3D RECONSTRUCTION PROJECT

VOXELSDK INSTALLATION

July 21, 2016

Yannick PORTO
Arjun BALAKRISHNAN

INTRODUCTION

This tutorial relates the easiest manners and most useful tools to install and use the voxel SDK in order to read and process the data of the T.I. OPT8241 TOF camera. It allows also a bridge to the Point Cloud Library (PCL) for visualizing and handling point clouds.

INSTALLATION ON WINDOWS

The voxel SDK package and its documentation can be found in github :
`https://github.com/3dtof/voxelsdk` . If you follow the wiki webpage, you will have all the information for installing the package into your system.

We have followed the installation on windows x64. This lists all the dependencies the voxel SDK requires. In this list are mentioned *Qt 5 for mscv13* and *PCL 1.7.2.*, so it explicitly says that you need to have Visual Studio 2013 installed on your PC. You can get it for free with the following link : `https://www.visualstudio.com/fr-fr/downloads/download-visual-studio-vs.aspx`

All the step mentioned in this wiki page have been followed, except that we have just taken a different PCL installer that they mentioned, because the structure of the dependencies folder (e.g. boost) was not the same and this was easier to find the libraries automatically. The one we have downloaded is the *"PCL 1.7.2 All-in-one Installer MSVC2013 x64"* from the following website : `http://unanancyowen.com/?p=1255&lang=en`.

The version of the voxel SDK we have downloaded is the *Voxel-SDK-0.6.0-win64-debug.exe*

If you encounter any warnings during the installation saying that the path in the environment variables is too long, make sure you have :

- **VOXEL_SDK_PATH** = %MAIN_VOXELSDK_FOLDER_PATH%
- **PCL_ROOT** = %MAIN_PCL_FOLDER_PATH%
- **PATH** = %MAIN_VOXELSDK_FOLDER_PATH%/bin;
  %MAIN_VOXELSDK_FOLDER_PATH%/lib;%MAIN_PCL_FOLDER_PATH%/bin;
  %MAIN_PCL_FOLDER_PATH%/lib;%PATH%

CODING IN VISUAL STUDIO

Once the voxel SDK is installed, here are the steps to start coding using its libraries on Windows.

- Open Visual Studio 2013
- Create a **new Project**
- Select **console application win32**
- In the pop-up window, check **"Empty project"** ("Projet vide")
- In the tool bar, go to "**Build**" ("Generer") and click on "**Configuration Manager**" ("Gestionnaire de configuration")
- Go to "**Active solution platform**" ("Plateforme de solution active") and select "**New**" ("Nouveau")
- Choose the new platform "**x64**", instead of "ARM", and validate.
- Go inside the "**Property Manager**" ("Gestionnaire de propriete"). This can be found in the pannel on the right.
- Right click on "**Debug | x64**" and "**Add New Project Property Sheet**" ("Ajouter une nouvelle feuille de proprietes de projet")
- Add a new **.props** file ; you can name it "voxelsdk.props"
- Open this file
- In "**C/C++->General**", write the path to the voxelsdk folder containing the header files inside "**Additional Include Directories**" ("Autres Repertoires Include") which should be somthing like $C:\backslash ProgramFiles\backslash VoxelSDK0.6.0\backslash include\backslash voxel-0.6.0$
- In "**Linker->General**" ("Editeur de liens->General"), give the path to the libraries in "**Additional Library Directories**" ("Repertoires de bibliotheques supplementaires"), which should be something like $C:\backslash ProgramFiles\backslash VoxelSDK0.6.0\backslash lib$
- In "**Linker->Input**", ("Editeur de liens -> entree") give all the name of libraries you need in "**Additional Dependencies**" ("Dependances supplementaires") like *ti3dtof.lib; voxel.lib; voxelpcl.lib*

Notes

Some errors or warnings can occurs while using certain voxelsdk functions. You can avoid them by going to the "**Properties**" of the source file in question, then "**C/C++->Preprocessor**". Just add *_CRT_SECURE_NO_WARNINGS;* to the "**Preprocessor definitions**" line.

INSTALLATION ON LINUX

Access to the same wiki page than before in `https://github.com/3dtof/voxelsdk` and follow the **Installation on Linux** this time. We have downloaded the *libvoxel-0.6.1-all.tar.gz* but any other version is fine.

For PCL, the version they suggest (from apt-get) is not suitable. Voxelsdk requires the C++11 standard to compile without error and some of the functions of PCL, working with Boost (like the filter module), will crash with this version. So PCL needs to be compiled from source.

To do so, the following website provides all the steps to compile PCL from source properly:
`https://larrylisky.com/2014/03/03/installing-pcl-on-ubuntu/`
The only thing to change is the variable **CMAKE_CXX_FLAGS** in cmake. For that, you can run **cmake-gui** instead of the terminal command *cmake -DCMAKE_BUILD_TYPE=None ...* at the end of the tutorial.
Inside the gui, just reference the folder ".../pcl" as the source code, and the folder ".../pcl/release" to store the binaries. Press **Configure** and you will get many variables. Write **-std=c++11** in the variable mention before to change the flag of the c++ standard. We encountered some problems with OPENNI during the installation so we recommend to uncheck the two variables **WITH_OPENNI** and **WITH_OPENNI2**. Then you can **Generate** and follow the rest of the tutorial online.

CODING IN QT (LINUX)

The main advantage to program under Qt is the fact that GUI's are easy to implement. We did not find the way to build a proper application under Visual Studio with the libraries Voxelsdk and PCL. If Qt is not installed yet in your system you can install it by the terminal command: **sudo apt-get install qt-sdk**.

Once **qt-creator** is opened, open a new console project or application. As you might want to use the module **QVTKWidget** from vtk to display a PCLVisualizer inside an application, you have to switch your kit from Qt5 to Qt4. Go into **Tools->Options->Build and Run->Qt Versions**. If Qt4 is not automatically detected, add a new version and look for the **qmake** executable into the main folder of qt4. This should be inside a folder like *"/usr/lib/x86_64-linux-gnu/qt4/bin/qmake"*. Then go to the panel **Kits** and change the **Qt version** of the current Kit.

For starting coding you just have to reference the source code and library locations in the **.pro** file of your project :

- The first line you can add is **QMAKE_CXXFLAGS += -std=c++11** makes Qt use the c++ standard 11 (for Voxelsdk).
- Add all the source code locations after **INCLUDEPATH +=**
  pcl, voxel, eigen3, boost and vtk need to be included and all these folders should be located inside **/usr/include**, except pcl which should be inside **/usr/local/include** since it has been compiled from source.
- Add all the libraries locations after **LIBS +=**
  They should all be located inside **/usr/lib** and **/usr/local/lib**
  (Write **-L** before the libraries paths
- Add all the libraries names after **LIBS +=**
  You also have to write **-l** following by the name of the library like it is written on the **.so** file. For example, for including **libpcl_common.so**, you have to write **-lpcl_common** .

The project is now ready to include all those libraries and to build an application.