October 6, 2011

RITK
THE RANGE IMAGING TOOLKIT

RITK Plugin Tutorial

Jakob Wasza, Sebastian Bauer, Sven Haase

October 2011

1 Why do I need a plugin?

As described in the RITK paper [Was11], the RITK is split up into several parts, that are called plugins. At the beginning there must be a plugin to provide the measured data (*Source Plugin*), in the middle one can add plugins to manipulate this data (*Filter Plugins*) and at the end there should be at least one plugin that represents the application layer of your program (*Application Plugin*). (see Fig. 1)

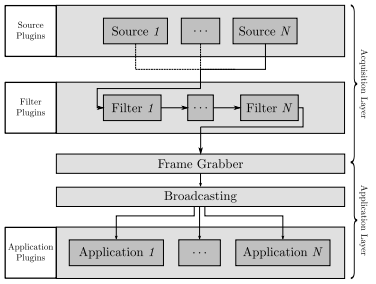

Figure 1: The Plugin Architecture.

2 Where is the plugin creator?

The plugin creator is integrated into the RITK application. Therefore, you can find it either in the source code in the module *ritkCore*, if you want to modify the creator itself anything or, in the case you just want to use it and create your own plugin, you will find the plugin creator in the menu of RITK under "Plugins" → "Plugin Creator".

3 What plugin do I need?

If you want to acquire data from a 3-D sensor that is not yet supported by RITK then you will need a *Source Plugin*. This kind of plugin is placed at the beginning of a RITK pipeline and provides the data for all downstream plugins of the current pipeline.
If you want to manipulate any data read from a file or acquired by 3-D sensor and provided by an already implemented *Source Plugin* then you might need a *Filter Plugin*. This kind of plugin is placed anywhere between a source plugin and an application plugin and can preprocess any payload delivered by the

previous plugin. *Filter Plugins* are optional and not needed to a complete RITK pipeline.
If you just want to present the data in a special way or evaluate the data at the end of the RITK pipeline then you will need an *Application Plugin*. This kind of plugin is placed at the end of the RITK plugin. It provides the GUI and shows data that was acquired and preprocessed by the pipeline.

4 How do I create my own plugin?

After opening the plugin creator, a dialog should pop up (Fig. 2) that gives you all the options needed to create your own plugin. At first you have to define the type of your plugin. Afterwards, you have to name your plugin (note: the extension *plugin* is added to your name automatically) and give a short description. Both of these information will be shown in the plugin manager, when RITK is launched. Although RITK itself is open source developers may also differentiate between open source and closed source plugins. This will have an influence where the sources are created. The last option offers the opportunity to select the RITK modules and external libraries you want to link against.
After creating the plugin, RITK will create a folder in your RITKPlugins source directory named after your plugin and add this plugin to the CMake file. Now you just have to regenerate the *RITKPlugins* solution with CMake and your new plugin will be ready (note: MSVC is able of detecting modifications of CMake files and does this automatically).

5 Where do I write my code?

The created plugin sources are very basic and have no functionality but can be compiled and used straight forward. If you want to add additional GUI elements into your widgets please be aware of the different thread contexts. Therefore, we is recommend to use the signal-slot-system of QT.

5.1 Source Plugin

What additional functions you want to add to your source plugin depends on the 3-D sensor and its interface. At least the function *GenerateOutputInformation()* of *MySourcePluginSourcePluginRImageSource.cxx* must be modified with the correct intrinsic parameters needed to calculate the 3-D data afterwards. Furthermore the same class includes the function *GenerateRImage()* where you have to acquire the data from your sensor and set it as an output of the current plugin. Additionally, one can modify the data before providing it to the pipeline in the file *MySourcePluginSourcePlugin.cxx* in the function *RequestFrame*. If you want to add some additional payload to the current RImage (e.g. the skeleton of the Kinect) you can do this by implementing a new RImage and include this new RImage into your plugin.

5.2 Filter Plugin

In most cases you want to filter the range values of your input. Therefore the class *MyFilterPluginImageFilter* was created that represents a wrapper class for an itk image filter. In the function *ThreadedGenerateData()* of this class you can implement your filter algorithm. If you want to filter other values as well (e.g. RGB or WorldCoords) then you can implement those procedures into the *DoFiltering()* function of *MyFilterPluginRImageFilter.txx*.

5.3 Application Plugin

The generated plugin just includes a visualization widget. If you want to add processing algorithms before representing the data, you will need to do this in the *ProcessEvent(ritk::Event::Pointer EventP)* function of *MyApplicationPluginPlugin.cxx*. For any additional help please take a look at existing application plugins.

