# 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 16, 2010

Harvard Medical School, Megason lab

**Abstract**

This document describes a project that can compare many images simultaneously in a synchronized manner. Such an application can be used in any ITK/VTK image processing pipelines; the programmer can use it to quickly visualize and compare intermediate results of used filters within the pipeline.The goals of this project are two folds: (i) to be integrated in gdb pretty debuggers [5] (for visual debugging of ITK pipelines), and, (ii) to provide a visual comparison widget for image-to-image plugins (ITK or VTK based filters).
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.

## Contents

Based on Qt libraries [2], ITK [4], VTK [3], and MegaVTK (a variant of vtkINRIA3D engine [6]), we developed an application for multiple image visualization and comparison. We provide the following items in this submission:

- a set of classes and widgets for simple integration in a generic program developed by a user,
- an executable for command line usage (compareexample).

## 1 Principle

In order to make these classes easy to use and modify, we created an instance of `QWidget` using a form (.ui file) for the GUI using Qt Designer. We also created a manager class that takes care of widgets creation, deletion and synchronization. The manager class (`QObject`) automatically takes care of the following issues:

- Support for ITK images
- Support for VTK images
- Camera synchronization for fine comparison of results.
- QWidget inheritance for integration to an external application with Qt-based GUI.

## 2 Examples and Documentation

### Documentation

The library is composed of six classes :

**QGoSynchronizedView** abstract class for QGoSynchronizedView2D and QGoSynchronizedView3D.

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

**QGoSynchronizedView2DCallbacks** This object takes a list of QGoSynchronizedView objects and synchronizes their cameras by setting up appropriate callbacks. It is recommended to let the QGoSynchronizedViewManager deal with synchronization of objects since it simplifies the burden on the user significantly.

**QGoSynchronizedView3D** class is used to display a QWidget containing a three dimensional vtkimagedata* or itkimage*. QGoSynchronizedView3D provides the interface to synchronize cameras among several GoSynchronizedView3D obhects.

**QGoSynchronizedView3DCallbacks** This object takes a list of QGoSynchronizedView objects and synchronizes their cameras by setting up appropriate callbacks. It is recommended to let the QGoSynchronizedViewManager deal with synchronization of objects since it simplifies the burden on the user significantly.

**QGoSynchronizedViewManager** High level class for QGoSynchronizedView2D, QGoSynchronizedView2DCallbacks, QGoSynchronizedView3D, QGoSynchronizedView3DCallbacks. This class deals with QGoSynchronizedViews for correct synchronization and 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 extensively 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 several tests 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...)

