ROS Tutorial

Jaeyong Sung
Feb 4th-5th 2013

ROS Filesystem

- Packages
  - libraries, tools, executable, etc.
- Manifest
  - description of packages
  - list dependencies
- Stacks
  - collection of packages
- Stack Manifest

Navigating ROS Filesystem

- roscd = ros + cd
- rosls = ros + ls
- rospack find [package]

  e.g.

  ```
  rospack find roscpp
  /opt/ros/fuerte/share/roscpp
  ```

- roscreate-pkg [name] [depend1] [depend2]
- rosmake [name]
- rosrun [package] [executable]

ROS Nodes

- Nodes
  - an executable that uses ROS to communicate with other nodes.
- Messages
  - ROS data type used when subscribing or publishing to a topic.
- Topics
  - Nodes can publish messages to a topic as well as subscribe to a topic to receive messages.
- Master
  - Name service for ROS (i.e. helps nodes find each other)
  - "roscore"

ROS Nodes

- rosnode list

```
/rosout
```

- rosnode info [node_name]

```
Node [/rosout]
Publications:
 * /rosout_agg [rosgraph_msgs/Log]

Subscriptions:
 * /rosout [unknown type]

Services:
 * /rosout/set_logger_level
 * /rosout/get_loggers
```

- rosnode ping [node_name]

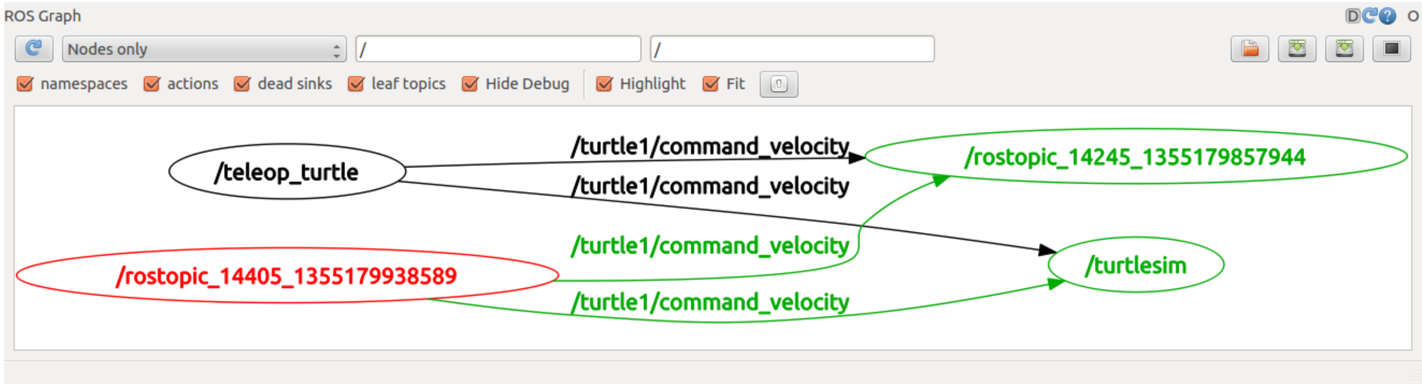
- rosrun [package_name] [executable]

ROS Topic

- rostopic echo [topic_name]

```
$ rostopic echo /turtle1/command_velocity
---
linear: 2.0
angular: 0.0
---
linear: 2.0
angular: 0.0
---
linear: 2.0
angular: 0.0
---
linear: 2.0
angular: 0.0
---
linear: 2.0
angular: 0.0
```

- rxgraph


ROS Graph
Nodes only
/
/
namespaces
actions
dead sinks
leaf topics
Hide Debug
Highlight
Fit
/teleop_turtle
/turtle1/command_velocity
/turtle1/command_velocity
/rostopic_14245_1355179857944
/turtle1/command_velocity
/rostopic_14405_1355179938589
/turtle1/command_velocity
/turtlesim

Etc.

- rosservice
- roslaunch
  - allows launching multiple nodes at once

ROS Tutorial

http://www.ros.org/wiki/ROS/Tutorials

Upson 317/319 Lab

- Door lock :
- Ubuntu 12.04
- ROS Fuerte/Groovy
    - Use Fuerte!

Lab

- Use Gazebo simulator to bring up PR2
- Use Keyboard to drive PR2 around
- Create your own package that allows same functionality
- Use Rviz 3D visualization tool to visualize sensor data from PR2

Lab Part 1

1. Open terminal (default: Ctrl + Alt + T)

2. Add this line to end of ~/.bashrc

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

3. Launch PR2

```
roslaunch pr2_gazebo pr2_table_object.launch
```

(If you just want PR2: roslaunch pr2_gazebo pr2_table_object.launch)

4. Launch PR2 Keyboard controller

```
roslaunch pr2_teleop_general pr2_teleop_general_keyboard.launch
```

5. Drive PR2 around

Lab Part 2

1. Create new directory
```
mkdir ~/rosws
```

2. Add it as ROS package. Add following ~/.bashrc
```
export ROS_PACKAGE_PATH=$ROS_PACKAGE_PATH:~/rosws
```

3. Verify it is corrected added
```
echo $ROS_PACKAGE_PATH
```

4. Create ROS package
```
roscreate-pkg drive_base_tutorial roscpp geometry_msgs
```

5. Verify ROS is able to navigate to your new package
```
roscd drive_base_tutorial
```

6. Create new source file

```
gedit src/drive_base.cpp
```

7. Copy and paste code from

   http://pastebin.com/4XVer5HR

8. Adding following line to CMakeLists.txt

```
rosbuild_add_executable(drive_base src/drive_base.cpp)
```

9. Compile your package

```
rosmake drive_base_tutorial
```

10. Let's run it!

```
rosrun drive_base_tutorial drive_base
```

Visualizing with Rviz

1. Make sure your PR2 is still running in PR2
2. Launch Rviz

```
rosrun rviz rviz
```

3. Change "**Fixed Frame**" to "**/base_footprint**"
4. Click "**Add**" -> "**RobotModel**". You should see PR2.
5. Click "**Add**" -> "**PointCloud2**".
   Click on "**Topic**" and select
   "**head_mount_kinect/ depth_registered/points**"

6. Try moving PR2 and its head around and observe point cloud changes