2 Guide to Building Regard3D

This section describes in detail the steps I took to get Regard3D to build on a Linux system.

2.1 Hardware and OS

I am using the KVM system to host various virtual machines. The one for this project is configured to have 6 CPUs and 16GB of memory. The file system is ext4 which is sitting on a QEMU qcow2 file. The /home file system is an NFS mount to another virtual machine which manages access to a 1TB SSD.

The OS is Bodhi Linux. This was chosen for its lightweight Enlightenment desktop environment and because it uses Ubuntu underneath. It appears image processing software is more readily available for Ubuntu [and perhaps Debian] than other operating systems.

I have found that during building and running that over 10GB of RAM is required, so 12GB is probably the minimum for this project. Six is probably also a reasonable minimum for CPUs - more would have been better. I would also avoid using NFS - some of the code is not very efficient when using it.

2.2 Installed Packages

The following packages were installed from the repositories using the usual "sudo apt-get install" command:

git	p7zip	cmake
g++	gfortran	gdb
valgrind	qtbase5-dev	libqt5svg5-dev
libwxgtk3.0-gtk3-dev	libboost-dev	libboost-all-dev
libopenscenegraph-dev	libeigen3-dev	libopencv-dev
libceres-dev	libassimp-dev	libpng-dev
libjpeg-dev	libtiff-dev	libxxf86vm1
libxxf86vm-dev	libxi-dev	libxrandr-dev
graphviz	libflann-dev	libgsl-dev
libf2c2-dev	libfreefem++-dev	

You might also like to install MeshLab. [sudo apt-get install meshlab]

2.3 Downloads

Download Regard3D and OpenMVG from
https://sourceforge.net/projects/regard3d/files/Regard3D/1.0.0/

These are p7zip files, but if you have installed p7zip your archive manager should be able to handle them. I used Regard3D-master, but I am not seeing that on the SourceForge site now. I will repeat my build steps using Regard3D_src_1.0.0.7z

2.3.1 Support Programs

It was not obvious initially that there are additional programs necessary. You will need to get the following:

cmvs-fix2.tar.gz

graclus1.2.tar.gz

mve-master.zip

mvs-texturing-master.zip

PoissonRecon-master.zip

smvs-master.zip

The sites for these are:

http://www.di.ens.fr/cmvs/

https://www.gcc.tu-darmstadt.de/home/proj/mve/index.en.jsp

https://github.com/flanggut/smvs

http://www.cs.jhu.edu/~misha/Code/PoissonRecon

https://www.gcc.tu-darmstadt.de/home/proj/fssr/index.en.jsp

https://www.gcc.tu-darmstadt.de/home/proj/texrecon/

2.4 Building

I have all my downloads for this project in a Packages directory.

Create a directory, r3d, and set up an environment variable to point to it:

```
mkdir r3d
cd r3d
export R3D=$(pwd)
```

2.4.1 OpenMVG

Unpack the zip file and create a short cut.

[Be careful with p7zip, it will delete the input file without the -k option.]

```
cd $R3D
p7zip -d -k $PACKAGES/openMVG-1.4_r3d.7z
ln -s openMVG-1.4_r3d openMVG
cd openMVG
```

I needed to make a few changes to the build instructions.

Edit the src/CMakeLists.txt file. This change should only be for private use; if you are doing it for commercial reasons then you should seek appropriate guidance. Find and change:

```
# Included for research purpose only
#add_subdirectory(nonFree)
add_subdirectory(nonFree)
```

Edit src/openMVG/multiview/CMakeLists.txt. Find and change:

```
target_link_libraries(openMVG_multiview
  PUBLIC
    openMVG_numeric
  PRIVATE
    openMVG_graph
    ${CERES_LIBRARIES}
    minilog
)
```

Edit src/openMVG/system/CMakeLists.txt. Find and change:

```
target_link_libraries(openMVG_progress_test
INTERFACE ${OPENMVG_LIBRARY_DEPENDENCIES} minilog)
```

Edit src/openMVG_Samples/cameras_undisto_Brown/CMakeLists.txt. Find and change:

```
target_link_libraries(openMVG_sample_cameras_undistoBrown
  openMVG_geometry
  openMVG_image
  openMVG_multiview
  minilog
  ${STLPLUS_LIBRARY})
```

Create two directories, one for the build objects, and one to install into:

```
cd $R3D
mkdir openMVG_Build
mkdir openMVG_Install
```

Create a build script, build.sh with the following and place it into openMVG_Build. The script should contain:

```
#!/bin/bash

cmake -DCMAKE_BUILD_TYPE=RELEASE \
    -DOpenMVG_BUILD_EXAMPLES=ON \
    -DOpenMVG_BUILD_TESTS=ON \
    -DCMAKE_INSTALL_PREFIX:STRING="$R3D/openMVG_Install" \
        ../openMVG/src/ &> build.log
```

Make it executable:

```
cd $R3D/openMVG_Build
chmod +x build.sh
```

Configure the project (takes a little while - you can monitor build.log to see how its going)

```
./build.sh
```

Review build.log. There will be warnings about Doxygen, Sphinx and COIN - these are related to document generation.

