

# 机器人 DIY 项目指导书——Turtlebot 篇

## 目录

1.	项目介绍 .....	2
2.	学习资料 .....	2
3.	DO IT YOURSELF .....	3
3.1	环境配置篇 .....	3
3.1.1	安装 Ubuntu 14.04.....	3
3.1.2	安装 Indigo.....	3
3.1.3	安装 Turtlebot 相关包 .....	3
3.1.4	网络配置 .....	4
3.1.5	测试网络配置是否成功.....	4
3.2	Demo 运行篇.....	5
3.2.1	获取并显示机载 Kinect 图像数据.....	5
3.2.2	遥控 Turtlebot 机器人.....	7
3.2.3	Turtlebot 机器人 Follower.....	8
3.2.4	使用 Turtlebot 机器人创建室内地图 .....	9
3.2.5	Turtlebot 自主导航 .....	10
3.3	提高篇.....	12
3.3.1	创建 ROS 工作区间 .....	13
3.3.2	创建 ROS 程序包 .....	14
3.3.3	编译 demo 程序 .....	14
3.3.4	在模拟器中运行 demo 程序 .....	14
3.3.5	修改 demo 程序 .....	15
3.3.6	运行修改后 demo 程序 .....	15

## 1. 项目介绍



该DIY项目使用 Turtlebot2 机器人作为项目执行平台，希望通过该项目让大家了解 ROS (Robot operation system) 的一些基础知识，实现智能机器人的入门级开发。项目主要分为环境配置篇，Demo 运行篇和提高篇。

环境配置篇，大家将会体验 ROS 系统，Turtlebot 相关程序包安装全过程，以及 pc 机与 Turtlebot 端的连接过程。

Demo 运行篇，大家将会运行 Turtlebot 相关的 demo 程序包，完成一些非常有趣的实验，如：遥控机器人，建立室内地图，室内导航等。

提篇中，我们将会提供给大家一个 demo 程序，并教会大家如何运行该 demo 程序并查看效果，大家可以对该 demo 程序进行修改，完成更加丰富的功能。

## 2. 学习资料

ROS 相关资料：<http://www.ros.org/>

ROS 入门教程：<http://wiki.ros.org/ROS/Tutorials>

Turtlebot 相关资料：<http://learn.turtlebot.com/>

<http://wiki.ros.org/Robots/TurtleBot>

以上资料供大家学习参考，在本项目中，即使大家没有这些基础知识也可以继续完成项目任务，因为后面会详细介绍完成本项目的具体步骤。如果大家对智能机器人开发有浓厚的兴趣，那么认真学习以上资料是非常有必要的。

## 3. DO IT YOURSELF

### 3.1 环境配置篇

我们把 Turtlebot 自带笔记本电脑叫做 Turtlebot Netbook，用户使用的 PC 机叫做 Workstation，以下配置皆在 Workstation 上进行，Turtlebot 端所有环境已经配置好。

#### 3.1.1 安装 Ubuntu 14.04

关于 Ubuntu 14.04 的安装方法，网上很多资料，请自行查阅相关资料。

#### 3.1.2 安装 Indigo

```
$ 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://pool.sks-keyservers.net --recv-key
0xB01FA116
$ sudo apt-get update
$ sudo apt-get install ros-indigo-desktop-full
$ sudo rosdep init
$ rosdep update
$ echo "source /opt/ros/indigo/setup.bash" >> ~/.bashrc
$ source ~/.bashrc
$ sudo apt-get install python-rosinstall
```

#### 3.1.3 安装 Turtlebot 相关包

```
$ sudo apt-get install ros-indigo-turtlebot-*
```

### 3.1.4 网络配置

在 Turtlebot 端:

```
$ ifconfig
```

获取 IP\_OF\_TURTLEBOT

在 Workstation 端:

```
$ ifconfig
```

获取 IP\_OF\_PC

```
$ echo export ROS_MASTER_URI=http://IP_OF_TURTLEBOT:11311 >> ~/.bashrc
```

```
$ echo export ROS_HOSTNAME=IP_OF_PC >> ~/.bashrc
```

```
$ ssh turtlebot@[IP_OF_TURTLEBOT]
```

