Plug-In System

Jian Liu

Vaa3D

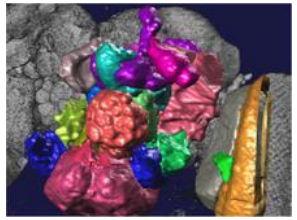
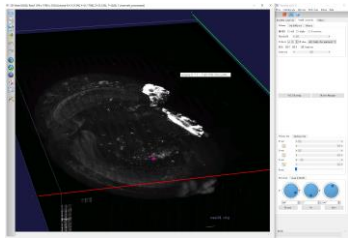
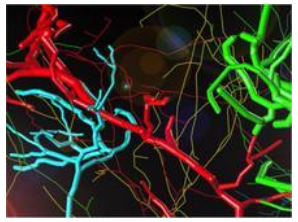


Image visualization

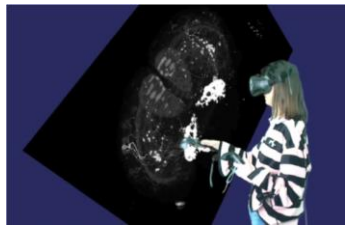


Terafly

Vaa3D



Neuron morphology analysis



TeraVR

Plug-In System

- Useful plugins
  - Data analyzing
  - Image acquisition
  - Neuron tracing
  - Neuronal morphology analysis
  - Vaa3D-ITK plugins

  (https://github.com/Vaa3D/Vaa3D_Wiki/wiki/PluginList.wiki)

- Extensibility
  - Custom plug-in development interface

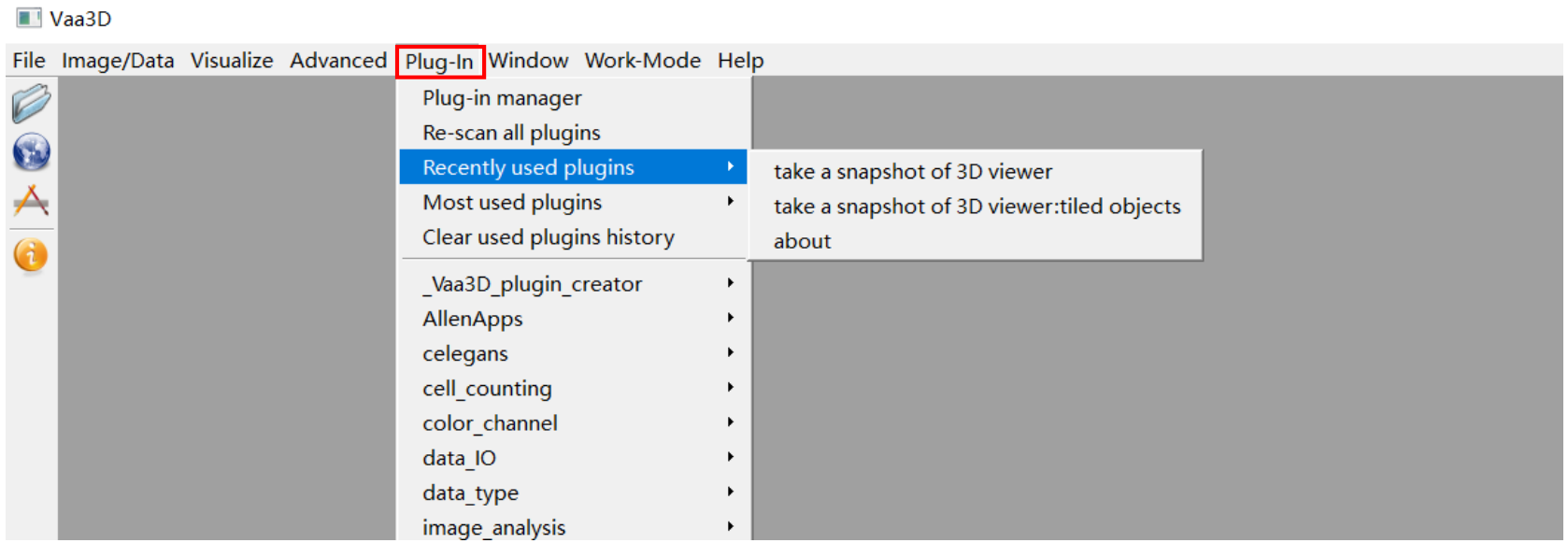
Start from Vaa3D main window

Where to find Vaa3D plugins' executable?

