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)

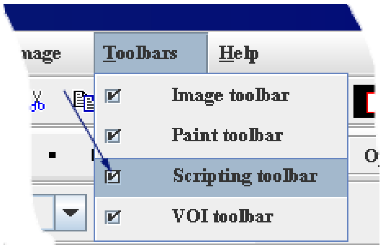
Setting up scripting

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

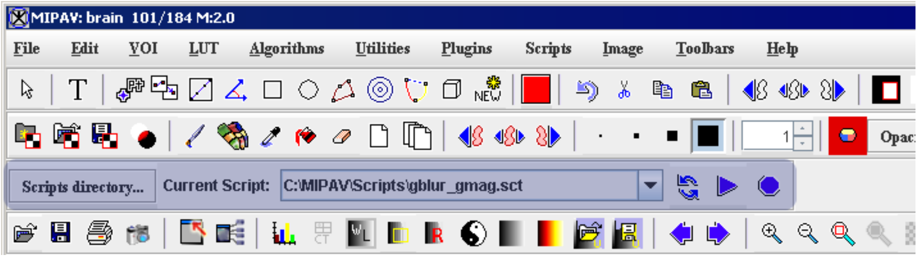
Like the paint toolbar, you can choose to show it on an as-needed basis or not to show it on an as-needed basis. To display the scripting toolbar on an as-needed basis, select Toolbars > Scripting Toolbar (Figure 285) MIPAV.



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.



Current Script: C:\MIPAV\Scripts\gblur_gmag.sct

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.

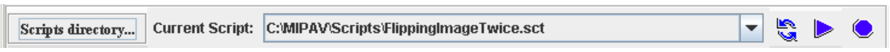
To set up Scripts Home

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.



Current Script: C:\MIPAV\Scripts\FlippingImageTwice.sct

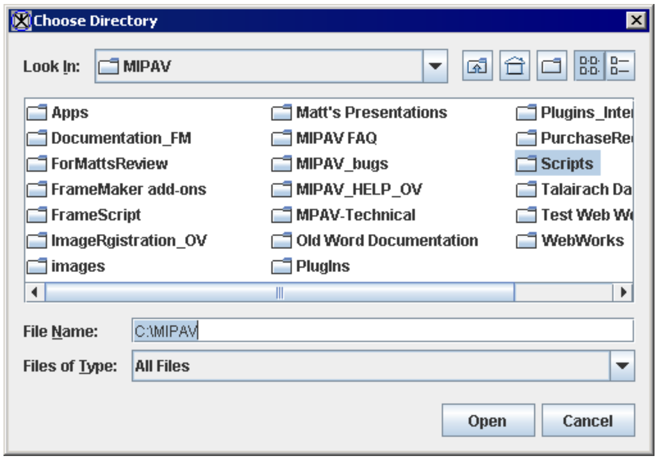
Refresh the script listing

▶ 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.



Look In: MIPAV

File Name: C:\MIPAV

Files of Type: All Files

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

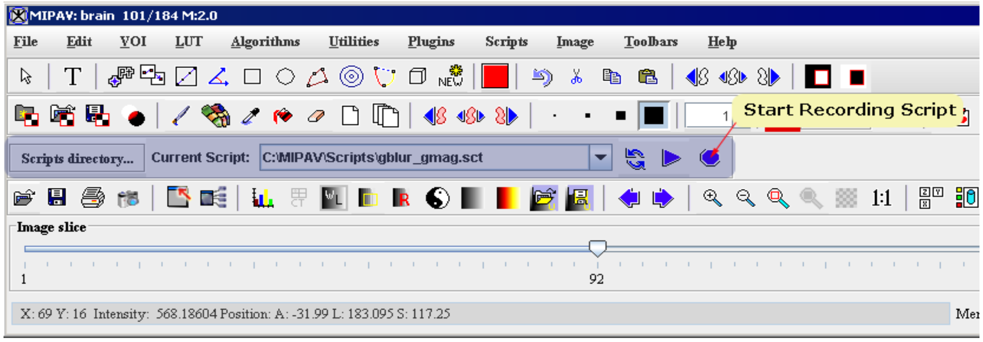
Although MIPAV allows you to easily create scripts, it is important to plan scripts before creating them, particularly if the scripts perform actions on VOIs. First, however, you need to understand:

- 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.



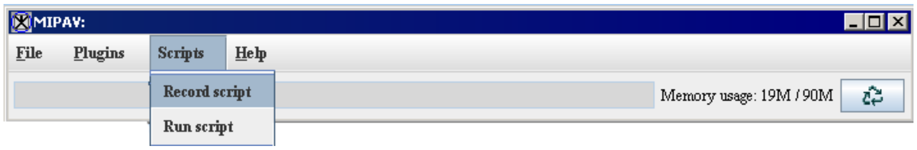
Current Script: C:\MIPAV\Scripts\gblur_gmag.sct

X: 69 Y: 16 Intensity: 568.18604 Position: A: -31.99 L: 183.095 S: 117.25

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.



Memory usage: 19M / 90M

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.

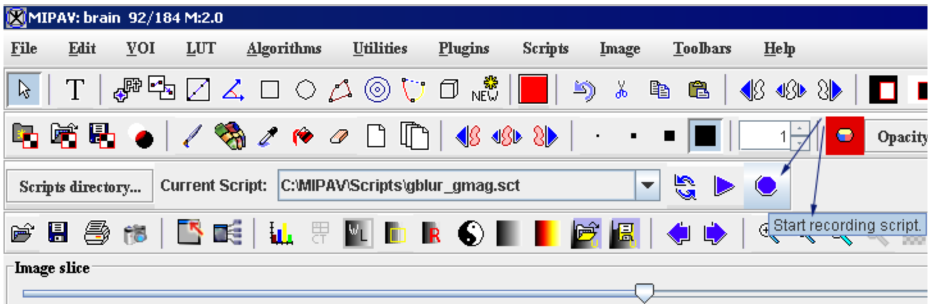
Recording scripts

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.



