     *   @param srcImg              Source image model.
     *   @param iters               Number of iterations of the median filter.
     *   @param kSize               Kernel size: dimension of the kernel (i.e., 5 = 5x5,
     *                              7 = 7x7, 9 = 9x9, etc.).
     *   @param kShape              Kernel shape: element neighbors to include when finding
     *                              the median.
     *   @param stdDev              Inner-bounds by which to process pixels (pixel values
     *                              outside this bound will be median filtered).
     *   @param sliceBySlice        Each slice in a volume image is to be filtered separately
     *                              (when true), else the volume will use a kernel with 3
     *                              dimensions.
     *   @param maskFlag            Flag that indicates that the median filtering will be
     *                              performed for the whole image if equal to true.
     */
         public PlugInAlgorithmMedian(ModelImage destImg, ModelImage srcImg, int iters,
                                  int kSize, int kShape, float stdDev, boolean sliceBySlice,
                                  boolean maskFlag) {

         super(destImg, srcImg);
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```java
            if ( srcImg.isColorImage() ) {
                  isColorImage = true;
                  valuesPerPixel = 4;
            }
            // else, already false
            entireImage  = maskFlag;
            iterations   = iters;
            kernelSize   = kSize;                //  dimension of the kernel
            kernelShape  = kShape;               //  set up the mask (kernel) used to
                                                 //  filter

            stdDevLimit  = stdDev;               //  inside magnitude bounds of pixel
                                                 //  value to adjust
            sliceFiltering = sliceBySlice;
            numberOfSlices = srcImage.getExtents()[2];
            makeKernel();
      }

   /**
   *   Constructor for 2D images in which changes are placed in a predetermined
   *   destination image.
   *   @param destImg      Image model where result image is to stored.
   *   @param srcImg       Source image model.
   *   @param iters        Number of iterations of the median filter.
   *   @param kSize        Kernel size: dimension of the kernel (i.e., 5 = 5x5, 7 = 7x7,
   *                        9 = 9x9, etc.).
   *   @param kShape       Kernel shape: element neighbors to include when finding the
   *                        median.
   *   @param stdDe        Inner-bounds by which to process pixels (pixel values outside
   *                        this bound will be median filtered).
   *   @param maskFlag     Flag that indicates that the median filtering will be
   *                       performed for the whole image if equal to true.
   */
      public PlugInAlgorithmMedian(ModelImage destImg, ModelImage srcImg, int iters,
                         int kSize, int kShape, float stdDev, boolean maskFlag) {

        super(destImg, srcImg);
          if ( srcImg.isColorImage() ) {
                  isColorImage = true;
                  valuesPerPixel = 4;
          }
          // else, already false
          entireImage  = maskFlag;
          iterations   = iters;
          kernelSize   = kSize;   //  dimension of the kernel
          kernelShape  = kShape;  //  set up the mask (kernel) used to filter
          stdDevLimit  = stdDev;  //  inside magnitude bounds of pixel value to adjust
          sliceFiltering = true;  //  as a default--this doesn't make much sense in a 2D
                                //  application.
          numberOfSlices = 1;     //  2D images may only have 1 slice.
          makeKernel();
      }
   /**
   *   Constructor for 3D images in which changes are returned to the source image.
   *   @param srcImg       Source image model.
   *   @param iters        Number of iterations of the median filter.
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
 *   @param kSize        Kernel size: dimension of the kernel (i.e., 5 = 5x5, 7 = 7x7,
 *                        9 = 9x9, etc.).
 *   @param kShape       Kernel shape: element neighbors to include when finding the
 *                        median.
 *   @param stdDev       Inner-bounds by which to process pixels (pixel values outside
 *                        this bound will be median filtered).
 *   @param sliceBySlice Each slice in a volume image is to be filtered separately
 *                       (when
 *                         true), else the volume will use a kernel with 3 dimensions.
 *   @param maskFlag     Flag that indicates that the median filtering will be
 *                       performed for the whole image if equal to true.
 */
    public PlugInAlgorithmMedian(ModelImage srcImg, int iters, int kSize, int kShape,
                          float stdDev, boolean sliceBySlice, boolean maskFlag) {
        super(null, srcImg);
          if ( srcImg.isColorImage() ) {
                  isColorImage = true;
                  valuesPerPixel = 4;
          }
          // else, already false
          entireImage  = maskFlag;
          iterations   = iters;
          kernelSize   = kSize;   //  dimension of the kernel
          kernelShape  = kShape;  //  set up the mask (kernel) used to filter
          stdDevLimit  = stdDev;  //  inside magnitude bounds of pixel value to adjust
          sliceFiltering = sliceBySlice;
          numberOfSlices = srcImage.getExtents()[2];
          makeKernel();
      }

/**
 *   Constructor for 2D images in which changes are returned to the source image.
 *   @param srcImg        Source image model.
 *   @param iters        Number of iterations of the median filter.
 *   @param kSize        Kernel size: dimension of the kernel (i.e., 5 = 5x5, 7 = 7x7,
 *                        9 = 9x9, etc.).
 *   @param kShape     Kernel shape: element neighbors to include when finding the
 *                        median.
 *   @param stdDev       Inner-bounds by which to process pixels (pixel values outside
 *                        this bound will be median filtered).
 *   @param maskFlag     Flag that indicates that the median filtering will be
 *                       performed for the whole image if equal to true.
 */
    public PlugInAlgorithmMedian(ModelImage srcImg, int iters, int kSize, int kShape,
                               float stdDev, boolean maskFlag) {
        super(null, srcImg);
          if ( srcImg.isColorImage() ) {
                  isColorImage = true;
                  valuesPerPixel = 4;
          }
          // else, already false
          entireImage  = maskFlag;
          iterations   = iters;
          kernelSize   = kSize;   //  dimension of the kernel
          kernelShape  = kShape;  //  set up the mask (kernel) used to filter
          stdDevLimit  = stdDev;  //  inside magnitude bounds of pixel value to adjust
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
            sliceFiltering = true;  //  as a default--though a different value doesn't
                                    //  make much sense in a 2D application.
                                      //  (calculates makeKernel() )
            numberOfSlices = 1;     //  2D images may only have 1 slice.
            makeKernel();
        }

    /**
    *   RGB images are median filtered by "channel."  That is, each color,
    *   red, blue and green, is filtered independently of the other two colors.
    *   This median filter permits selectively filtering any combination of the
    *   three channels instead of simply filtering all three. True for any of
    *   the arguments enforces filtering that channel.
    *   @param rFilter red channel.
    *   @param gFilter green channel.
    *   @param bFilter blue channel.
    */
        public void setRGBChannelFilter(boolean r, boolean g, boolean b) {
            if (isColorImage) {        // just in case somebody called for a mono image
                rChannel = r;
                gChannel = g;
                bChannel = b;
            }
        }

    /**
    *      Prepares this class for destruction.
    */
        public void finalize(){
            destImage   = null;
            srcImage    = null;
            super.finalize();
        }

    /**
    *    Constructs a string of the construction parameters and outputs the string to the
    *    message frame if the logging procedure is turned on.
    */
     private void constructLog() {
          historyString = new String( "Median(" +
                                         String.valueOf(kernelShape) + ", " +
                                         String.valueOf(kernelSize)  + ", " +
                                         String.valueOf(iterations)  + ", " +
                                         String.valueOf(entireImage) + ")\n");
      }

    /**
    *   Starts the algorithm.
    */
        public void runAlgorithm() {

          if (srcImage  == null) {
              displayError("Source Image is null");
              return;
          }

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
        constructLog();
        if (destImage != null){     // if there exists a destination image
            if (srcImage.getNDims() == 2){
                 calcStoreInDest2D();
              }
              else if (srcImage.getNDims() > 2) {
                  calcStoreInDest3D();
              }
        }
        else {                      // there is no image but the original source.
            if (srcImage.getNDims() == 2){
                 calcInPlace2D();
              }
              else if (srcImage.getNDims() > 2) {
                  calcInPlace3D();
              }
        }
    } // end runAlgorithm()