### 3.1.5 测试网络配置是否成功

1. 登录 Turtlebot 主机

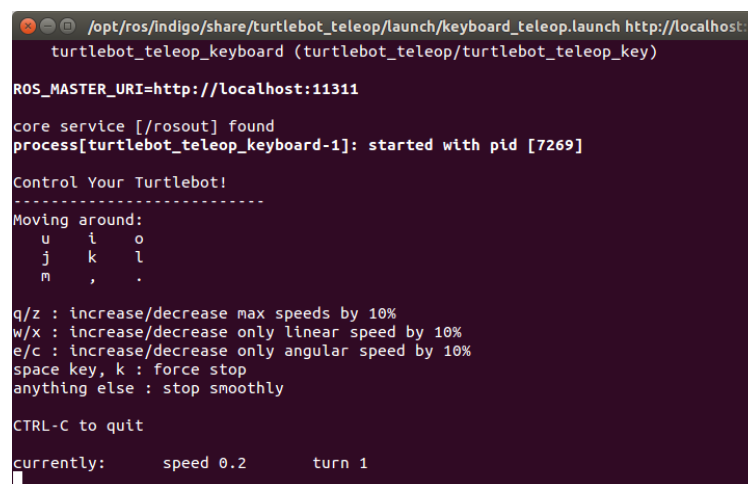
```
$ ssh turtlebot@[IP_OF_TURTLEBOT]
```

2. 启动 Turtlebot

```
$ roslaunch turtlebot_bringup minimal.launch
```

4. 在 Workstation 端另起一个终端运行下面命令:

```
$ roslaunch turtlebot_teleop keyboard_teleop.launch
```



```
/opt/ros/indigo/share/turtlebot_teleop/launch/keyboard_teleop.launch http://localhost:11311
turtlebot_teleop_keyboard (turtlebot_teleop/turtlebot_teleop_key)

ROS_MASTER_URI=http://localhost:11311
core service [/rosout] found
process[turtlebot_teleop_keyboard-1]: started with pid [7269]

Control Your Turtlebot!
-----
Moving around:
  u   i   o
  j   k   l
  m   ,   .

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
space key, k : force stop
anything else : stop smoothly

CTRL-C to quit

currently:      speed 0.2      turn 1
```

此时，能够根据终端提示，通过键盘控制 turtlebot 则表示网络配置成功。

## 3.2 Demo 运行篇

在进行这一步之前关闭 Turtlebot 端和 workstation 端所有终端。

### 3.2.1 获取并显示机载 Kinect 图像数据

在该部分我们尝试使用不同的方式来显示机载 Kinect 图像数据(深度图像和彩色图像)。

如果没有启动 Turtlebot，则执行下面两步启动 Turtlebot；如果 Turtlebot 已经启动，则不需要执行启动步骤。

登录 Turtlebot 主机：

```
ssh turtlebot@[IP_OF_TURTLEBOT]
```

在 Turtlebot 端运行下面命令启动 Turtlebot：

```
$ roslaunch turtlebot_bringup minimal.launch
```

#### 1. 通过 OpenCV 框架显示图像

启动另外一个终端登入 Turtlebot 主机：

```
ssh turtlebot@[IP_OF_TURTLEBOT]
```

在 Turtlebot 端运行下面命令启动 Kinect 摄像头：

```
$ roslaunch openni_launch openni.launch
```

在 workstation 端运行下面命令显示 RGB 图像：

```
$ rosrn image_view image_view image:=/camera/rgb/image_raw
```



停止 workstation 端程序 (Ctrl+C), 在 workstation 端运行下面命令显示深度图像:

```
$ rosrn image_view image_view image:=/camera/depth/image
```