In the Vaa3D executable package, the default Vaa3D plugins are released under the "plugins" folder, parallel to the Vaa3D executable.

You can also find the available plugins using the following three convenient methods:

- Go to Vaa3D GUI main menu and select the "Plug-In" menu, then select "Plugin Manager". You will see a tree-list of all plugins, including their locations and version information.
- Go to Vaa3D GUI main window, left side control button and select the "A" button (means "applications"). Then you can see a toolbar appears on the top of the main window. You can then click this toolbar button, and you can define short-cuts to all plugins and their internal menu functions.
- On command line console (e.g. Linux , Mac or Windows terminal), type the command (assuming you are running Linux. If you run Mac, you may need to use 'vaa3d64.app/Contents/MacOS/vaa3d64' to replace 'vaa3d' below).



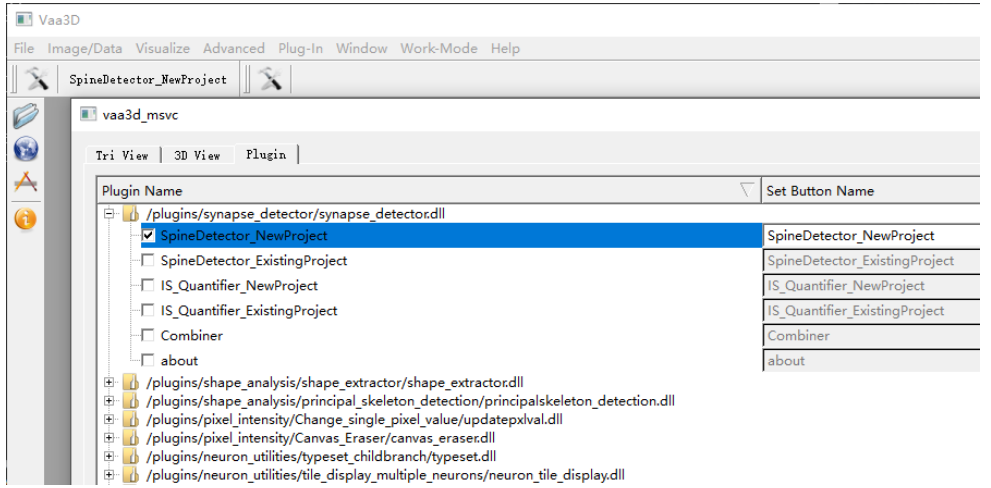
Start from Vaa3D main window

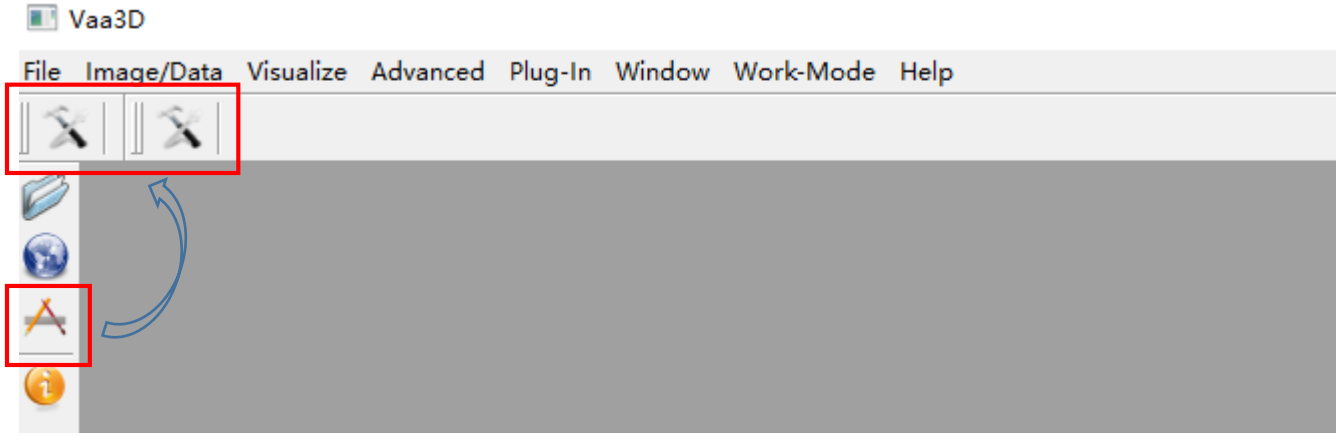
Where to find Vaa3D plugins' executable?

