ECE 4703
Mobile Autonomous Robots

Jenny Zhen Yu
zhenyu@cpp.edu
Department of Electrical and Computer Engineering
California State Polytechnic University, Pomona

Lecture 4: Robot, Sensor, Motor

Outline

- **Mobile Autonomous Robots Introduction**
  - Sensor packages
  - Camera
  - Depth camera
  - Laser distance sensor
  - Dynamixel
  - How to use open packages

Sensor Package

Type of Sensor Package

- **1D Range Finders**
  - Infrared linear distance sensor that can be used to make low-cost robots
- **2D Range Finders**
  - Sensors that can measure the distance on 2D plane, and is mainly used for navigation
- **3D Sensors**
  - Sensors used in 3D distance measurement such as Intel's RealSense, Microsoft's Kinect, ASUS's Xtion
- **Audio/Speech Recognition**
  - Currently, there are few voice recognition related parts, but it seems to be added continuously
- **Cameras**
  - Camera driver used for object recognition, face recognition, character recognition, etc. and various application packages
- **Sensor Interfaces**
  - Very few sensors support USB and web protocols
  - There are still many sensors that can acquire data from a microprocessor
  - These sensors can be used with UART in MCU, or ROS in mini PC.

Sensor Package Practice #1 (USB Camera)

```
$ sudo apt-get install ros-kinetic-uvc-camera

$ rosrun uvc_camera uvc_camera_node
$ rosrun uvc_camera uvc_camera_node _device:=/dev/video?

$ rosrun image_view image_view image:=/image_raw
$ rqt_image_view image:=/image_raw
$ rviz
```

If there are more than two cameras,
Enter the device number you want to use
instead of the question mark
(Especially, for notebooks)

Three ways to view image messages

* Change the display options of RViz

1) Change fixed frame

   Global Options > Fixed Frame = camera

2) Add image display

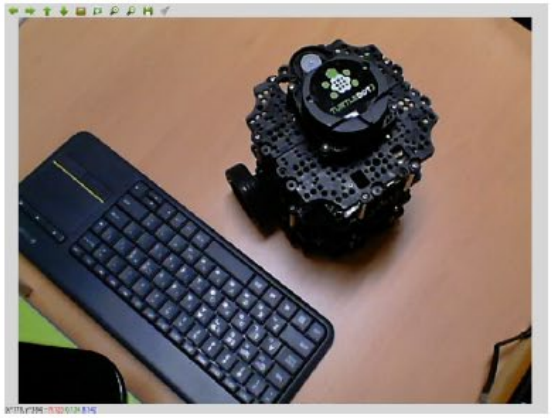
   Click 'Add' in the bottom left corner of Rviz, then select Image

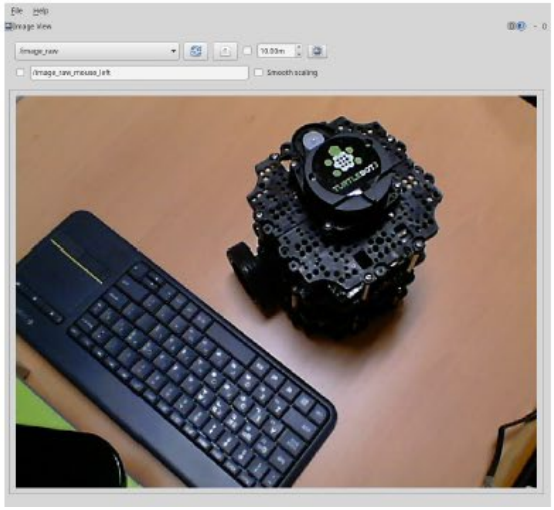
   (Add > by display > Rviz > Image)

