ROS Tutorial

Me133a

Joseph & Daniel

11/01/2017

Introduction to ROS

2D Turtle Simulation

3D Turtlebot Simulation

Real Turtlebot Demo

What is ROS

- "ROS is an open-source, meta-operating system for your robot"
  - open-source: all code is public. Most people share their code as to be used with ROS
  - meta-operating system: contains many of the components expected in an OS: hardware abstraction, low-level control, package management
- We can use C++ or Python
- We will cover the basics with some examples. Follow the tutorials to understand more

  http://wiki.ros.org/ROS/Tutorials
- Note: each version of ROS works with a different Ubuntu (the virtual machine has Ubuntu 14 and ROS Indigo)
  - ROS Indigo works with Ubuntu 14.04
  - ROS Kinect works with Ubuntu 16.04

ROS Terminology

- **Package**: a collection of software bundle together
- **Nodes:** a process
  - $ rosrun <packageName> <nodeName>
  - $ roscore
- **Topics:** labelled buses to exchange data between nodes
  - $ rostopic list
- **Messages:** data structures.
  - $ rostopic type <topic>
  - $ rosmsg show <messageType>
- **Launch file**: it can run several nodes at once with specific parameters
  - $ roslaunch <packageName> <launchFile>

ROS Tools

- Comes prepackaged with some useful stuff
  - $ rqt
- Debugging
  - $ rqt_graph   node-topic interaction
  - $ roswtf   general troubleshooter (v. useful)
- Visualization
  - $ rqt_plot   2D plot
  - $ rviz   3D plot

Why do we use ROS?

- We use ROS to
  - Interact between different programs (threads) running in parallel
  - Interact with robot hardware
  - Display data in real time
  - Record and replay sensor data
- Advantages of ROS
  - It is a easy way to share and use code from others
  - There are already many drivers and programs to use
  - It hides the complexity to use several computers talking to each other
  - We can use the speed of C++ in some parts and the flexibility of Python in other parts.
  - It is becoming the de-facto standard in for robotics in industry and academia, you should learn it!

Basic linux commands

- Open a new terminal (ctrl + alt + T)
- Navigate your filesystem using
  - $ cd ‘path’ where ‘path’ is the folder you want to go
  - $ cd .. to go back one folder
  - $ ls to display the contents in the current folder
  - $ ls -l adding the argument ‘-l’ gives more info
- Use key TAB to autocomplete results
- ‘~’ denotes the Home directory
  - $ cd ~/Documents

Introduction to ROS

2D Turtle Simulation

3D Turtlebot Simulation

Real Turtlebot Demo

ROS setup

- Open the virtual machine
- For each command open a new terminal (ctrl + alt + T)
- Start ROS core

```
$ roscore
```

2D Turtle Simulation

- ROS tutorial: http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics
- Start turtlesim node

```
$ rosrun turtlesim turtlesim_node
```

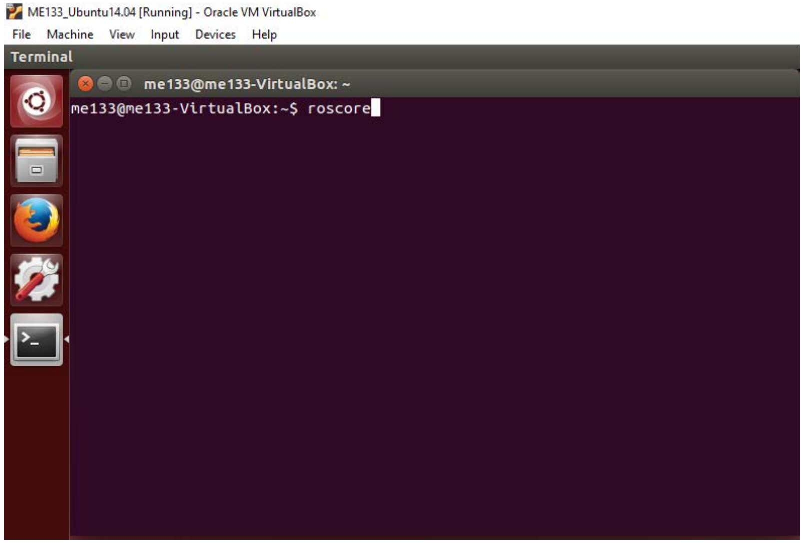
- Start keyboard teleoperation node

```
$ rosrun turtlesim turtle_teleop_key
```

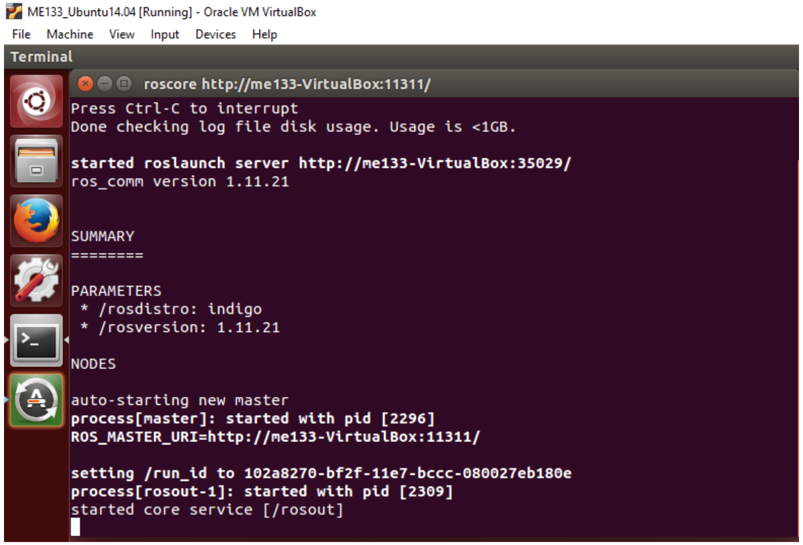
- Visualize the node graph

```
$ rqt_graph
```