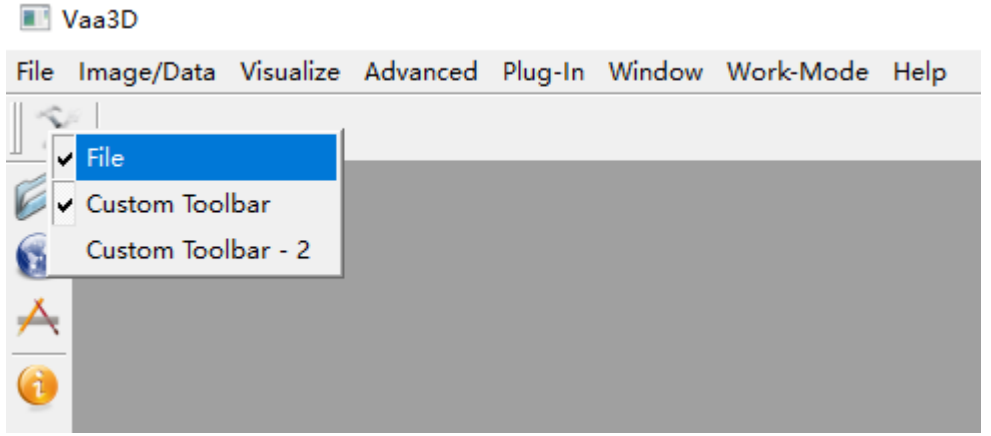
In the Vaa3D executable package, the default Vaa3D plugins are released under the "plugins" folder, parallel to the Vaa3D executable.

You can also find the available plugins using the following three convenient methods:

- Go to Vaa3D GUI main menu and select the "Plug-In" menu, then select "Plugin Manager". You will see a tree-list of all plugins, including their locations and version information.
- Go to Vaa3D GUI main window, left side control button and select the "A" button (means "applications"). Then you can see a toolbar appears on the top of the main window. You can then click this toolbar button, and you can define short-cuts to all plugins and their internal menu functions.
- On command line console (e.g. Linux , Mac or Windows terminal), type the command (assuming you are running Linux. If you run Mac, you may need to use 'vaa3d64.app/Contents/MacOS/vaa3d64' to replace 'vaa3d' below).