/**
*   Median filters the source image.  Replaces the original image with the filtered
*   image.
*/
    private void calcInPlace2D(){

        int length;                     // total number of data-elements (pixels) in image
        float buffer[];             // data-buffer (for pixel data) which is the "heart"
                                // of the image
        float resultBuffer[];       // copy-to buffer (for pixel data) for image-data
                                // after filtering

        try {
            if (!isColorImage) {
                // image length is length in 2 dims
                length = srcImage.getExtents()[0]
                        *srcImage.getExtents()[1];
            }
       else {  // if (isColorImage) {
                // image length is length in 2 dims
                // by 4 color elements per pixel
                length = srcImage.getExtents()[0]
                        *srcImage.getExtents()[1]
                        *4; // 1 each for ARGB
            }
            buffer          = new float[length];
            resultBuffer    = new float[length];
            srcImage.exportData(0,length, buffer); // locks and releases lock
        }
        catch (IOException error) {
            buffer = null;
            resultBuffer = null;
            errorCleanUp("Algorithm Median: source image locked", true);
            return;
        }
            catch (OutOfMemoryError e){
            buffer = null;
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
            resultBuffer = null;
            errorCleanUp("Algorithm Median reports: Out of memory when creating image
                    buffer", true);
            return;
        }

        this.buildProgressBar();                      // let user know what is going on
        this.sliceFilter(buffer, resultBuffer, 0, "image");   // filter this slice
        disposeProgressBar();                               // filtering work should be done.

        if (threadStopped) {
        finalize();
        return;
        }

        try {   // place buffer data into the image
            srcImage.importData(0, resultBuffer, true);
        }
        catch (IOException error) {
            buffer = null;
            resultBuffer = null;
            errorCleanUp("Algorithm Median: Source image locked", true);
            return;
        }

        setCompleted(true);
    }

/**
*   Median filters the source image and replaces the source image with the median
*   filtered image.
*/
    private void calcInPlace3D(){

        int imageSliceLength = srcImage.getExtents()[0]*srcImage.getExtents()[1];
        int length;
        float buffer[];
        float resultBuffer[];
    try {
            if (!isColorImage) {
                // image length is length in 3 dims
                length = srcImage.getExtents()[0]
                        *srcImage.getExtents()[1]
                        *srcImage.getExtents()[2];
            }
    else {  // if (isColorImage) {
                // image length is length in 3 dims
                // by 4 color elements per pixel
                length = srcImage.getExtents()[0]
                        *srcImage.getExtents()[1]
                        *srcImage.getExtents()[2]
                        *4; // 1 each for ARGB
            }
            buffer          = new float[length];
            resultBuffer    = new float[length];
            srcImage.exportData(0,length, buffer); // locks and releases lock
            this.buildProgressBar();
            }
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
                            catch (IOException error) {
                buffer = null;
                resultBuffer = null;
                errorCleanUp("Algorithm Median: Source image locked", true);
                return;
            }
            catch (OutOfMemoryError e){
                buffer       = null;
                resultBuffer = null;
                errorCleanUp("Algorithm Median: Out of memory", true);
                return;
            }

            if (sliceFiltering){
                for ( currentSlice = 0; currentSlice < numberOfSlices && !threadStopped;
                        currentSlice++){
                    sliceFilter(buffer, resultBuffer, currentSlice*imageSliceLength,
                                "slice " + String.valueOf(currentSlice+1));
                }
            }
            else {          // volume kernel requested
                volumeFilter(buffer, resultBuffer);
            }

            if (threadStopped) {
                  finalize();
                  return;
            }

            try {
                srcImage.importData(0, resultBuffer, true);
            }
            catch (IOException error) {
                buffer = null;
                resultBuffer = null;
                errorCleanUp("Algorithm Median: source image locked", true);
                setThreadStopped(true);
                return;
            }

            progressBar.dispose();
            setCompleted(true);
        }

    /**
    *     This function produces a new image that has been median filtered and places
    *     filtered image in the destination image.
    */
        private void calcStoreInDest2D(){

            int length;                  // total number of data-elements (pixels) in
                                     // image
            float buffer[];               // data-buffer (for pixel data) which is the
                                     // "heart" of
                                          // the image
            float resultBuffer[];         // copy-to buffer (for pixel data) for image data
                                     // after filtering

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
    try { destImage.setLock(ModelStorageBase.RW_LOCKED); }
           catch (IOException error){
               errorCleanUp("Algorithm Median reports: destination image locked", false);
               return;
           }
           try {
               if (!isColorImage) {
                   // image length is length in 2 dims
                   length = srcImage.getExtents()[0]
                           *srcImage.getExtents()[1];
               }
               else {  // if (isColorImage) {
                   // image length is length in 2 dims
                   // by 4 color elements per pixel
                   length = srcImage.getExtents()[0]
                           *srcImage.getExtents()[1]
                           *4; // 1 each for ARGB
               }
               buffer       = new float[length];
               resultBuffer = new float[length];
               srcImage.exportData(0,length, buffer); // locks and releases lock
           }
           catch (IOException error) {
               buffer = null;
               resultBuffer = null;
               errorCleanUp("Algorithm Median reports: source image locked", true);
               return;
           }
    catch (OutOfMemoryError e){
               buffer = null;
               resultBuffer = null;
               errorCleanUp("Algorithm Median reports: out of memory", true);
               return;
           }

           this.buildProgressBar();
           sliceFilter(buffer, resultBuffer, 0, "image");    // filter image based on provided
                                                        // info
           destImage.releaseLock();                     // we didn't want to allow the image
                                                           // to be adjusted by someone else
           progressBar.dispose();
           if (threadStopped) {
           finalize();
           return;
           }

           try {                                        // but now place buffer data into the
                                                 // image
               destImage.importData(0, resultBuffer, true);
           }
           catch (IOException error) {
               buffer = null;
               resultBuffer = null;
               errorCleanUp("Algorithm Median reports: destination image still locked",
                      true);
               return;
           }

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
        setCompleted(true);
    }

/**
*   This function produces a new volume image that has been median filtered.
*   Image can be filtered by filtering each slice individually, or by filtering using
*   a kernel-volume.
*/
    private void calcStoreInDest3D(){

        int length;
        int imageSliceLength = valuesPerPixel *
             srcImage.getExtents()[0]*srcImage.getExtents()[1]; // cover case of color image
       float buffer[];
        float resultBuffer[];

        try { destImage.setLock(ModelStorageBase.RW_LOCKED); }
        catch (IOException error){
            errorCleanUp("Algorithm Median reports: destination image locked", false);
            return;
       }
        try {
            if (!isColorImage) {
                // image length is length in 3 dims
                length = srcImage.getExtents()[0]
                        *srcImage.getExtents()[1]
                        *srcImage.getExtents()[2];
            }
            else {  // if (isColorImage) {
                // image length is length in 3 dims
                // by 4 color elements per pixel
                length = srcImage.getExtents()[0]
                        *srcImage.getExtents()[1]
                        *srcImage.getExtents()[2]
                        *4; // 1 each for ARGB
            }
            buffer = new float[length];
            srcImage.exportData(0,length, buffer); // locks and releases lock
            this.buildProgressBar();
        }
        catch (IOException error) {
            buffer = null;
            resultBuffer = null;
            errorCleanUp("Algorithm Median: source image locked", true);
            return;
        }
        catch (OutOfMemoryError e){
         buffer = null;
            resultBuffer = null;
            errorCleanUp("Algorithm Median: Out of memory creating process buffer",
                   true);
            return;
        }

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
        try { resultBuffer = new float[length];}
        catch(OutOfMemoryError e){
            buffer = null;
            resultBuffer = null;
            errorCleanUp(“Algorithm Median reports: Out of memory because of
                    resultBuffer", true);
            return;
        }

        if (sliceFiltering){
            for ( currentSlice = 0; currentSlice < numberOfSlices && !threadStopped;
                    currentSlice++) {
                sliceFilter(buffer, resultBuffer, currentSlice*imageSliceLength,
                            "slice "+String.valueOf(currentSlice+1));
            }
        }
        else {          // requested volume filter
            if (isColorImage)   // for color image
                volumeColorFilter(buffer, resultBuffer);
            else                // for mono image
               volumeFilter(buffer, resultBuffer);
        }

        destImage.releaseLock();

        if (threadStopped) {
        finalize();
        return;
        }

        try{destImage.importData(0, resultBuffer, true);}
        catch  (IOException e)
        {
            buffer = null;
            resultBuffer = null;
            errorCleanUp(“Algorithm Median reports: destination image still locked”,
                   true);
            return;
        }
        progressBar.dispose();
        setCompleted(true);
    }

