从零开始跑ORB_SLAM2(三) 使用Realsense D435相机跑ORB_SLAM2生成稀疏点阵

前期准备

从零开始跑ORB_SLAM2(零) 安装Ubuntu16.04 LTS
从零开始跑ORB_SLAM2(一) 前期准备与环境配置
从零开始跑ORB_SLAM2(二) 配置ROS Kinetic环境并测试Realsense D435相机

工作环境

PC：i5-8265U 8GRAM 核显 ~~弟弟CPU~~
相机：Intel-Realsense D435
环境：Ubuntu16.04 LTS+ ROS Kinetic

创造ROS工作空间

国际惯例，GitHub的官方文档发一波:ROS Wrapper for Intel® RealSense™ Devices
如果懒得看英文也可以跟我一步步走。
在开始这一步之前，需要完成的事情分别是:

```
安装ROS Kinetic
安装Realsense SDK
```

毫无疑问我们已经完成了这两步，为了安装基于ROS使用RealSense的包，现在开始我们要创造一个ROS的工作空间并且配置它。

Ctrl + Alt + T 新建一个命令行:
新建一个文件夹命名为catkin_ws，在目录下新建一个文件夹src并进入:

```
mkdir -p ~/catkin_ws/src
cd ~/catkin_ws/src/
```

将catkin_ws/src设置为工作空间:

```
catkin_init_workspace
```

这个工作空间设置完了之后最神奇的地方在于，虽然里面什么也没有，但是仍然可以进行编译:

```
catkin_make clean
catkin_make -DCATKIN_ENABLE_TESTING=False -DCMAKE_BUILD_TYPE=Release
catkin_make install
```

编译完毕之后你可以在根目录下看见build等文件。

将**最新的ntel® RealSense™ ROS包**git到src下: 注意这一步，直接git不一定能得到最新的(虽然我也不知道为什么)，我比较建议你们进入Github官网直接把包download到本地。

```
cd src
git clone https://github.com/IntelRealSense/realsense-ros.git
cd realsense-ros/
git checkout `git tag | sort -V | grep -P "^\d+\.\d+\.\d+" | tail -1`
cd ..
```

现在需要对工作空间再进行一次编译，但是一部分人会出现编译失败(也包括我...)，请看一条报错:

```
missing: ddynamic_reconfigure_DIR
```

缺少了ddynamic_reconfigure这个依赖包，进入Github下的Readme中查阅到:

> Make sure all dependent packages are installed. You can check .travis.yml file for reference.
> Specifically, make sure that the ros package ddynamic_reconfigure is installed. If ddynamic_reconfigure cannot be installed using APT, you may clone it into your workspace ‘catkin_ws/src/’ from here (Version 0.2.0)

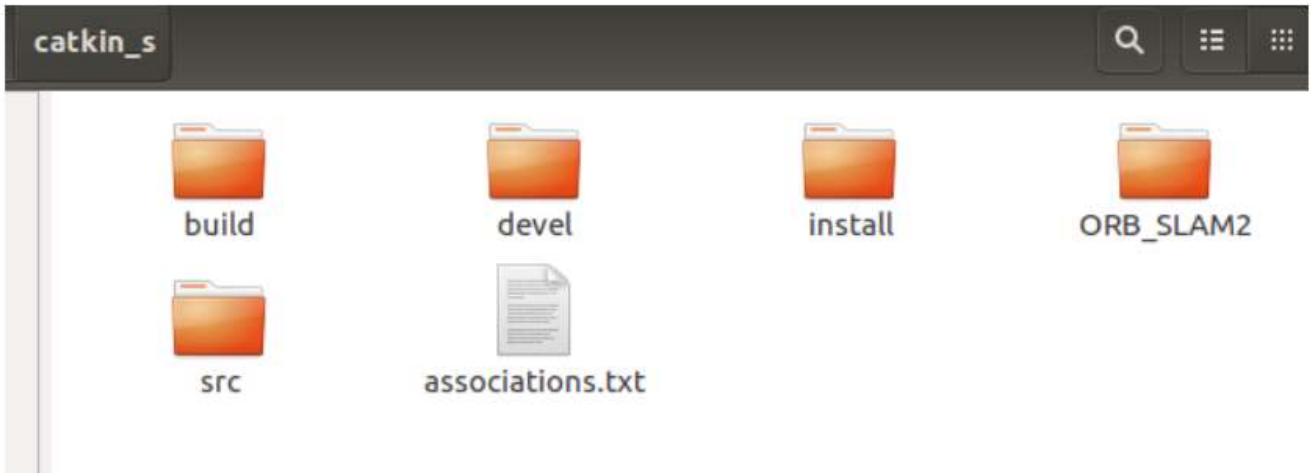
所以需要点击此处下载依赖包，解压后将其提取到src下，再重新编译一次:

```
catkin_make clean
catkin_make -DCATKIN_ENABLE_TESTING=False -DCMAKE_BUILD_TYPE=Release
catkin_make install
```