Current Script: C:\MIPAV\Scripts\gblur_gmag.sct

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.

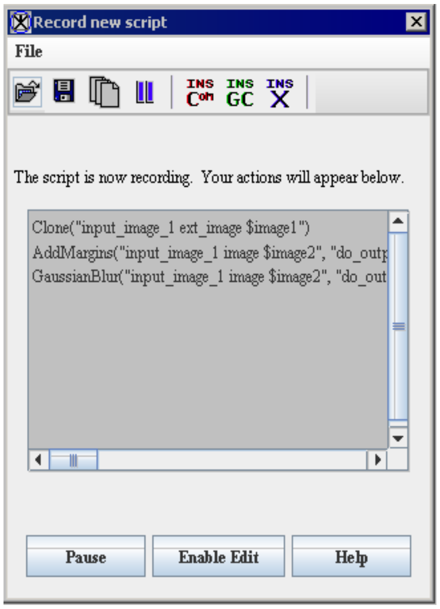
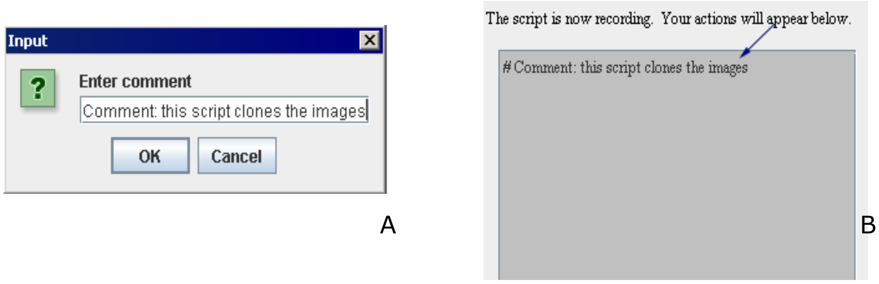
File	**Open**—Opens a previously saved script. When you select this command, the Open dialog box appears. **Save**—Saves the script under the specified name. When you select this command, the Save dialog box opens. **Exit**—Closes this dialog box without saving the script.  Record new script File The script is now recording. Your actions will appear below. Clone("input_image_1 ext_image $image1") AddMargins("input_image_1 image $image2", "do_outp GaussianBlur("input_image_1 image $image2", "do_out Pause Enable Edit Help
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*.  Enter comment Comment: this script clones the images The script is now recording. Your actions will appear below. #Comment: this script clones the images **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.*