  /**
  *   Allows a single slice to be filtered.  Note that a progressBar must be created
  *   first.
  *   @param  srcBuffer           Source buffer.
  *   @param  destBuffer          Destination Buffer.
  *   @param  bufferStartingPoint Starting point for the buffer.
  *   @param  msgString           A text message that can be displayed as a message text
  *                               in the progressBar.
  */
    private final void sliceFilter(float srcBuffer[],
                                   float destBuffer[],
                                   int bufferStartingPoint,
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
                                 String msgString)  {
        int i, a, pass;                                  // counting....   i is the offset
                                                         // from the bufferStartingPoint
        // a adds support for 3D filtering by counting is as the pixel at the starting
             point plus the counter offset
        int buffStart = bufferStartingPoint;         // data element at the buffer. a =
                                                       // bufferStartingPoint+i
        int sliceLength = srcImage.getSliceSize();
        int imageSliceLength = sliceLength * valuesPerPixel;  // since there are 4 values
                                                       // for every color pixel.
        int kCenter = maskCenter;                    // to find the middle pixel of the
                                                // kernel-mask
        int width =  srcImage.getExtents()[0];       // width of slice in number of pixels (
        int height = srcImage.getExtents()[1];      // height of slice in number of pixels
        int sliceWidth =  width * valuesPerPixel;    // width of slice, which, in color
                                                // images is (4*width)
        int sliceHeight = height;                    // height of image, which, actually
                                                     // doesn't change
initialIndex = 0;                        // first element is alpha

        float tempBuffer[];

        float average;            // arithmetic mean
        float sigma;              // standard deviation

        float maskedList[];                              // list of buffer-values that were
                                                         // showing inside the mask

        int row, col;                                    // row and column vars for easier
                                                         // reading [(0,0) is in the top-
                                                  // left corner]
        int mod;                                         // 1% length of slice for percent
                                                         // complete

        // these bounds "frame" the interior of the slice which may be filtered
        // (&adjusted);
        // image outside the frame may not
        int upperBound, lowerBound,      // bounds on the row
            leftBound, rightBound;       // bounds on the column

        if (isColorImage) {
            upperBound = halfK;
            leftBound  = halfK*4;
            lowerBound = sliceHeight - halfK - 1;
            rightBound = sliceWidth - halfK*4 - 1;

            // data element at the buffer (a = i+bufferStartingPoint) must start on an
          // alpha value
           buffStart = bufferStartingPoint - bufferStartingPoint%4;  // & no effect if
                                                     // bufferStartingPoint%4 == 0 !!!

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
    // copy all alpha values in this slice
                setCopyColorText("alpha");
                for (a = buffStart, i = 0; i < imageSliceLength; a+=4, i+=4) {
                    destBuffer[a] = srcBuffer[a];       // copy alpha;
                }

            }
            else {      // monochrome image
                upperBound = leftBound = halfK;
                rightBound = sliceWidth - halfK - 1;
                lowerBound = sliceHeight - halfK -1;
            }
            mod = (imageSliceLength*numberOfSlices)/100;  // mod is 1 percent of length of
                                                          // slice + the number of slices.

            BitSet mask = srcImage.generateVOIMask();

            for (pass = 0; pass < iterations && !threadStopped; pass++) {
                a = buffStart;                              // set/reset a to address pixels
                                                            // from the beginning of this
                                                            // buffer.
                if (isColorImage) {                         // color image dealt with in
                                                            // special way
                                                           // choose i so proper colors go
                                                           // copy only needed RGB values
                    initialIndex = 0;                       // start with alpha on each pass
                                                            // (routine moved so we don't do
                                                            // it for each pass)
                    while (initialIndex < 3 && !threadStopped) {   // alpha:0, R:1, G:2,
                                                            // B:3.  But alpha must
                                                            // be copied
                        ++initialIndex;                     // next initial index
                        a += initialIndex;                  // keep the pixel location up with
                                                            // color indexed to

                        if (numberOfSlices > 1 && pBarVisible == true) {   // 3D image update
                                                            // progressBar
                                                           // do a progress bar update
                                       progressBar.setValue(Math.round
                                (( (   (float)(3*currentSlice*iterations + 3*pass +
                                          (initialIndex - 1))/(3*iterations*numberOfSlices))*
                                           100)));
                        }

                        if (!rChannel && initialIndex==1) {
                          // when looking at the image reds but we're not filtering the red channel
                            // copy all red values
                            setCopyColorText("red");
                            for (i = initialIndex; i < imageSliceLength; a+= 4, i+=4) {
                                destBuffer[a] = srcBuffer[a];
                            }
                        }
    else if (!gChannel && initialIndex==2) {
                            // when looking at the image greens but we're not filtering the
                          // greens channel

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
// copy all greens values
                        setCopyColorText("green");
                        for (i = initialIndex; i < imageSliceLength; a+=4, i+=4) {
                            destBuffer[a] = srcBuffer[a];
                        }
                    }
                    else if (!bChannel && initialIndex==3) {
                        // when looking at the image blues but we're not filtering the
                      // blues channel
                        // copy all blue values
                        setCopyColorText("blue");
                        for (i = initialIndex; i < imageSliceLength; a+=4, i+=4) {
                            destBuffer[a] = srcBuffer[a];
                        }
                    }
                    else {
                        if (pBarVisible == true) {
                            progressBar.setMessage("Filtering "+ msgString +" (pass "+
                              String.valueOf(pass+1) +" of "+ iterations +") ...");
                        }
                        // if we needed to filter the image, we dropped through the
                      // selection to filter the
                        // color given by ints initialIndex
                        for (i = initialIndex; i < imageSliceLength && !threadStopped;
                                     a+=4, i+=4){
                            if (numberOfSlices == 1) {   // 2D image  update progressBar
                                if (i%mod == 0 && pBarVisible == true) {
                                    progressBar.setValue(Math.round
                                    ( (float)(3*(pass*sliceLength) + (initialIndex-
                                            1)*sliceLength + i/4)/
                                            (3*iterations*(sliceLength-1))*100) );
                                }
                            }
 if (entireImage == true || mask.get(a/4) ) {// may have problems
                                                            // in masking ...
                             row = i/sliceWidth;
                                col = i%sliceWidth;
                                if ( (row < upperBound) || (row > lowerBound) ) {
                                    destBuffer[a] = srcBuffer[a]; // row too far up or
                                                       // down--out of bounds
                                }
                                else if ((col < leftBound) || (col > rightBound)) {
                                    destBuffer[a] = srcBuffer[a]; // column too far left
                                                       // or right--out of bounds
                                }
                                else {          // in bounds
                                    maskedList = getNeighborList(a, srcBuffer, true);
                                    // verify that this element is an outlier
                                    if (stdDevLimit == 0.0) {   // anything is an outlier
                                        shellSort(maskedList);
                                        destBuffer[a] = median(maskedList);
                                    }
                                else {      // look for outlierness
                                    average = mean(maskedList);
                                    sigma = standardDeviation(maskedList, average);
                                    if ((maskedList[kCenter] > (average +
                                          stdDevLimit*sigma)) ||
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
                                                (maskedList[kCenter] < (average -
                                         stdDevLimit*sigma))) {
                                                 shellSort(maskedList);
                                                 destBuffer[a] = median(maskedList);
                                             }
                                       else {  // if element was not an outlier, pixel is fine.
                                             destBuffer[a] = srcBuffer[a];
                                         }
                                     }
                                 }
                            }
                            else {     // not part of the VOI so just copy this into the
destination
                                   // buffer.
                                destBuffer[a] = srcBuffer[a];
                            }
                        }
                    }
                    a = buffStart;                 // reset the index back to the beginning of
                                                // the filterarea
                }
            }
            else {          // monochrome image
                if (pBarVisible) {
                    progressBar.setMessage("Filtering "+ msgString +" (pass "+
                                       String.valueOf(pass+1) +" of "+ iterations +") ...");
                    if (numberOfSlices > 1) {    // 3D image update progressBar
                        // do a progress bar update
                        progressBar.setValue(Math.round
                            (( (  (float)(currentSlice*iterations + pass)/
                                     (iterations*numberOfSlices))* 100)));
                    }
                }
                for ( i = 0; i < imageSliceLength && !threadStopped; i++){
                    if (numberOfSlices == 1) {   // 2D image update progressBar
                        if (i%mod == 0 && pBarVisible == true) {
                            progressBar.setValue(Math.round
                            ( (  (float)( (pass*imageSliceLength)+i)
                            /(iterations*(imageSliceLength -1))*100)) );
                        }
                    }
                    if (entireImage == true || mask.get(a) ) {
                        // Median stuff here
                        row = i/width;
                        col = i%width;
                        if ( (row < upperBound) || (row > lowerBound) ) {
                            destBuffer[a] = srcBuffer[a]; // row too far up or down--out of
                                                      // bounds
                        }
                        else if ((col < leftBound) || (col > rightBound)) {
                            destBuffer[a] = srcBuffer[a]; // column too far left or right-
                                                      // -out of bounds
                        }
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
    else {          // in bounds
                            maskedList = getNeighborList(a, srcBuffer, true);
                            // verify that this element is an outlier
                            if (stdDevLimit == 0.0) {   // anything is an outlier
                                shellSort(maskedList);
                                destBuffer[a] = median(maskedList);
                            }
                            else {      // look for outlierness
                                average = mean(maskedList);
                                sigma = standardDeviation(maskedList, average);
                                if ((maskedList[kCenter] > (average + stdDevLimit*sigma)) ||
                                    (maskedList[kCenter] < (average - stdDevLimit*sigma))) {
                                        shellSort(maskedList);
                                        destBuffer[a] = median(maskedList);
                                    }
                                else {  // if element was not an outlier, pixel is fine.
                                    destBuffer[a] = srcBuffer[a];
                                }
                            }
                        }
                    }
                    else {     //  not part of the VOI so just copy this into the destination
                               buffer.
                    destBuffer[a] = srcBuffer[a];
                    }
                    a++;        // address the next data element from the bufferStartingPoint
                }
            }

