MeVisHub User Guide

2009

Table of Contents

1 What is MeVisHub?

For end-users: MeVisHub is a transparent bridge between OsiriX and MeVisLab image processing networks. With the help of MeVisHub you can invoke a MeVisLab image processing network from OsiriX and send the image back to OsiriX.
For developers: MeVisHub is an easy way to convert your image processing network to a function module of the PACS workstation software OsiriX.

A little background: MeVisLab and OsiriX
OsiriX is an open-source PACS workstation software for medical image research. It is an ideal platform to handle medical images and provide a user-friendly GUI for medical staffs to organize and view medical images. More important is its compatibility with different types of scanners and PACS system. OsiriX also provides some simple image processing methods and visualization tools for clinical usage, however it is not enough for various research projects. The plugin interface is an important extension to this system, which allow third-part provided algorithm running for special tasks. But developing plugin requires a lot of coding, compiling and testing. It can take very long time.
MeVisLab, on the other side, provides a graphic-based programming and testing environment. This prototyping tool includes hundreds of modules that allow a developer to design a complicate workflow without writing a single code. Another great thing is when you are changing parameters you needed to recompile the software which is very annoying when building a software from C code. A shortcoming of MeVisLab is lack of user-friendly data organization system. When you have more cases saved in the file system, it will become more and more difficult to locate a desired file. Inspired by Felix Ritter's OsiriX MeVisLab Bridge, I started to build this "transparent" bridge between OsiriX and MeVisLab. The idea is to use OsiriX as a front-end GUI for doctors and make MeVisLab running in background invisible to them. Meanwhile the developer can design and test they image processing network inside MeVisLab without knowing how to programming inside

OsiriX.

2 How does it work?

2.1 On MeVisLab side

Image processing networks can be developed and test as they should be. However after you finish a normal mlab file, you need replace the load image module to OsiriXImporter and link all the result module to the OsiriXExporter module. See section 3, for more detail examples.

2.2 On OsiriX side

A developer can configure a user interface using the "configuration" window, where you can define how many parameters and what kinds of drawing tools are allowed in the GUI. After restart OsiriX, the new module will be shown up in the Plugin menu.
For users, they can just run the function by choosing them from the menu and it is just like running other plugins. Most time they will not even notice they are using MeVisLab.

2.3 At run time

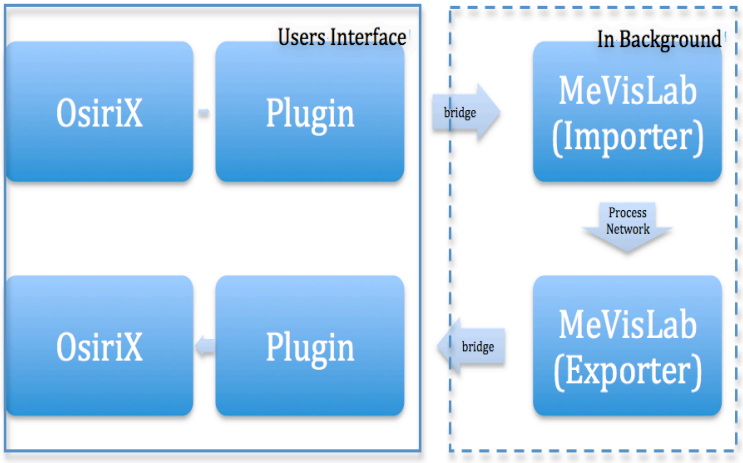

Figure 1. Data flow of a typical operation. Image data and user input is all collected by OsiriX and send to MeVisLab. A multiple-step operation may require the data exchanging for several times.

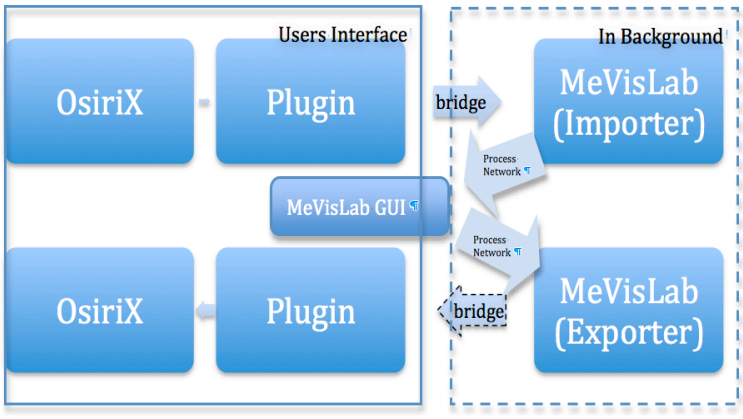

Figure 2. Another possibility of the data flow. Image data is passed down to MeVisLab, and MeVisLab will take the user input and showing the result. Results can be send back to OsiriX if necessary.

