A simple Qt based comparison program for ITK and VTK images.

Release 0.00

Antonin Perrot-Audet, Arnaud Gelas, Kishore Mosaliganti, Nicolas Rannou, Lydie Souhait, Sean Megason

August 12, 2010

Harvard Medical School, Megason lab

Abstract

This document describes a project aimed at simple comparison of images. Such an application can be used in the process of developing an ITK or VTK image processing or data visualization process : the programmer can use it to quickly visualize and compare results of used filters in different parts of the pipeline.
This project is part of the Gofigure2 [1] development effort, an open-source, cross-platform application for visualizing, processing and analyzing of multidimensional microscopy data. The aims of this project are two folds : to be integrated in gdb pretty debuggers [5] (for visual debugging of ITK pipelines), and to provide visual comparison widgets for image to image plugins (ITK or VTK based filters).

Latest version available at the Insight Journal [http://hdl.handle.net/10380/1338]

Contents

Based on Qt libraries [2], ITK [4], VTK [3], and MegaVTK (a variant of vtkINRIA3D engine [6]), we developed an application for images visualization and comparison. We provide:

- a set of classes and widgets for simple integration in a program,
- an executable for command line usage.

1 Principle

The library has been made to simplify the integration of Qt visualization widgets in ITK and VTK programs. Those classes should be easy to use and modify. To achieve this goal, we created the QWidgets using a form (.ui file) for the GUI, which is easy to modify using Qt Designer. We also created a manager class that takes care of widgets creation, deletion and synchronization.

This project includes:

- support for ITK images (see supported format for itk images support details)
- support for VTK images
- a camera synchronization for fine comparison of results.
- QWidget inheritance for integration to a Qt GUI.

2 Examples and Documentation

Documentation

The library is composed by six classes :

QGoSynchronizedView abstract class for QGoSynchronizedView2D and QGoSynchronizedView3D.

QGoSynchronizedView2D class used to display a QWidget containing a two dimensional a vtkimagedata* or an itkimage*. QGoSynchronizedView2D provide the interface to synchronize cameras among several GoSynchronizedView2D.

QGoSynchronizedView2DCallbacks This object takes a list of QGoSynchronizedView and synchronize their cameras by setting up callbacks. It is recommended to let the QGoSynchronizedViewManager deal with SynchronizedView synchronization.

QGoSynchronizedView3D class used to display a QWidget containing a two dimensional vtkimagedata* or itkimage*. QGoSynchronizedView3D provide the interface to synchronize cameras among several GoSynchronizedView3D.

QGoSynchronizedView3DCallbacks This object takes a list of QGoSynchronizedView and synchronize their cameras by setting up callbacks. It is recommended to let the QGoSynchronizedViewManager deal with SynchronizedView synchronization.

QGoSynchronizedViewManager High level class for QGoSynchronizedView2D, QGoSynchronized-View2DCallbacks, QGoSynchronizedView3D, QGoSynchronizedView3DCallbacks. This class deals with QGoSynchronizedViews for correct synchronization and handling provides a simple interface to create/delete/synchronize QGoSynchronizedViews. This class should be used with any class using QGoSynchronizedView and QGoSynchronize.

For implementation details, the reader is directed to the Doxygen documentation and the source code which is heavily documented.

Code Snippets

We introduce here the high level functions for creating QWidgets views and synchronizing visualizations :

Creation of the visualization manager object

```
/* we simply create a new manager that will take care of
 * creation/deletion of visualization and callbacks for us.
 */
QGoSynchronizedViewManager* ViewManager = new QGoSynchronizedViewManager();

// Visualize some images, process etc...

// Remember to delete ViewManager when no visualization is needed
delete ViewManager;
```

Visualization of a VTK image

```
/*  the synchronization manager can create visualization windows given
 *  a valid pointer to a VTK image and
 *  a string encoding the name of the visualization.
 */
ViewManager->newSynchronizedView("My VTK View", VTKSmartPointerToImage);
ViewManager->Update();
ViewManager->show();
```

Visualization of an ITK image

```
/*  the synchronization manager can create visualization windows given
 *  a valid pointer to an ITK image,
 *  the template argument representing the image pixel type,
 *  a string encoding the name of the visualization.
 */
ViewManager->newSynchronizedView<InputPixelType>
  ("My ITK View", ITKSmartPointerToImage);
ViewManager->Update();
ViewManager->show();
```