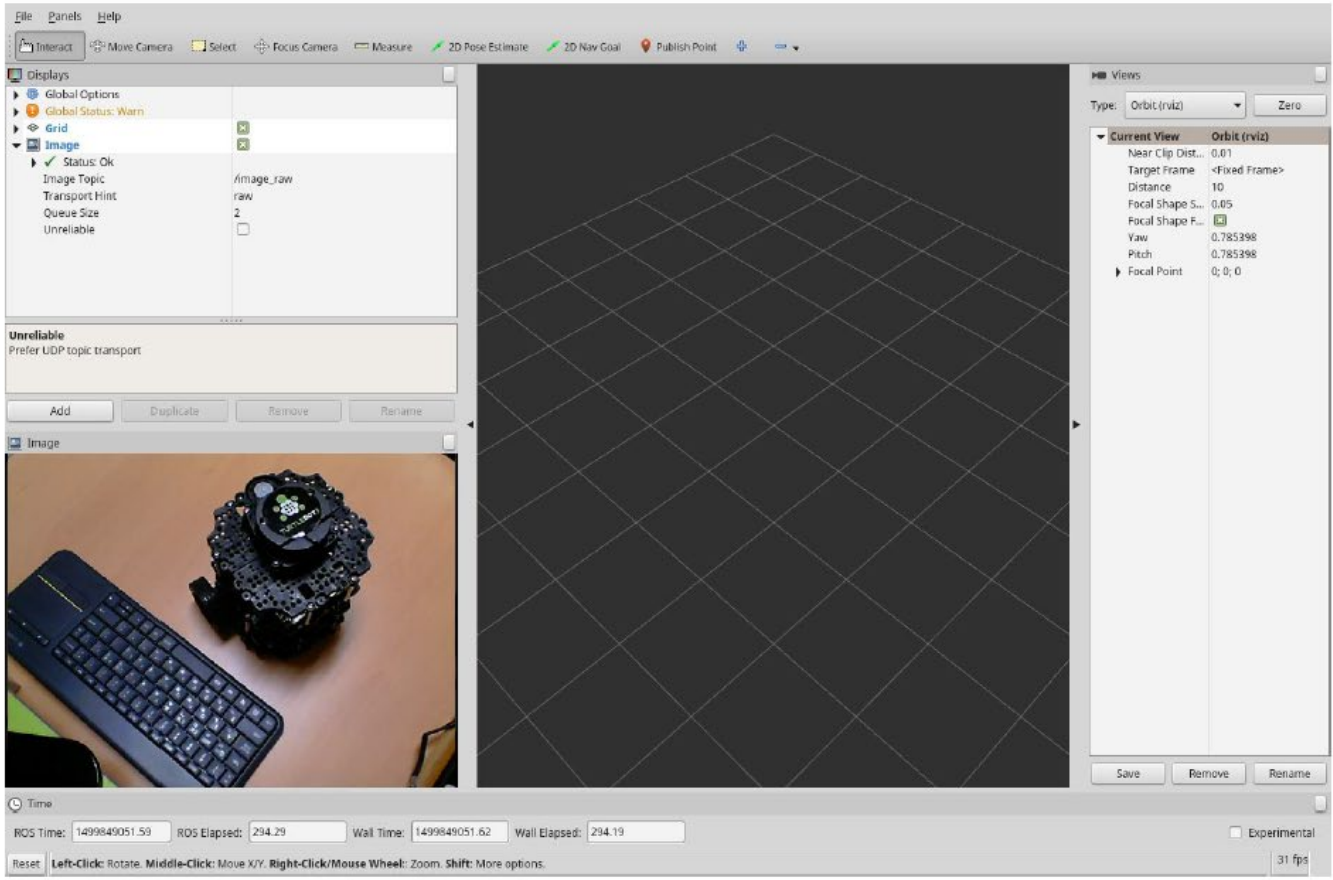
3) Change topic value

   Change the value of 'Image > Image Topic' to "/image_raw"

