



智能机器人系统实验报告

|  |  |
| --- | --- |
| 专 业 班 级 | 计算机创新实验18-1班 |
| 学生姓名及学号 | 2018213106 刘嘉伟 |
| 任 课 教 师 | 方宝富老师 |
| 实验指导教师 | 方宝富老师 |
| 实验地点 | D502 |
| 2020 ~2021 学年第二学期 | |

## 实验 1 熟悉 ROS 操作系统、学习 ROS 常用指令

### 实验目的

掌握 ROS 的安装以及常用命令，具体内容如下：

（1）学会在 ubuntu18.04 系统下安装 ROS。

（2）了解 ROS 基础命令。

（3）运行海龟示例。

### 实验设备

硬件环境：DELL笔记本

系统环境：VmWare、Ubuntu18.04、ROS Melodic

### 实验内容

1. 安装ROS

**（a）设置安装源**

|  |
| --- |
| $ sudo sh -c echo deb http://packages.ros.org/ros/ubuntu $(lsb\_release -sc) main /etc/apt/sources.list.d/ros-latest.list |

**（b）设置Key**

|  |
| --- |
| $ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654 |

**（c）安装**

|  |
| --- |
| $ sudo apt update  $ sudo apt install ros-melodic-desktop-full |

**（d）配置环境变量**

|  |
| --- |
| $ echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc  $ source ~/.bashrc |

**（e）安装依赖**

|  |
| --- |
| $ sudo apt install python-rosinstall python-rosinstall-generator python-wstool build-essential |

1. ROS 常用指令以及小海龟示例

* roscore命令

文本

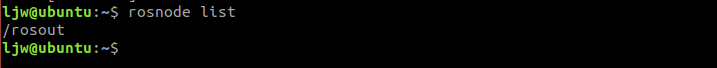
描述已自动生成

* rosrun 命令

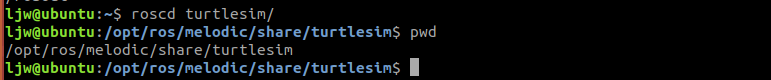
图形用户界面, 应用程序

描述已自动生成

* rosnode 命令



* roscd 命令

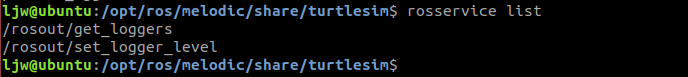


* rostopic 命令

文本

描述已自动生成

* rosservice 命令



* roslaunch 命令

文本

描述已自动生成

### 动手实现

动手实现1：通过手动给/turtle1/cmd\_vel 话题发送消息，控制小海龟移动。

**方法：通过rostopic的pub命令来向/turtle1/cmd\_vel发送运动消息，就能使海龟运动。**

文本

描述已自动生成

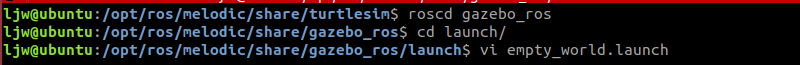
电脑萤幕的截图

描述已自动生成

动手实现2：当使用 roslaunch 启动节点后，再运行 roscore会报错，因为只能同时开启一个roscore，而roslaunch会自动打开roscore，所以如果再运行一个roscore将会报错。