                         // now set up for the repeat for multiple iterations.
            // But only bother with copying over if there are more iterations.
            if (pass < iterations - 1) {
                tempBuffer = destBuffer;    // swap dest & src buffers
                destBuffer = srcBuffer;
                srcBuffer = tempBuffer;
            }
        }
         // destBuffer should now be copied over for the size of imageSliceLength.
        // You may return.

    }

    /**
    *   Filter a 3D image with a 3D kernel.  Allows median filtering to include the
    *   picture elements at greater depths than only the current slice.
    *   <p><em>Note that this volume filter will correctly filter color images on all
    *    bands (aRGB) because the neighbor list is correct (see getNeighborList()).
    *   This means, however, it will not selectivly filter any bands (one may not
    *   filter only the Red channel, for instance), and will also filter all alpha
    *   values as well.  Of course, progress bar updates will not include any
    *   color information. For these reasons it a useable, but limited color
    *   filter.</em>
    *   @param  srcBufferSource image.
    *   @param  destBufferDestination image.
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
     *   @see    volumeColorFilter
     *   @see    getNeighborList
     *
     */
     // some code has been left in to allow this method to properly filter
     // color images, although the other method is included.

private void    volumeFilter(float srcBuffer[], float destBuffer[]) {
        int i, pass;        // counting the current element
        int row,            // ease of reading to find the row, column and slice
             column,         // (all starting at 0) associated with the current element
            slice;          // [(0,0,0) starts at the closest upper-left corner]
        int imageSliceLength = srcImage.getSliceSize() * valuesPerPixel;
        int imageLength = imageSliceLength * numberOfSlices;
        int kCenter = maskCenter;
         int width =  srcImage.getExtents()[0];     // width of slice in number of pixels
         int height = srcImage.getExtents()[1];      // height of slice in number of pixels
        int sliceWidth = width*valuesPerPixel;     // width of slice in number of intensity
                                                     // values (as in colors per pixel (1 for
                                           // mono, 4 for color))
        float tempBuffer[];

        float average;            // arithmetic mean
        float sigma;              // standard deviation

        float maskedList[];       // list of buffer-values that were showing inside the
        // mask these bounds "frame" the interior of the slice which may be filtered
       // (&adjusted); image outside the frame may not
        int leftBound, rightBound,      // bounds on the column
            upperBound, lowerBound,     // bounds on the row
            aheadBound, behindBound;    // bounds on the slice
        // (a note on orientation: object front is facing in the same direction as
        // viewer, thus ahead of viewer is into monitor, behind is out of monitor and
        // a more positive number of slices is farther forward.)
        upperBound =  halfK;
        lowerBound =  height - halfK -1;
        behindBound = halfK;
        aheadBound =  numberOfSlices - halfK - 1;
        // we may say that each column is a pixel intensity: mono images have 1 per
       // pixel, 4 in color;
        // these calculations are done separately for color & mono images in
       // sliceFilter().
       leftBound =   halfK * valuesPerPixel;
        rightBound =  sliceWidth - valuesPerPixel*halfK - 1; // in color: (4*width - 4*halfK
                                                     // - 1); mono: (width -
                                                    // halfK - 1)
   int mod = (imageLength)/100;  // mod is 1 percent of length of slice * the number
                             // of slices.

        BitSet mask = srcImage.generateVOIMask();
    for (pass = 0; pass < iterations && !threadStopped; pass++) {
            if (pBarVisible == true) {
               progressBar.setMessage("Filtering image (pass "+ String.valueOf(pass+1) +" of "+
              iterations +") ...");
            }

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
for ( i = 0; i < imageLength && !threadStopped; i++){
                if (i%mod == 0 && pBarVisible == true) {
                    progressBar.setValue(Math.round
                    ( (  (float)( (pass*imageLength)+i)/(iterations*(imageLength-1))*100)) );
                }

                if (entireImage == true || mask.get(i/valuesPerPixel) ) {
                    // Median stuff here
                    slice = i/imageSliceLength;
                    row = (i%imageSliceLength)/sliceWidth;
                    column = i%sliceWidth;

                    if ( (row < upperBound) || (row > lowerBound) ) {
                        destBuffer[i] = srcBuffer[i]; // row too far up or down--out of
                                                         // bounds
                    }
                    else if ((column < leftBound) || (column > rightBound)) {
                        destBuffer[i] = srcBuffer[i]; // column too far left or right--out
                                                       // of bounds
                    }
                    else if ((slice < behindBound) || (slice > aheadBound)) {
                        destBuffer[i] = srcBuffer[i]; // slice too far ahead or behind--out
                                                      // of bounds
                    }
                    else {          // in bounds
                        maskedList = getNeighborList(i, srcBuffer, false);
                        // verify that this element is an outlier
                        if (stdDevLimit == 0.0) {   // anything is an outlier
                            shellSort(maskedList);
                            destBuffer[i] = median(maskedList);
                        }
                        else {      // look for outlierness
                            average = mean(maskedList);
                            sigma = standardDeviation(maskedList, average);
                            if ((maskedList[kCenter] > (average + stdDevLimit*sigma)) ||
                                (maskedList[kCenter] < (average - stdDevLimit*sigma))) {
                                    shellSort(maskedList);
                                    destBuffer[i] = median(maskedList);
                            }
                            else { // if element was not an outlier, pixel is fine.
                                    destBuffer[i] = srcBuffer[i];
                            }
                        }
                    }
                }
                             else {     //  not part of the VOI so just copy this
                                            //  into the destination buffer.
                destBuffer[i] = srcBuffer[i];
                }
            }
            // now set up for the repeat for multiple iterations.
            // But only bother with copying over if there are more iterations.
            if (pass < iterations - 1) {
                tempBuffer = destBuffer;    // swap src & dest buffer
                destBuffer = srcBuffer;
                srcBuffer = tempBuffer;
            }
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
            }
        }
         /**
        *   Filter a Color 3D image with a 3D kernel.  Allows median filtering to
        *   include the picture elements at greater depths than only the current
        *   slice.  This method allows selected band filtering, and does not filter
        *   the alpha band.
        *   @param  srcBufferSource image.
        *   @param  destBufferDestination image.
        *   @see    volumeFilter
        *
        */
        private void    volumeColorFilter(float srcBuffer[], float destBuffer[]) {
           int i, pass;        // counting the current element
            int initialIndex;    // reference to the color band being filtered/copied: aRBG:
                              // 0, 1, 2, 3;
                                 // it is an offset to the identified pixel, or column, of
                              // the slice
           int row,            // ease of reading to find the row, column and slice
                column,         // (all starting at 0) associated with the current element
               slice;          // [(0,0,0) starts at the closest upper-left corner]
           int kCenter = maskCenter;
            int width =  srcImage.getExtents()[0];      // width of slice in number of pixels
            int height = srcImage.getExtents()[1];      // height of slice in number of pixels
           int sliceWidth = width*valuesPerPixel;     // width of slice in number of
                                                    // intensity values (as in colors per
                                                    // pixel (1 for mono, 4 for color))
            int sliceSize = width * height;                            // in pixels (or elements)
            int imageSliceLength = width * height * valuesPerPixel; // in values-pixels
            int imageSize = sliceSize * numberOfSlices;             // in pixels (or elements)
            int imageLength = imageSliceLength * numberOfSlices;    // in (values-pixels)
            float tempBuffer[];