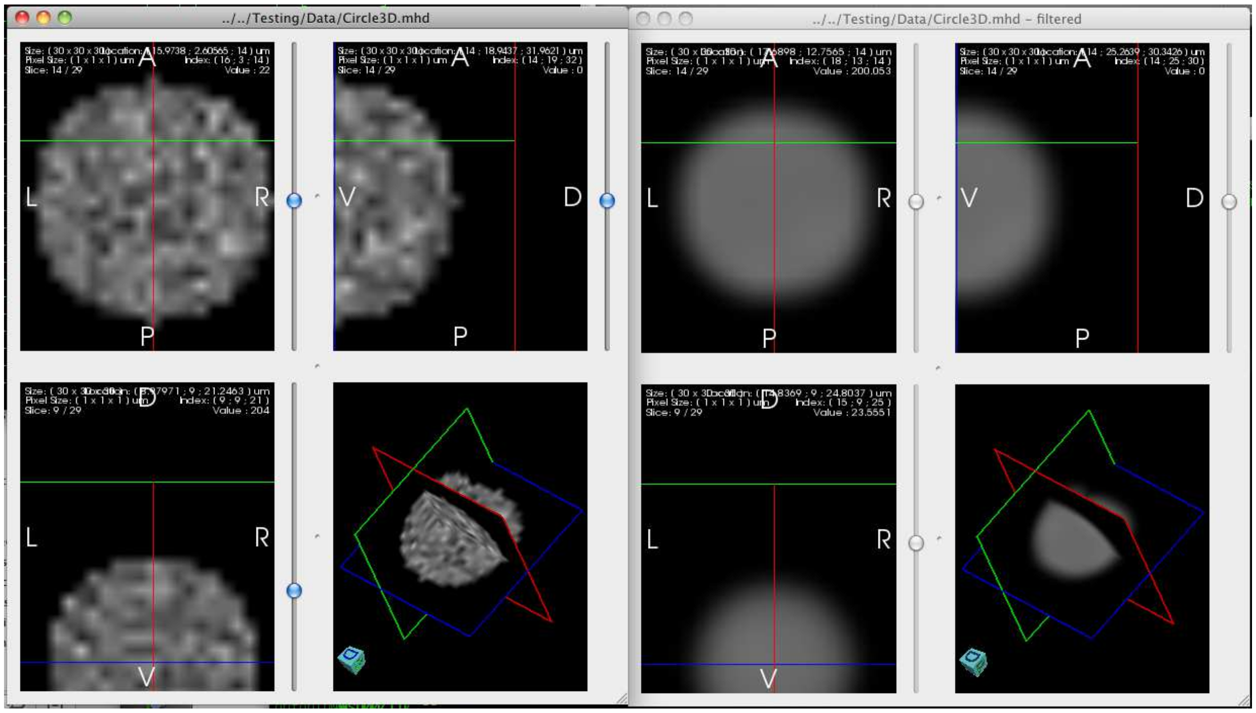
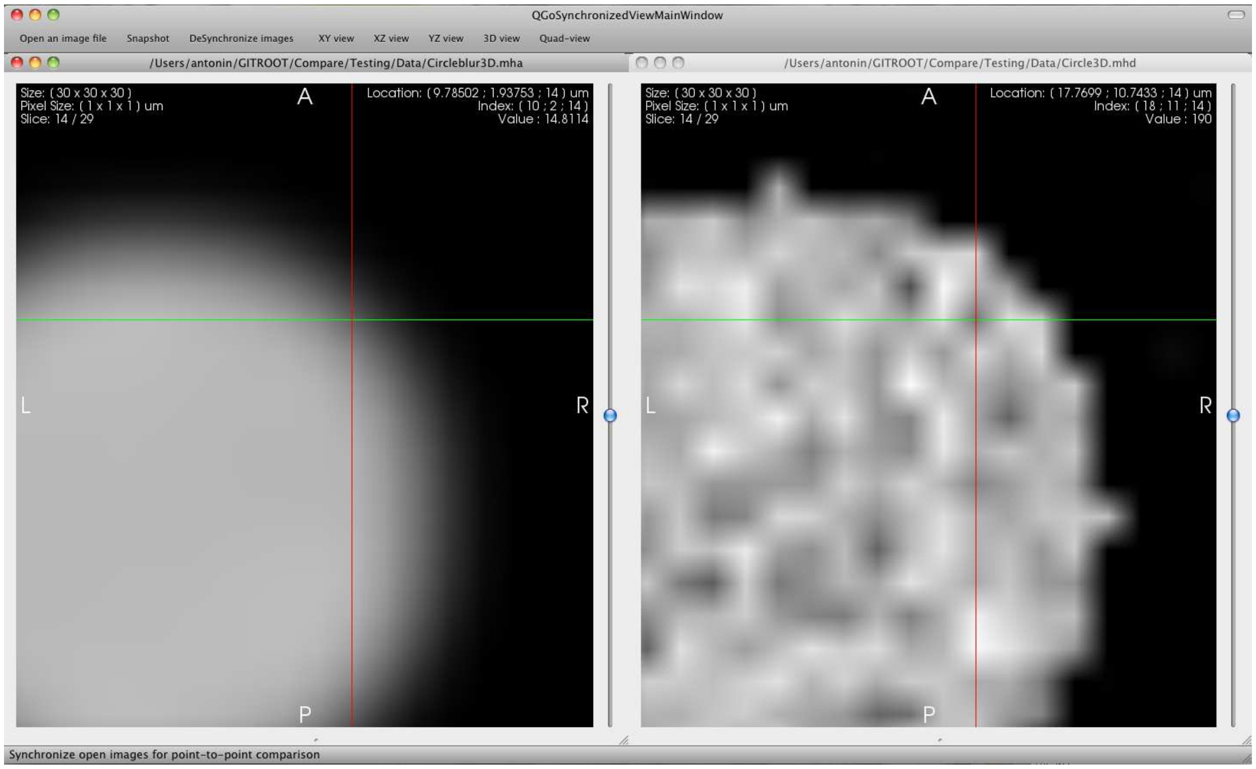

../../Testing/Data/Circle3D.mhd
../../Testing/Data/Circle3D.mhd – filtered
A
P
L
R
V
D

Figure 1: "./compareguiexample" on MacOsX 10.6. The user compares two 3D data sets and uses the quadview to compare them. The quadview provides three the orthogonal views and one isometric 3D view.


QGoSynchronizedViewMainWindow
Open an image file
Snapshot
DeSynchronize images
XY view
XZ view
YZ view
3D view
Quad-view
/Users/antonin/GITROOT/Compare/Testing/Data/Circleblur3D.mha
/Users/antonin/GITROOT/Compare/Testing/Data/Circle3D.mhd
A
P
L
R
Synchronize open images for point-to-point comparison

Size: ( 30 x 30 x 30 )
Pixel Size: ( 1 x 1 x 1 ) um
Slice: 14 / 29
Location: ( 9.78502 ; 1.93753 ; 14 ) um
Index: ( 10 ; 2 ; 14 )
Value : 14.8114

Size: ( 30 x 30 x 30 )
Pixel Size: ( 1 x 1 x 1 ) um
Slice: 14 / 29
Location: ( 17.7699 ; 10.7433 ; 14 ) um
Index: ( 18 ; 11 ; 14 )
Value : 190

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

## 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, Kitware, Inc, 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*, 2007. (document)