动手实现3：

通过roscd进入文件夹下修改empty\_world.launch



通过rosed直接修改empty\_world.launch



## 实验 2 熟悉ROS话题通信架构，实现话题通信过程

### 一、实验目的

（1）了解 ROS 节点间的话题通信过程

（2）编写节点实现话题通信

### 二、实验设备

硬件环境：DELL笔记本

系统环境：VmWare、Ubuntu18.04、ROS Melodic

### 三、实验内容

（1）了解 ROS 节点间的通信过程：

* 使用 rqt\_graph 查看 ROS 节点关系

图形用户界面, 文本, 应用程序

描述已自动生成

* 使用 rostopic 查看话题信息

文本

描述已自动生成

（2）编写节点实现话题通信

* C++版本

**talker.cpp**

文本

描述已自动生成

文本

描述已自动生成

**listener.cpp**

文本

描述已自动生成

* Python版本

**talker.py**

文本

描述已自动生成

**listener.py**

文本

描述已自动生成

### 四、动手实践

动手实现1：利用实验一所学知识，查看 turtle1/cmd\_vel 话题的消息类型 Twist 的内容是 什么。

文本

描述已自动生成

动手实现2：利用所学知识，编写 launch 文件，通过 roslaunch 同时启动 talker 与 listener 节点。

**launch文件：**

模糊的截图图