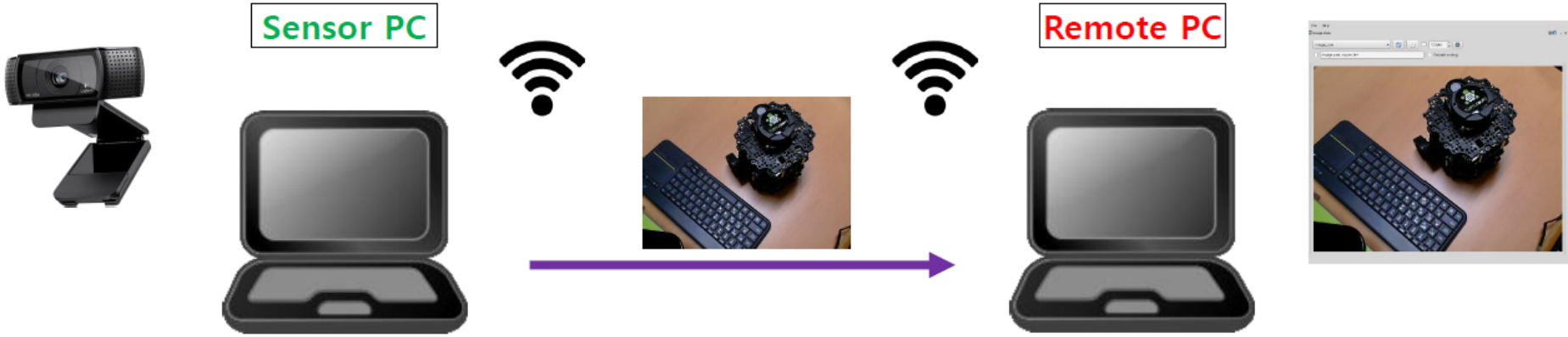
Sensor Package Practice #1 (USB Camera)







Sensor Package Practice #2 (Transfer images remotely)



ROS_MASTER_URI = http://IP_OF_REMOTE_PC:11311
ROS_HOSTNAME = IP_OF_SENSOR_PC

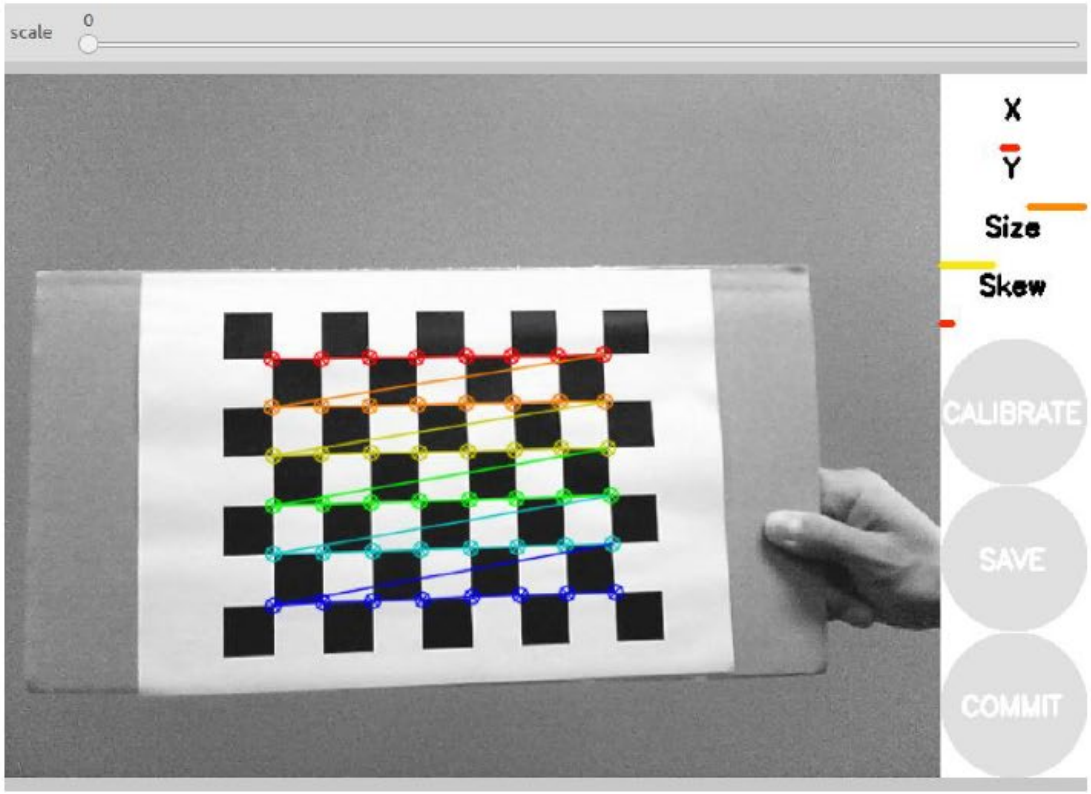
ROS_MASTER_URI = http://IP_OF_REMOTE_PC:11311
ROS_HOSTNAME = IP_OF_REMOTE_PC