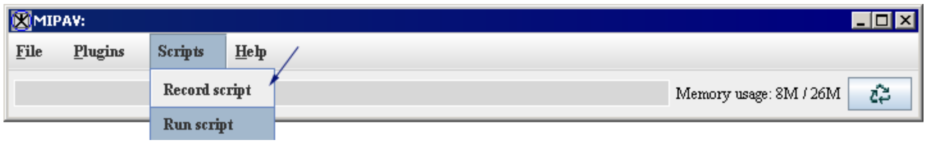


Figure 294. Recording a script via Scripts > Record Script menu

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")
```

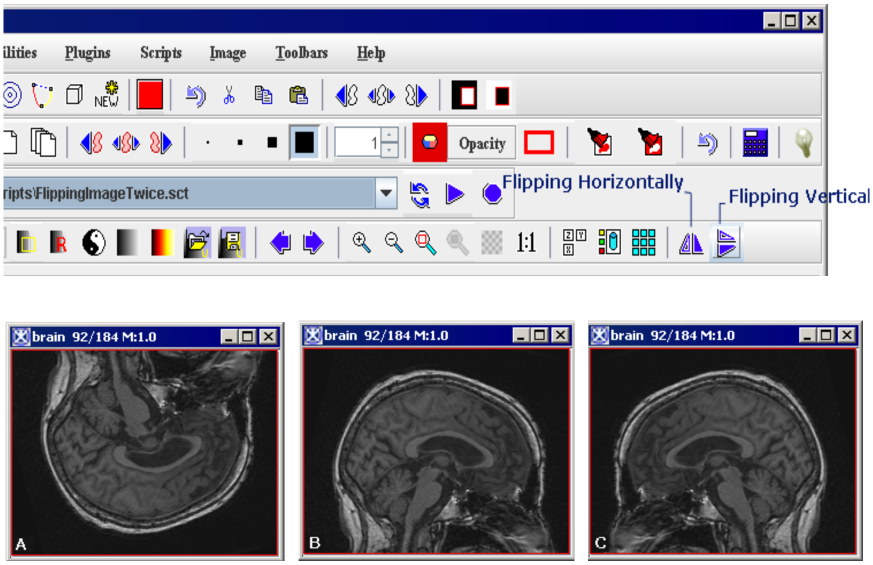


Figure 295. At the top: the flipping options available via the Image Toolbar. Below: the original image (A), the same image flipped vertically (B) and then horizontally (C)

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.

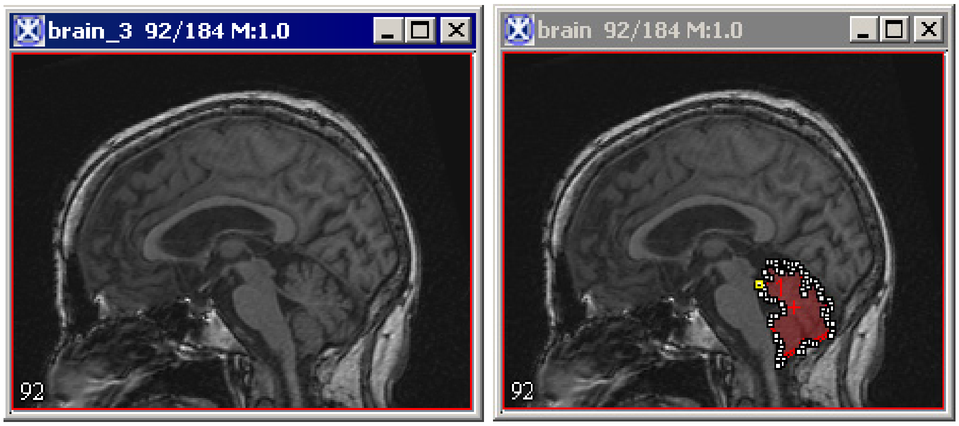
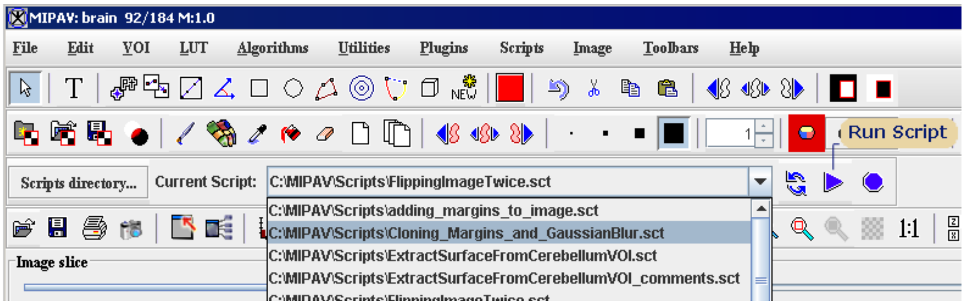


Figure 296. An active image, which bar is highlighted (on your left) and inactive image, which bar is dimmed and grey (on your right)

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.



Current Script: C:\MIPAV\Scripts\FlippingImageTwice.sct

C:\MIPAV\Scripts\adding_margins_to_image.sct
C:\MIPAV\Scripts\Cloning_Margins_and_GaussianBlur.sct
C:\MIPAV\Scripts\ExtractSurfaceFromCerebellumVOI.sct
C:\MIPAV\Scripts\ExtractSurfaceFromCerebellumVOI_comments.sct
C:\MIPAV\Scripts\FlippingImageTwice.sct

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.

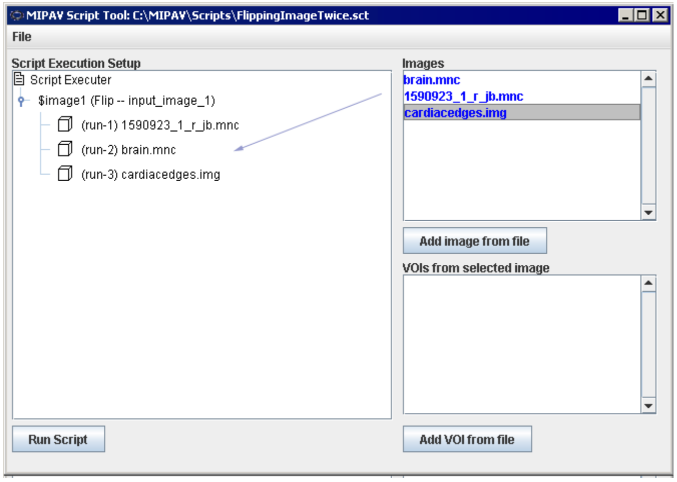
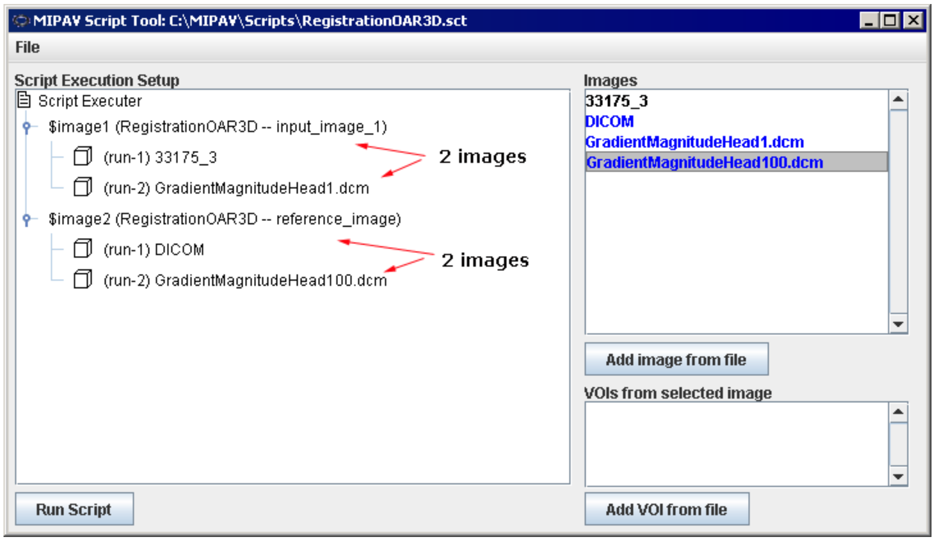


Figure 298. The MIPAV Script Tool dialog box

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.



MIPAV Script Tool: C:\MIPAV\Scripts\RegistrationOAR3D.sct

Script Execution Setup

- Script Executer
  - $image1 (RegistrationOAR3D -- input_image_1)
    - (run-1) 33175_3
    - (run-2) GradientMagnitudeHead1.dcm
  - $image2 (RegistrationOAR3D -- reference_image)
    - (run-1) DICOM
    - (run-2) GradientMagnitudeHead100.dcm

Images

33175_3
DICOM
GradientMagnitudeHead1.dcm
GradientMagnitudeHead100.dcm

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.

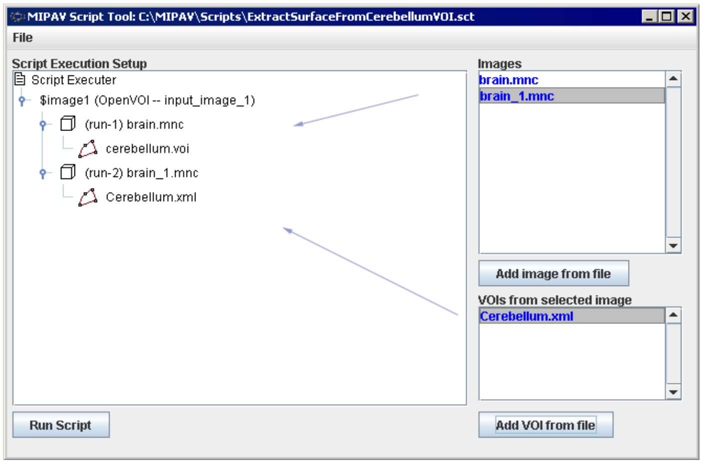
Adding VOIs to the Script Execution Setup

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.



MIPAV Script Tool: C:\MIPAV\Scripts\ExtractSurfaceFromCerebellumVOI.sct