描述已自动生成

**实验结果：**

文本

描述已自动生成

动手实现 3：当话题通信时，使用实验一学习的命令，查看话题的信息，比如 rosnode、rostopic、rosservice。

**rosnode：**

文本

描述已自动生成

**rostopic：**

文本

描述已自动生成

**rosservice：**

文本

描述已自动生成

## 实验三：熟悉ROS服务通信架构，实现服务通信过程

### 实验目的

（1）了解 ROS 节点间的服务通信过程

（2）编写节点实现服务通信

### 2、实验设备

硬件环境：DELL笔记本

系统环境：VmWare、Ubuntu18.04、ROS Melodic

### 3、实验内容

（1）了解服务通信过程

文本

描述已自动生成

（2）编写节点实现服务通信

* **srv消息文件**

图形用户界面, 文本, 应用程序

描述已自动生成

* **C++版本**

**Service.cpp**

文本

描述已自动生成

文本

描述已自动生成

**Client.cpp**

文本

描述已自动生成

* **Python版本**

**Service.py**

文本

描述已自动生成

**Client.py**

文本

描述已自动生成

### 4、动手实现

1、手动调用/turtle1/set\_pen 服务设置画笔为红色，宽度为 10，然后 控制海龟移动查看效果。



图表, 雷达图

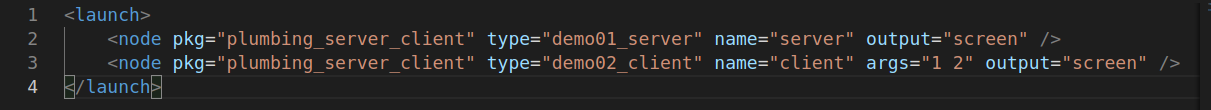