Synchronization of the camera for several images

```
/*  the synchronization manager can synchronize the opened images
 *  with a simple function call
 */
ViewManager->synchronizeOpenSynchronizedViews();
```

Code Examples

The code source is delivered with a bunch of test and examples, located in Examples/GUI/lib/ Three examples illustrate the code snippets introduced in this article :

compareexample takes a list of 2D or 3D images as an input and displays them in synchronized viewer widgets.

comparepipelineexample takes a 2D or 3D image as an argument and displays this images before and after filtering by a Gaussian filter.

compareguiexample shows how to create a very basic GUI using the functionalities provided by the qgosynchronized classes. Figure 1 and 2 are screenshots of this application.

3 Installation

Software Requirements

You need to have the following software installed:

- Insight Toolkit 3.18 (or higher)
- Visualization Toolkit 5.6.0 (or higher)
- CMake 2.4 (or higher)

Compiling from sources on Linux/MacOsX

Get the latest version of the program from the GIT repository :

```
$ git clone git://github.com/antonin07130/itkCompareProject.git
$ cd itkCompareProject
```

Create a build directory where the compare examples will be compiled

```
$ mkdir BUILD
```

Launch cmake

```
$ cd BUILD
$ ccmake path/to/source/directory
```

Build

```
$ make
```

Test

```
$ ctest
```

The binaries are located in BUILD/bin/

4 Future work

In the future, we plan to :

- integrate this application into gdb pretty debugger for ITK,

- add more pixel types for itk images (vector, tensor images),
- add more features (generate the difference of two images, overlays...)

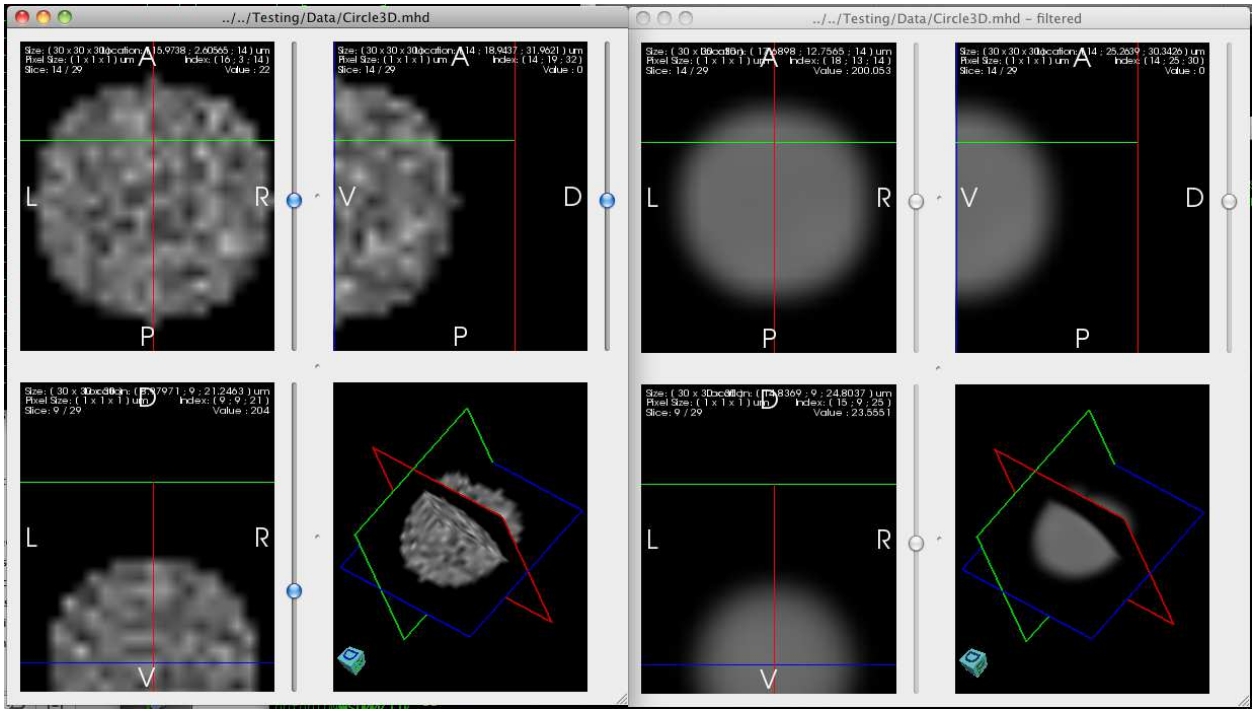

Figure 1: "./compareguiexample" on MacOsX 10.6. The user compares two 3D data sets and uses the quadview to compare them