```
Script Executer
  $image1 (OpenVOI -- input_image_1)
    (run-1) brain.mnc
      cerebellum.voi
    (run-2) brain_1.mnc
      Cerebellum.xml
```

Images: brain.mnc, brain_1.mnc

VOIs from selected image: Cerebellum.xml

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

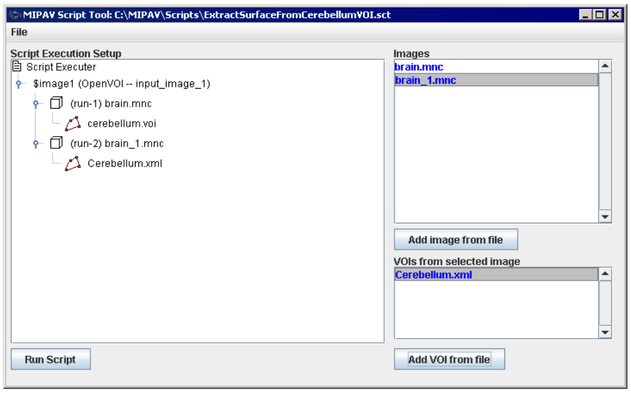
Script Execution Setup window	Contains a list of images and associated VOIs.	
Images	Lists the images that appear in the script	
Add image from file	Allows you to add one or more images on which to run the script.	
VOIs from selected file	Lists any saved VOIs that are necessary when you run the script.	
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.

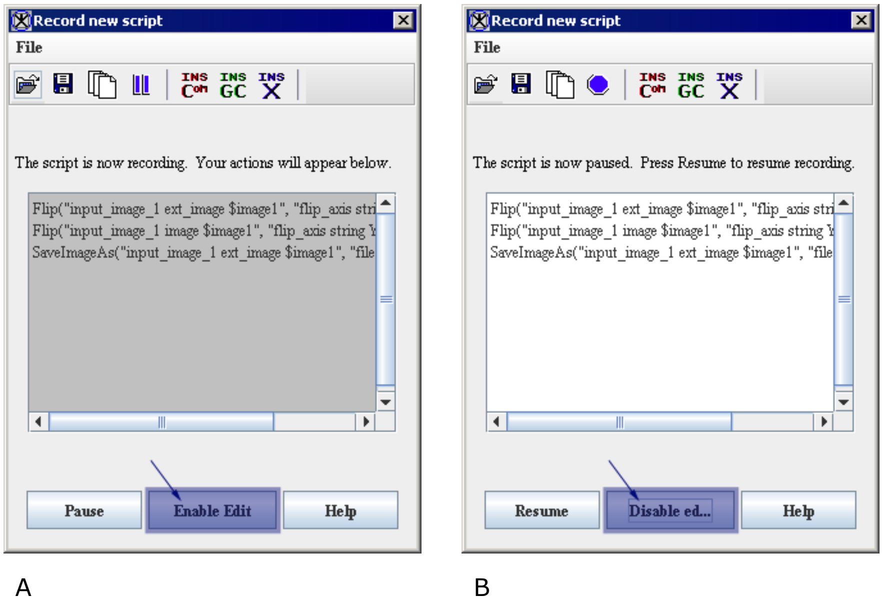
Caution: Editing scripts may cause serious problems if the script commands are not entered correctly. This task is recommended for more advanced users.

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.



The script is now recording. Your actions will appear below.

```
Flip("input_image_1 ext_image $image1", "flip_axis stri
Flip("input_image_1 image $image1", "flip_axis string Y
SaveImageAs("input_image_1 ext_image $image1", "file
```

The script is now paused. Press Resume to resume recording.

```
Flip("input_image_1 ext_image $image1", "flip_axis stri
Flip("input_image_1 image $image1", "flip_axis string Y
SaveImageAs("input_image_1 ext_image $image1", "file
```

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;

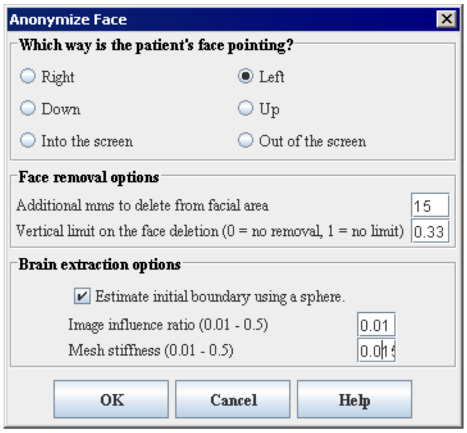
3 And finally, performing brain segmentation using the Fuzzy C-Means > Single Channel algorithm.

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.


Anonymize Face

Which way is the patient's face pointing?

○ Right ● Left

○ Down ○ Up

○ Into the screen ○ Out of the screen

Face removal options

Additional mms to delete from facial area 15

Vertical limit on the face deletion (0 = no removal, 1 = no limit) 0.33

Brain extraction options

☑ Estimate initial boundary using a sphere.

Image influence ratio (0.01 - 0.5) 0.01

Mesh stiffness (0.01 - 0.5) 0.015