描述已自动生成

1. 尝试改写服务通信的服务功能，比如拼接客户端传来的参数并返回。

**在实验内容中已经写进代码里了，我在srv文件中加了一个string类型的变量，用来拼接运算的字符串，并将拼接结果返回给client，由client输出结果。**

1. 思考是否能够将 server 与 listener 使用 launch 启动？若可以请实现。

**可以。**



**结果：**

文本

描述已自动生成

## 实验 4 了解 ROS 中常用组件：launch 启动文件、Qt 工具箱和 rviz 三维可视化环境

### 实验目的

了解 ROS 中常用组件，具体内容如下：

（1）学会使用 launch 启动文件同时启动多个节点。

（2）学会使用 Qt 工具箱中的常用工具。

（3）学会使用 rviz 相关插件。

### 2、实验设备

硬件环境：DELL笔记本

系统环境：VmWare、Ubuntu18.04、ROS Melodic

### 3、实验内容

（1）launch启动文件

屏幕上有字

描述已自动生成

电脑萤幕的截图

描述已自动生成

（2）Qt 工具箱

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

（3）节点可视化工具：rqt\_graph

文本

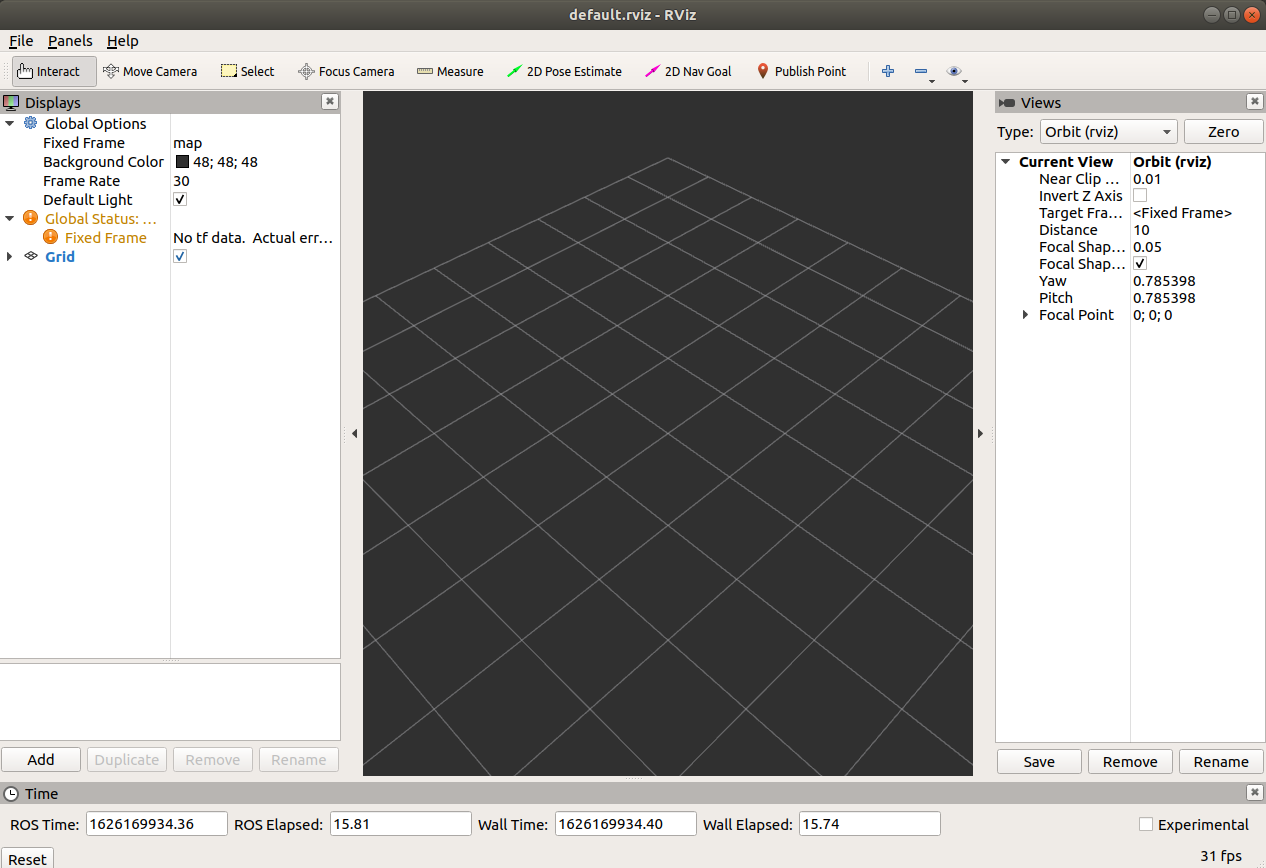
低可信度描述已自动生成

（4）数据绘图工具：rqt\_plot

图片包含 表格

描述已自动生成