* Example of running ROS Master on a remote PC

- Modify '~/.bashrc' for each PC (ROS_MASTER_URI and ROS_HOSTNAME)
- Run '**roscore**' & '**rqt_image_view image:=/image_raw**' on the remote PC
- Run '**rosrun uvc_camera**', '**uvc_camera_node**' on the sensor PC

Sensor Package Practice #3 (Camera Calibration)

```
$ sudo apt-get install ros-kinetic-camera-calibration
$ rosrun uvc_camera uvc_camera_node
$ rosrun camera_calibration cameracalibrator.py --size 8x6 --square 0.024 image:=/image_raw camera:=/camera
```



Sensor Package Practice #4 (Depth Camera)

```
$ sudo apt-get install ros-kinetic-openni2-camera ros-kinetic-openni2-launch
$ tar -xvf Sensor-Bin-Linux-x64-v5.1.0.41.tar.bz2
$ cd Sensor-Bin-Linux-x64-v5.1.0.41/
$ sudo sh install.sh
$ roslaunch openni2_launch openni2.launch
```

(In case of ASUS's Xtion)

```
$ sudo apt-get install ros-kinetic-astra-camera ros-kinetic-astra-launch
$ wget https://raw.githubusercontent.com/tfoote/ros_astra_camera/master/orbbec-usb.rules
$ wget https://raw.githubusercontent.com/tfoote/ros_astra_camera/master/install.sh
$ sudo ./install.sh
$ roslaunch astra_launch astra.launch
```

(In case of ASTRA)

* Change the display options of RViz

1) Change fixed frame
   Change 'Global Options > Fixed Frame' to "camera_depth_frame"

2) Add & configure PointCloud2
   Click 'Add' at the bottom left of rviz, then select PointCloud2

3) Change topic name & detail settings

Sensor Package Practice #4 (Depth Camera)

(In case of RealSense)

```
$ sudo apt-get install ros-kinetic-librealsense ros-kinetic-realsense-camera
$ roslaunch realsense_camera r200_nodelet_default.launch
$ rosrun rviz rviz -d rviz/realsenseRvizConfiguration1.rviz
```