```
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")
```

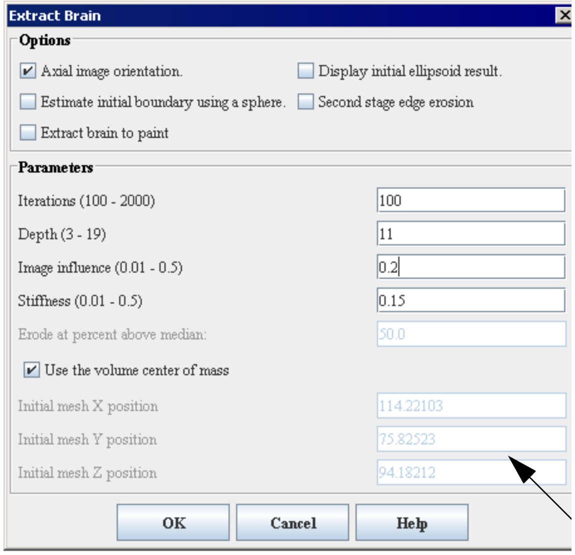
Figure 303. The Anonymize Face dialog box options and the corresponding script text. Here, "face_orientation int 2" **is an image specific and corresponds to the Left option chosen in the "Which way is the patient's face pointing?" dialog**

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 screen, make the result image active and call the last algorithm – Fuzzy C-Means > Single Channel.



☑ Axial image orientation.
☐ Display initial ellipsoid result.
☐ Estimate initial boundary using a sphere.
☐ Second stage edge erosion
☐ Extract brain to paint

Iterations (100 - 2000): 100
Depth (3 - 19): 11
Image influence (0.01 - 0.5): 0.2
Stiffness (0.01 - 0.5): 0.15
Erode at percent above median: 50.0
☑ Use the volume center of mass
Initial mesh X position: 114.22103
Initial mesh Y position: 75.82523
Initial mesh Z position: 94.18212

```
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**

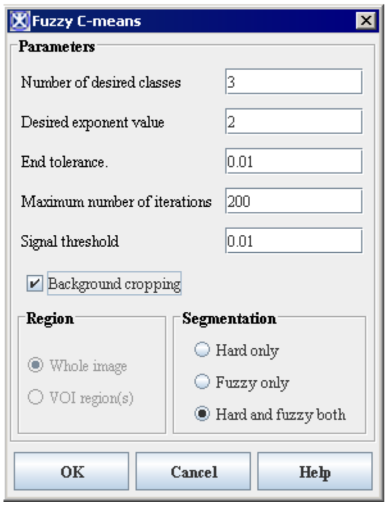
Recording the Fuzzy C-Means > Single Channel algorithm execution:

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;

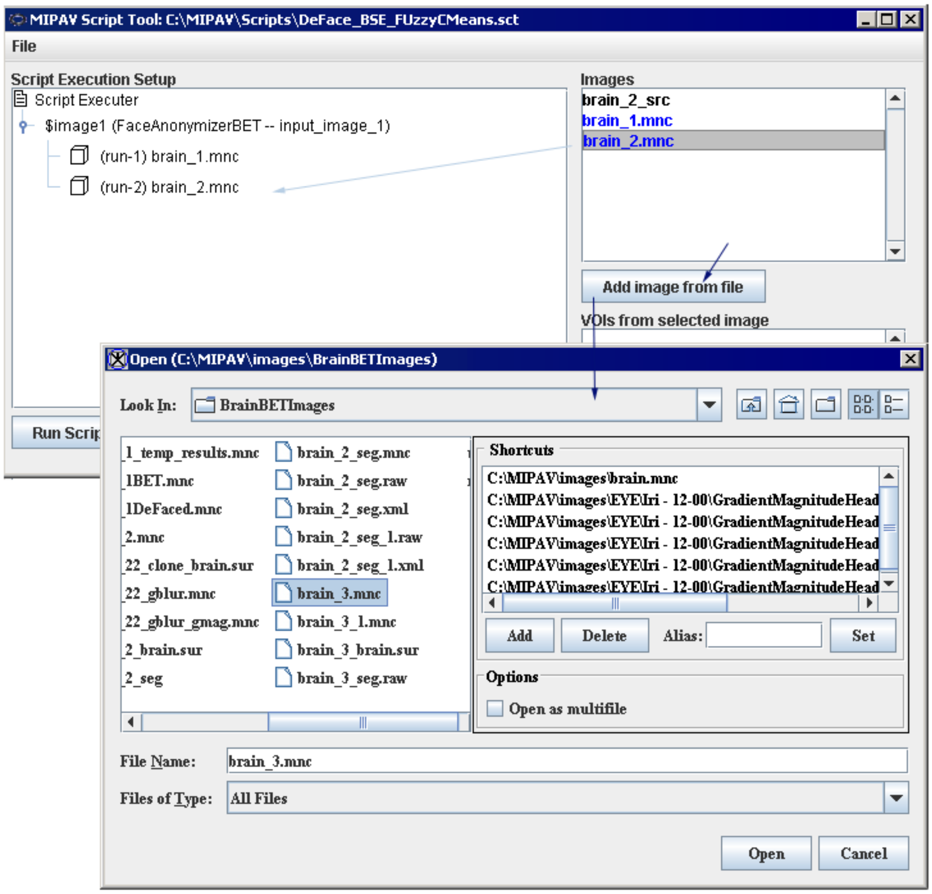
2 Select the script you've just recorded, and press OK;

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.



