Drone Scanning Installation Guide

Install Ubuntu on VM or natively (the project was developed on Ubuntu).

Install GCC:

```
sudo apt install build-essential
```

Install GIT:

```
sudo apt install git
```

Python3:

```
sudo apt install python3.8
```

OpenCV:

Step 1: Update packages

```
sudo apt-get update
sudo apt-get upgrade
```

Step 2: Install OS libraries

```
# Remove any previous installations of x264</h3>
sudo apt-get remove x264 libx264-dev

# We will Install dependencies now

sudo apt-get install build-essential checkinstall cmake pkg-config yasm
sudo apt-get install git gfortran
sudo apt-get install libjpeg8-dev
sudo add-apt-repository "deb http://security.ubuntu.com/ubuntu
xenial-security main"
sudo apt update
sudo apt install libjasper1 libjasper-dev
sudo apt-get install libpng-dev
```

```
# If you are using Ubuntu 14.04
sudo apt-get install libtiff4-dev
# If you are using Ubuntu 16.04
sudo apt-get install libtiff5-dev

sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev
libdc1394-22-dev
sudo apt-get install libxine2-dev libv4l-dev
sudo apt install -y libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev
sudo apt-get install qt5-default libgtk2.0-dev libtbb-dev
sudo apt-get install libatlas-base-dev
sudo apt-get install libfaac-dev libmp3lame-dev libtheora-dev
sudo apt-get install libvorbis-dev libxvidcore-dev
sudo apt-get install libopencore-amrnb-dev libopencore-amrwb-dev
sudo apt-get install x264 v4l-utils

# Optional dependencies (also download these)
sudo apt-get install libprotobuf-dev protobuf-compiler
sudo apt-get install libgoogle-glog-dev libgflags-dev
sudo apt-get install libgphoto2-dev libeigen3-dev libhdf5-dev doxygen
```

Step 3: Install Python libraries

```
sudo apt-get install python3-dev python3-pip
sudo -H pip3 install -U pip numpy
(if there was a warning or error, just continue its ok)

sudo pip3 install virtualenv virtualenvwrapper
echo "# Virtual Environment Wrapper" >> ~/.bashrc echo "export
VIRTUALENVWRAPPER_PYTHON='/usr/bin/python3'" >> ~/.bashrc
echo "source /usr/local/bin/virtualenvwrapper.sh" >> ~/.bashrc
source ~/.bashrc

mkvirtualenv facecourse-py3 -p python3
workon facecourse-py3
pip install numpy scipy matplotlib scikit-image scikit-learn ipython
deactivate
```

Step 4: Download OpenCV and OpenCV_contrib

```
git clone https://github.com/opencv/opencv.git
cd opencv
git checkout 3.3.1
cd ..
```

```
git clone https://github.com/opencv/opencv_contrib.git
cd opencv_contrib
git checkout 3.3.1
cd ..
```

Step 5: Compile and install OpenCV with contrib modules

open file "opencv/modules/python/src2/cv2.cpp" and on line 856, change the variable from "char" to "const char".
open file "opencv/modules/videoio/src/cap_ffmpeg_impl.hpp" and add these 3 lines at the top:

```
#define AV_CODEC_FLAG_GLOBAL_HEADER (1 << 22)
#define CODEC_FLAG_GLOBAL_HEADER AV_CODEC_FLAG_GLOBAL_HEADER
#define AVFMT_RAWPICTURE 0x0020
```

Step 5.1: Create a build directory

```
cd opencv
mkdir build
cd build
```

Step 5.2: Run CMake

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
      -D CMAKE_INSTALL_PREFIX=/usr/local \
      -D INSTALL_C_EXAMPLES=ON \
      -D INSTALL_PYTHON_EXAMPLES=ON \
      -D WITH_TBB=ON \
      -D WITH_V4L=ON \
      -D WITH_QT=ON \
      -D WITH_OPENGL=ON \
      -D OPENCV_EXTRA_MODULES_PATH=../../opencv_contrib/modules \
      -D BUILD_EXAMPLES=ON ..
```

Step 5.3: Compile and Install

```
# find out number of CPU cores in your machine
nproc

# substitute 4 by output of nproc
make -j4
sudo make install
sudo sh -c 'echo "/usr/local/lib" >> /etc/ld.so.conf.d/opencv.conf'
sudo ldconfig
```

Pangolin:

```
git clone https://github.com/stevenlovegrove/Pangolin.git

sudo apt install libgl1-mesa-dev
sudo apt install libglew-dev
sudo apt install cmake

cd Pangolin
mkdir build
cd build
cmake ..
cmake --build
make
sudo make install
```

Eigen3:

```
sudo apt install libeigen3-dev
```

BLAS & LAPACK & OpenVSLAM:

```
sudo apt-get install libblas-dev
sudo apt-get install liblapack-dev
sudo apt-get install libboost-all-dev
```

ORB_SLAM2:

```
git clone https://github.com/faresfaresCS/ORB_SLAM2.git
```

Open file "ORB_SLAM2/include/LoopClosing.h", there find these lines:

```
typedef map<KeyFrame*, g2o::Sim3, std::less<KeyFrame*>,
Eigen::aligned_allocator<std::pair<const KeyFrame*, g20::sim3> > >
KeyFrameAndPose;
```

and change it with this:

```
typedef map<KeyFrame*, g2o::Sim3, std::less<KeyFrame*>,
Eigen::aligned_allocator<std::pair<KeyFrame* const, g20::sim3> > >
KeyFrameAndPose;
```

Change ‘mono_tum.cc’ file:
Go to ORB_SLAM2/Examples/Monocular.
Replace ‘mono_tum.cc’ file with our version from the github repo.
Add ‘DRONE_PARAMS.yaml’ file to the same directory.

Now we build ORB_SLAM2:

```
cd ORB_SLAM2
chmod +x build.sh
./build.sh
```

DONE !!