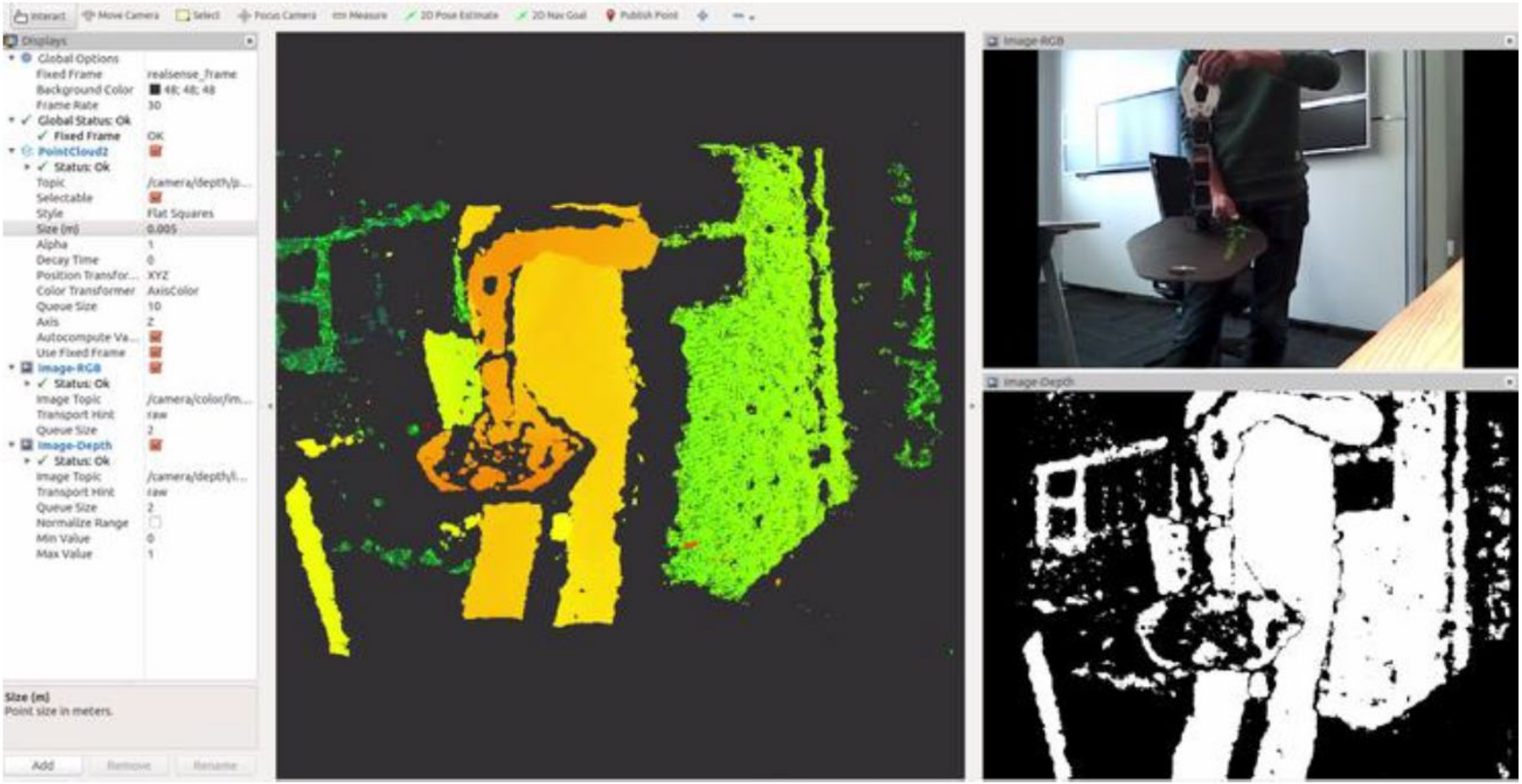
* Change the display options of RViz

1) Change fixed frame
   Change 'Global Options > Fixed Frame' to "camera_depth_frame"

2) Add & configure PointCloud2
   Click 'Add' at the bottom left of rviz, then select PointCloud2

3) Change topic name & detail settings

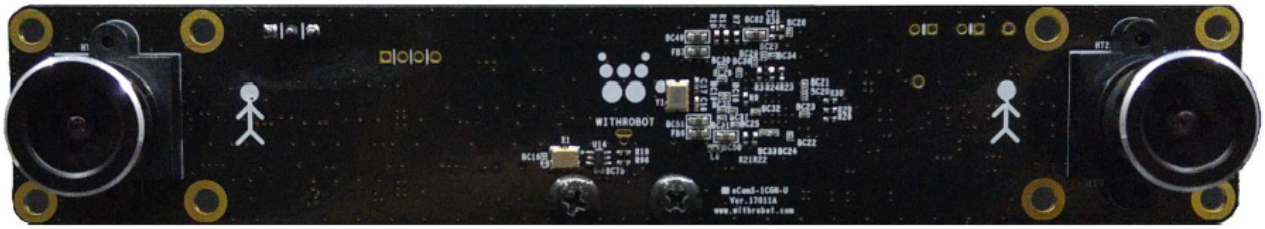
Sensor Package Practice #4 (Depth Camera)