如无意外这次可以成功。在src下不需要放其他的东西，ORB_SLAM2的包放在catkin_ws下就行了，如下图。(我的catkin_ws写成了catkin_s，不要介意细节)


catkin_s
src
ddynamic_
reconfigure-kinetic-
devel
realsense-ros
CMakeLists.txt
https://blog.csdn.net/weixin_41732319

主目录下文结构:


catkin_s
build
devel
install
ORB_SLAM2
src
associations.txt

至此，ROS工作空间的创造就完成了，如果对工作空间感兴趣可移步这里了解更多信息。
下面配置相机节点并测试。

相关配置

插上相机，Ctrl + Alt + T 新建一个命令行：
在任意目录下输入：

```
roslaunch realsense2_camera rs_rgbd.launch
```

请看一条可能的报错：

```
[rs_rgbd.launch] is neither a launch file in package [realsense2_camera] nor is [realsense2_camera] a launch file na
The traceback for the exception was written to the log file
```

这是由于你没有把devel下的bash文件添加到bashrc的缘故
为了每次启动命令行都能自动添加，则直接修改bashrc文件：

```
echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
echo "source ~/catkin_ws/devel/setup.sh" >> ~/.bashrc
source ~/.bashrc
```

至此，你的bashrc文件中应该添加了以下路径：

```
source /opt/ros/kinetic/setup.bash
```

```
source ~/catkin_ws/devel/setup.sh
source ~/catkin_ws/devel/setup.bash
export ROS_PACKAGE_PATH=~/catkin_s:/home/(用户名)/catkin_ws/src:/opt/ros/kinetic/share
```