            float average;            // arithmetic mean
            float sigma;              // standard deviation

             float maskedList[];       // list of buffer-values that were showing inside the
                                   // mask
            // these bounds "frame" the interior of the slice which may be
           // filtered (&adjusted); image outside the frame may not
            int leftBound, rightBound,      // bounds on the column
                upperBound, lowerBound,     // bounds on the row
                aheadBound, behindBound;    // bounds on the slice
           // (a note on orientation: object front is facing in the same direction as
            // viewer, thus ahead of viewer is into monitor, behind is out of monitor and
           // a more positive number of slices is farther forward.)
       upperBound =  halfK;
            lowerBound =  height - halfK -1;
            behindBound = halfK;
            aheadBound =  numberOfSlices - halfK - 1;
            // we may say that each column is a pixel intensity: mono images have 1 per
           // pixel, 4 in color; these calculations are done separately for color &
           // mono images in sliceFilter().
            leftBound =   halfK * valuesPerPixel;
            rightBound =  sliceWidth - valuesPerPixel*halfK - 1; // in color: (4*width -
                                                               // 4*halfK - 1); mono:
                                                                // (width - halfK - 1)
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
    int mod = (imageSize)/100;  // mod is 1 percent of length of slice * the number
                                // of slices.

             BitSet mask = srcImage.generateVOIMask();

        // copy all alpha values in the image
        setCopyColorText("alpha");
        for (i = 0; i < imageLength; i+=4) {
            destBuffer[i] = srcBuffer[i];       // copy alpha;
        }

        // choose i so the proper colors go alongside the initial index
        // so we get the right output statements in the progress bar
        // copy only needed RGB values
        initialIndex = 0;    // start with alpha on each pass (routine moved so we don't
                            // do it for each pass)
        while (initialIndex < 3 && !threadStopped) {      // alpha:0, R:1, G:2, B:3. But
                                                   // alpha must be copied
            ++initialIndex;             // next initial index

            if (!rChannel && initialIndex==1) {
                // when looking at the image reds but we're not filtering the red
              // channel copy all red values
                setCopyColorText("red");
                for (i = initialIndex; i < imageLength; i+=4) {
                    destBuffer[i] = srcBuffer[i];
                }
            }
             else if (!gChannel && initialIndex==2) {  // when looking at the image greens
                                                // but we're not filtering the
                                                // greens channel copy all greens
                                                // values
              setCopyColorText("green");
                for (i = initialIndex; i < imageLength; i+=4) {
                    destBuffer[i] = srcBuffer[i];
                }
            }
            else if (!bChannel && initialIndex==3) {
                // when looking at the image blues but we're not filtering the
              //  blues channel copy all blue values
                setCopyColorText("blue");
                for (i = initialIndex; i < imageLength; i+=4) {
                    destBuffer[i] = srcBuffer[i];
                }
            }
            else {
                for (pass = 0; pass < iterations && !threadStopped; pass++) {
                    if (pBarVisible == true) {
                    if (initialIndex == 1) {
                        progressBar.setMessage("Filtering red channel (pass "+
                        String.valueOf(pass+1)  +" of "+ iterations +") ...");
                        }
            else if (initialIndex == 2) {
                        progressBar.setMessage("Filtering green channel (pass "+
                        String.valueOf(pass+1) +" of "+ iterations +") ...");
                        }
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
    else if (initialIndex == 3) {
                            progressBar.setMessage("Filtering blue channel (pass "+
                            String.valueOf(pass+1) +" of "+ iterations +") ...");
                        }
                    }
                    // if we needed to filter the image, we dropped through the
                  // selection to filter the color given by ints initialIndex
                    for ( i = initialIndex; i < imageLength && !threadStopped; i+=4){
                        if (i%mod == 0 && pBarVisible == true) {
                            progressBar.setValue(Math.round
                          ( (  (float)(iterations*(initialIndex - 1)*imageSize + imageSize*pass
                              + i/4)/(3*iterations*(imageSize-1))*100)) );
                        }

                        if (entireImage == true || mask.get(i/valuesPerPixel) ) {
                            // Median stuff here
                            slice = i/imageSliceLength;
                            row = (i%imageSliceLength)/sliceWidth;
                            column = i%sliceWidth;

                            if ( (row < upperBound) || (row > lowerBound) ) {
                               destBuffer[i] = srcBuffer[i]; // row too far up or down--out of
                                                        // bounds
                            }
                            else if ((column < leftBound) || (column > rightBound)) {
                               destBuffer[i] = srcBuffer[i]; // column too far left or right--
                                                        // out of bounds
                            }
                            else if ((slice < behindBound) || (slice > aheadBound)) {
                              destBuffer[i] = srcBuffer[i]; // slice too far ahead or behind--
                                                        // out of bounds
                            }
                            else {          // in bounds
                                maskedList = getNeighborList(i, srcBuffer, false);
                                // verify that this element is an outlier
                                if (stdDevLimit == 0.0) {   // anything is an outlier
                                    shellSort(maskedList);
                                    destBuffer[i] = median(maskedList);
                                }
                                else {      // look for outlierness
                                    average = mean(maskedList);
                                    sigma = standardDeviation(maskedList, average);
                                  if ((maskedList[kCenter] > (average + stdDevLimit*sigma)) ||
                                      (maskedList[kCenter] < (average - stdDevLimit*sigma))) {
                                            shellSort(maskedList);
                                            destBuffer[i] = median(maskedList);
                                    }
                                    else { // if element was not an outlier, pixel is fine.
                                            destBuffer[i] = srcBuffer[i];
                                    }
                                }
                                                  }
                        }
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
                         else {     // not part of the VOI so just copy this into the
                               // destination buffer.
                            destBuffer[i] = srcBuffer[i];
                        }
                    }
                    // now set up for the repeat for multiple iterations.
                 // But only bother with copying over if there are more iterations.
                    if (pass < iterations - 1) {
                        tempBuffer = destBuffer;    // swap src & dest buffer
                        destBuffer = srcBuffer;
                        srcBuffer  = tempBuffer;
                    }
                }
                if (iterations%2 == 0) {        // if even number of iterations, then
                    tempBuffer = destBuffer;    // swap src & dest buffer is necessary
                    destBuffer = srcBuffer;     // to keep other colors not-yet-filtered
                                          // from
                    srcBuffer  = tempBuffer;    // filtering from the wrong buffer,
                                          // overwriting the real src
                }
            }
        }
    }

    /**
    * Forms kernel.  Note that the <bold>kernel</bold> uses the 0th place, unlike the
    *   kernelMask where counting starts at 1.
    *
    */
    private void makeKernel(){
        try {
            if (sliceFiltering)
                kernel = new byte[kernelSize*kernelSize];
            else if (!sliceFiltering)
                kernel = new byte[kernelSize*kernelSize*kernelSize];
        }
        catch (OutOfMemoryError e) {
           displayError("Algorithm Median reports: not enough memory to form a kernel mask.");
            setCompleted(false);
            setThreadStopped(true);
            return;
        }
        setKernel();
        makeKernelMask();
    }