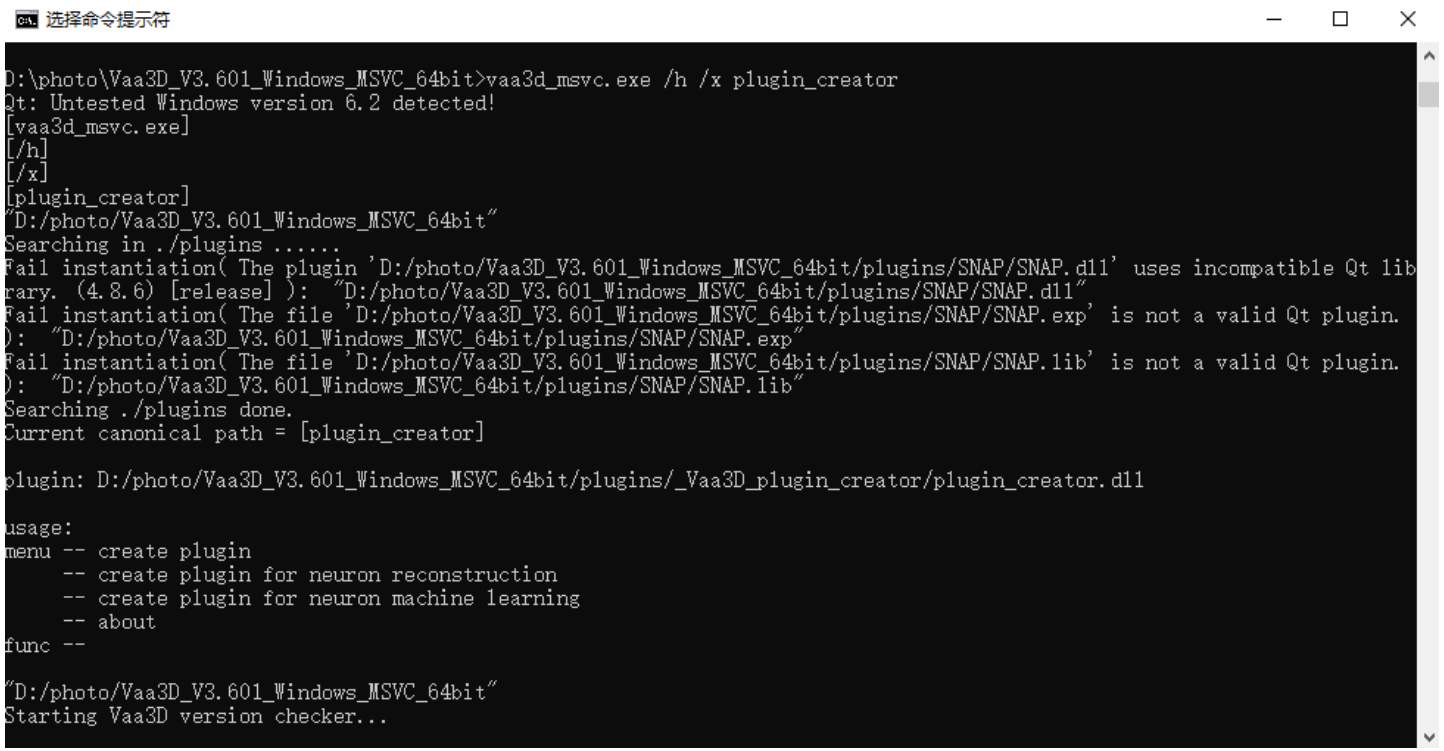
Start from command line

- Get complete list of all available plugins:

vaa3d -h (Mac OS and Linux)
vaa3d_msvc.exe /h (Windows)

- Get individual help information of a plugin:

vaa3d -h -x <plugin_name> (Mac OS and Linux)
vaa3d_msvc.exe /h /x <plugin_name> (Windows)



How to write a plugin?