此时再启动相机节点，

```
roslaunch realsense2_camera rs_rgbd.launch
```

成功则有以下提示:

```
... logging to /home/ushio/.ros/log/207a371a-eb1a-11e9-9dd5-207918827b84/roslaunch-ushio-Lenovo-XiaoXin-Air-13IWL-10250.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://ushio-Lenovo-XiaoXin-Air-13IWL:44349/

SUMMARY
========

PARAMETERS
 * /camera/realsense2_camera/accel_fps: 250
 * /camera/realsense2_camera/accel_frame_id: camera_accel_frame
 * /camera/realsense2_camera/accel_optical_frame_id: camera_accel_opti...
 * /camera/realsense2_camera/align_depth: True
 * /camera/realsense2_camera/aligned_depth_to_color_frame_id: camera_aligned_de...
 * /camera/realsense2_camera/aligned_depth_to_fisheye1_frame_id: camera_aligned_de...
 * /camera/realsense2_camera/aligned_depth_to_fisheye2_frame_id: camera_aligned_de...
 * /camera/realsense2_camera/aligned_depth_to_fisheye_frame_id: camera_aligned_de...
 * /camera/realsense2_camera/aligned_depth_to_infra1_frame_id: camera_aligned_de...
 * /camera/realsense2_camera/aligned_depth_to_infra2_frame_id: camera_aligned_de...
 * /camera/realsense2_camera/allow_no_texture_points: False
```

```
 * /camera/realsense2_camera/infra2_optical_frame_id: camera_infra2_opt...
 * /camera/realsense2_camera/infra_fps: 30
 * /camera/realsense2_camera/infra_height: 480
 * /camera/realsense2_camera/infra_width: 640
 * /camera/realsense2_camera/initial_reset: False
 * /camera/realsense2_camera/json_file_path:
 * /camera/realsense2_camera/linear_accel_cov: 0.01
 * /camera/realsense2_camera/odom_frame_id: camera_odom_frame
 * /camera/realsense2_camera/pointcloud_texture_index: 0
 * /camera/realsense2_camera/pointcloud_texture_stream: RS2_STREAM_COLOR
 * /camera/realsense2_camera/pose_frame_id: camera_pose_frame
 * /camera/realsense2_camera/pose_optical_frame_id: camera_pose_optic...
 * /camera/realsense2_camera/publish_odom_tf: True
 * /camera/realsense2_camera/rosbag_filename:
 * /camera/realsense2_camera/serial_no:
 * /camera/realsense2_camera/topic_odom_in: camera/odom_in
 * /camera/realsense2_camera/unite_imu_method: none
 * /rosdistro: kinetic
 * /rosversion: 1.12.14

NODES
  /camera/
    color_rectify_color (nodelet/nodelet)
    points_xyzrgb_hw_registered (nodelet/nodelet)
    realsense2_camera (nodelet/nodelet)
    realsense2_camera_manager (nodelet/nodelet)

auto-starting new master
process[master]: started with pid [10260]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 207a371a-eb1a-11e9-9dd5-207918827b84
process[rosout-1]: started with pid [10273]
started core service [/rosout]
process[camera/realsense2_camera_manager-2]: started with pid [10290]
process[camera/realsense2_camera-3]: started with pid [10291]
process[camera/color_rectify_color-4]: started with pid [10292]
process[camera/points_xyzrgb_hw_registered-5]: started with pid [10298]
[ INFO] [1570683306.447900662]: Initializing nodelet with 8 worker threads.
[ INFO] [1570683306.500915771]: RealSense ROS v2.2.8
[ INFO] [1570683306.500990563]: Running with LibRealSense v2.25.0
```

```
 * /camera/realsense2_camera/infra2_optical_frame_id: camera_infra2_opt...
 * /camera/realsense2_camera/infra_fps: 30
 * /camera/realsense2_camera/infra_height: 480
 * /camera/realsense2_camera/infra_width: 640
 * /camera/realsense2_camera/initial_reset: False
 * /camera/realsense2_camera/json_file_path:
 * /camera/realsense2_camera/linear_accel_cov: 0.01
 * /camera/realsense2_camera/odom_frame_id: camera_odom_frame
 * /camera/realsense2_camera/pointcloud_texture_index: 0
 * /camera/realsense2_camera/pointcloud_texture_stream: RS2_STREAM_COLOR
 * /camera/realsense2_camera/pose_frame_id: camera_pose_frame
 * /camera/realsense2_camera/pose_optical_frame_id: camera_pose_optic...
 * /camera/realsense2_camera/publish_odom_tf: True
 * /camera/realsense2_camera/rosbag_filename:
 * /camera/realsense2_camera/serial_no:
 * /camera/realsense2_camera/topic_odom_in: camera/odom_in
 * /camera/realsense2_camera/unite_imu_method: none
 * /rosdistro: kinetic
 * /rosversion: 1.12.14

NODES
  /camera/
    color_rectify_color (nodelet/nodelet)
    points_xyzrgb_hw_registered (nodelet/nodelet)
    realsense2_camera (nodelet/nodelet)
    realsense2_camera_manager (nodelet/nodelet)

auto-starting new master
process[master]: started with pid [10260]
ROS_MASTER_URI=http://localhost:11311

setting /run_id to 207a371a-eb1a-11e9-9dd5-207918827b84
process[rosout-1]: started with pid [10273]
started core service [/rosout]
process[camera/realsense2_camera_manager-2]: started with pid [10290]
process[camera/realsense2_camera-3]: started with pid [10291]
process[camera/color_rectify_color-4]: started with pid [10292]
process[camera/points_xyzrgb_hw_registered-5]: started with pid [10298]
[ INFO] [1570683306.447900662]: Initializing nodelet with 8 worker threads.
[ INFO] [1570683306.500915771]: RealSense ROS v2.2.8
[ INFO] [1570683306.500990563]: Running with LibRealSense v2.25.0
```

如果你忘记插上相机，则会有warning提示:

```
[ WARN] [1570683306.945951510]: No RealSense devices were found!
```