As shown in Figure 1, a data flow of a typical operation starts from OsiriX, where the image data and users interaction are transferred from OsiriX to MeVisLab and transferred back after processing. The data-transfer procedure is transparent to the end user who will feel it is OsiriX that does everything. But sometime the interaction between users and image processing software could be very complicated. An alternative way would be directly present the user interface of the image processing software. In this case the bridge will only serve as image loader and result exporting function of the image-processing module (figure 2). The user will recognize the switching between applications, however the switching is much smoother compare to manually run the MeVisLab networks, as the invoke of the function and data transfer are automatically done by OsiriX.

2.4 What happens when MeVisLab and OsiriX are on different computers

If these two computer is located in a local network, OsiriX should still be able connect to the MeVisLab network without any extra configuration. However you should aware, in such case OsiriX will not be able to start MeVisLab on another machine, if it is not running already. And also shared memory technique is not supported in this case, you should disable it from the configuration window.

3 A quick tutorial:

3.1 Example I:Add MeVisLab's thresholding module to OsiriX

Step 1.

a. Create processing network in MeVisLab as shown in figure 3
b. Set “Operation Name” of importer and exporter to “Thresholding”.
c. Set “Para Name0” of importer to “Threshold”
d. Connect “Parameter0Value” of importer to “Threshold” of threshold module
e. save the file to test1.mlab

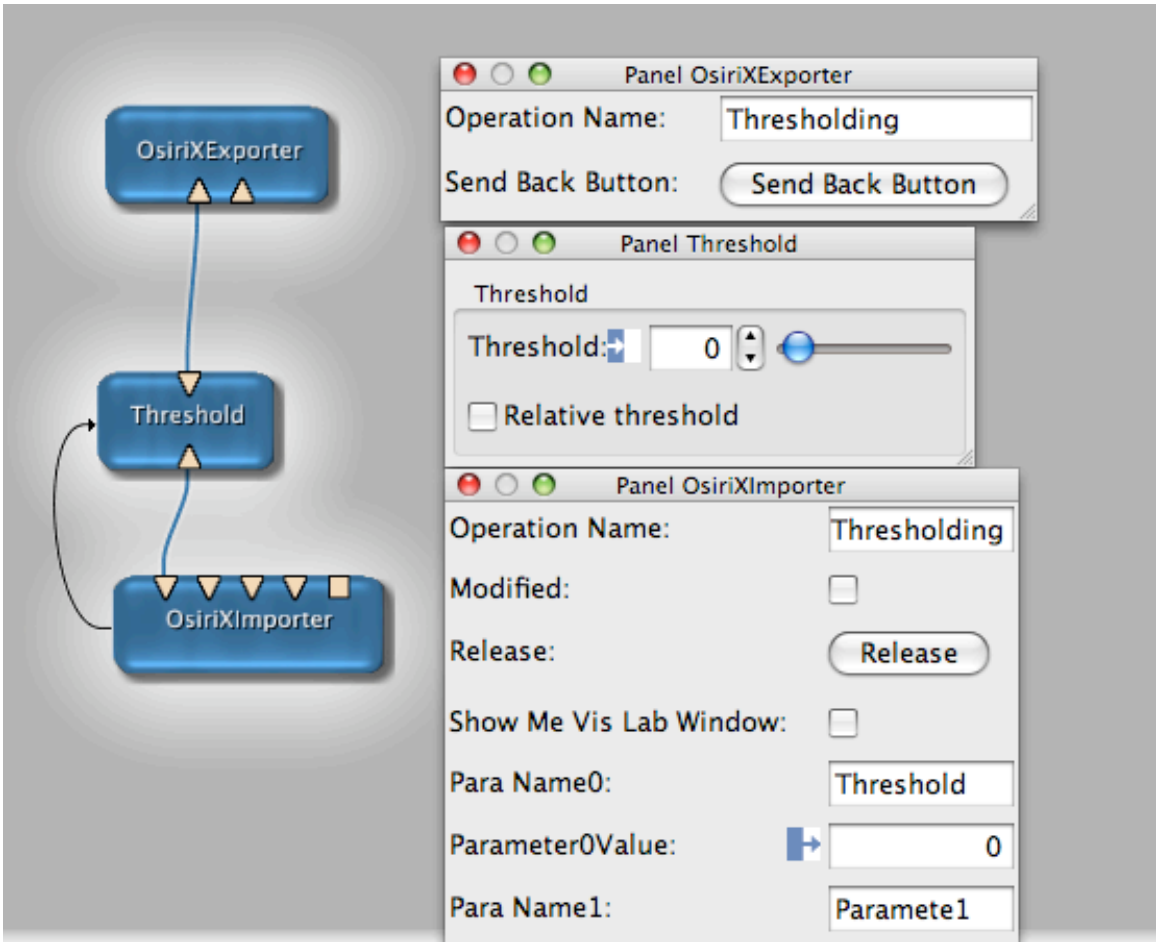

Figure 3. Network in MeVisLab

Step 2.