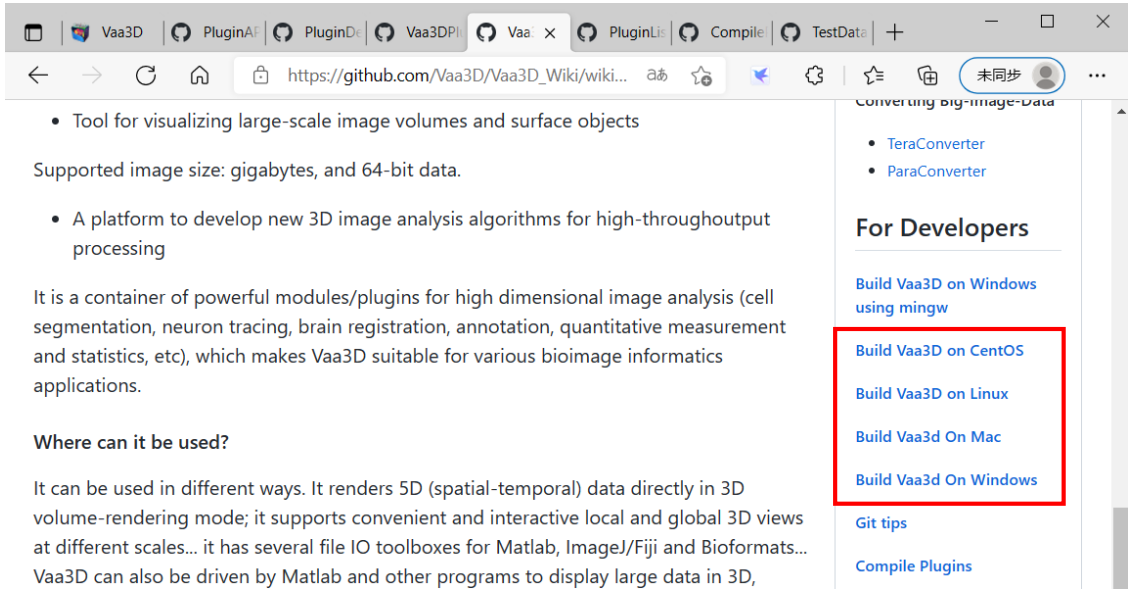
- Structure of a plugin project
  - plugin.h
  - plugin.cpp
  - plugin.pro

- Templates automatically generate
  - _Vaa3D_plugin_creator

- Plugin API
  - https://github.com/Vaa3D/Vaa3D_Wiki/wiki/PluginAPI.wiki

Preparation

- Qt properly installed;
- Vaa3D source code;
- a C++ compiler (e.g. gcc)

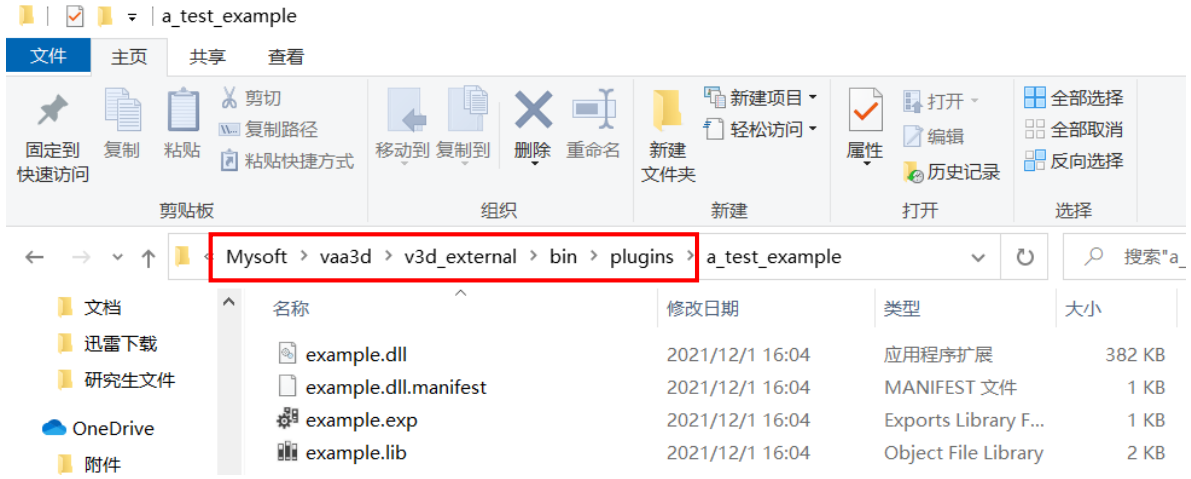


Compile

- Run the command line Terminal of Visual Studio 2013

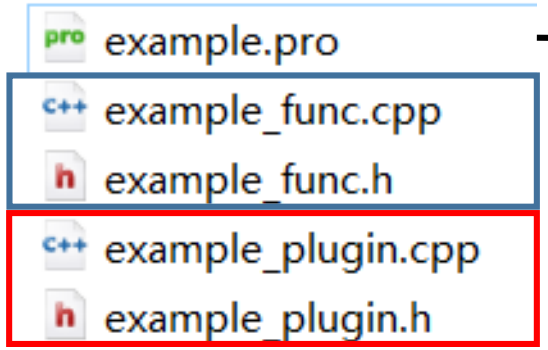
```
qmake <plugin_project_file>
nmake -f Makefile.Release          # Compile the plugin.
```