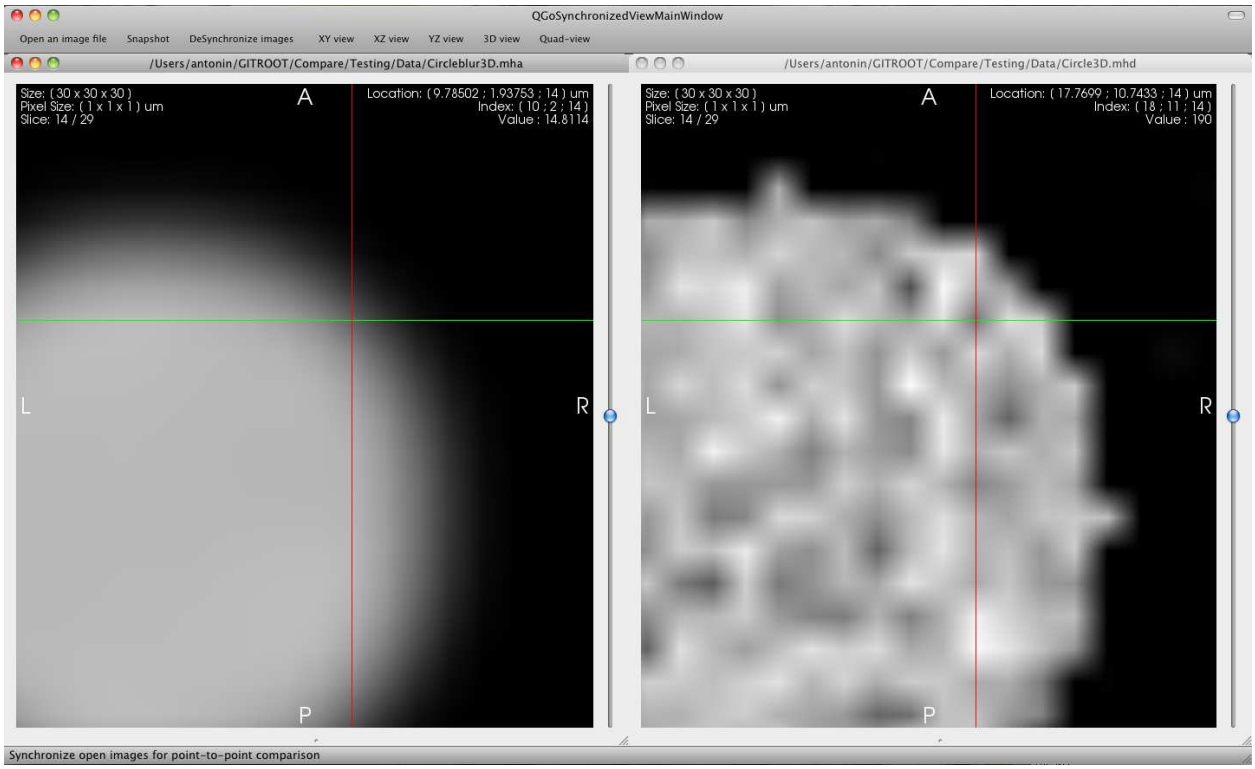

Figure 2: "./compareguiexample" on MacOsX 10.6. The user compares two 3D data sets by visualizing the XY view

References

[1] Gofigure2 image analysis. (document)

[2] Qt - cross-platform application an ui framework. (document)

[3] D. Doria and VTK community. Vtk examples. http://www.vtk.org/Wiki/VTK/Examples. (document)

[4] L. Ibanez, W. Schroeder, L. Ng, and J. Cates. *The ITK Software Guide*. Kitware, Inc. ISBN 1-930934-10-6, http://www.itk.org/ItkSoftwareGuide.pdf, first edition, 2003. (document)

[5] M. McCormick. Visual debugging of itk. Technical report, http://www.kitware.com/products/thesource.html, April 2010. (document)

[6] N. Toussaint, M. Sermesant, and P. Fillard. vtkinria3d: A vtk extension for spatiotemporal data synchronization, visualization and management. In *Proc. of Workshop on Open Source and Open Data for MICCAI*. (document)