    /**
    * Fill in the mask for which pixels are used in filtering.
    *
    */
    private void setKernel() {
        int i;
        int halfK = kernelSize/2;

        //  square/box
        if ( (kernelShape == SQUARE_KERNEL) ||
             (kernelShape == CUBE_KERNEL)) {
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
            for (i=0; i < kernel.length; i++)
                    kernel[i] = 1;

    } // end square/cube kernel

      // cross/axial
    else if ((kernelShape == CROSS_KERNEL) ||
                    (kernelShape == AXIAL_KERNEL)) {
                int row;    // indicates current row
                int col;    // indicates current column
                if (sliceFiltering) {
                    for (i = 0; i < kernel.length; i++) {
                        row = i/kernelSize;
                        col = i%kernelSize;

                        if (col == halfK)       {kernel[i] = 1;}
                        else if (row == halfK)  {kernel[i] = 1;}  // should be for a cross ->
                                                          // else if
                                                            // (row == halfK)  {kernel[i] = 1;}
                        else                    {kernel[i] = 0;}
                    }
                }
                else {      // volume filtering
                    int slice;
                    for (i = 0; i < kernel.length; i++) {
                        slice = i/(kernelSize*kernelSize);
                        row = (i%(kernelSize*kernelSize))/kernelSize;
                        col = i%kernelSize;

                        if (slice == halfK) {
                            if (col == halfK)       {kernel[i] = 1;}
                            else if (row == halfK)  {kernel[i] = 1;}
                            else                    {kernel[i] = 0;}
                        }
                        else if ((row == halfK) && (col == halfK)) {
                            kernel[i] = 1;
                        }
                        else {kernel[i] = 0;}
                    }
                }
            } // end cross/axial
            else if ( kernelShape == VERT_KERNEL) {
                int row;    // indicates current row
                int col;    // indicates current column
        if (sliceFiltering) {
                    for (i = 0; i < kernel.length; i++) {
                        row = i/kernelSize;
                        col = i%kernelSize;

                        if (col == halfK)       {kernel[i] = 1;}
                        else if (row == halfK)  {kernel[i] = 0;}
                        else                    {kernel[i] = 0;}
                    }
                }
                else {      // volume filtering
                }
            } // end vert

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
                     else if ( kernelShape == HORZ_KERNEL) {
                  int row;    // indicates current row
                  int col;    // indicates current column
                  if (sliceFiltering) {
                      for (i = 0; i < kernel.length; i++) {
                          row = i/kernelSize;
                          col = i%kernelSize;

                          if (col == halfK)       {kernel[i] = 0;}
                          else if (row == halfK)  {kernel[i] = 1;}
                          else                    {kernel[i] = 0;}
                      }
                  }
                  else {      // volume filtering
                  }
              } // end vert

              // 'x' kernel
              else if (kernelShape == X_KERNEL) {
                  int row;    // indicates current row
                  int col;    // indicates current column
                  int revcol; // runs opposite of the col.
                  if (sliceFiltering) {
                      for (i = 0; i < kernel.length; i++) {
                          row = i/kernelSize;
                          col = i%kernelSize;
                          revcol = kernelSize - 1 - col;

                          if      (row == col)        {kernel[i] = 1;}
                          else if (row == revcol)     {kernel[i] = 1;}
                          else                        {kernel[i] = 0;}
                      }
                  }
                  else {      // volume filtering
                      int slice;
                      for (i = 0; i < kernel.length; i++) {
                          slice = i/(kernelSize*kernelSize);
                          row = (i%(kernelSize*kernelSize))/kernelSize;
                          col = i%kernelSize;
                          revcol = kernelSize - 1 - col;

                          if ((slice == col) ||
                              (slice == revcol)) {
                              if      (row == col)    {kernel[i] = 1;}
                              else if (row == revcol) {kernel[i] = 1;}
                              else                    {kernel[i] = 0;}
                          }
                                 else {kernel[i] = 0;}
                      }
                  }
              } // end 'x' kernel

          }

          /**

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
 *     Makes the kernel mask.
      *  The kernel mask is the list of values pulled from the
      *  image which will be used to find the median of the
      *  central pixel.  Its length is
      *  <i>(number of pixels to be used to determine median) + 1</i>.
      *  <p>
      *  Thus the kernel center (decided here), has the value of
      *  the location of the central pixel shown in the window.
      *  The value of the kernel center is the number of pixels picked
      *  up to median sort.
      *
      *  Since the kernel mask is <i>number of pixels + 1</i>,
      *  the maskCenter must be
      */
    private void makeKernelMask() {
        halfK = kernelSize/2;
        // figure how many kernel elements are actually in the kernel-mask
        int count = 1; // start counting from one, since sort starts with element 1
                    // (even empty mask must have 1 element!)
        for (int m = 0; m < kernel.length; m++) {
            if (kernel[m] != 0) // if this element is marked 'on'
                count++;
        }
        kernelMask = new float[count];    // must have the leading element empty: the sort
                                        // starts with element 1
        if (sliceFiltering) {// 2D

            if (kernelShape == SQUARE_KERNEL){
                kernelCenter = count/2 - 1;  // whole square
                maskCenter = halfK*(kernelSize + 1) + 1;        // count/2 : I feel dumb
            }
            else if (kernelShape == CROSS_KERNEL || kernelShape == VERT_KERNEL || kernelShape
==
                                HORZ_KERNEL) {
                kernelCenter = halfK*(kernelSize + 1);
                maskCenter = kernelSize;
            }
            else if  (kernelShape == X_KERNEL) { // sizeof kernel is same as CROSS_KERNEL
                kernelCenter = halfK*(kernelSize + 1);  // whole square -- (count/2-1)??
                maskCenter = kernelSize;
            }
        }
        else {//3D
            if (kernelShape == CUBE_KERNEL) {
                kernelCenter = count/2 - 1;
                maskCenter = halfK*(kernelSize*kernelSize + kernelSize + 1) + 1;
            }
            else if (kernelShape == AXIAL_KERNEL) {
                kernelCenter = (kernelSize*kernelSize*kernelSize)/2;    // whole cube
          maskCenter = count/2;           // i feel dumb...
            }
            else if  (kernelShape == X_KERNEL) { // sizeof kernel is same as AXIAL_KERNEL
                kernelCenter = (kernelSize*kernelSize*kernelSize)/2;    // whole cube
                maskCenter = count/2;           // i feel dumb...
            }
        }
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```java
// not entirely dumb.  mc = count/2 because of the symmetry of the mask. Custom masks
// may be diff. & i'd like to include custom masks someday....
    }

    /**
     *   Compiles a list of the values neighboring the desired pixel, that are defined
     *   in the kernel.  Be careful because although the kernel starts its index at 0,
     *   the list that is returned starts indexing at 1.
     *   <p>
     *   Color images are processed differently from the monochrome images because
     *   although color images use the same size kernel as mono images, it fills the
     *   kernel with brightness levels that are spread out in the data set.  The
     *   Neighbor list still reports the monochromatic brightness values. That is, for
     *   a color image: the neighbors of the central pixel with the same color are
     *   returned in the neighbor list's kernel.
     *   @param  i       The central pixel to find neighbors for.
     *   @param  data    Image data
     *   @param  is2D    True indicates that the neighbors are found along a
     *                   2D slice (or 2D image) instead of neighbors in a 3D volume.
     *   @return The neighboring pixel list, where the list starts at 1 (leaving the
      initial
     *   element 0), and corresponds to the kernel chosen.
     */
    private final float[] getNeighborList(int i, float[] data, boolean is2D) {
        int row, col;
        int kCenter = kernelCenter;    // index to the central element of the kernel
                                       // (this is the mask for which elements in data are
                                  // used.)
        int width = 0;                 // width of slice in number of pixels
        int height = 0;                // height of slice in number of pixels

        try {
            width  = srcImage.getExtents()[0];
            height = srcImage.getExtents()[1];
        } catch (NullPointerException npe) {
            Preferences.debug("AlgorithmMedian: null pointer while making neighbor list.");
            setThreadStopped(true);
            setCompleted(false);
        }
        int sliceWidth =  width * valuesPerPixel;             // width of slice in number of
                                                         // elements