## 2. 通过 Rviz 显示图像

在 Turtlebot 端停止 openni 程序 (Ctrl+C), 并停止 workstation 端所有程序。

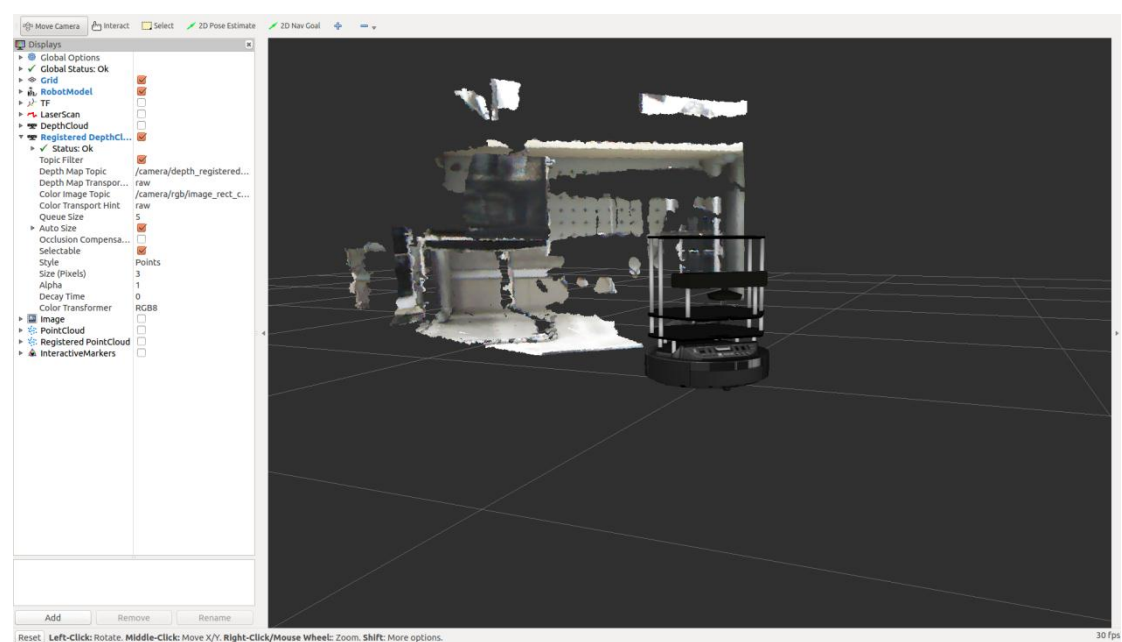
在 Turtlebot 端运行下面命令:

```
$ roslaunch turtlebot_bringup 3dsensor.launch
```

在 workstation 端运行下面命令:

```
$ roslaunch turtlebot_rviz_launchers view_robot.launch
```

在界面中勾选 Registered DepthCloud 选项, 显示深度图像和彩色图像。



### 3.2.2 遥控 Turtlebot 机器人

在该部分我们分别使用键盘和 Rviz 来遥控 Turtlebot 机器人。

停掉所有在 Turtlebot 端（turtlebot 启动程序除外）和 workstation 端运行的程序(Ctrl+C)。

如果没有启动 Turtlebot，则执行下面两步启动 Turtlebot；如果 Turtlebot 已经启动，则不需要执行启动步骤。

登录 Turtlebot 主机：

```
ssh turtlebot@[IP_OF_TURTLEBOT]
```

在 Turtlebot 端运行下面命令启动 Turtlebot：

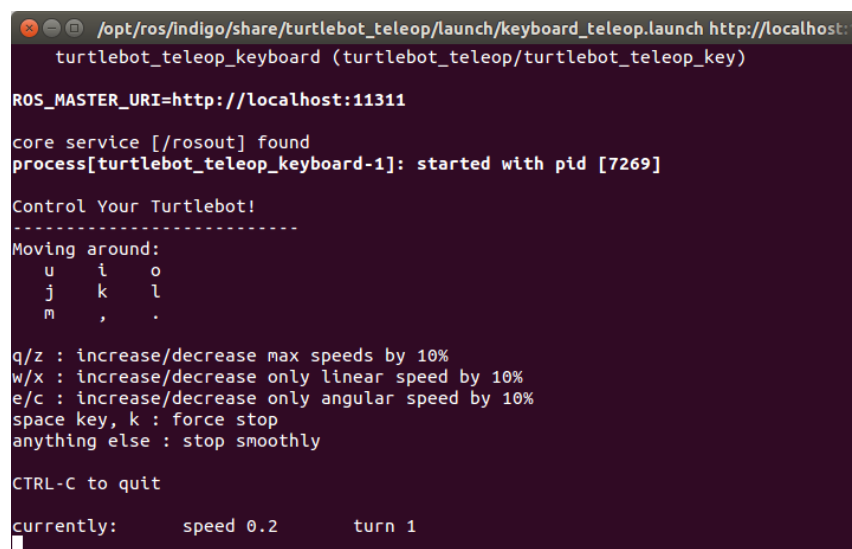
```
$ roslaunch turtlebot_bringup minimal.launch
```