MIPAV Script Tool: C:\MIPAV\Scripts\DeFace_BSE_FUzzyCMeans.sct

Script Execution Setup

- Script Executer
  - $image1 (FaceAnonymizerBET -- input_image_1)
    - (run-1) brain_1.mnc
    - (run-2) brain_2.mnc

Images

- brain_2_src
- brain_1.mnc
- brain_2.mnc

Open (C:\MIPAV\images\BrainBETImages)

Look In: BrainBETImages

- 1_temp_results.mnc
- 1BET.mnc
- 1DeFaced.mnc
- 2.mnc
- 22_clone_brain.sur
- 22_gblur.mnc
- 22_gblur_gmag.mnc
- 2_brain.sur
- 2_seg
- brain_2_seg.mnc
- brain_2_seg.raw
- brain_2_seg.xml
- brain_2_seg_1.raw
- brain_2_seg_1.xml
- brain_3.mnc
- brain_3_1.mnc
- brain_3_brain.sur
- brain_3_seg.raw

Shortcuts

- C:\MIPAV\images\brain.mnc
- C:\MIPAV\images\EYE\Iri - 12-00\GradientMagnitudeHead
- C:\MIPAV\images\EYE\Iri - 12-00\GradientMagnitudeHead
- C:\MIPAV\images\EYE\Iri - 12-00\GradientMagnitudeHead
- C:\MIPAV\images\EYE\Iri - 12-00\GradientMagnitudeHead
- C:\MIPAV\images\EYE\Iri - 12-00\GradientMagnitudeHead

Options

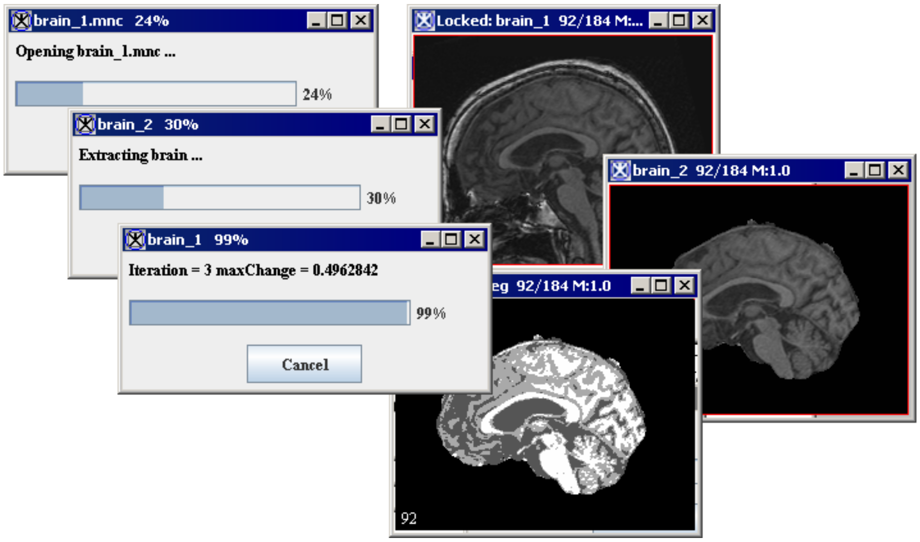
☐ Open as multifile

File Name: brain_3.mnc

Files of Type: All Files

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.



Opening brain_1.mnc ...

Extracting brain ...

Iteration = 3 maxChange = 0.4962842

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,

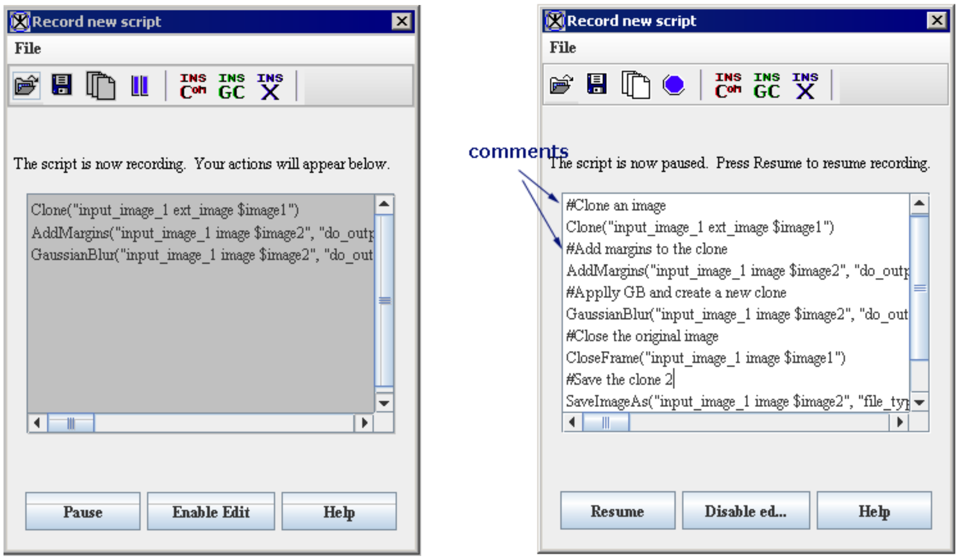
open an image of interest and then, press Start Recording Script button. The Record New Script dialog box appears indicating that recording a script is started, see Figure 308.

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.



The script is now recording. Your actions will appear below.

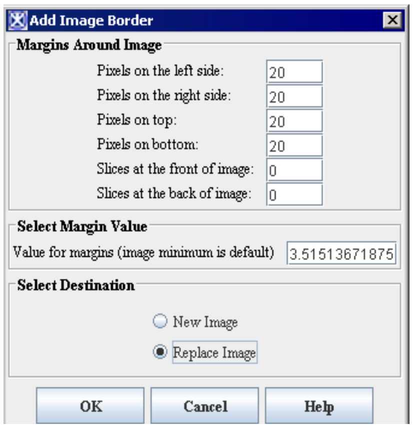
```
Clone("input_image_1 ext_image $image1")
AddMargins("input_image_1 image $image2", "do_outp
GaussianBlur("input_image_1 image $image2", "do_out
```

The script is now paused. Press Resume to resume recording.

```
#Clone an image
Clone("input_image_1 ext_image $image1")
#Add margins to the clone
AddMargins("input_image_1 image $image2", "do_outp
#Applly GB and create a new clone
GaussianBlur("input_image_1 image $image2", "do_out
#Close the original image
CloseFrame("input_image_1 image $image1")
#Save the clone 2
SaveImageAs("input_image_1 image $image2", "file_typ
```

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.