        // place all the masked 'on' elements into the data-list
        int count  = 1;
        // color images are different from the mono images in that though color images
       // use the same size kernel as mono images, but fill it with brightness levels
       // that are spread out in the data set.

```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
   int kcol;
              int leftBound = -halfK * 4;
              int rightBound = halfK * 4;
              if (is2D) {
                  for (row = -halfK; row <= halfK; row++) {   // go through all rows
                      for (col = leftBound, kcol = -halfK; col <= rightBound; col += 4, kcol++) {
                           // go through every 4th column
                          if (kernel[kCenter+kcol+row*kernelSize] != 0) {// but don't bother
                                                                         // copying into the list
                                                                         // if we don't want that
                                                                            // that element (the
                                                                         // kernel's pixl is zero)
                                  kernelMask[count++] = data[i+col+row*sliceWidth];
                          }
                      }
                  }
              }
              else {           // find neighbors in a volume
                  int slice;
                  // halfK-number of kernelSize slices (to get to the center slice)
                  for (slice = -halfK; slice <= halfK; slice++) {
                      for (row = -halfK; row <= halfK; row++) {
                          for (col = leftBound,  kcol = -halfK; col <= rightBound; col += 4,
kcol++) {
                              if (kernel[kCenter+kcol+row*kernelSize+slice*kernelSize*kernelSize]
                               != 0) {
                                  kernelMask[count++] =
                               data[i+col+row*sliceWidth+slice*sliceWidth*height];
                              }
                          }
                      }
                  }
              }
          }
          else {  // a mono image
              if (is2D) {
                  for (row = -halfK; row <= halfK; row++) {   // go through all rows
                      for (col = -halfK; col <= halfK; col++) {   // go through all columns
                          if (kernel[kCenter+col+row*kernelSize] != 0) {  // but don't bother
                                                                          // copying into the list
                                                                            // if we don't want
                                                                            // that element (the
                                                                            // kernel's pixl is zero)
                              kernelMask[count++] = data[i+col+row*width];
                          }
                      }
                  }
              }
    if (isColorImage) {         // 2D filtering of color images is a little different
                                  // than of mono images
                        else {          // find neighbors in a volume
                  int slice;
                  // halfK-number of kernelSize slices (to get to the center slice)
                  for (slice = -halfK; slice <= halfK; slice++) {
                      for (row = -halfK; row <= halfK; row++) {
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
                               for (col = -halfK; col <= halfK; col++) {
                           if (kernel[kCenter+col+row*kernelSize+slice*kernelSize*kernelSize]
    !=
                             0) {
                               kernelMask[count++] = data[i+col+row*width+slice*width*height];
                          }
                      }
                  }
              }
          }
      }
      return (kernelMask);
  }

  /**
  * Sorts a list of values.  Taken from Numerical Recipes in C, 2nd ed.  William H.
  * Press, et al, page 332.  Chose shell sort over a quicksort because both shell
  * and quick are about the same speed for the middle range of sizes of the list.
  * The list is more likely during a slice-filter operation to be smaller than the
  * maximum 121 length.  The list could be as much as 1331 elements, but according
  * to Numerical Recipes, it still runs fast enough at only N**1.25 an average for
  * N < 60000).  My guess is that sliceFilter is more useful than a volumeFilter
  * and will be plenty fast enough to not necessitate a quicksort for a
  * debatable speed-increase.
  *
  *   @param float a[]    The list to sort.
  */
 private final void shellSort(float a[]) {
      int N = a.length - 1;
      int i, j;
      int inc = 1;
      float val;

      do {
          inc *=3;
          inc++;
      } while (inc <= N);
      do {
          inc /=3;
          for (i = inc + 1; i <= N; i++) {
              val = a[i];
              j = i;
              while (a[j - inc] > val) {
                  a[j] = a[j - inc];
                  j -= inc;
                  if (j <= inc) break;
           }
              a[j] = val;
          }
         }while (inc > 1);
  }
 /**
  *  Finds the median value of the list.  Median assumes the list of values starts
  *   at index 1, not an index of 0. (i.e., 1st element is not included.)
  *   @param  listList of numbers
  *   @return The median.
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
    *   @author parsonsd
    */
private final float median(float list[])    {
        int N;
        float med;

        N = list.length - 1;

        if ((N%2) != 0) {
            med = list[N/2];
        }
        else {
            med = (list[N/2] + list[N/2+1])/2;
        }
        return (med);
    }

    /**
    *   Finds the mean value (average) in the list.  Mean assumes the list of
    *   values starts at index 1, not an index of 0. (i.e., 1st element is not
    *   included.)
    *   @param  listList of numbers
    *   @return floatThe mean.
    *   @author parsonsd
    */
    private final float mean(float list[]) {
        int i;
        float sum = 0;

        for (i = 1; i < list.length; i++) {
            sum += list[i];
        }
        return (float)(sum/(list.length - 1));  // length-1 because list goes from
                                          // [1 ... N]
    }

        /**
    *   Finds the standard deviation of the values in the input list
    *   (defined as:   s = [(1/(N-2))*SUM (from 1 to N-1)[ (Xi - <bold>X</
    *   bold>)^2]]^(1/2))
    *   @param listThe list of numbers.
    *   @param average  Arithmetic mean of the values in list.
    *   @return The standard deviation.
    */
```

Figure 1. PlugInAlgorithmMedian.java (continued)

```
private final float standardDeviation(float list[], float average) {
        int i;
        int N = list.length;

        double sum = 0.0;

        for (i = 1; i < N; i++) {
            sum += (list[i] - average)*(list[i] - average);
        }
        return ((float) Math.sqrt(sum/(N-2)));        // sqrt((1/(N-2)) * sum)
    }


    /**
    *   Creates the standard progressBar.  Stores in the class-global, progressBar.
    */
    private void buildProgressBar(){
        try {
            if (pBarVisible == true) {
                progressBar = new ViewJProgressBar(srcImage.getImageName(), “Filtering
                image ...”,
                                                0, 100, true, this, this);
                int xScreen = Toolkit.getDefaultToolkit().getScreenSize().width;
                int yScreen = Toolkit.getDefaultToolkit().getScreenSize().height;
                progressBar.setLocation(xScreen/2, yScreen/2);
                progressBar.setVisible(true);
            }
        } catch (NullPointerException npe) {
            if (Preferences.isDebug()) {
                Preferences.debug("AlgrithmMedian: NullPointerException found while building
                                progress bar.");
            }
        }
    }

    /**
     *If the progress bar is visible, sets the text to: <br><tt>Copying all <i>color</i> values
     *  ... </tt>
     *  @param colorTextThe color to use.  E.g., "red" or "blue".
     */
    private void setCopyColorText(String colorText)
    {
        try {
            if (pBarVisible == true) {
                progressBar.setMessage(“Copying all " + colorText + " values ... ");
            }
        } catch (NullPointerException npe) {
            if (Preferences.isDebug()) {
                Preferences.debug("AlgrithmMedian: NullPointerException found while setting
                progress bar text.");
            }
        }
    }
```

Figure 1. PlugInAlgorithmMedian.java (continued)

Glossary

This glossary defines all acronyms and selected terms used in this guide.

Numerics

2D. Two dimensional.

3D. Three dimensional.

4D. Four dimensional.

5D. Five dimensional.

A

ACR. American College of Radiology. The ACR, in conjunction with National Electrical Manufacturers Association, developed the Digital Image Communication in Medicine standard.

AE. Application entity.

Analyze. Unix-based medical-imaging display and analysis software developed by the Mayo Foundation. MIPAV allows researchers to process, analyze, and visualize Analyze-formatted image files on virtually any platform.

API. Application program interface. Pieces of code that are used to perform common tasks, such as generate a standard window frame. Software developers often incorporate these pieces of code into their programs. An API is analogous to a package of form letters; APIs reduce programming time because common functions have already been written.