a. Open Configuration window of MeVisLabHub in OsiriX and add a new operation named "Thresholding" (it has to be the same as the on in step1.b)

b. Create a command line parameter by click "create" button and choose the saved "test1.mlab" file (the plugin make a copy to its recourse folder)

c. Choose Run GUI From OsiriX and add one parameter named "Threshold" and check Realtime option which means updates the result in real time. (Figure 4)

d. save the configuration and restart OsiriX. A new menu item "Thresholding" will show up in the menu list. Run the function a new window will open as in Figure 5

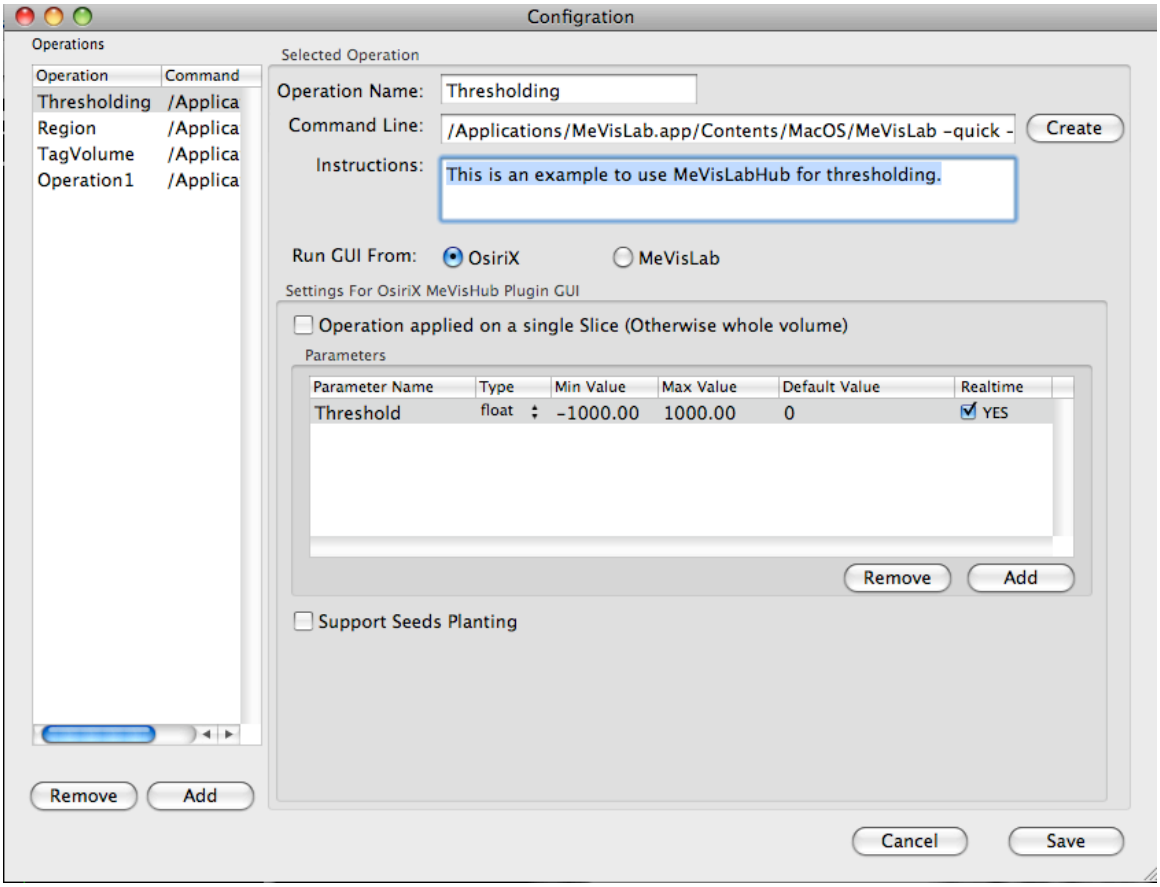

Figure 4. the configuration of MeVisLabHub.