Now we can compile with

```
make -j3 2>&1 | tee make.log
```

Search make.log for the string "error:". This should come up empty.

Search make.log for the string "warning:". This will return a lot of warnings (60 on my build) and I strongly suspect some are real problems.

Edit src/dependencies/cereal/include/cereal/external/rapidjson/document.h. Find and change the following lines (590, 753):

```
explicit GenericValue(Type type) CEREAL_RAPIDJSON_NOEXCEPT : data_() {
 change to
explicit GenericValue(Type type) /*CEREAL_RAPIDJSON_NOEXCEPT*/ : data_() {
```

```
GenericValue& operator=(GenericValue& rhs) CEREAL_RAPIDJSON_NOEXCEPT {
   change to
GenericValue& operator=(GenericValue& rhs) /*CEREAL_RAPIDJSON_NOEXCEPT*/ {
```

Edit src/dependencies/osi_clp/CoinUtils/src/CoinMessageHandler.cpp at line 823:

```
sprintf(messageOut_,format_+2);
      change to
sprintf(messageOut_,"%s", (char *)(format_+2));
```

Edit src/nonFree/sift/vl/generic.c at line 517 and src/nonFree/sift/vl/host.c at line 523:

```
int length = 0 ;
      change to
size_t length = 0 ;
```

Edit src/dependencies/osi_clp/CoinUtils/src/CoinLpIO.cpp from line 80 to reorder the initialisation to match the declaration of the class:

```
CoinLpIO::CoinLpIO(const CoinLpIO& rhs)
    :
    problemName_(CoinStrdup("")),
    defaultHandler_(true),
    numberRows_(0),
    numberColumns_(0),
    numberElements_(0),
    matrixByColumn_(NULL),
    matrixByRow_(NULL),
    rowlower_(NULL),
    rowupper_(NULL),
    collower_(NULL),
    colupper_(NULL),
    rhs_(NULL),
    rowrange_(NULL),
    rowsense_(NULL),
    objective_(NULL),
    objectiveOffset_(0.0),
    integerType_(NULL),
    fileName_(CoinStrdup("")),
    infinity_(COIN_DBL_MAX),
    epsilon_(1e-5),
    numberAcross_(10),
    objName_(NULL)
{
```

Edit src/dependencies/osi_clp/Clp/src/ClpModel.cpp on lines 1369, 1380, 3496, 3533, 3619, 3647, 3880, 3896, 3920, and 3936:

```
char name[9];
      change to
char name[16];
```

Edit src/dependencies/osi_clp/Clp/src/ClpPdco.cpp at line 319

```
char solver[6];
      change to
char solver[8];
```

Edit src/dependencies/osi_clp/CoinUtils/src/CoinMpsIO.cpp.

At line 3993

```
char newName[9];
      change to
char newName[16];
```

At lines 4935, 4941, and 4992

```
rowNames[i] = reinterpret_cast<char *> (malloc (9 * sizeof(char)));
      change to
rowNames[i] = reinterpret_cast<char *> (malloc (16 * sizeof(char)));
```

At lines 4951, 4957, 4967, and 5002

```
columnNames[i] = reinterpret_cast<char *> (malloc (9 * sizeof(char)));
    change to
columnNames[i] = reinterpret_cast<char *> (malloc (16 * sizeof(char)));
```

Edit src/dependencies/osi_clp/CoinUtils/src/CoinWarmStartBasis.cpp from line 494:

```
      if (status==CoinWarmStartBasis::basic) {
        setStructStatus(i,atLowerBound);
        numberBasic--;
        if (numberBasic==numArtificial_)
          break;
      }
```

Edit src/dependencies/osi_clp/Osi/src/Osi/OsiColCut.hpp.