#### 1. 键盘控制机器人

在 workstation 端运行：

```
$ roslaunch turtlebot_teleop keyboard_teleop.launch
```

根据界面提示控制机器人前进后退以及旋转。



```
/opt/ros/indigo/share/turtlebot_teleop/launch/keyboard_teleop.launch http://localhost:11311
turtlebot_teleop_keyboard (turtlebot_teleop/turtlebot_teleop_key)

ROS_MASTER_URI=http://localhost:11311

core service [/rosout] found
process[turtlebot_teleop_keyboard-1]: started with pid [7269]

Control Your Turtlebot!
-----
Moving around:
  u    i    o
  j    k    l
  m    ,    .

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
space key, k : force stop
anything else : stop smoothly

CTRL-C to quit

currently:      speed 0.2      turn 1
```

#### 2. 通过 Rviz 界面交互控制机器人

停止 workstation 端所有程序(Ctr+C)。

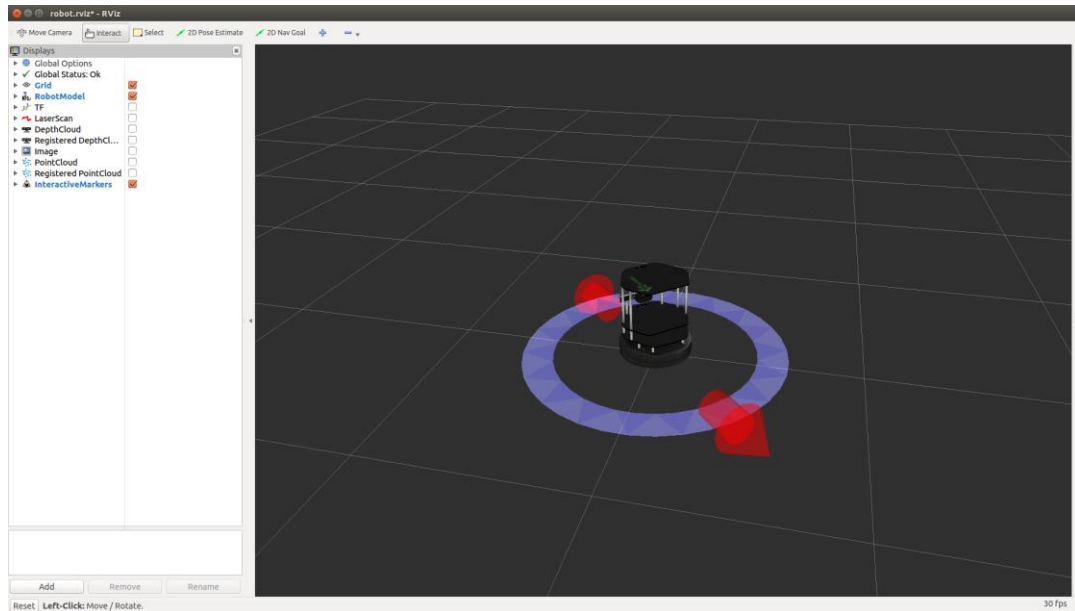
在 workstation 端运行 turtlebot\_maker\_server：

```
roslaunch turtlebot_interactive_markers interactive_markers.launch
```

在 workstation 端运行 rviz

```
roslaunch turtlebot_rviz_launchers view_robot.launch
```

交互操作：选中工具栏中的“interactive marker”并点击 interact 来操作 turtlebot。红色箭头控制机器人前进后退，紫色圆环控制机器人旋转。



### 3.2.3 Turtlebot 机器人 Follower

在该部分我们让 TurtleBot 寻找其前方一定范围内的物体，并把物体的中心作为他的视点，如果物体中心离它太远，TurtleBot 会向物体中心开去方向，如果物体中心离它太近，TurtleBot 会向后退，如果中心偏移，TurtleBot 会旋转。

停掉所有在 Turtlebot 端（turtlebot 启动程序除外）和 workstation 端运行的程序(Ctrl+C)。

如果没有启动 Turtlebot，则执行下面两步启动 Turtlebot；如果 Turtlebot 已经启动，则不需要执行启动步骤。

登录 Turtlebot 主机：

```
ssh turtlebot@[IP_OF_TURTLEBOT]
```



在 Turtlebot 端运行下面命令启动 Turtlebot:

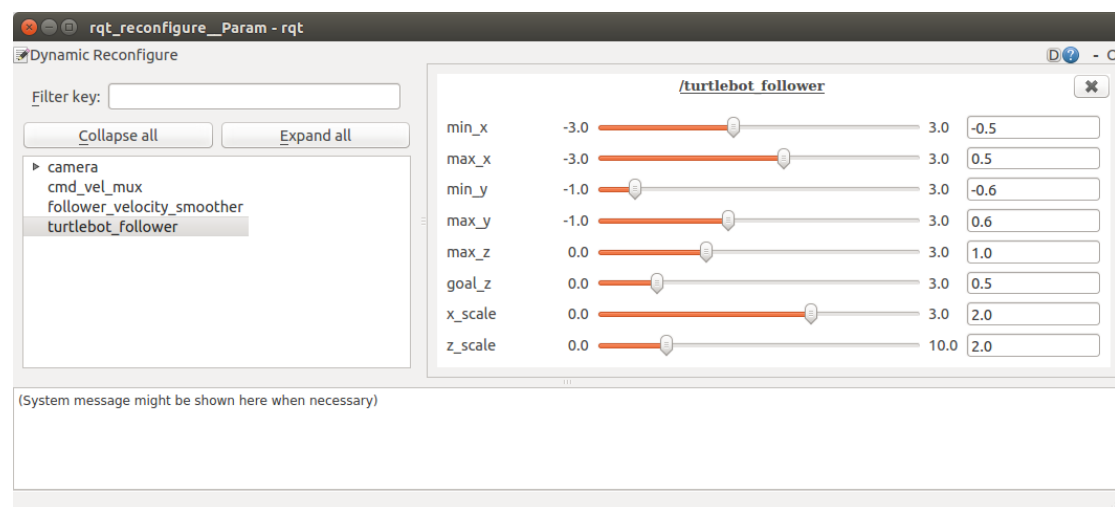
```
$ roslaunch turtlebot_bringup minimal.launch
```

打开另一终端, 登入 Turtlebot 主机, 运行下面命令启动 follower 程序:

```
roslaunch turtlebot_follower follower.launch
```

在 workstation 端运行:

```
roslaunch rqt_reconfigure rqt_reconfigure
```



我们可以通过这个调试表来调试 turtlebot 跟随的很多参数。

### 3.2.4 使用 Turtlebot 机器人创建室内地图

在该部分我们控制 Turtlebot 机器人扫描房间, 建立房间地图并保存。

停掉所有在 Turtlebot 端 (turtlebot 启动程序除外) 和 workstation 端运行的程序 (Ctrl+C)。

如果没有启动 Turtlebot, 则执行下面两步启动 Turtlebot; 如果 Turtlebot 已经启动, 则不需要执行启动步骤。

登录 Turtlebot 主机:

```
ssh turtlebot@[IP_OF_TURTLEBOT]
```

在 Turtlebot 端运行下面命令启动 Turtlebot:

```
$ roslaunch turtlebot_bringup minimal.launch
```

打开另一个终端，登录 Turtlebot 主机后

```
ssh turtlebot@[IP_OF_TURTLEBOT]
```

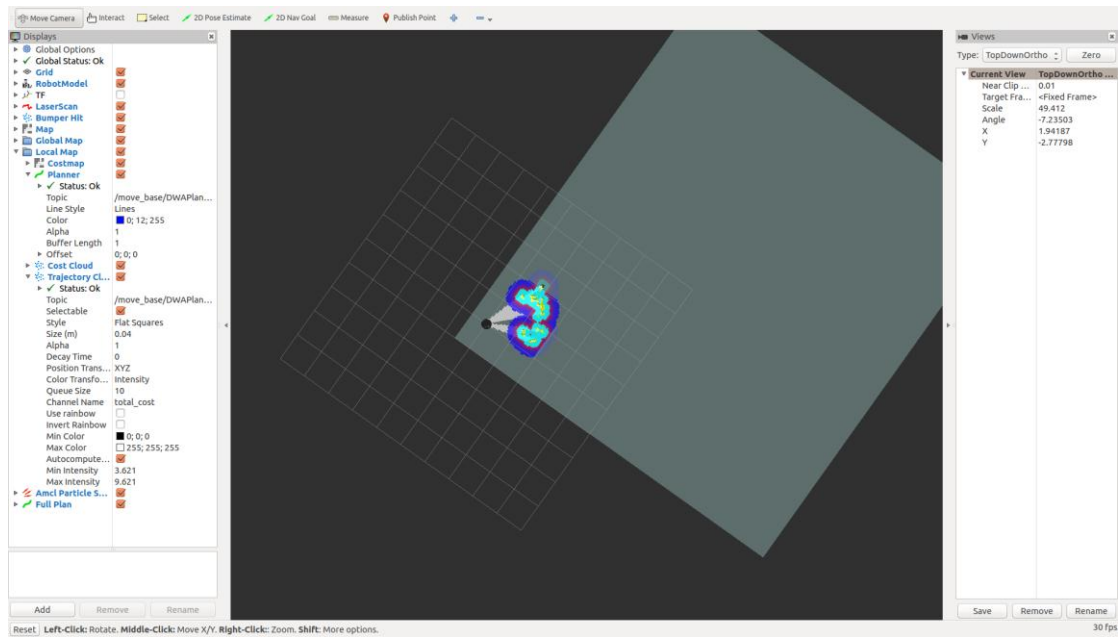
运行下面命令启动建地图程序：

```
$ roslaunch turtlebot_navigation gmapping_demo.launch
```