- Copy the compiled plugin dynamic library file to plugin folder



Example

- File structure:

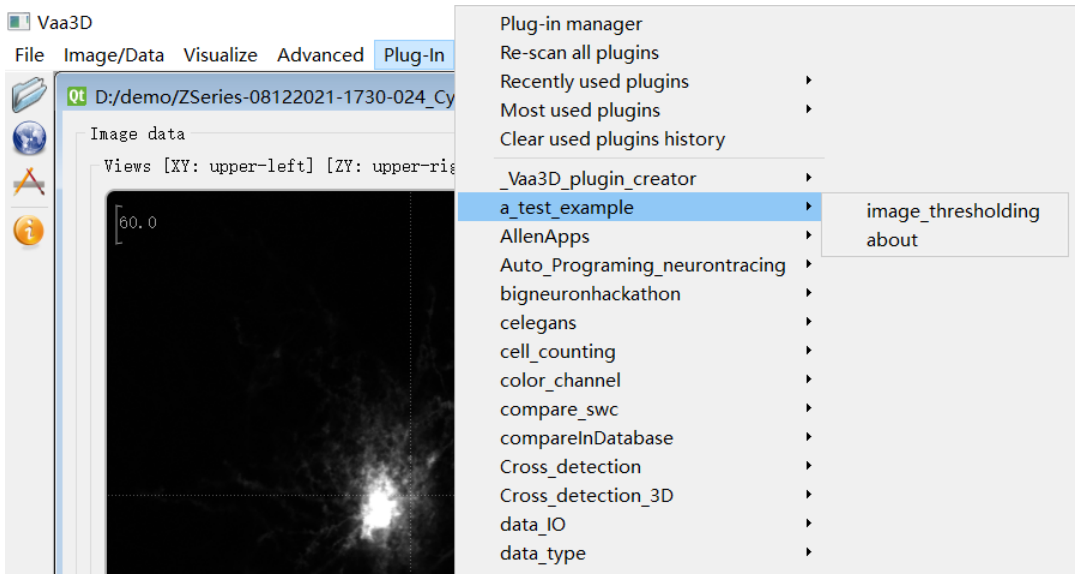


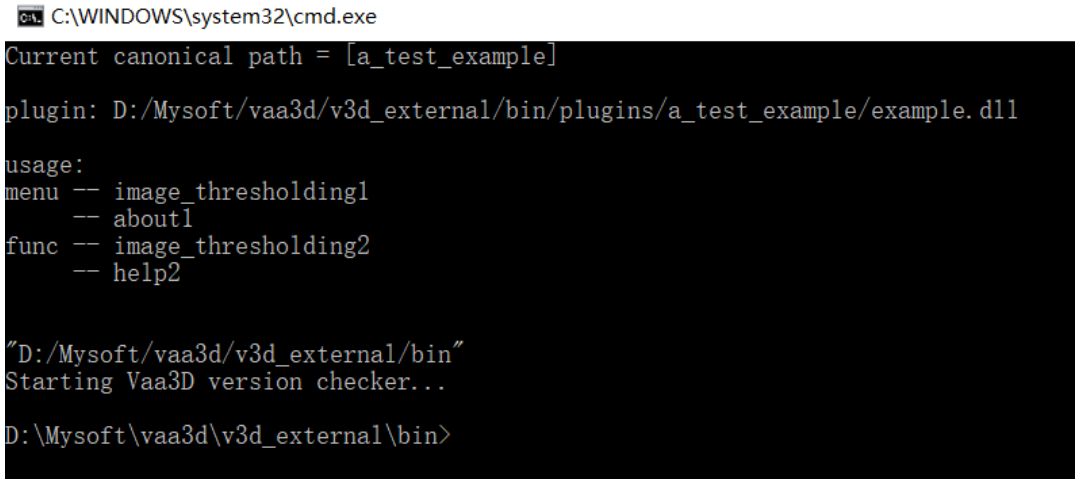

```
#an example plugin project file
TEMPLATE        = lib
CONFIG  += qt plugin warn_off
#CONFIG += x86_64

#set the Vaa3D main path
V3DMAINPATH     =  ../../../v3d_external/v3d_main

#include necessary paths
INCLUDEPATH     += $$V3DMAINPATH/basic_c_fun
INCLUDEPATH     += $$V3DMAINPATH/common_lib/include
LIBS += -L. -lv3dtiff -L$$V3DMAINPATH/common_lib/lib

#include the headers used in the project
HEADERS += example_plugin.h
HEADERS += example_func.h

#include the source files used in the project
SOURCES = example_plugin.cpp
SOURCES += example_func.cpp
SOURCES += $$V3DMAINPATH/basic_c_fun/v3d_message.cpp
SOURCES += $$V3DMAINPATH/basic_c_fun/stackutil.cpp
SOURCES += $$V3DMAINPATH/basic_c_fun/mg_image_lib.cpp
SOURCES += $$V3DMAINPATH/basic_c_fun/mg_utilities.cpp
SOURCES += $$V3DMAINPATH/basic_c_fun/basic_memory.cpp


#specify target name and directory
TARGET  = $$qtLibraryTarget(example)
DESTDIR = ../../v3d/plugins/example/
```