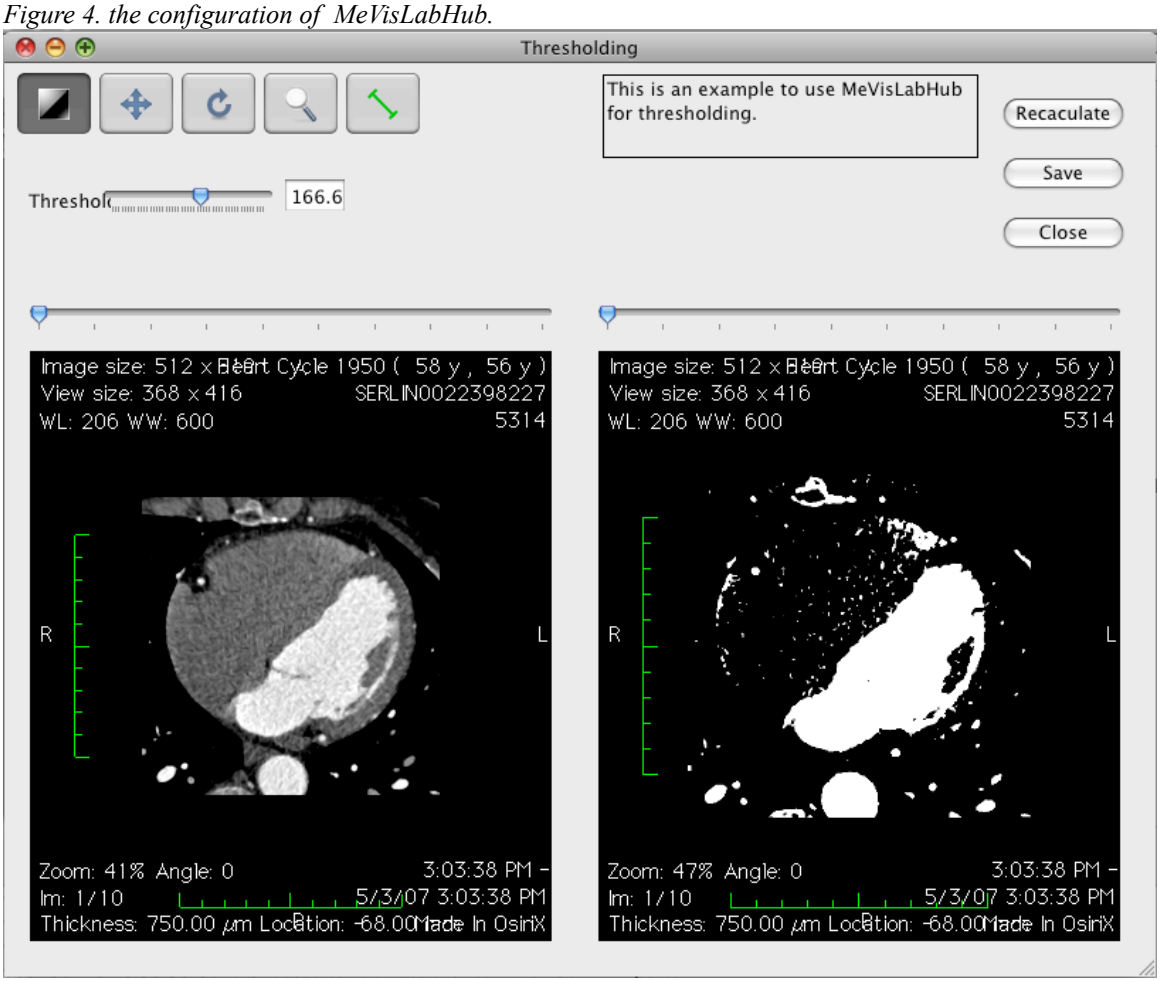

Figure 5 The user interface of "thresholding".

Alternatively, the output of "Threshold" module can be linked to the second input port

"OsiriXExporter" without changing anything else, the thresholding result will be shown as a mask instead Black/White image.

3.2 Example II:Add MeVisLab's RegionGrowing module to OsiriX

Step 1.

a. Create processing network in MeVisLab as shown in figure 6
b. Set "Operation Name" of importer and exporter to "Region".
c. Set "Para Name0" of importer to "Interval"
d. Connect "Parameter0Value" of importer to "Interval" of RegionGrowing module
e. save the file to test2.mlab