6 How can I load my plugin in RITK?

First of all, you don't have to modify RITK itself in any way to get your plugin loaded. After building the *INSTALL* project ot *RITKPlugins* your plugin is automatically loaded and added to the plugin manager. The plugin manager can either be called from the menu "Plugins" → "Plugin Manager" or by using the shortcut "Ctrl+M". The dialogue should look like Fig. 3 and provides the functionality to start or stop any plugin loaded.
Furthermore, you can find a plugin configuration file in your RITK binary directory in the folder "PluginConfigs". This functionality is not yet fully supported, but allows you to load a prepared pipeline to safe you time for loading plugins and parameter adjustment. During the startup of RITK the "default.xml" is loaded. If you want to have additional configuration files, you can load them from the menu "Plugins" → "Load Plugin Configuration".

References

[Was11] S.; Haase S.; Schmid M.; Reichert S.; Hornegger J. Wasza, J.; Bauer. RITK: The Range Imaging Toolkit A Framework for 3-D Range Image Stream Processing. In *Proceedings of Vision, Modeling, and Visualization*, 2011.

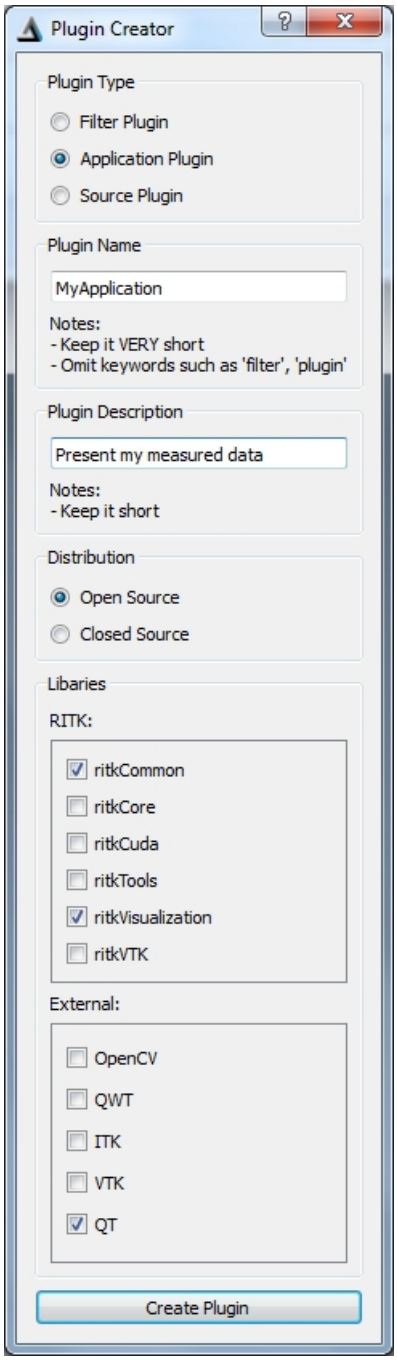

Figure 2: The Plugin Creator of RITK (future version).

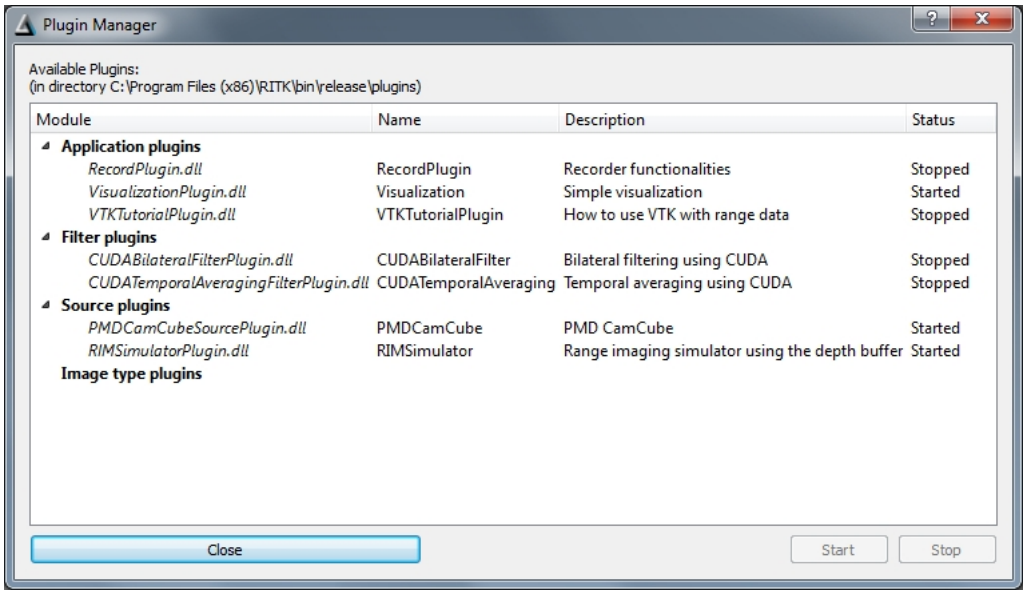

Figure 3: The Plugin Manager of RITK.