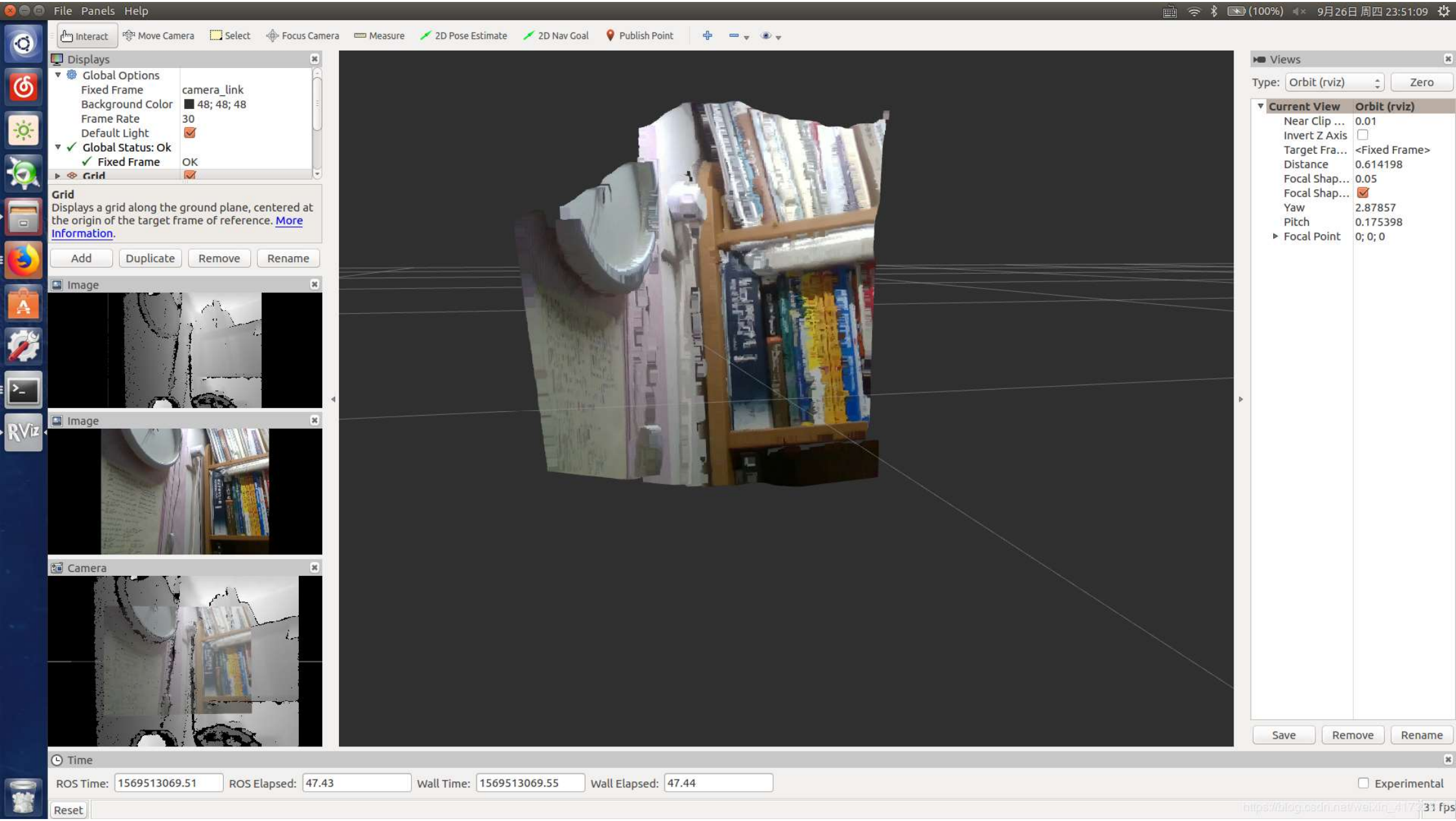
此时按Ctrl+C强制退出，插上相机再执行一次就行。

Ctrl + Alt + T 新建一个命令行:

```
sudo apt-get install rviz
rviz
```

> 此时并不能看见什么结果
> 左上角 Displays 中 Fixed Frame 选项中，下拉菜单选择 camera_link
> 这是主要到Global Status变成了绿色
> 点击该框中的Add -> 上方点击 By topic -> /depth_registered 下的 /points 下的/PointCloud2
> 点击该框中的Add -> 上方点击 By topic -> /color 下的 /image_raw 下的image

你也可以添加更多的信息(深度图)等，成功则有以下图片，拖动鼠标左右键可以查看生成的点云。附上笔者乱糟糟书柜一张。


File Panels Help
(100%) 9月26日 周四 23:51:09
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Fixed Frame camera_link
Background Color 48; 48; 48
Frame Rate 30
Default Light
Global Status: Ok
Fixed Frame OK
Grid
Grid
Displays a grid along the ground plane, centered at the origin of the target frame of reference. More Information.
Add
Duplicate
Remove
Rename
Image
Image
Camera
Views
Type: Orbit (rviz)
Zero
Current View Orbit (rviz)
Near Clip ... 0.01
Invert Z Axis
Target Fra... <Fixed Frame>
Distance 0.614198
Focal Shap... 0.05
Focal Shap...
Yaw 2.87857
Pitch 0.175398
Focal Point 0; 0; 0
Save
Remove
Rename
Time
ROS Time: 1569513069.51
ROS Elapsed: 47.43
Wall Time: 1569513069.55
Wall Elapsed: 47.44
Experimental
Reset
31 fps