Sensor Package Practice #5 (Stereo Camera)

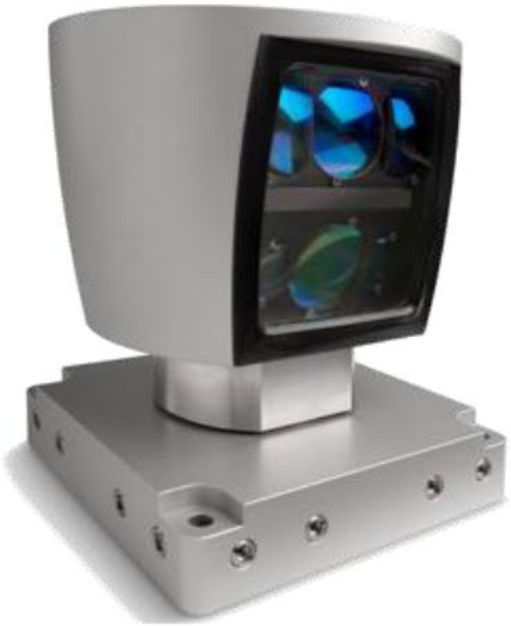
```
$ sudo apt-get install libv4l-dev libudev-dev ros-kinetic-rtabmap*
$ cd ~/catkin_ws/src/
$ svn export https://github.com/withrobot/oCam/trunk/Software/oCamS_ROS_Package/ocams
$ cd ~/catkin_ws/ && catkin_make
$ sudo gedit /etc/udev/rules.d/99-ttyacms.rules
ATTRS{idVendor}=="04b4" ATTRS{idProduct}=="00f9", MODE="0666", ENV{ID_MM_DEVICE_IGNORE}="1"
ATTRS{idVendor}=="04b4" ATTRS{idProduct}=="00f8", MODE="0666", ENV{ID_MM_DEVICE_IGNORE}="1"
$ sudo udevadm control --reload-rules
$ roslaunch ocams pointcloud.launch
```

(In case of oCam-Stereo)



https://github.com/withrobot/oCam/tree/master/Products/oCamS-1CGN-U

Sensor Package Practice #6 (LDS)



Sensor Package Practice #6 (LDS)

(In case of LDS)

```
$ cs
$ git clone https://github.com/ROBOTIS-GIT/hls_lfcd_lds_driver.git
$ cm
$ sudo chmod a+rw /dev/ttyUSB0
$ roslaunch hls_lfcd_lds_driver view_hlds_laser.launch
```

(In case of RPLiDAR)

```
$ cs
$ git clone https://github.com/robopeak/rplidar_ros.git
$ cm
$ sudo chmod a+rw /dev/ttyUSB0
$ roslaunch rplidar_ros rplidar.launch
```