From line 304:

```
   if ( cutUbs.isExistingIndex(colIndx) ) {
      if ( cutUbs[colIndx] < newUb ) newUb = cutUbs[colIndx];
      if ( newLb > newUb )
        return true;
   }
```

From line 316 (after the previous edit)

```
    if ( cutLbs.isExistingIndex(colIndx) ) {
      if ( cutLbs[colIndx] > newLb ) newLb = cutLbs[colIndx];
      if ( newUb < newLb )
        return true;
    }
```

Now clean up the previous build and re-make:

```
cd $R3D/openMVG_Build
make clean
make -j3 2>&1 | tee make.log
```

There are still a few warnings, but I will leave it to others to rectify, since they don't seem to affect the working of the program.

Finish off with

```
make install
```

2.4.2 Regard3D

Create a directory and unpack the zip file:

```
cd $R3D
mkdir regard
cd regard
p7zip -d -k $PACKAGES/Regard3D_src_1.0.0.7z
```

A few edits are required to the build instructions:

Edit src/CMakeLists.txt

The first change turns off some code in the Eigen3 library that causes Regard3D to crash as soon as you start to look for matches. You may not have the bug in libeigen3 in which case you don't need this change.

Add after line 160:

```
INCLUDE_DIRECTORIES( ${EIGEN3_INCLUDE_DIR} )
ADD_DEFINITIONS(-DEIGEN_HAS_RVALUE_REFERENCES=0)
```

The next change allows the build to find some include directories. Find and change:

```
# The above doesn't work, this is an ugly workaround:
#INCLUDE_DIRECTORIES("${wxWidgets_ROOT_DIR}/include/openMVG_dependencies")
#INCLUDE_DIRECTORIES("${wxWidgets_ROOT_DIR}/include/openMVG_dependencies/cereal/include")
INCLUDE_DIRECTORIES("${R3D}/openMVG_Install/include/openMVG_dependencies")
INCLUDE_DIRECTORIES("${R3D}/openMVG_Install/include/openMVG_dependencies/cereal/include")
```

The next is just a spelling/case error. At line 671

```
INCLUDE_DIRECTORIES(thirdparty/OpenMVG)
      change to
INCLUDE_DIRECTORIES(thirdparty/openMVG)
```

Finally the libraries need a bit of re-ordering. At line 738:

```
SET(REGARD3D_LINK_LIBRARIES ${REGARD3D_LINK_LIBRARIES} ${OPENSCENEGRAPH_LIBRARIES}
        AKAZE fast-akaze liop kgraph sqlite ${OpenCV_LIBRARIES}
        ${Boost_LIBRARIES}
        OpenMVG::openMVG_sfm OpenMVG::openMVG_stlplus OpenMVG::openMVG_fast
         OpenMVG::minilog
        OpenMVG::openMVG_exif OpenMVG::openMVG_features OpenMVG::openMVG_image
        OpenMVG::openMVG_linearProgramming OpenMVG::openMVG_lInftyComputerVision
        OpenMVG::openMVG_geometry OpenMVG::openMVG_matching OpenMVG::openMVG_kvld
        OpenMVG::openMVG_matching_image_collection OpenMVG::openMVG_multiview
```

```
        OpenMVG::openMVG_numeric OpenMVG::openMVG_robust_estimation
        OpenMVG::openMVG_system
        ${CERES_LIBRARIES} ${GLOG_LIBRARIES} ${SuiteSparse_LIBRARIES}
        ${FLANN_LIBRARIES}
        ${ASSIMP_LIBRARIES} ${OPENGL_LIBRARIES}
        X11 dl
        debug; tinyply cpuid
)
```

Create a file, build.sh, with the following contents and place in the $R3D/regard directory.

```
#!/bin/bash

# script to build regard3d

cmake -G "Unix Makefiles" \
      -DCMAKE_BUILD_TYPE="Debug" \
      -DR3D=$R3D \
      -DCERES_DIR=$R3D/openMVG-1.4_r3d/third_party/ceres-solver \
      -DOpenMVG_DIR=$R3D/openMVG_Install/share/openMVG/cmake \
      ../src
```

Now generate the Makefiles:

```
cd $R3D/regard
chmod +x build.sh
mkdir build
cd build
../build.sh
```

In order to get it to build a few edits are required.

Edit src/thirdparty/kgraph/kgraph-data.h at line 109 change:

```
data = (char *)memalign(A, row * stride); // SSE instruction needs data to be aligned
if (!data) throw runtime_error("memalign");
           change to
int rc = posix_memalign(&data, A, row * stride); // SSE instruction needs data to be aligned
if (rc) throw runtime_error("memalign");
```

Edit src/threads/PreviewGeneratorThread.cpp.

This problem arises because opencv2 has not been updated to use the enum class. Apparently some compilers are able to interpret a class or struct that only contains an enum as an enum class but gcc fairly correctly rejects that.