（5）rviz 三维可视化环境



### 4、动手实现

文本

描述已自动生成

电脑萤幕的截图

描述已自动生成

## 实验5 ROS简单应用：激光SLAM

### 1、实验目的

（1）了解机器人和 SLAM 之间关系

（2）了解激光 SLAM

（3）了解 gmapping

### 2、实验设备

硬件环境：DELL笔记本

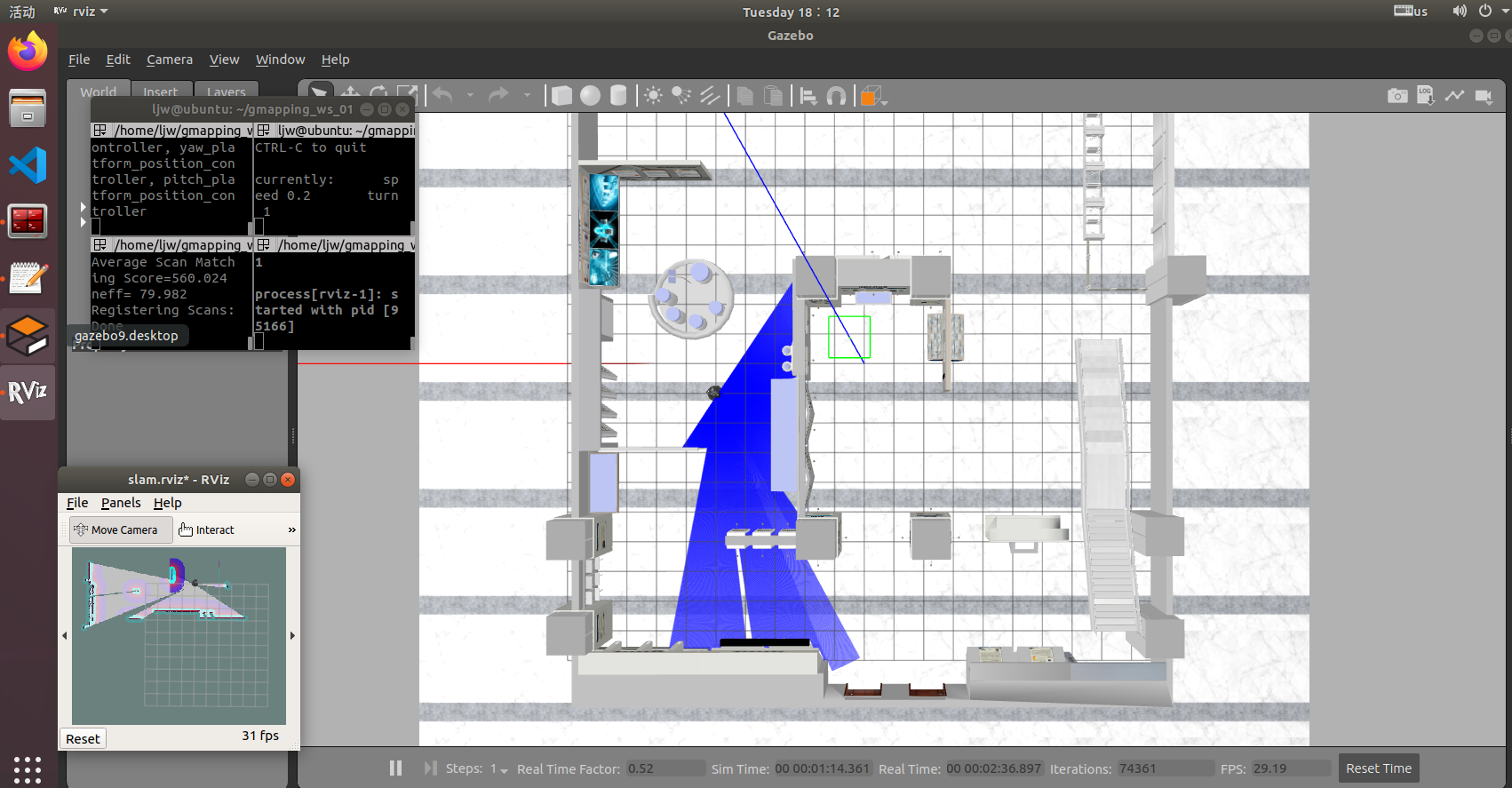
系统环境：VmWare、Ubuntu18.04、ROS Melodic

### 3、实验内容

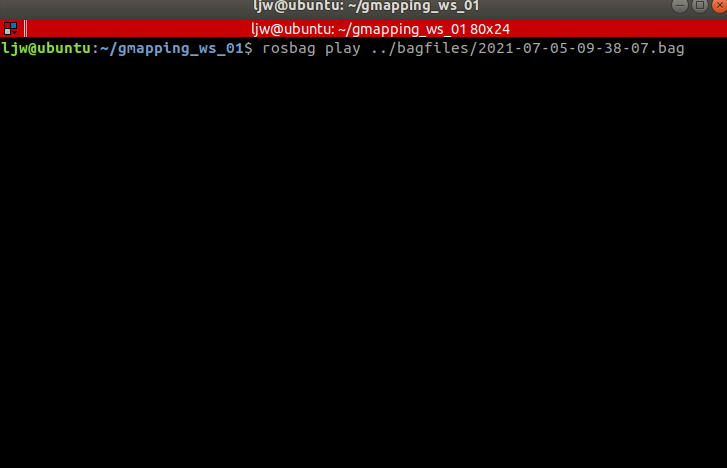
电脑萤幕的截图

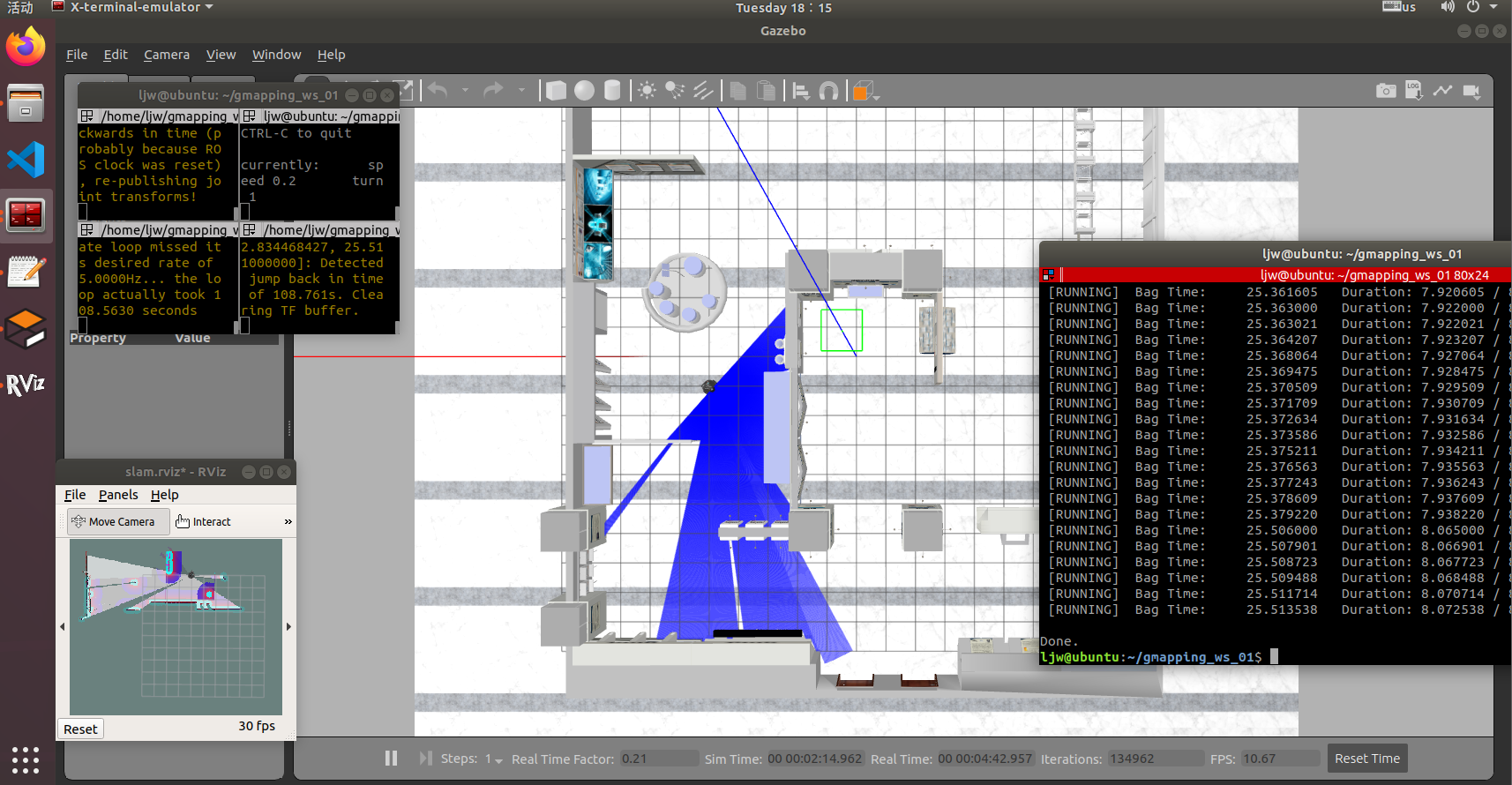
描述已自动生成

### 动手实现



使用bag文件重现运动轨迹





## 实验6 ROS简单应用：视觉SLAM

### 1、实验目的

（1）了解视觉 SLAM

（2）了解 ORB-SLAM

### 2、实验设备

硬件环境：DELL笔记本

系统环境：VmWare、Ubuntu18.04、ROS Melodic

### 3、动手实现

尝试了解 ORB-SLAM2 相关工作原理，并尝试编译并运行 ORB-SLAM2 源码。

* ORB-SLAM2工作原理

ORB-SLAM2算法使用ORB算子进行特征点的提取与描述，其算法原理来自于文章《ORB an efficient alternative to SIFT or SURF》。