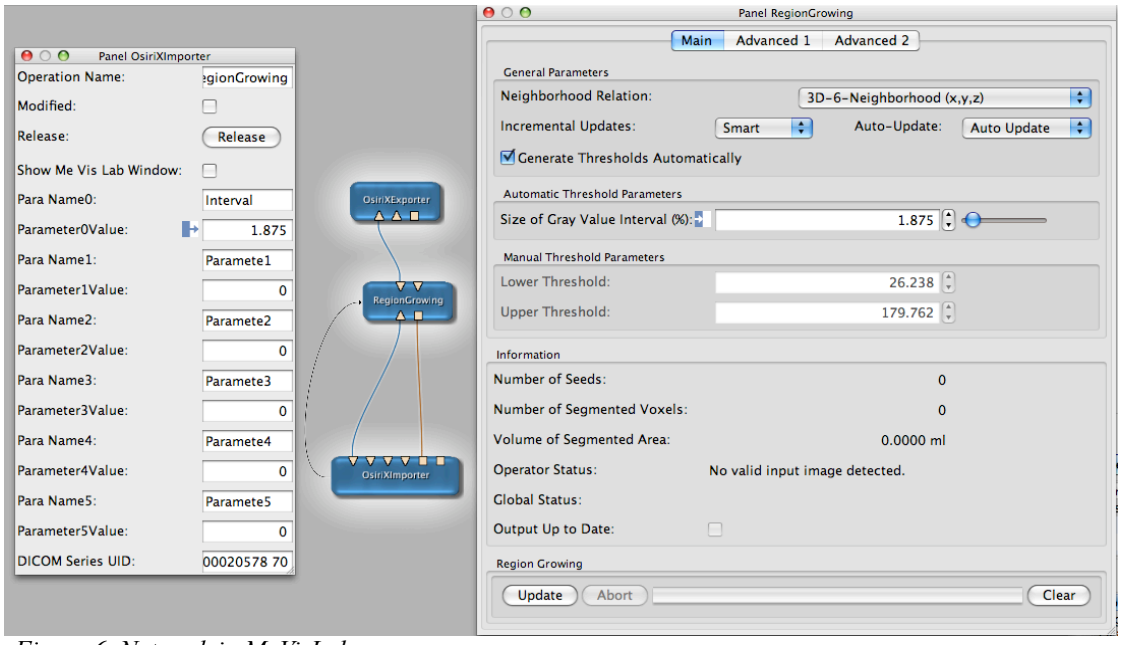

Figure 6. Network in MeVisLab

Step 2.

a. Open Configuration window of MeVisLabHub in OsiriX and add a new operation named "Region" (it has to be the same as the on in step1.b)
b. Create a command line parameter by click "create" button and choose the saved "test2.mlab" file (the plugin make a copy to its recourse folder)
c. Choose Run GUI From OsiriX and add one parameter named "Interval", set default value to 10.0 and check Realtime option which means updates the result in real time. (Figure 7)
d. Check support seeding tools option and create an object named "Object1". And check Realtime option.
e. save the configuration and restart OsiriX. A new menu item "Region" will show up in the menu list. Run the function a new window will open as in Figure 8. It will first give a "failed" warning because no result image is sent back. Ignore it and choose the "Object1" button and click a point on the left window. The region growing result will show in the right window as a blue mask.