相机节点测试完毕。

运行ORB_SLAM2

终于到~~令人兴奋的~~ 实战环节了!其实到这一步就已经没什么坑了，按照官网的步骤基本都可以成功，但有些细节需要注意，我大概讲一讲。

首先要在工作空间内放置一个Pangolin优化库，这就是我在之前的博客中提到的Pangolin可能需要重新安装的原因，你可以把之前的先卸载了。

在任意目录下安装一些依赖:

```
sudo apt-get install libglew-dev
sudo apt-get install libboost-dev libboost-thread-dev libboost-filesystem-dev
sudo apt-get install libpython2.7-dev
sudo apt-get install build-essential
```

cd到catkin_ws/ORBSLAM2下，克隆Pangolin代码到本地仓库并编译:

```
git clone https://github.com/stevenlovegrove/Pangolin.git
cd Pangolin
mkdir build
cd build
cmake -DCPP11_NO_BOOST=1 ..
make -j
```

克隆ORB_SLAM2代码到本地仓库:

```
cd catkin_ws
git clone https://github.com/raulmur/ORB_SLAM2
```

修改订阅topic话题:进入`catkin_ws/ORB_SLAM2/Examples/ROS/ORB_SLAM2/src`路径下，找到`ros_rgbd.cc`，找到以下语句:

```
message_filters::Subscriber<sensor_msgs::Image> rgb_sub(nh, "/camera/rgb/image_raw", 1);
message_filters::Subscriber<sensor_msgs::Image> depth_sub(nh, "camera/depth_registered/image_raw", 1);
```

将其修改为:

```
message_filters::Subscriber<sensor_msgs::Image> rgb_sub(nh, "/camera/color/image_raw", 1);
message_filters::Subscriber<sensor_msgs::Image> depth_sub(nh, "/camera/aligned_depth_to_color/image_raw", 1);
```

如下图:

```
//message_filters::Subscriber<sensor_msgs::Image> rgb_sub(nh, "/camera/rgb/image_raw", 1);
//message_filters::Subscriber<sensor_msgs::Image> depth_sub(nh, "camera/depth_registered/image_raw", 1);

message_filters::Subscriber<sensor_msgs::Image> rgb_sub(nh, "/camera/color/image_raw", 1);
message_filters::Subscriber<sensor_msgs::Image> depth_sub(nh, "/camera/aligned_depth_to_color/image_raw", 1);
```

老规矩，**别忘了在bashrc下添加ORB_SLAM2的工作路径:**

```
用gedit或vim打开后在里面自行添加:
export ROS_PACKAGE_PATH=${ROS_PACKAGE_PATH}:/home/(用户名)/catkin_ws/ORB_SLAM2/Examples/ROS
```

现在你的bashrc应该总共有了以下新的路径信息:

```
source /opt/ros/kinetic/setup.bash
source ~/catkin_ws/devel/setup.sh
source ~/catkin_ws/devel/setup.bash
export ROS_PACKAGE_PATH=${ROS_PACKAGE_PATH}:/home/(用户名)/catkin_ws/ORB_SLAM2/Examples/ROS
export ROS_PACKAGE_PATH=~/catkin_s:/home/(用户名)/catkin_ws/src:/opt/ros/kinetic/share
```

开始编译，回到catkin_ws/ORB_SLAM2下:

```
ls
```

你会发现有两个可运行脚本，一个是build.sh，一个是build_ros.sh，但是都处于不能运行状态。
赋予编译权限:

```
chmod +x build.sh
chmod +x build_ros.sh
```

现在如果运行 `ls` 命令，则发现两个文件已经高亮标注，可以运行了。
运行第一个脚本:

```
./build.sh
```

这同样是一个漫长的过程，如果你的电脑因为运行内存不足而溢出报错(原因请看我之前的博客)，请在gedit或者vim中修改build.sh的最后一行代码将 `make -j` 改为 `make -j2` 或 `make`；你也可以释放更多的虚拟内存来防止溢出。后面的build_ros.sh同理。

编译完后继续编译ros版本脚本:

```
./build_ros.sh
```

下面内容来自其他博主，我没有遇到这个错误，供参考:

> 在/Examples/ROS/ORB-SLAM2/CMakeLists.txt文件下修改 加-lboost_system(没加这个的时候会有关于stl的错误)

```
set(LIBS
${OpenCV_LIBS}
${EIGEN3_LIBS}
${Pangolin_LIBRARIES}
${PROJECT_SOURCE_DIR}/.../.../.../Thirdparty/DBoW2/lib/libDBoW2.so
${PROJECT_SOURCE_DIR}/.../.../.../Thirdparty/g2o/lib/libg2o.so
${PROJECT_SOURCE_DIR}/.../.../.../lib/libORB_SLAM2.so
-lboost_system #此处
)
```

编译结束后启动相机节点:

```
roslaunch realsense2_camera rs_rgbd.launch
```

Ctrl + Alt + T 新建一个命令行:
我们先用一个yaml文件试一下能不能运行

```
cd catkin_ws/ORB_SLAM2
rosrun ORB_SLAM2 RGBD Vocabulary/ORBvoc.txt Examples/RGB-D/TUM1.yaml
```