在 workstation 端打开两个终端分别运行：

```
$ roslaunch turtlebot_rviz_launchers view_navigation.launch
```

```
$ roslaunch turtlebot_teleop keyboard_teleop.launch
```



通过键盘遥控 Turtlebot 机器人扫描房间，当地图建立完全后，登入 Turtlebot 主机，在 Turtlebot 端运行下面命令保存地图：

```
$ rosrn map_server map_saver -f /tmp/my_map
```

运行下面命令：

```
$ ls /tmp/
```

我们能够看到/tmp 文件夹下有两个文件 my\_map.pgm 和 my\_map.yaml，表明地图保存成功。

### 3.2.5 Turtlebot 自主导航

停掉所有在 Turtlebot 端（turtlebot 启动程序除外）和 workstation 端运行的

程序(Ctrl+C)。

如果没有启动 Turtlebot，则执行下面两步启动 Turtlebot；如果 Turtlebot 已经启动，则不需要执行启动步骤。

登录 Turtlebot 主机：

```
ssh turtlebot@[IP_OF_TURTLEBOT]
```

在 Turtlebot 端运行下面命令启动 Turtlebot：

```
$ roslaunch turtlebot_bringup minimal.launch
```

打开另一个终端，登入 Turtlebot 主机，运行下面命令：

```
$ roslaunch turtlebot_navigation amcl_demo.launch  
  map_file:=/tmp/my_map.yaml
```

如果看到“odom received!”提示信息，可以进行下一步操作。如果看到“Waiting on transform...”提示，尝试重新启动 minimal.launch 和 amcl\_demo.launch，可能需要重新启动多次才可成功。也可以通过关闭 Kobuki base 然后重新打开来解决问题。

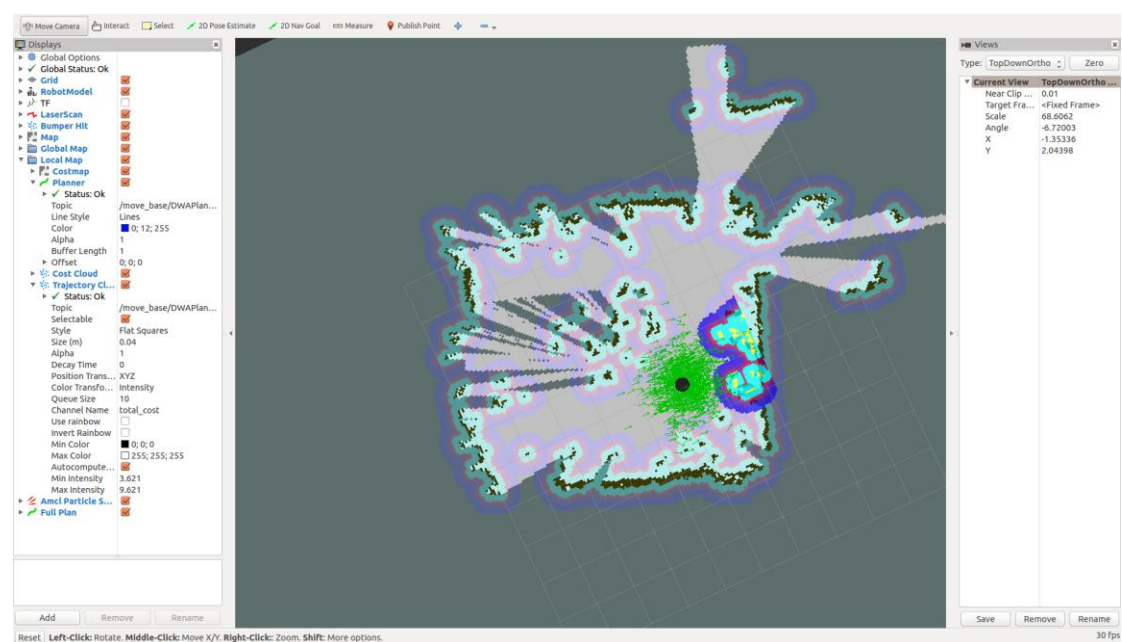
在 workstation 端运行：

```
$ roslaunch turtlebot_rviz_launchers view_navigation.launch --screen
```