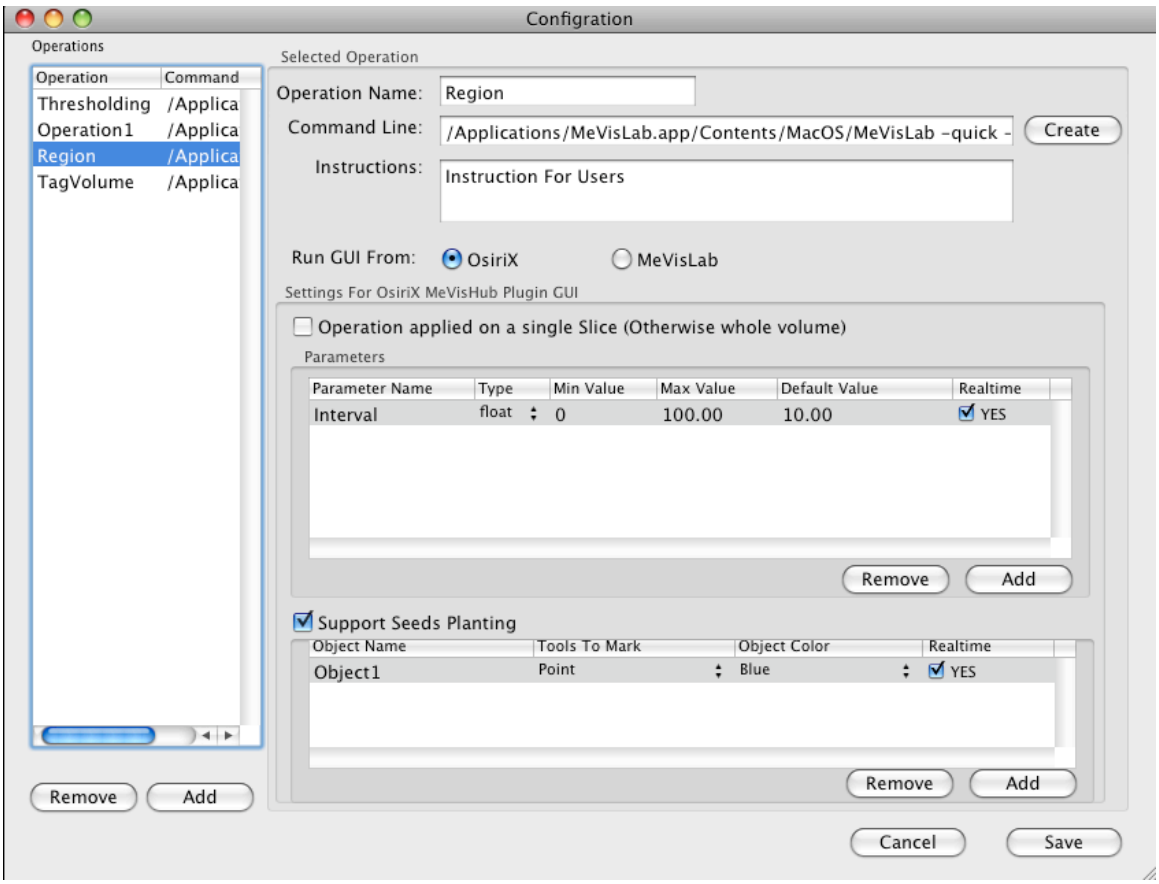

Figure 7. the configuration of MeVisLabHub.

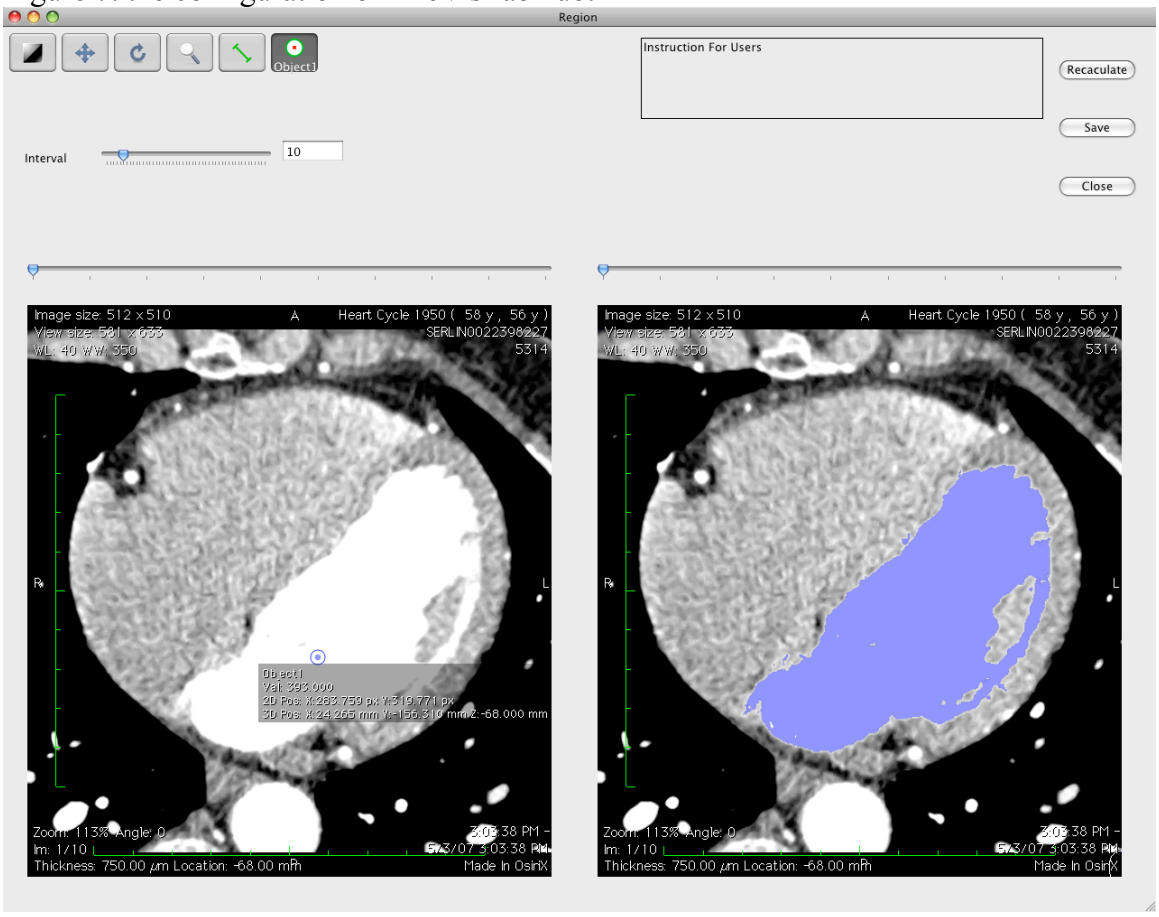

Figure 8 The user interface of “Region”.

3.3 Example III:Add MeVisLab's Surface Rendering module to OsiriX

Step 1.

a. Create processing network in MeVisLab as shown in figure 9.
b. Set “Operation Name” of importer and exporter to “SurfaceRender”.
c. Connect “modified” to “view all” button of SoExaminerViewer.
d. Add two parameters “RotateX” and “RotateY” and link them the RotateXYZ
e. Add Threshold parameter to “ISO value” of IsoSurface
f. Save the file as “test3.mlab”

Important: You should leave the render window open all the time.