Example

- Plugin framework:

```
/* example_plugin.cpp
 * This is an example plugin perform binary thresholding on
 * 2012-02-10 : by Yinan Wan
 */

#include "v3d_message.h"

#include "example_plugin.h"
#include "example_func.h"

// 1- Export the plugin class to a target, the first item in
Q_EXPORT_PLUGIN2(example, ExamplePlugin);


// 2- Set up the items in plugin domenu
QStringList ExamplePlugin::menulist() const
{
        return QStringList()
                <<tr("image_thresholding")
                <<tr("about");
}

// 3 - Set up the function list in plugin dofunc
QStringList ExamplePlugin::funclist() const
{
        return QStringList()
                <<tr("image_thresholding")
                <<tr("help");
}
```





Example

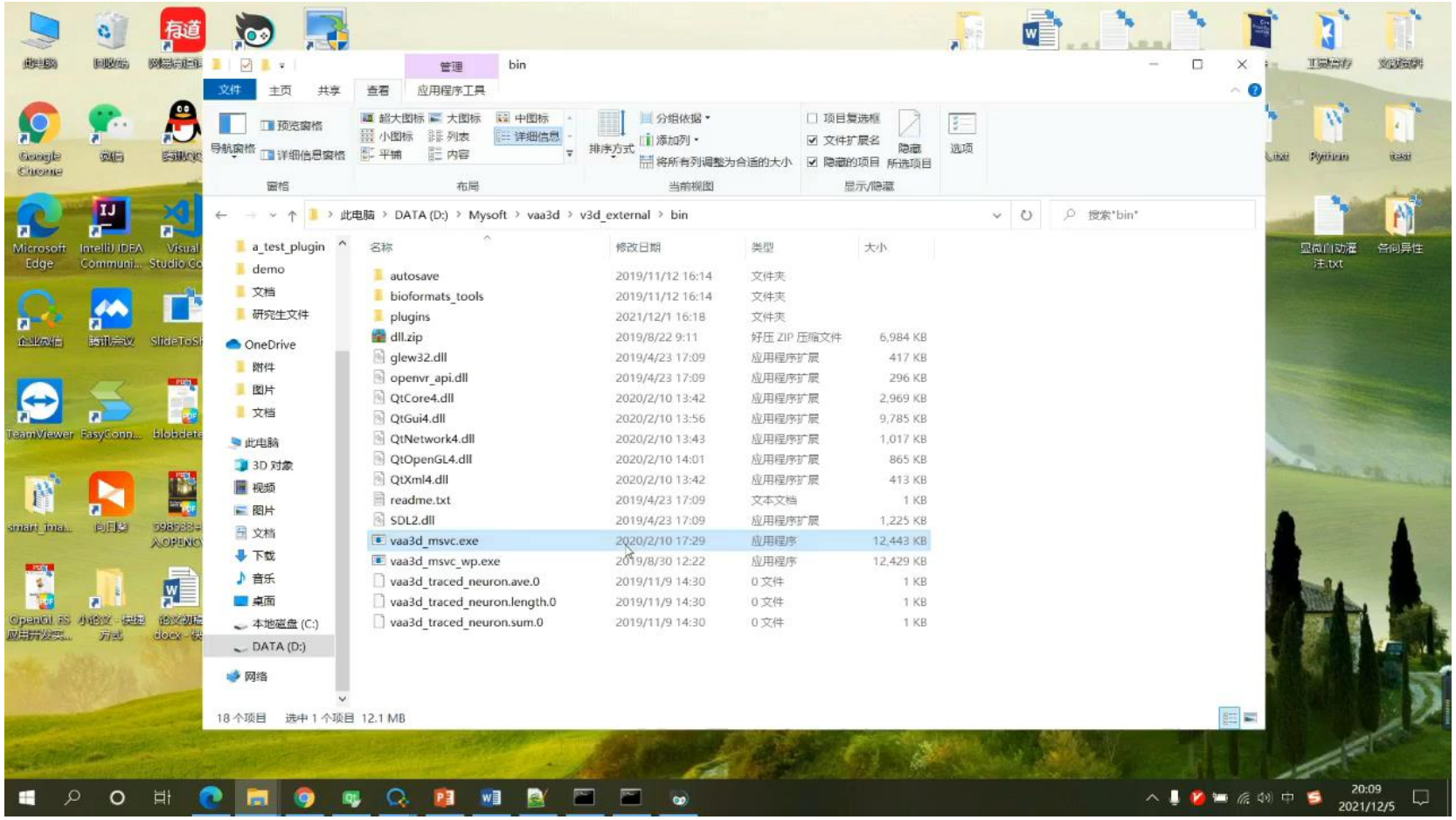
- Plugin framework:

```
/* example_plugin.cpp
 * This is an example plugin perform binary thresholding on
 * 2012-02-10 : by Yinan Wan
 */

#include "v3d_message.h"

#include "example_plugin.h"
#include "example_func.h"

// 1- Export the plugin class to a target, the first item in
Q_EXPORT_PLUGIN2(example, ExamplePlugin);


// 2- Set up the items in plugin domenu
QStringList ExamplePlugin::menulist() const
{
        return QStringList()
                <<tr("image_thresholding")
                <<tr("about");
}

// 3 - Set up the function list in plugin dofunc
QStringList ExamplePlugin::funclist() const
{
        return QStringList()
