This guide contains instructions on how to configure a computer to build and run the Firefighter Indoor Navigation projects. This guide assumes a basic level of familiarity with install procedures and general computer usage. The projects are designed to be built and run on Ubuntu 16.04, other operating systems and versions may be usable, but the setup for these is outside the scope of this guide.

1: Download and install the appropriate Ubuntu 16.04 package from the official site: http://releases.ubuntu.com/16.04/

- Note: If you are installing on a virtual machine, you may need to take additional steps to allow the virtual OS to access the Intel RealSense d435i camera properly. Consult the documentation for the virtualization software for assistance.

2: Ensure that the Ubuntu install is up to date by running the following command:

- sudo apt -y update && sudo apt -y upgrade

3: Install Qt Creator by downloading the online installer from the official site: https://www.qt.io/download-qt-installer. Once downloaded, run the installer and install the Qt Creator component, and the latest LTS version of the Qt Libraries.

- Note: If you are low on available space, the install size can be reduced by not installing unnecessary library components. The only required component is the desktop gcc library.
- Note: Other IDEs may be usable, but the application project files are for Qt Creator, and the application requires the Qt Libraries regardless of the IDE used.

4: Install ROS 10(Kinetic Kame) by running the following commands:

- sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
- sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
- sudo apt -y update
- sudo apt -y install ros-kinetic-desktop-full
- sudo rosdep init
- rosdep update
- echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
- source ~/.bashrc

5: Install the Intel RealSense SDK libraries by running the following commands:

- sudo apt-key adv --keyserver keys.gnupg.net --recv-key F6E65AC044F831AC80A06380C8B3A55A6F3EFCDE || sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-key F6E65AC044F831AC80A06380C8B3A55A6F3EFCDE
- sudo add-apt-repository "deb

http://realsense-hw-public.s3.amazonaws.com/Debian/apt-repo xenial main" -u
- sudo apt -y update
- sudo apt -y install librealsense2-dkms librealsense2-utils librealsense2-dev librealsense2-dbg

6: Install GLFW by running the following command:
- sudo apt -y install libglfw3-dev

7: Install Pangolin by navigating to the directory in which you would like to install it(I recommend /home/$USER/Libraries/) and running the following commands:
- sudo apt -y install libgl1-mesa-dev libglew-dev libegl1-mesa-dev libwayland-dev libxkbcommon-dev wayland-protocols ffmpeg libavcodec-dev libavutil-dev libavformat-dev libswscale-dev libavdevice-dev libjpeg-dev libpng12-dev libtiff5-dev libopenexr-dev
- git clone https://github.com/stevenlovegrove/Pangolin.git
- cd Pangolin
- mkdir build
- cd build
- cmake ..
- cmake --build .

After installing Pangolin, it will likely need to be added to your $LD_LIBRARY_PATH in order to be used by executables. To do so, assuming you are using the default bash shell, open the file ~/.bashrc and add the following line to it, replacing YOUR_LIBRARY_PATH with the location you ran the install commands from:
- export LD_LIBRARY_PATH+=:/YOUR_LIBRARY_PATH/Pangolin/build/src

In addition, because Pangolin does not have a fixed path relative to the project files, you must update the PANGOLIN_PATH variable in FireFighterBaseUnit.pro to point to the location of the libpangolin.so file(/home/$USER/Libraries/Pangolin/build/src/ in the example install).

8: Install the Point Cloud Library by running the following commands:
- wget https://github.com/PointCloudLibrary/pcl/archive/pcl-1.9.1.tar.gz
- tar xf pcl-1.9.1.tar.gz
- cd pcl-pcl-1.9.1 && mkdir build && cd build
- cmake ..
- make
- sudo make install
- cd ../..
- sudo rm pcl-1.9.1.tar.gz
- sudo rm -r pcl-pcl-1.9.1

9: Install the xdo library by running the following commands:
- sudo apt -y install libxdo-dev

10: This project uses a custom version of the ORB_SLAM2 library. The source code and build files for this library are included in the project folder for the firefighterbaseunit application. In order to run the application, this library must be built. To do so, open a terminal and navigate to the ORB_SLAM2_firefighter directory inside the firefighterbaseunit project directory(the directory containing the file FireFighterBaseUnit.pro) and run the following command:

- ./build.sh

This command will build the two third party libraries(g2o and DBoW2) and then build the custom ORB_SLAM2 library. Note: you may need to edit the custom ORB_SLAM2 src or include files in the course of the project. If you do, you must repeat the build command to recompile the library. You will also need to add the location of the ORB_SLAM2 library to your $LD_LIBRARY_PATH. If you are using the standard bash shell, this can be done by adding the following line to your ~/.bashrc file, replacing PROJECT_PATH with the location of the firefighterbaseunit folder:

- export LD_LIBRARY_PATH+=:/PROJECT_PATH/firefighterbaseunit/ORB_SLAM2_firefighter/lib

11: Before building with Qt, the vocabulary file for the custom ORB_SLAM2 library must be uncompressed into its original text format. This can be done by navigating the the Vocabulary subfolder of the ORB_SLAM2_firefighter folder and running the following command:

- tar xf ORBvoc.txt.tar.gz