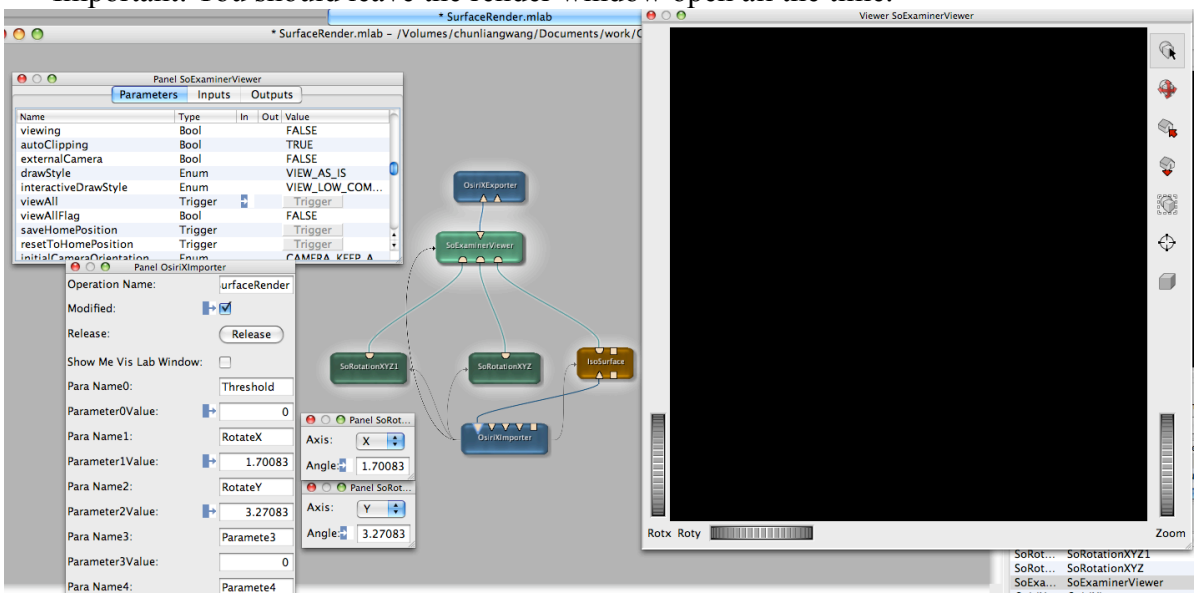

Figure 9. Network in MeVisLab

Step 2.

a. Open Configuration window of MeVisLabHub in OsiriX and add a new operation named “Surface” (it has to be the same as the on in step1.b)
b. Create a command line parameter by click “create” button and choose the saved “test3.mlab” file (the plugin make a copy to its recourse folder)
c. configure the rest parameter as shown in Figure 10

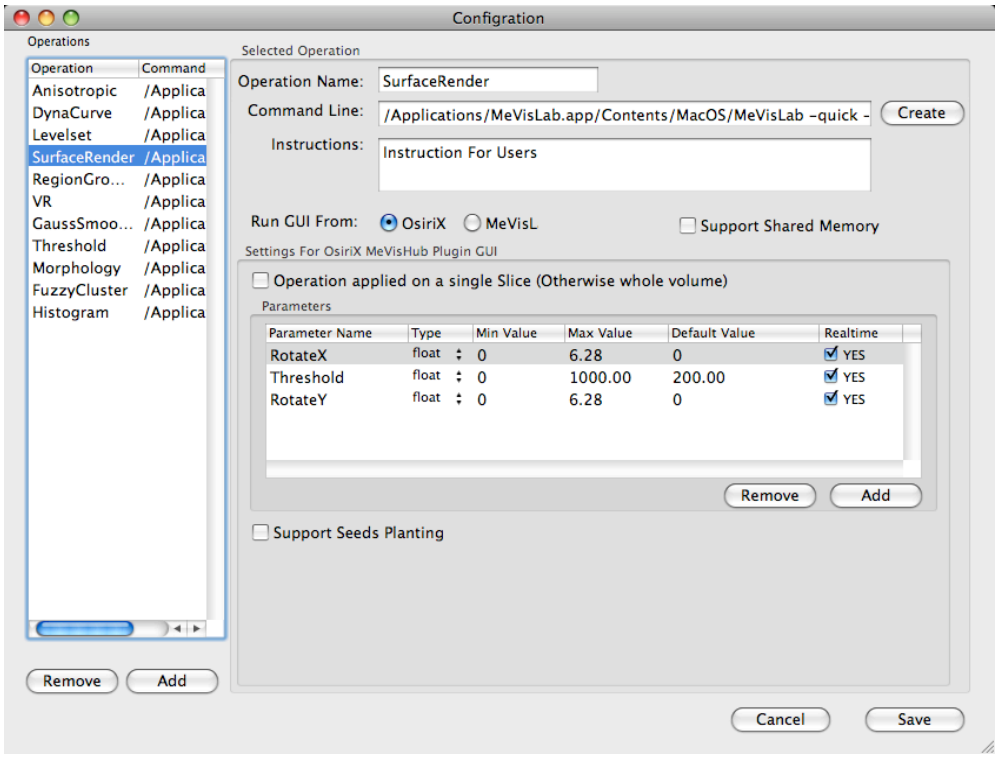

Figure 10. the configuration of MeVisLabHub.

