69156 Simultaneous Localization and Mapping

Lab 0: Setting up the environment for SLAM labs

1 Introduction

The code has been tested in Ubuntu 20.04 but it should work with no problem in other Linux distributions. Installation on other operating systems may require some more effort. You will need CMake (at least 3.15 version) and a C++ compiler compatible with the C++14 standard.

For developing the assignment, we recommend that you use an IDE that you are familiar with and that provides powerful tools for C++ development, such as CMake compilation, debugging, and profiling tools, etc.

Download link: `https://releases.ubuntu.com/focal/`

- Minimal requirements: 2 CPU / 4GB RAM / 25GB HDD
- Recommended: 4 CPU / 8GB RAM / 50GB HDD

1.1 Mini-SLAM

Mini-SLAM is KeyFrame-based visual SLAM system implemented in C++, which uses ORB features. In essence, it is a simplified version of ORB-SLAM, without place recognition, and without threads.

```
unzip Mini-SLAM.zip
cd Mini-SLAM
```

Listing 1: Download “Mini-SLAM” and unzip the folder

2 Installation

- build-essential (includes g++, gcc, make...)
- Cmake
- Git
- OpenGL
- SuiteSparse
- Eigen3

```
sudo apt install build-essential cmake git mesa-common-dev freeglut3-dev \
  libsuitesparse-dev libeigen3-dev
```

Listing 2: Install build dependencies

2.1 Build OpenCV (version 4.x)

Skip this step if you already have OpenCV 4.0 or later installed on your computer.

```
sudo apt update && sudo apt install -y cmake g++ wget unzip
```

Listing 3: Install minimal prerequisites

```
sudo apt install libjpeg-dev libpng-dev libtiff-dev libavcodec-dev \
  libavformat-dev libswscale-dev libv4l-dev libxvidcore-dev \
  libx264-dev libgtk-3-dev libatlas-base-dev gfortran
```

Listing 4: Install other useful dependencies

```
mkdir Thirdparty/OpenCV && cd Thirdparty/OpenCV
wget -O opencv.zip https://github.com/opencv/opencv/archive/4.9.0.zip
wget -O opencv_contrib.zip \
  https://github.com/opencv/opencv_contrib/archive/4.9.0.zip
unzip opencv.zip && rm opencv.zip
unzip opencv_contrib.zip && rm opencv_contrib.zip
```

Listing 5: Download and unpack sources

```
mkdir -p build && cd build
cmake -DOPENCV_EXTRA_MODULES_PATH=../opencv_contrib-4.9.0/modules \
  ../opencv-4.9.0
cmake --build .
```

Listing 6: Configure and build OpenCV

2.1.1 Install OpenCV system-wide

This step is optional. The provided CMake project is configured to use the local build of OpenCV 4.x inside the *Thirdparty* folder. However, if you do not have any previous version on your computer, you may consider a system-wide installation.

This command will copy the new OpenCV binaries to `/usr/local` so that you can use this library in future projects without having to build it again:

```
sudo make install
```

2.2 Build Pangolin

Reference link: `https://github.com/stevenlovegrove/Pangolin`

```
cd Thirdparty
git clone --recursive https://github.com/stevenlovegrove/Pangolin.git
cd Pangolin
./scripts/install_prerequisites.sh required
sudo apt install python3-dev python3-pip
cmake -B build
cmake --build build
```

3 Running the code

Once you have installed the dependencies, you need to build the Thirdparty libraries using the script provided:

```
./build_thirdparty.sh
```

The last step is to build Mini-SLAM. This last step uses CMake so if your IDE is compatible with it, it may build everything without the scripts. You can always do it manually by running the script:

```
./build_SLAM.sh
```

Binary files should have been generated inside the *Apps* folder:

```
./Apps/mono_tumrgbd
./Apps/mono_euroc
```

Every time you change something in the code, you can recompile the binaries with:

```
./build_SLAM.sh
```

We provide a virtual machine with the code already installed for those who cannot install it.