ORB算法相比于SIFT具有相似的匹配能力，运算量更小，受噪声的影响更小，能够用于实时运算。作者的主要目的是将该算法更为广泛地应用到各种图像处理场景中，譬如用没有GPU的低功耗设备实现全景拼接和图像跟踪，或者减少在PC上执行基于特征的物体检测运算时间。在这些应用中该算法表现和SIFT一样好（优于SURF），同时运行速度比SIFT快两个数量级。ORB算法基于FAST角点检测算法和BRIEF描述子，因此称为ORB（Oriented FAST and Rotated BRIEF）。这两种方法都具有高性能低运算量的特点。

**ORB算法流程：**

（1）特征点的检测

ORB采用FAST（features from accelerated segment test）算法来检测特征点。FAST核心思想就是找出那些卓尔不群的点，即拿一个点跟它周围的点比较，如果它和其中大部分的点都不一样就可以认为它是一个特征点。

FAST具体计算过程：

1. 从图片中选取一个像素点P，下面我们将判断它是否是一个特征点。我们首先把它的密度（即灰度值）设为Ip。

2. 设定一个合适的阙值t ：当2个点的灰度值之差的绝对值大于t时，我们认为这2个点不相同。

3. 考虑该像素点周围的16个像素。

4. 现在如果这16个点中有连续的n个点都和点不同，那么它就是一个角点。 这里n设定为12。

5. 我们现在提出一个高效的测试，来快速排除一大部分非特征点的点。该测试仅仅检查在位置1、9、5和13四个位置的像素（首先检查1和9，看它们是否和点相同。如果是，再检查5和13）。如果是一个角点，那么上述四个像素点中至少有3个应该和点相同。如果都不满足，那么不可能是一个角点。

（2）特征点的描述

得到特征点后我们需要以某种方式F描述这些特征点的属性。这些属性的输出我们称之为该特征点的描述子（Feature DescritorS）.ORB采用BRIEF算法来计算一个特征点的描述子。BRIEF算法的核心思想是在关键点P的周围以一定模式选取N个点对，把这N个点对的比较结果组合起来作为描述子。

（3）特征点的匹配

ORB算法最大的特点就是计算速度快 。 这首先得益于使用FAST检测特征点，FAST的检测速度正如它的名字一样是出了名的快。再次是使用BRIEF算法计算描述子，该描述子特有的2进制串的表现形式不仅节约了存储空间，而且大大缩短了匹配的时间。