BMP. Extension for Windows Bitmap formatted files. BMP is the standard bitmap graphics file format that is used in the MS Windows environment.

boolean. This data type refers to data that represents symbolic relationships between entities, such as those implied by the logical operators AND, NOT, and OR. Examples of valid boolean values are TRUE and FALSE.

bytecode. Compiled format for Java code. Bytecode is analogous to object code. When the Java program is written and compiled, the compiled program is written in bytecode. When you execute the bytecode program, it is interpreted by the platform-specific Java Virtual Machine, which serves as an interface between your platform and the platform-independent bytecode. Java bytecode can be ported to almost any platform and executed, provided the correct Java Virtual Machine has been installed.

CIT. Center for Information Technology. CIT provides, coordinates, and manages information technology so that computational science at the National Institutes of Health is advanced.

color 24. Color 24 is commonly referred to as 24-bit color images. Full RGB color requires that the intensities of three color components be specified for each and every pixel. It is common for each component intensity to be stored as an 8-bit integer, and so each pixel requires 24 bits to completely and accurately specify its color. Image formats that store a full 24 bits to describe the color of each and every pixel are therefore known as 24-bit color images.

CR. Computed radiography.

C-STORE. Composite Storage.

CT. Computed Tomography.

data type. A set of values from which a variable, constant, function, or expression may take its value. MIPAV accommodates the following data types: Boolean, Signed Byte, Unsigned Byte, Signed Short, Unsigned Short, Integer, Long, Float, Double, and Color 24.

DCB. Division of Computational Bioscience. DCB is a research and development organization that provides engineering and computer science expertise to support biomedical research activities at the National Institutes of Health (NIH). DCB applies image processing and medical imaging technologies, high-performance parallel computing, high-speed networking, signal processing, state-of-the-art optical and electronic devices, bioinformatics, database technology, mathematical and statistical techniques, and modern hardware and software engineering principles to help solve biomedical research problems at NIH.

DICOM. Digital Image Communication in Medicine. Standard protocol developed by the American College of Radiology (ACR) and National Electrical Manufacturers Association (NEMA). Specifies a standard method of communication between two devices.

Double. Primitive, 64-bit, data type. Double is a floating point data type that accommodates decimal values, up to 14 or 15 significant digits of accuracy. Valid values can range from -1.7×10^{308} to 1.7×10^{308}.

Endian. Data organization strategy. Refers to the way computer processors store data in memory. Big-endian format stores the most significant byte (MSB) first. Little-endian format stores the least significant byte (LSB) first.

FF. Feet first.

Float. Primitive, 32-bit, data type. Float is a floating point data type that accommodates decimal values, up to 6 or 7 significant digits of accuracy. Valid values can range from -3.4×10^{38} to 3.4×10^{38}.

FTP. File Transfer Protocol.

GIF. Graphic Interchange Format. A compressed, bit mapped, graphics file format that supports color and various resolutions.

GUI. Graphical user interface. A user interface that is based on graphics rather than text.

header offset. Space reserved at the beginning of some graphic files that contain non-image data.

HF. Head First

HP. Hewlett-Packard.

HSB. Hue Saturation Brightness. In this color model, hue is the color, saturation is the purity of the color; and brightness indicates the brightness or darkness of the color.

ID. Identifier.

IE. Information Entity.

integer. Primitive, 32-bit, data type. Integer is sometimes abbreviated as int. Integer accommodates values that are whole numbers. Valid values range from -2,147,483,648 to +2,147,483,648.

interlaced. A display technique that increases resolution. Half of the horizontal lines are drawn on the monitor during the first pass; the other half are drawn during the second pass. For example, the odd numbered lines may be drawn during the first pass and the even numbered lines during the second pass.

interpolation. The generation of intermediate values based on known values.

IOD. Information Object Definition

IOD. Information Object Definition. Provides an abstract definition of real-world objects applicable to the communication of digital medical information.

IP. Internet Protocol.

Java. High-level, object-oriented, platform-independent programming language developed by Sun Microsystems.

Java VM. Java Virtual Machine.

JIT. Just-In-Time compiler. The JIT converts Java bytecode into machine language instructions.

JPEG. Extension for Joint Photographics Experts Group formatted files. Also refers to a compression type.

Linux. An operating system that is an open source implementation of UNIX.

Long. Primitive, 64-bit data type. Long is a variation of the integer data type. Long accommodates values that are whole numbers. Valid values range from -9,223,372,036,854,775,808 to +9,223,372,036,854,775,808.

LSB. Least Significant Byte. Also see endian.

LUT. Lookup Table.

Mac OS. Macintosh Operating System

MB. Megabyte

MIPAV. Medical Image Processing, Analysis, and Visualization program. MIPAV is an n-dimensional, general-purpose, extensive image processing, and visualization program. It is platform-independent and assists researchers with extracting quantitative information from various medical imaging modalities.

MR. Magnetic Resonance.

MSB. Most Significant Byte. See endian for more details.

MSEE. Master of Science in Electrical Engineering

MTX. Extension for MIPAV's transform matrix files.

NEMA. National Electrical Manufacturers Association.

NIH. National Institutes of Health.

NM. Nuclear medicine.

OS. Operating system

PACS. Picture Archiving System.

PCX. Extension for PC Paintbrush formatted graphic files.

PDU. Protocol Data Unit.

PET. Positron Emission Tomography.

PICT. Extension for Macintosh formatted graphic files.

PLT. Extension for MIPAV's graphics files.

PNG. Extension for Portable Network Graphic formatted graphic files.

PSD. Extension for Adobe Photoshop formatted graphic files.

RAM. Random Access Memory

Raster. Bitmap file type.

Raw. File type.

resolution. The sharpness and clarity of an image.

RGB. Red Green Blue.

RIS. TBD.

RLE (Run Length Encoding). The file extension for graphics that have been reduced using run-length encoding. RLE is a compression method that converts consecutive identical characters into a code consisting of the character and the number marking the length of the run.

ROI. Region of Interest.

RS. Extension for Sun Raster formatted graphics files.

SCP. Service Class Provider.

SCU. Service Class User.

SGI. Silicon Graphics Incorporated.

short. Primitive, 16-bit data type. Short is a variation of the integer data type. Short accommodates values that are whole numbers. Valid values range from 0 to +32,767.

signed byte. Primitive, 8-bit, data type. Signed byte is a variation of the integer data type. The signed byte data type signifies that valid values fall within a range of whole numbers. Valid values range from -128 to +128. Negative values (indicated by the negative sign) are permitted, hence the term, signed byte.

signed short. Primitive, 16-bit data type. Signed short is a variation of the integer data type. The signed short data type signifies that valid values fall within a range of whole numbers. Valid values range from -32,768 to +32,767. Negative values (indicated by the negative sign) are permitted, hence the term, signed short.

Solaris. Unix-based operating environment that was developed by Sun Microsystems. Solaris consists of the Sun operating system and a windowing system.

SOP. Service Object Pair

SOP. Service Object Pair.

SPECT. TBD.

TCP/IP. Transmission Control Protocol/Internet Protocol. The suite of communications protocols used to connect hosts on the Internet.

TGA. Extension for Truevision Graphics Adapter formatted graphics files.

TIFF. Extension for Tag Image File Format formatted graphics files.

UID. Unique Identifier.

UNIX. Multi-tasking, multi-user operating system developed by Bell Labs. Many versions of UNIX abound, including Linux.

unsigned byte. Primitive, 8-bit, data type. Unsigned byte is a variation of the integer data type. The unsigned byte data type signifies that valid values must fall within a specified range of positive, whole-number values. Valid values range from 0 to +128. Negative values (indicated by the negative sign) are not valid, hence the term, unsigned byte.

unsigned short. Primitive, 16-bit data type. Unsigned short is a variation of the integer data type. The unsigned short data type signifies that valid values must fall within a specified range of positive, whole-number values. Valid values range from 0 to +32,767. Note that negative values (indicated by the negative sign) are not valid, hence the term, unsigned byte.

US. Ultrasound

VM. Virtual Machine.

VOI. Volume of interest (used interchangeably with ROI).

voxel. Smallest distinguishable cube-shaped part of a 3D image.

XBM. X BitMap file format.

XPM. X PixMap file format.