At line 293

```
cv::DrawMatchesFlags flags = cv::DrawMatchesFlags::DEFAULT;
       change to
int flags = cv::DrawMatchesFlags::DEFAULT;
```

At line 403

```
cv::DrawMatchesFlags flags = cv::DrawMatchesFlags::DEFAULT;
if(previewInfo.keypointType_ == PreviewInfo::PIKPTRichKeypoints)
      flags = cv::DrawMatchesFlags::DRAW_RICH_KEYPOINTS;
if(!previewInfo.showSingleKeypoints_)
      flags |= cv::DrawMatchesFlags::NOT_DRAW_SINGLE_POINTS;

      change to

int flags = cv::DrawMatchesFlags::DEFAULT;
if(previewInfo.keypointType_ == PreviewInfo::PIKPTRichKeypoints)
      flags = (int)cv::DrawMatchesFlags::DRAW_RICH_KEYPOINTS;
if(!previewInfo.showSingleKeypoints_)
      flags |= (int)cv::DrawMatchesFlags::NOT_DRAW_SINGLE_POINTS;
```

Edit src/utils/OpenMVGHelper.cpp.

At line 2782

```
double z = Depth(iterPose->second.rotation(),
                 iterPose->second.translation(), X);
      change to
double z = openMVG::Depth(iterPose->second.rotation(),
                 iterPose->second.translation(), X);
```

Edit src/utils/CameraDBLookup.cpp.

I do not understand this error. How is it even possible for a semicolon to be the separator in a CSV file ?

At line 175

```
wxStringTokenizer tokenizer(line, wxT(";"));
      change to
wxStringTokenizer tokenizer(line, wxT(","));
```

Now build it

```
cd $R3D/regard/build
make -j3 2>&1 | tee make.log
```

There are still a few warnings, but nothing that would cause the program to fail.

You can now run Regard3D but it won't be very happy. It needs the camera database and its third party programs. You might also find that it just displays a series of messages "ClientToScreen cannot work when toplevel window is not shown". This appears to be caused by a race condition in the start up code. After a few tries it might actually start. When it does it will often leave the centre panel empty [as in you can see the underlying desktop.] Slightly moving its window will prompt it to refresh with the centre OpenGL panel.

Next we need to build the third party programs. This is the easy part.

2.4.3 GRACLUS

Unpack it:

```
cd $R3D
tar -xzf $PACKAGES/graclus1.2.tar.gz
```

Edit Makefile.in

```
# What options to be used by the compiler
COPTIONS = -DNUMBITS=64
```

Build it:

```
cd $R3D/graclus1.2
make -j3 2>&1 | tee make.log
```

It has a few warnings that should be attended to, but nothing worth our trouble.

2.4.4 CMVS & PMVS2

Unpack it

```
cd $R3D
tar -xzf $PACKAGES/cmvs-fix2.tar.gz
```

Its Makefile needs you to enter a few paths.

```
cd $R3D/cmvs/program/main
```

Edit Makefile:

```
# Your INCLUDE path (e.g., -I/usr/include)
YOUR_INCLUDE_PATH = -I/usr/include

# Your metis directory (contains header files under
graclus1.2/metisLib/)
YOUR_INCLUDE_METIS_PATH = -I${R3D}/graclus1.2/metisLib

# Your LDLIBRARY path (e.g., -L/usr/lib)
YOUR_LDLIB_PATH = -L/usr/lib -L${R3D}/graclus1.2
```

Edit cmvs/program/base/numeric/mylapack.cc.

At line 5:

```
extern "C" {
// #include <clapack/f2c.h>
// #include <clapack/clapack.h>
#include <f2c.h>
#include <freefem++/clapack.h>
};
```

At line 157:

```
void Cmylapack::lls(std::vector<float>& A,
                    std::vector<float>& b,
                    integer width, integer height) {
  char trans = 'N';
      change to
void Cmylapack::lls(std::vector<float>& A,
                    std::vector<float>& b,
                    long int w, long int h) {
  integer width = (int)w, height = (int)h;
  char trans = 'N';
```

At line 171

```
void Cmylapack::lls(std::vector<double>& A,
                    std::vector<double>& b,
                    integer width, integer height) {
  char trans = 'N';
      change to
void Cmylapack::lls(std::vector<double>& A,
                    std::vector<double>& b,
                    long int w, long int h) {
  integer width = (int)w, height = (int)h;
  char trans = 'N';
```

Edit cmvs/program/base/stann/dpoint.hpp.