Margins Around Image

Pixels on the left side:	20
Pixels on the right side:	20
Pixels on top:	20
Pixels on bottom:	20
Slices at the front of image:	0
Slices at the back of image:	0

Select Margin Value

Value for margins (image minimum is default) 3.51513671875

Select Destination

○ New Image

● Replace Image

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

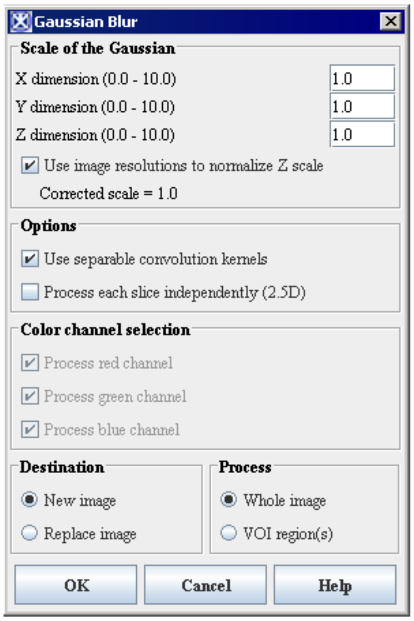
"front int 0", "back int 0" indicates that there are no slices added to the front and back of the image, refer to Figure 309

"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



Gaussian Blur

Scale of the Gaussian

X dimension (0.0 - 10.0) 1.0

Y dimension (0.0 - 10.0) 1.0

Z dimension (0.0 - 10.0) 1.0

☑ Use image resolutions to normalize Z scale

Corrected scale = 1.0

Options

☑ Use separable convolution kernels

□ Process each slice independently (2.5D)

Color channel selection

☑ Process red channel

☑ Process green channel

☑ Process blue channel

Destination

● New image

○ Replace image

Process

● Whole image

○ VOI region(s)

Figure 310. The Gaussian Blur dialog box

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

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.

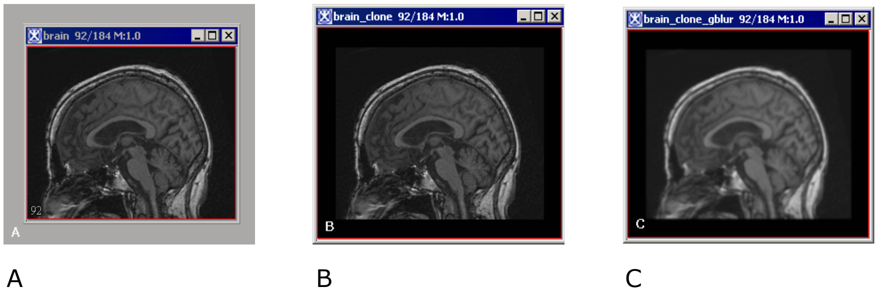


Figure 311. The original image (A), the first clone after applying margins (B), and the second clone (C) after applying Gaussian Blur

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:

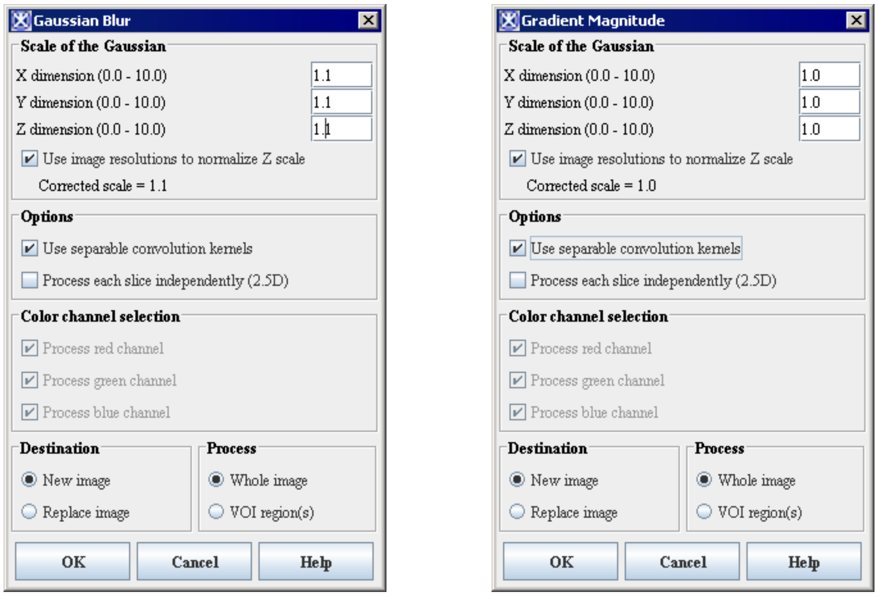
```
GradientMagnitude("input_image_1 image $image3", "do_output_new_image
boolean true", "do_process_whole_image boolean true", "do_process_in_2.5D
boolean false", "do_separable_convolution boolean true", "gauss_std_dev
list_float 1.0,1.0,1.0", "gauss_do_z_resolution_correction boolean true",
"do_process_r_g_b_channel 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.



Gaussian Blur

Scale of the Gaussian
X dimension (0.0 - 10.0) 1.1
Y dimension (0.0 - 10.0) 1.1
Z dimension (0.0 - 10.0) 1.1
☑ Use image resolutions to normalize Z scale
Corrected scale = 1.1

Options
☑ Use separable convolution kernels
□ Process each slice independently (2.5D)

Color channel selection
☑ Process red channel
☑ Process green channel
☑ Process blue channel

Destination
● New image
○ Replace image

Process
● Whole image
○ VOI region(s)

Gradient Magnitude

Scale of the Gaussian
X dimension (0.0 - 10.0) 1.0
Y dimension (0.0 - 10.0) 1.0
Z dimension (0.0 - 10.0) 1.0
☑ Use image resolutions to normalize Z scale
Corrected scale = 1.0

Options
☑ Use separable convolution kernels
□ Process each slice independently (2.5D)

Color channel selection
☑ Process red channel
☑ Process green channel
☑ Process blue channel

Destination
● New image
○ Replace image

Process
● Whole image
○ VOI region(s)

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.

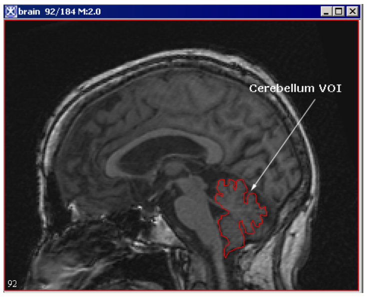
Tip: If the script doesn't appear under Current Script, click the Refresh Script Listing icon and try again.

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

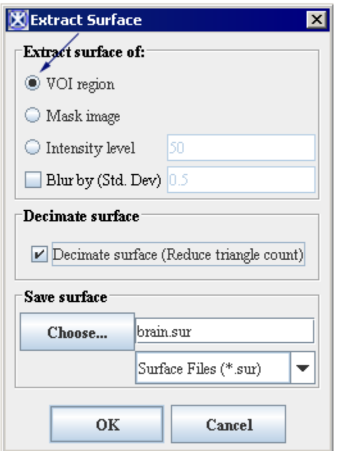
"extraction_type string VOI" corresponds to selecting the VOI region option in the dialog box and indicates that the algorithm is applied only to the image region bounded by the VOI

"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



Extract Surface

Extract surface of:

● VOI region

○ Mask image

○ Intensity level 50

□ Blur by (Std. Dev) 0.5

Decimate surface

☑ Decimate surface (Reduce triangle count)

Save surface

brain.sur

Surface Files (*.sur)

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
```

