

第一节课习题

binxxx

June 11, 2018

1 习题说明

2 熟悉Linux

1. 如何在Ubuntu中安装软件（命令行界面）？它们通常被安装在什么地方？
 - 命令行安装: `apt-get install xxx`。
 - 安装的软件通常在`/var/cache/apt/archives`。
2. Linux的环境变量是什么？我如何定义新的环境变量？
 - 环境变量是一个具有特定名字的对象，它包含了一个或者多个应用程序所将使用到的信息。
 - `export PATH = $PATH:xxx` 或者修改`.bashrc`文件
3. Linux根目录下面的目录结构是什么样的？至少说出三个目录的用途。
 - Linux根目录下有如下文件: `/boot`, `/bin`, `/etc`, `/dev`, `/home`, `/lib`, `/initrd`, `/misc`, `/mnt`, `/opt`, `/proc`, `/lost+found`, `/root`, `/sbin`, `/tmp`, `/usr`, `/var`。
 - `/bin`: 普通用户可执行的命令的文件夹。
 - `/etc`: 存放系统配置目录及文件。
 - `/lib`: 存放库文件。
 - `/opt`: 用来安装给所有用户使用的文件或程序。
 - `/root`: 超级用户`root`的缺省目录。
 - `/usr`: 存放应用程序文件。
 - `/var`: 存放系统中经常变化的文件。
4. 假设我要给`a.sh`加上可执行权限，该输入什么命令？
 - `chmod +x a.sh`
5. 假设我要将`a.sh`文件的所有者改成`xiang:xiang`，该输入什么命令？
 - `chown xiang:xiang a.sh`

3 SLAM综述文献阅读

1. SLAM会在哪些场合中用到？至少列举三个方向。
 - 无人驾驶(self driving)
 - 虚拟现实(augmented reality)
 - 仓储机器人
2. SLAM中定位与建图是什么关系？为什么在定位的同时需要建图？
 - 在SLAM中定位与建图是互相依靠的。完美的定位需要用到一个无偏差的地图，但这样的地图又需要精确的位置估测来描绘。SLAM可以认识是一个逐步迭代以减小与实际误差的迭代数学问题。
3. SLAM发展历史如何？我们可以将它划分成哪几个阶段？
 - Classical Age (1986-2004): 此阶段主要是引入了SLAM相关的概率模型，包括EKF，Particle Filter，MLE等方法。
 - Algorithmic-analysis Age (2004-2015): 此阶段主要是研究了SLAM基本的特征问题，包括可观测性，收敛性和连续性，与此同时，很多开源的SLAM的包在此阶段完成。
 - Robust-perception Age (2015-): 在解决了SLAM的基本问题之后，现今的问题在于如何使SLAM系统更加的稳定，主要的要求在于稳定的表现，高层次的理解，可得资源的了解和对任务的推断。
4. 列举三篇在SLAM领域的经典文献。
 - Parallel Tracking and Mapping for Small AR Workspaces
 - LSD-SLAM: Large-Scale Direct Monocular SLAM
 - ORB-SLAM: a Versatile and Accurate Monocular SLAM System

4 CMake练习

5 理解ORB-SLAM2框架

1. 从GitHub下载ORB-SLAM2代码
2.
 - ORB-SLAM2将编译出一个库文件ORB_SLAM2和六个可执行文件rgbd_tum, stereo_kitti, stereo_euroc, mono_tum, mono_kitti, mono_euroc。
 - ORB-SLAM2 include文件夹中包含了所有的头文件，src文件夹中包含了与头文件相对应的主文件，Examples文件夹包含了调用库文件的主程序。
 - ORB-SLAM2中可执行文件链接了OpenCV, Eigen3, Pangolin, DBoW2, g2o。

```

binx@x1-carbon:~/Documents/SLAM$ git clone https://github.com/raulmur/ORB_SLAM2.git ORB_SLAM2
Cloning into 'ORB_SLAM2'...
remote: Counting objects: 566, done.
remote: Total 566 (delta 0), reused 0 (delta 0), pack-reused 566
Receiving objects: 100% (566/566), 41.34 MiB | 3.88 MiB/s, done.
Resolving deltas: 100% (196/196), done.
Checking connectivity... done.
binx@x1-carbon:~/Documents/SLAM$ ls
opencv-2.4.13.5  opencv-2.4.13.5.zip  ORB_SLAM2  Pangolin
binx@x1-carbon:~/Documents/SLAM$ cd ORB_SLAM2/
binx@x1-carbon:~/Documents/SLAM/ORB_SLAM2$ ls
build_ros.sh  CMakeLists.txt  Dependencies.md  include  LICENSE.txt  src  Vocabulary
build.sh      cmake_modules  Examples         License-gpl.txt  README.md  Thirdparty

```

Figure 1: Terminal Snapshot