At line 492:

```
std::cerr << "Error Reading Point:"
             << is << std::endl;
      change to
std::cerr << "Error Reading Point:"
             << std::endl;
```

Edit cmvs/program/base/cmvs/bundle.cc at line 1:

```
#include <algorithm>
#include <numeric>
#include <fstream>
#include <iterator>
```

Build it

```
cd $R3D/cmvs/program/main
make -j3 2>&1 | tee make.log
```

2.4.5 Bundler

Unpack it:

```
cd $R3D
tar -xvzf $PACKAGES/bundler-v0.4-source.tar.gz
```

Edit src/BundlerApp.h at line 620:

Since SkeletalApp is derived from BundlerApp the BundlerApp constructor will be called automatically and does not need to be called explicitely.

```
    SkeletalApp() {
        // BundlerApp::BundlerApp();
        m_start_camera = -1;
```

Build it:

```
cd $R3D/bundler-v0.4-source
make -j3 2>&1 | tee make.log
```

This produces a mere 361 warnings but somehow seems to work.

2.4.6 MVE

Unpack it

```
cd $R3D
unzip $PACKAGES/mve-master
```

Build it:

```
cd $R3D/mve-master
make -j3 2>&1 | tee make.log
```

2.4.7 SMVS

Unpack it

```
cd $R3D
unzip $PACKAGES/smvs-master
```

Create shortcut to mve

```
ln -s mve-master mve
```

Build it

```
cd $R3D/smvs-master
make -j3 2>&1 | tee make.log
```

2.4.8 PoissonRecon

Unpack it:

```
cd $R3D
unzip $PACKAGES/PoissonRecon-master
```

Build it. Note that I have dropped the "-j3" option since this code pushes things to the limit in terms of memory use. You should have at least 12GB for this to compile.

```
cd $R3D/PoissonRecon-master
make 2>&1 | tee make.log
```

2.4.9 MVS-Texturing

Unpack it:

```
cd $R3D
unzip $PACKAGES/mvs-texturing-master
```

Build it:

```
cd mvs-texturing-master
mkdir build
cd build
cmake ..
make -j3 2>&1 | tee make.log
```

Note that this program does downloads, including of things we have already built (mve). It would appear that the entire process for constructing Regard3D needs to be examined and streamlined.

2.5 Installation

All the programs are statically linked so there is no messing about with shared libraries. Hence we can just copy the files to somewhere on the $PATH. My choice was /usr/local/bin.

Start with Regard3D itself:

```
cd $R3D
sudo cp $R3D/regard/build/Regard3D /usr/local/bin
```

2.5.1 Camera Database

Next we need the camera database:

This needs to be in the same directory as Regard3D, which is a rather naughty requirement on a Linux system. It should be in a configuration data directory somewhere.

```
unzip $PACKAGES/CameraSensorSizeDatabase-master
sudo cp CameraSensorSizeDatabase-master/sensor_database.csv /usr/local/bin
```

2.5.2 Support Programs

Now we need the support programs which need to be in subdirectories, which is also a rather non-standard requirement.

```
cd /usr/local/bin
sudo mkdir mve
sudo mkdir pmvs
sudo mkdir poisson
cd $R3D/mve/apps
sudo cp dmrecon/dmrecon fssrecon/fssrecon /usr/local/bin/mve/
sudo cp makescene/makescene meshclean/meshclean /usr/local/bin/mve
sudo cp scene2pset/scene2pset /usr/local/bin/mve
sudo cp $R3D/mvs-texturing-master/build/apps/texrecon/texrecon /usr/local/bin/mve
sudo cp $R3D/smvs-master/app/smvsrecon /usr/local/bin/mve
cd /usr/local/bin/mve
ln -s smvsrecon svmsrecon_SSE41
cd $R3D/cmvs/program/main
sudo cp cmvs genOption pmvs2 /usr/local/bin/pmvs
cd $R3D/PoissonRecon-master/Bin/Linux
sudo cp SurfaceTrimmer PoissonReco /usr/local/bin/poisson
```

2.5.3 Icon

Copy the program's icon to /usr/local/share:

```
sudo cp $R3D/regard/src/res/png/regard3d.png /usr/local/share/
```

2.5.4 Desktop File

If you want to be able to run it from your desktop menus etc then create a text file called Regard3d.desktop with the following contents.

```
[Desktop Entry]
Comment=A program for creating 3D models from photos.
Terminal=false
Name=Regard3D
Exec=/usr/local/bin/Regard3D
Type=Application
Icon=/usr/local/share/regard3d.png
Categories=Graphics;
```

Place this in your personal .local/share/applications or /usr/share/application.

```
sudo cp Regard3D.desktop /usr/share/applications.
```