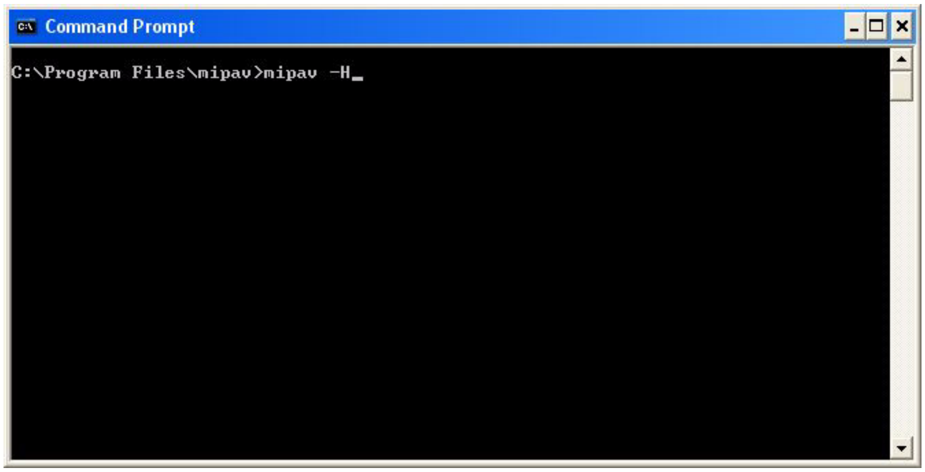
Example 5: Exiting the MIPAV program

```
> mipav Exit
```

Figure 315. Examples of using the mipav command

To display help for using the mipav command

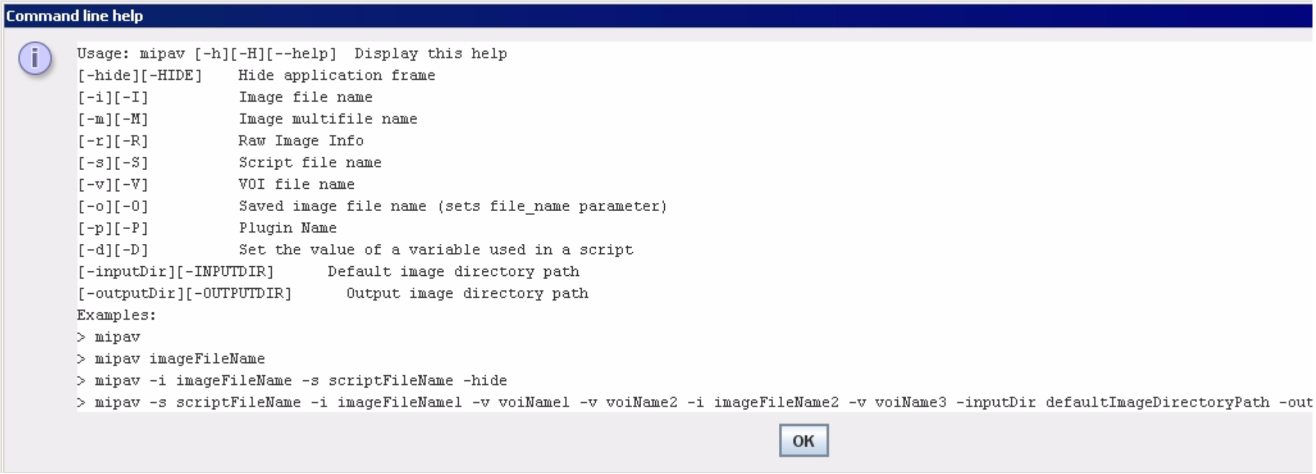
1. Navigate to the mipav directory on your computer.
2. Select Start > All Programs > Accessories > Command Prompt. The Command Prompt dialog box opens.
3. Type **mipav -H** (refer to Figure 316).



```
C:\Program Files\mipav>mipav -H_
```

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.



```
Usage: mipav [-h][-H][--help]  Display this help
[-hide][-HIDE]    Hide application frame
[-i][-I]          Image file name
[-m][-M]          Image multifile name
[-r][-R]          Raw Image Info
[-s][-S]          Script file name
[-v][-V]          VOI file name
[-o][-O]          Saved image file name (sets file_name parameter)
[-p][-P]          Plugin Name
[-d][-D]          Set the value of a variable used in a script
[-inputDir][-INPUTDIR]      Default image directory path
[-outputDir][-OUTPUTDIR]      Output image directory path
Examples:
> mipav
> mipav imageFileName
> mipav -i imageFileName -s scriptFileName -hide
> mipav -s scriptFileName -i imageFileName1 -v voiName1 -v voiName2 -i imageFileName2 -v voiName3 -inputDir defaultImageDirectoryPath -out
```

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.