一个常见的错误:

```
find: 探测到文件系统循环;
`/opt/ros/kinetic/share/ORB_SLAM2/ORB_SLAM2' 是与
 `/opt/ros/kinetic/share/ORB_SLAM2' 相同的文件系统循环的一部分。
[ERROR] [1570687975.286079105]:
[registerPublisher] Failed to contact master at [localhost:11311].  Retrying...
```

这是由于你忘了打开相机节点的缘故。

请看一条不常见的报错:

```
ushio@ushio-Lenovo-XiaoXin-Air-13IWL:~/catkin_s$ rosrun ORB_SLAM2 RGBD Vocabulary/ORBvoc.txt Examples/RGB-D/D435.yam
段错误 (核心已转储)
```

这是因为在catkin_ws下你有两个或更多的ORB_SLAM2包的缘故，请只留下已经编译完的版本。这个错误在以后使用改进版(with_pointcloud版本)的时候可能还会再冒出来一次。

再来看一条罕见的报错:

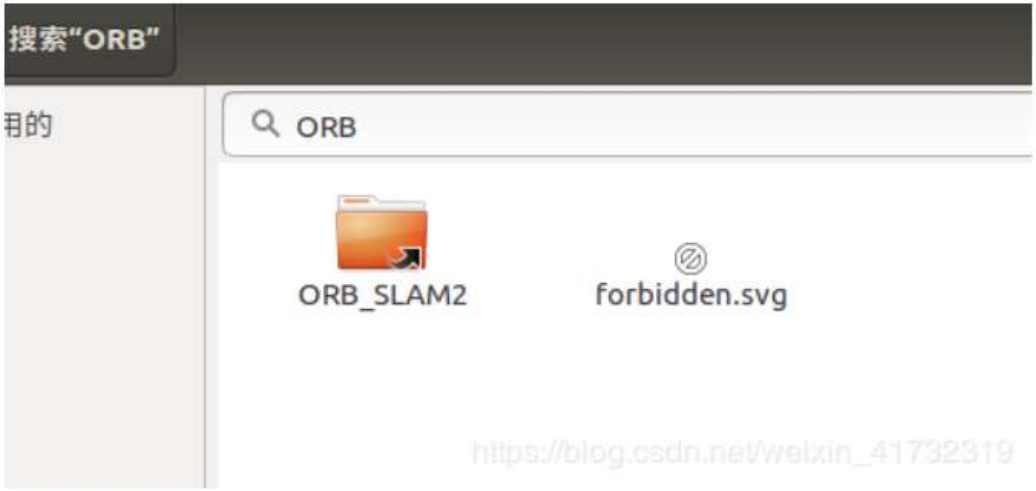
```
[rospack] Error: package 'ORB_SLAM2' not found
```

请进入文件路径 `/opt/ros/kinetic/share` ，查找关键词 `ORB` 。
如果你的结果是:

证明因为某种操作你的工作路径对应的链接丢失了，我们用 `sudo ln-s` 指令镜像一个新的:(类似于超链接)

```
sudo ln -s /home/ushio/catkin_s/ORB_SLAM2/Examples/ROS/ORB_SLAM2 /opt/ros/kinetic/share/ORB_SLAM2
```

此时对应路径下应该如下所示:

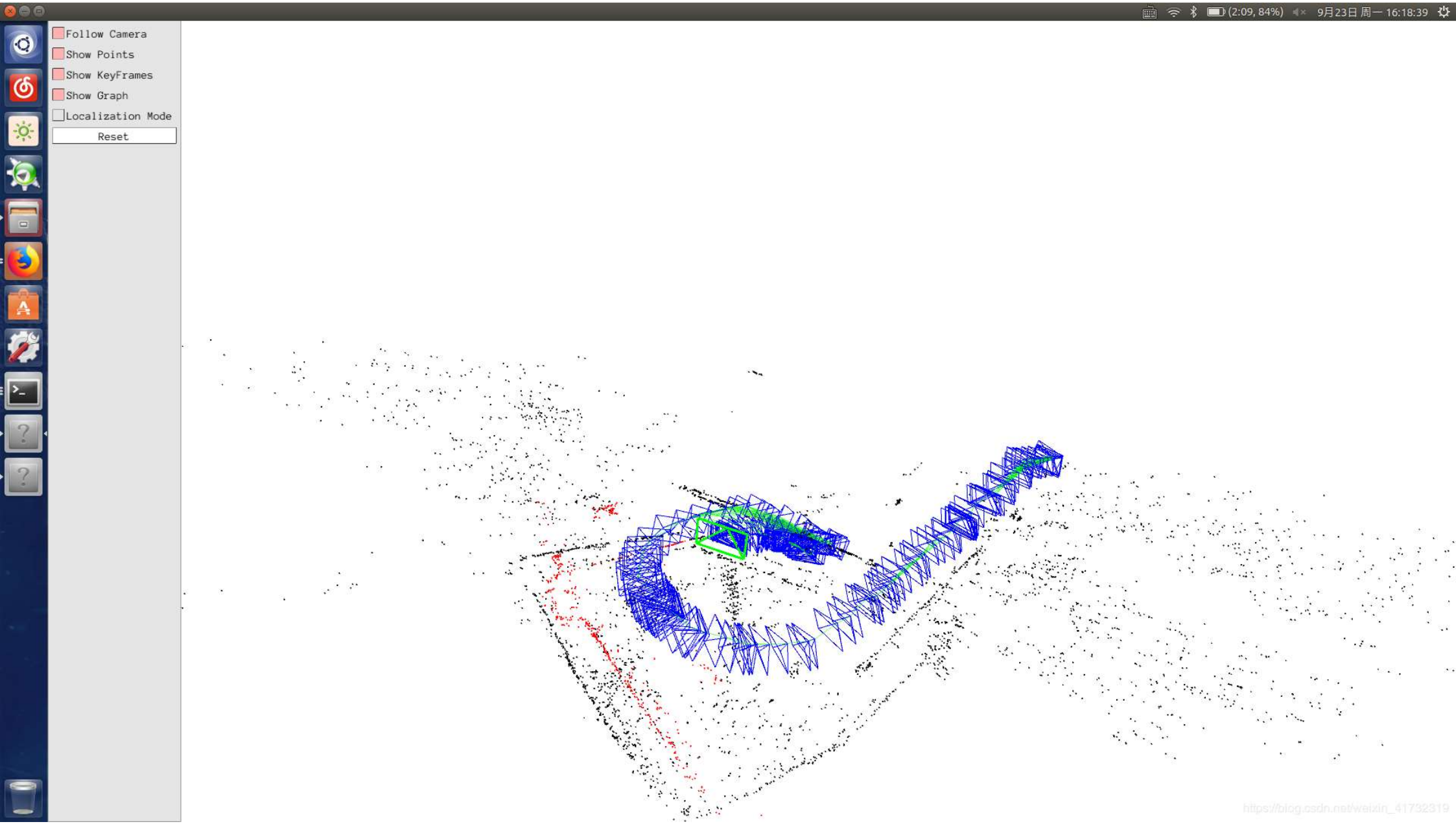


此时再执行 `rosrun ORB_SLAM2 RGBD Vocabulary/ORBvoc.txt Examples/RGB-D/TUM1.yaml` 应该不会有问题。

```
rosrun ORB_SLAM2 RGBD Vocabulary/ORBvoc.txt Examples/RGB-D/TUM1.yaml
```

运行结果如下:

```
ushio@ushio-Lenovo-XiaoXin-Air-13IWL:~/catkin_s/ORB_SLAM2$ rosrun ORB_SLAM2 RGBD Vocabulary/ORBvoc.txt Examples/RGB-D/D435.yar

ORB-SLAM2 Copyright (C) 2014-2016 Raul Mur-Artal, University of Zaragoza.
This program comes with ABSOLUTELY NO WARRANTY;
This is free software, and you are welcome to redistribute it
under certain conditions. See LICENSE.txt.

Input sensor was set to: RGB-D

Loading ORB Vocabulary. This could take a while...
Vocabulary loaded!


Camera Parameters:
- fx: 613.42
- fy: 613.439
- cx: 327.191
- cy: 244.335
- k1: 0
- k2: 0
- k3: 1.04
- p1: 0
- p2: 0
- fps: 30
- color order: RGB (ignored if grayscale)

ORB Extractor Parameters:
- Number of Features: 1000
- Scale Levels: 8
- Scale Factor: 1.2
- Initial Fast Threshold: 20
- Minimum Fast Threshold: 7

Depth Threshold (Close/Far Points): 2.5
New map created with 546 points
```


(2:09, 84%)
9月23日 周一 16:18:39
Follow Camera
Show Points
Show KeyFrames
Show Graph
Localization Mode
Reset
https://blog.csdn.net/weixin_41732319

成功!