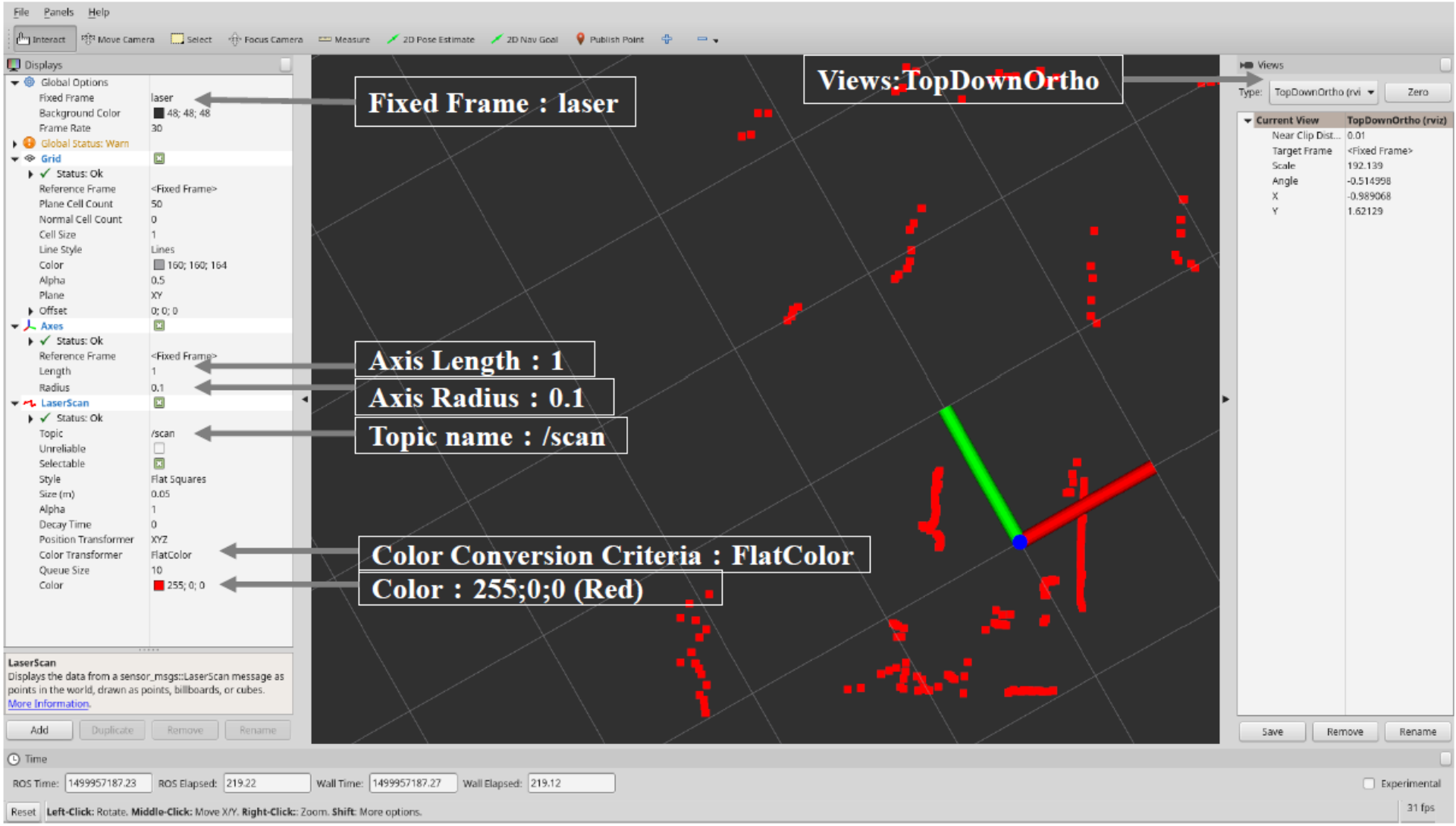
(In case of HOKUYO)

```
$ sudo apt-get install ros-kinetic-urg-node
$ sudo chmod a+rw /dev/ttyACM0
$ rosrun urg_node urg_node
```

* Change the display options of RViz

1) Change fixed frame: Global Options > Fixed Frame = laser
2) Add & configure Axes: Click 'Add' at the bottom left of rviz, then add Axes (Change 'Length' & 'Radius' is option)
3) Add & configure LaserScan: Click 'Add' at the bottom left of rviz, then add LaserScan
(Topic designation is required, 'Color Transformer', 'Color', etc. are options)