开始的时候 turtlebot 不知道他自己在地图的什么位置，所以我们要指示给他他的初始位置和面向。点击“2D Pose Estimate”按钮，通过交互设置 Turtlebot 在地图上的大致位置和朝向，具体做法为：点击地图上 Turtlebot 所在的位置然后拉箭头到 Turtlebot 面向的方向。

然后指定 Turtlebot 要去的位置，点击“2D Nav Goal”按钮，点击地图上任意一点你想让 Turtlebot 去的地方然后选择箭头的朝向来确定 Turtlebot 的面向。此时，Turtlebot 会自动行驶到你指定的地点。

**注意：**在此过程中一定要紧随 Turtlebot，防止碰撞到物体。因为一方面我们指定的 Turtlebot 初始位置可能不准确，另一方面，Kinect 对黑色的物体不敏感，可能不会把他作为障碍物。



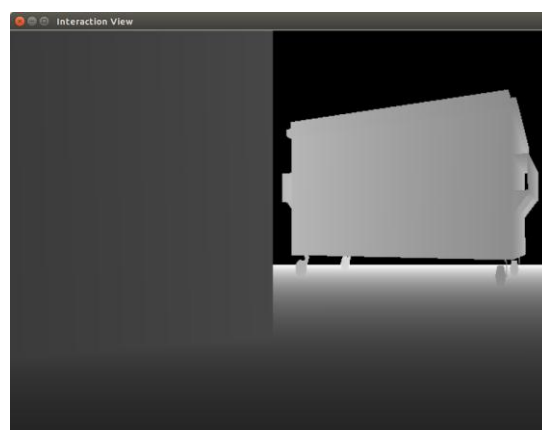
### 3.3 提高篇

在该部分我们需要编程实现以下的功能：

1. 在界面上显示 Turtlebot 上 kinect 采集到的数据。
2. 在界面上通过鼠标控制彩色图和深度图之间的转换。
3. 在界面上通过鼠标控制 Turtlebot 前进，后退以及左右旋转。



颜色图



深度图

**具体要求：**初始界面显示 Turtlebot 上 kinect 获取的彩色图像，鼠标左击界面的上半部分控制 turtlebot 前进，鼠标左击界面的下半部分控制 turtlebot 后退，鼠标右击界面的左半部分控制 turtlebot 向左旋转，鼠标右击界面的右半部分控制 turtlebot 向右旋转，点击鼠标中键，能够切换彩色图像和深度图像。

我们会提供给大家一个 demo 程序，该程序实现了以下功能：

1. 在界面上显示 Turtlebot 上 kinect 采集到的彩色图像。
3. 在界面上通过鼠标控制 Turtlebot 前进，后退。



**该 demo 程序实现的具体功能为：**初始界面显示 Turtlebot 上 kinect 获取的彩色图像，鼠标左击界面的上半部分控制 turtlebot 前进，鼠标左击界面的下半部分控制 turtlebot 后退。

所以大家需要实现的功能是在界面上通过鼠标控制 Turtlebot 的左右旋转以及在界面上通过鼠标控制彩色图和深度图之间的转换。

该部分向大家介绍了如何在模拟环境中测试该 demo 程序，大家可以在模拟器中进行调试，并最终在真机上进行运行。

### 3.3.1 创建 ROS 工作区间

这里我们使用 catkin 工作区间。

```
$ mkdir -p ~/catkin_ws/src
$ cd ~/catkin_ws/src
$ catkin_init_workspace
$ cd ~/catkin_ws/
$ catkin_make
$ echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
$ source ~/.bashrc
```