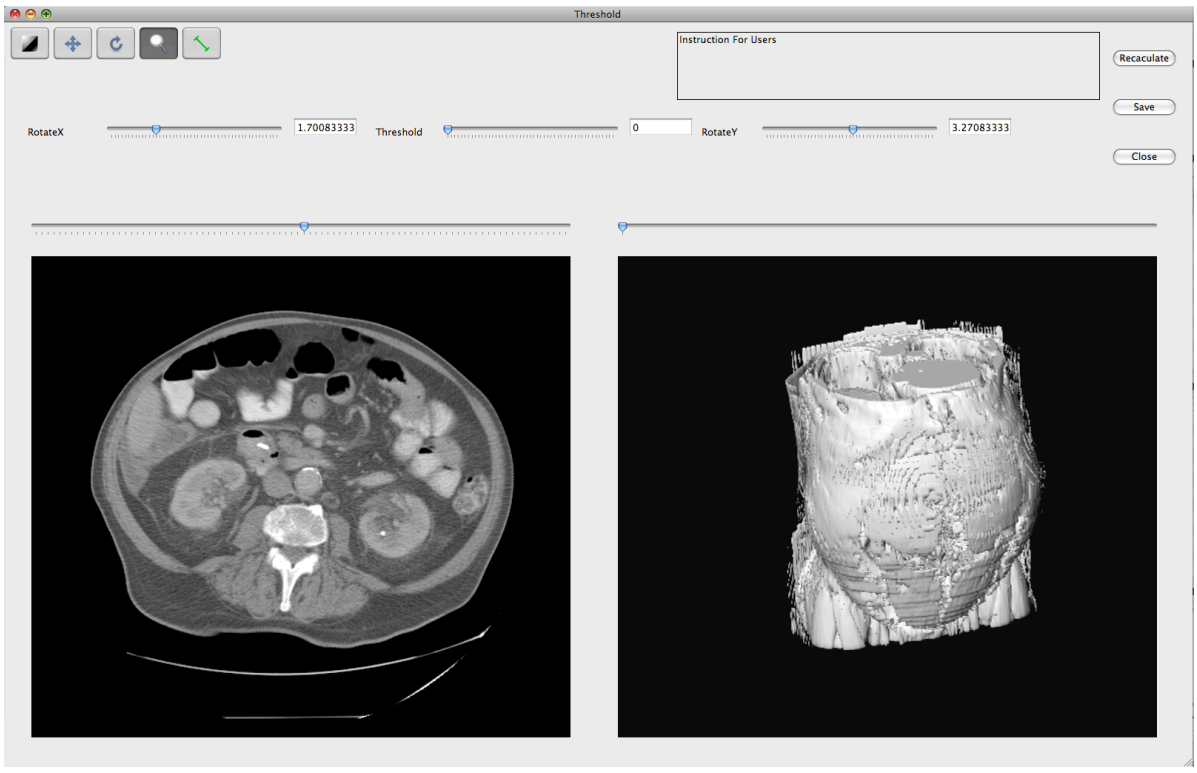

Figure 11 The surface rendering GUI

3.4 More Examples:

Download the 10 simiple examples from :
http://mevislabmodules.svn.sourceforge.net/viewvc/mevislabmodules/trunk/Community/General/Sources/ML/MLOsiriXI

mporter/ .

The 10 examples are list here:

Image Filtering
Gausion Filter Aniostropic Filter Morphological Operation

Image Segmentation
Fuzzy c-means method Region Growing Level-set method

Quantitative Measurement
Region Histogram Dynamic Enhancement Curve

3D Visualization
Iso-Surface Rendering Volume Rendering

4 Known Issues

Only Point and Rectangle ROI are supported so far. But we are working on other ROI marking tools.
The bridge can not handle color images. So even the render image (volume rendering/ geometric models/ dynamic curves) is color image, it will shown as grey image in OsiriX. This hopefull will also be fixed in the future.

Contact:

Chunliang Wang, PhD student
Center for Medical Image Science and Visualization(CMIV)
Linköping University Hospital
SE-581 85 Linköping
Sweden http://www.cmiv.liu.se
Tel. +46-13-228998
Email: chunliang.wang@liu.se