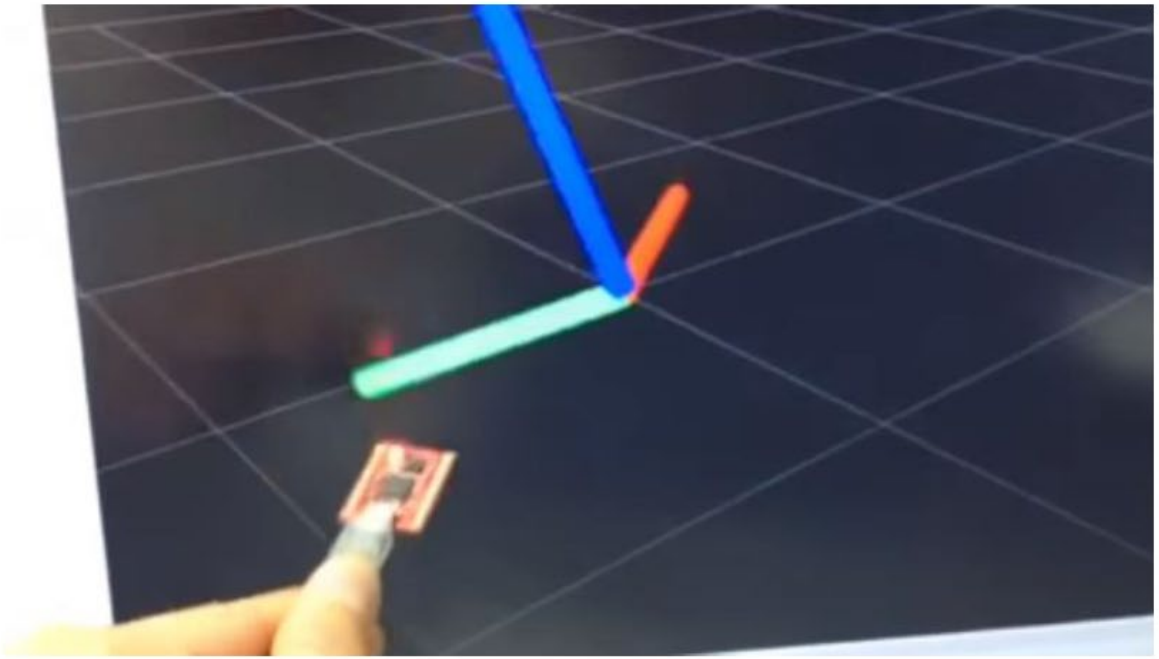
Sensor Package Practice #6 (LDS)



Sensor Package Practice #7 (IMU)

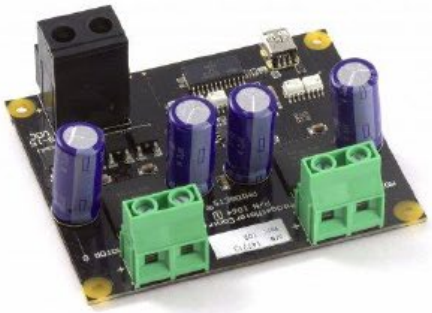
(In case of withrobot's myAHRS+)

```
$ cs
$ git clone https://github.com/robotpilot/myahrs_driver.git
$ cm
$ sudo chmod a+rw /dev/ttyACM0
$ roslaunch myahrs_driver myahrs_driver.launch
```



Motor Package

- PhidgetMotorControl HC
- Roboteq AX2550 Motor Controller
- ROBOTIS Dynamixel



ROS Enabled

ContorollingDynamixelwith ROS Package

- **DynamixelSDK** (http://wiki.ros.org/dynamixel_sdk)
  - Support 3 representative OS (Linux, Windows, MacOS)
  - Support programming language such as C, C++, C#, Python, Java, MATLAB, LabVIEW, etc.
  - Support ROS

DYNAMIXEL SDK

- **dynamixel_workbench** (http://wiki.ros.org/dynamixel_workbench)
  - Provide a variety of examples for ease of use in ROS
  - Provide GUI tool for ROS

SMPS2DYNAMIXEL

Dynamixel

Reference

✓ Download link

✓ Language:

English, Chinese, Japanese, Korean

"ROS Robot Programming"

A Handbook is written by TurtleBot3 Developers

Reference

- R. Siegwart, I. R. Nourbakhsh, D. Scaramuzza. Introduction to Autonomous Mobile Robots. MIT Press, 2nd Edition, 2011, ISBN-10: 0262015358.
- Y. Pyo, H. Cho, R. Jung, and T. Lim, ROS Robot Programming, ROBOTIS Co., Ltd., 2017, ISBN 979-11-962307-1-5
- J. O'Kane, A Gentle Introduction to ROS, CreateSpace Independent Publishing Platform, 2013, ISBN-13: 978-1492143239