### 3.3.2 创建 ROS 程序包

```
$ cd ~/catkin_ws/src
```

```
$ catkin_create_pkg turtlebot_teleop_with_image
```

此时解压附件中的 **turtlebot\_teleop\_with\_image.rar** 文件，

将附件 **turtlebot\_teleop\_with\_image** 文件夹下的所有文件拷贝到 `~/catkin_ws/src/turtlebot_teleop_with_image` 文件夹下并替换重复文件。

### 3.3.3 编译 demo 程序

```
$ cd ~/catkin_ws/
```

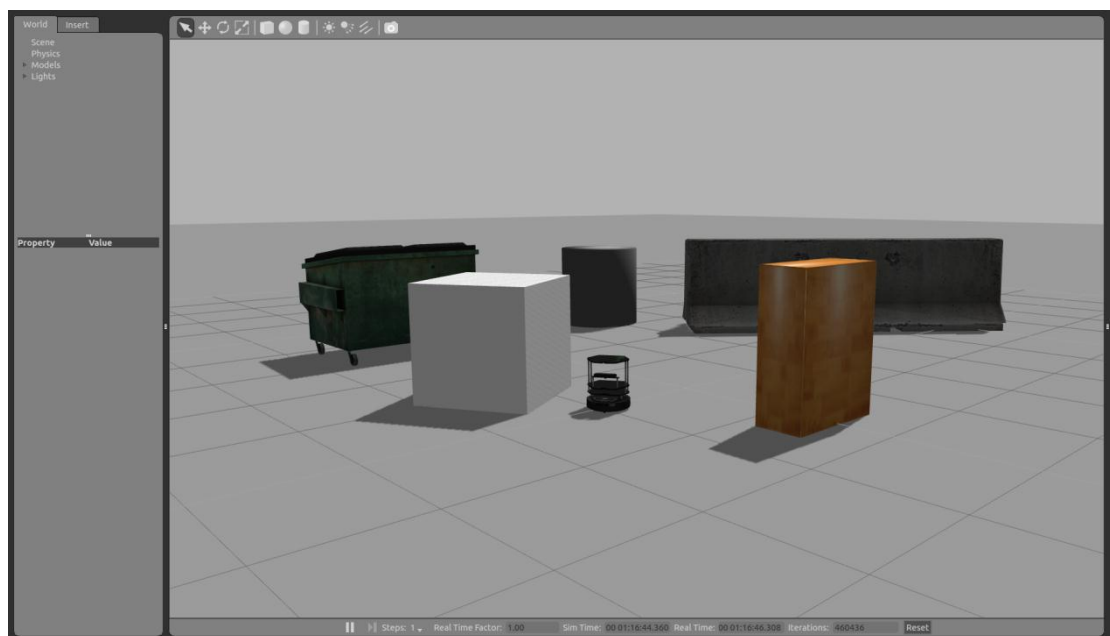
```
$ catkin_make
```

### 3.3.4 在模拟器中运行 demo 程序

#### 1. 运行模拟环境

```
$ roslaunch turtlebot_gazebo turtlebot_world.launch
```

此时会出现一个虚拟的世界，该空间有一些障碍物和 turtlebot 机器人。



#### 2. 运行 demo 程序

```
$ rosrn turtlebot_teleop_with_image move_listener.py
```

```
$ rosrn turtlebot_teleop_with_image interaction_view
```



此时会出现一个界面，该界面显示 Turtlebot 上 kinect 获取的彩色图像，鼠标左击界面的上半部分控制 turtlebot 前进，鼠标左击界面的下半部分控制 turtlebot 后退。

### 3.3.5 修改 demo 程序

**提示：**需要修改文件为 `~/catkin_ws/src/ turtlebot_teleop_with_image /script` 文件夹下的 `move_listener.py` 文件和 `~/catkin_ws/src/ turtlebot_teleop_with_image /src/nodelets` 文件夹下的 `image_nodelet.cpp` 文件。

### 3.3.6 运行修改后 demo 程序

在模拟器中测试修改后的 demo 程序，测试成功后，连接 turtlebot 机器人，测试程序。