我测试的是一个小楼梯，很明显可以看出边缘存在明显的畸变，这是没有使用自己的yaml的缘故，你需要**标定自己的相机**，并制作自己的yaml文件，yaml中存储的是你的相机内参。标定相机步骤略过。

制作yaml文件获取内参

国际惯例贴上Intel Realsense-D435的官方说明书，有兴趣可自己琢磨。不过我还是讲一下它在说什么。

首先确保**相机已经连上电脑并打开相机节点**，Ctrl + Alt + T 新建一个命令行:

```
rostopic echo /camera/color/camera_info
```

此时可以看见命令行在不停地刷新一些信息，大概长这样:

```
---
header:
  seq: 108
  stamp:
    secs: 1548140470
    nsecs: 392033543
  frame_id: "camera_color_optical_frame"
height: 480
width: 640
distortion_model: "plumb_bob"
D: [0.0, 0.0, 0.0, 0.0, 0.0]
K: [614.4114379882812, 0.0, 324.2138671875, 0.0, 614.7125244140625, 236.86329650878906, 0.0, 0.0, 1.0]
R: [1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0]
P: [614.4114379882812, 0.0, 324.2138671875, 0.0, 0.0, 614.7125244140625, 236.86329650878906, 0.0, 0.0, 0.0, 1.0, 0.0
```

你应该还记得在运行 `rosrun ORB_SLAM2 RGBD Vocabulary/ORBvoc.txt Examples/RGB-D/TUM1.yaml` 指令的时候，命令行界面显示的是:

```
ORB-SLAM2 Copyright (C) 2014-2016 Raul Mur-Artal, University of Zaragoza.
This program comes with ABSOLUTELY NO WARRANTY;
This is free software, and you are welcome to redistribute it
under certain conditions. See LICENSE.txt.

Input sensor was set to: RGB-D

Loading ORB Vocabulary. This could take a while...
Vocabulary loaded!


Camera Parameters:
- fx: 613.42
- fy: 613.439
- cx: 327.191
- cy: 244.335
- k1: 0
- k2: 0
- k3: 1.04
- p1: 0
- p2: 0
- fps: 30
- color order: RGB (ignored if grayscale)
```

我们关注这一行代码:

```
K: [614.4114379882812, 0.0, 324.2138671875, 0.0, 614.7125244140625, 236.86329650878906, 0.0, 0.0, 1.0]
```

K就是相机的内参矩阵写成行向量后的结果，四个非零数值依次对应运行 `rosrun ORB_SLAM2 RGBD Vocabulary/ORBvoc.txt Examples/RGB-D/TUM1.yaml` 指令后的这四行数值:

```
- fx: 613.42
- fy: 613.439
- cx: 327.191
- cy: 244.335
```

此时我们可以新建一个yaml后缀文件，以yaml1为模板修改这四个数值。

打开yaml1，另一个需要关心的数值如下:

```
# IR projector baseline times fx (aprox.)
Camera.bf: 30.720571899
```

翻阅到官方说明书第33页:

Table 3-42. Depth Camera SKU properties

D400 series Depth Cameras	Intel® RealSense™ Depth Camera D415	Intel® RealSense™ Depth Camera D435	Intel® RealSense™ Depth Camera D435i
Depth module	Intel® RealSense™ Depth module D415	Intel® RealSense™ Depth module D430	Intel® RealSense™ Depth module D430
Baseline	55mm	50mm	50mm
Left/Right Imagers Type	Standard	Wide	Wide
Depth FOV HD (degrees)	H:65±2 / V:40±1 / D:72±2	H:87±3 / V:58±1 / D:95±3	H:87±3 / V:58±1 / D:95±3
Depth FOV VGA (degrees)	H:50±2 / V:40±1 / D:61±2	H:75±3 / V:62±1 / D:89±3	H:75±3 / V:62±1 / D:89±3
IR Projector	Standard	Wide	Wide
IR Projector FOV	H:67 / V:41 / D:75	H:90 / V:63 / D:99	H:90 / V:63 / D:99
Color Sensor	OV2740	OV2740	OV2740
Color Camera FOV	H:69±1 /V:42±1 /D:77±1	H:69±1 /V:42±1 /D:77±1	H:69±1 /V:42±1 /D:77±1
IMU	NA	NA	6DoF

NOTE: H – Horizontal FOV, V – Vertical FOV, D – Diagonal FOV, X – Length, Y – Breadth, Z – Thickness

D435的Baseline基线长度为50毫米。`Camera.bf = ybaseline (in meters) * fx`。
根据此条公式计算出D435的bf值，写入你的yaml中。
最后文件可命名为D435.yaml

一切都完成后，**确保相机插入、重新启动相机节点**并在ORB_SLAM2下运行:

```
rosrun ORB_SLAM2 RGBD Vocabulary/ORBvoc.txt Examples/RGB-D/D435.yaml
```

现在可以开始SLAM建图了!