* 编译运行ORB-SLAM2源码

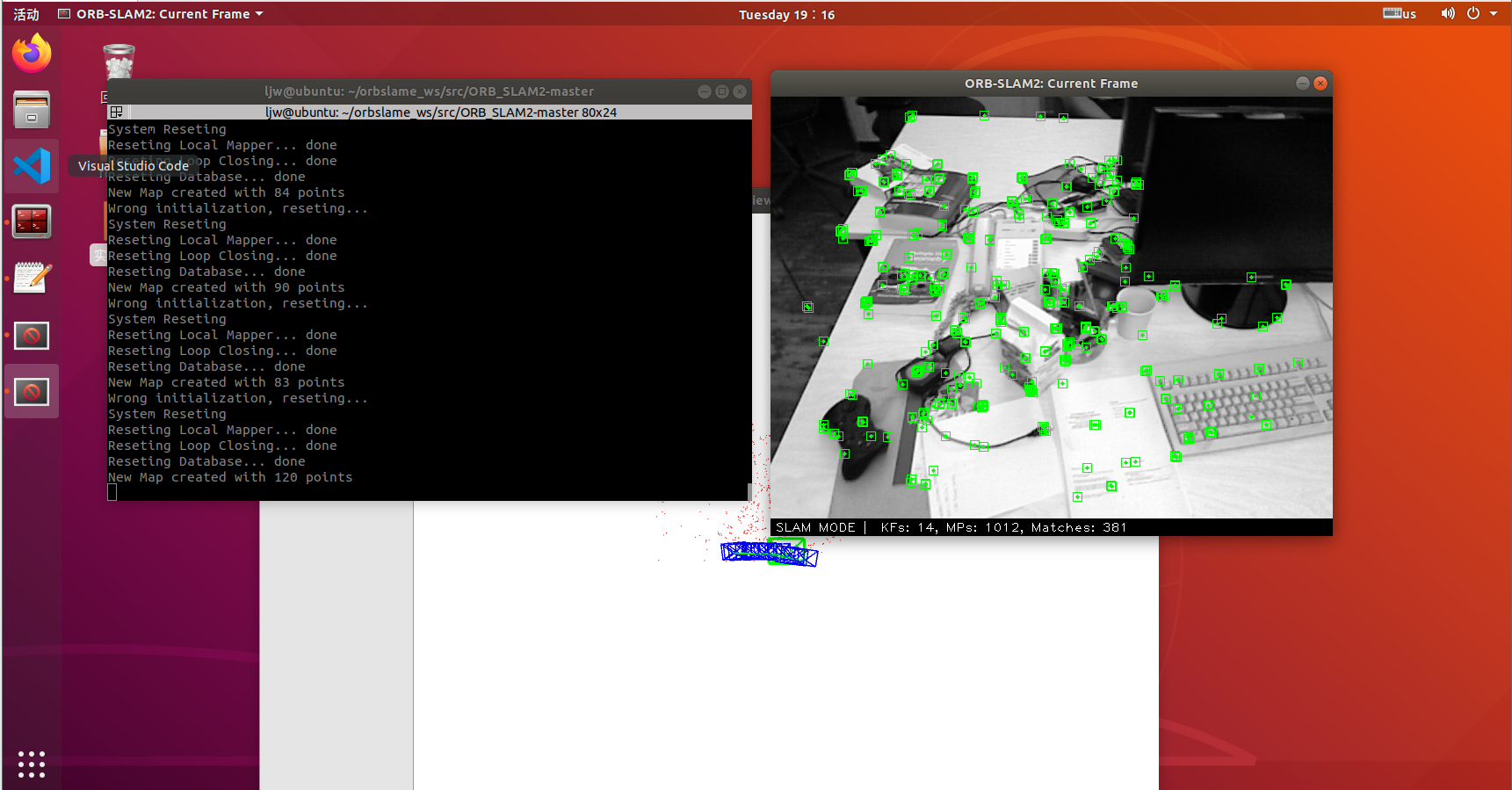
（a）安装Pangolin

|  |
| --- |
| $ sudo apt-get install libglew-dev  $ git clone https://github.com/stevenlovegrove/Pangolin.git  $ cd Pangolin  $ mkdir build  $ cd build  $ cmake ..  $ make -j |

（b）安装ORB\_SLAM2

|  |
| --- |
| $ cd ~/catkin\_ws/src/  $ git clone https://github.com/raulmur/ORB\_SLAM2.git ORB\_SLAM2  $ cd ORB\_SLAM2  $ chmod +x build.sh  $ ./build.sh  $ chmod +x build\_ros.sh  $ export ROS\_PACKAGE\_PATH=${ROS\_PACKAGE\_PATH}:~/catkin\_ws/src/ORB\_SLAM2/Examples/ROS  $ ./build\_ros.sh |

实验结果：



图形用户界面

描述已自动生成

由于虚拟机无法连接笔记本的内置摄像头，所以无法实时检测场景。

遇到的问题：

（1）ORB\_SLAM2/src/System.cc: error: ‘usleep’ was not declared in this scope usleep(5000);

解决办法：在所有缺失usleep的文件中添加一个头文件#include<unistd.h>

（2）CMakeFiles/RGBD.dir/build.make:197: recipe for target ‘…/RGBD’ failed

解决办法：修改home/catkin\_ws/src/ORB\_SLAM2/Examples/ROS/ORB\_SLAM2/文件夹下的CMakeLists.txt文件。在set（LIBS xxxxx 的后面加上-lboost\_system

（3）OPENCV版本不匹配

解决办法：修改CmakeLists.txt，将OPENCV版本修改为3.2.0（系统自带3.2.0版本）

图形用户界面, 文本

中度可信度描述已自动生成