                <<tr("image_thresholding")
                <<tr("help");
}
```

```
// 4 - Call the functions corresponding to the domenu items.
//     The functions may not necessarily be in example_func.cpp, but you are
//     to seperate the Interface from the core functions, and it is consistan
void ExamplePlugin::domenu(const QString &menu_name, V3DPluginCallback2 &call
{
        if (menu_name == tr("image_thresholding"))
        {
                image_threshold(callback,parent);
        }
        else if (menu_name == tr("about"))
        {
                v3d_msg(tr("This is a demo plugin to perform binary threshold
                        "Developed by Yinan Wan, 2012-02-10"));
        }
        else
        {
                v3d_msg(tr("This is a demo plugin to perform binary threshold
                        "Developed by Yinan Wan, 2012-02-10"));
        }
}

// 5 - Call the functions corresponding to dofunc
//     The functions may not necessarily be in example_func.cpp, but you are
//     to seperate the Interface from the core functions, and it is consistan
bool ExamplePlugin::dofunc(const QString & func_name, const V3DPluginArgList
{
        if (func_name == tr("image_thresholding"))
        {
                image_threshold(input, output);
        }
        else if (func_name == tr("help"))
        {
                printHelp();
        }
}
```

Example

- Plugin test:



Thank you!

Jian Liu