2D Turtle Simulation


ME133_Ubuntu14.04 [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
Terminal
me133@me133-VirtualBox: ~

```
me133@me133-VirtualBox:~$ roscore
```

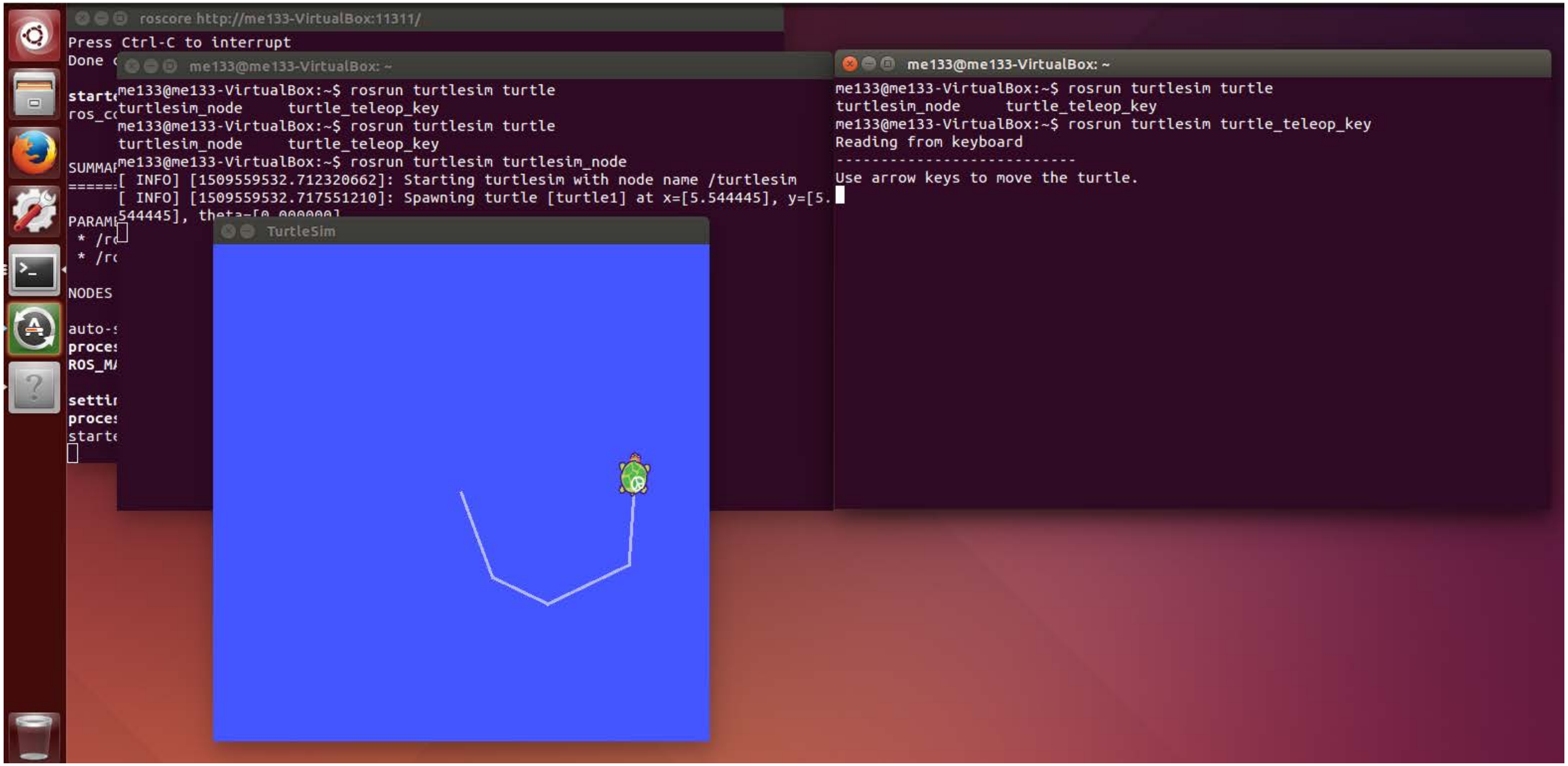

ME133_Ubuntu14.04 [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
Terminal
roscore http://me133-VirtualBox:11311/

```
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://me133-VirtualBox:35029/
ros_comm version 1.11.21


SUMMARY
========

PARAMETERS
 * /rosdistro: indigo
 * /rosversion: 1.11.21

NODES

auto-starting new master
process[master]: started with pid [2296]
ROS_MASTER_URI=http://me133-VirtualBox:11311/

setting /run_id to 102a8270-bf2f-11e7-bccc-080027eb180e
process[rosout-1]: started with pid [2309]
started core service [/rosout]
```

Caltech

2D Turtle Simulation

You need at least 3 terminal windows

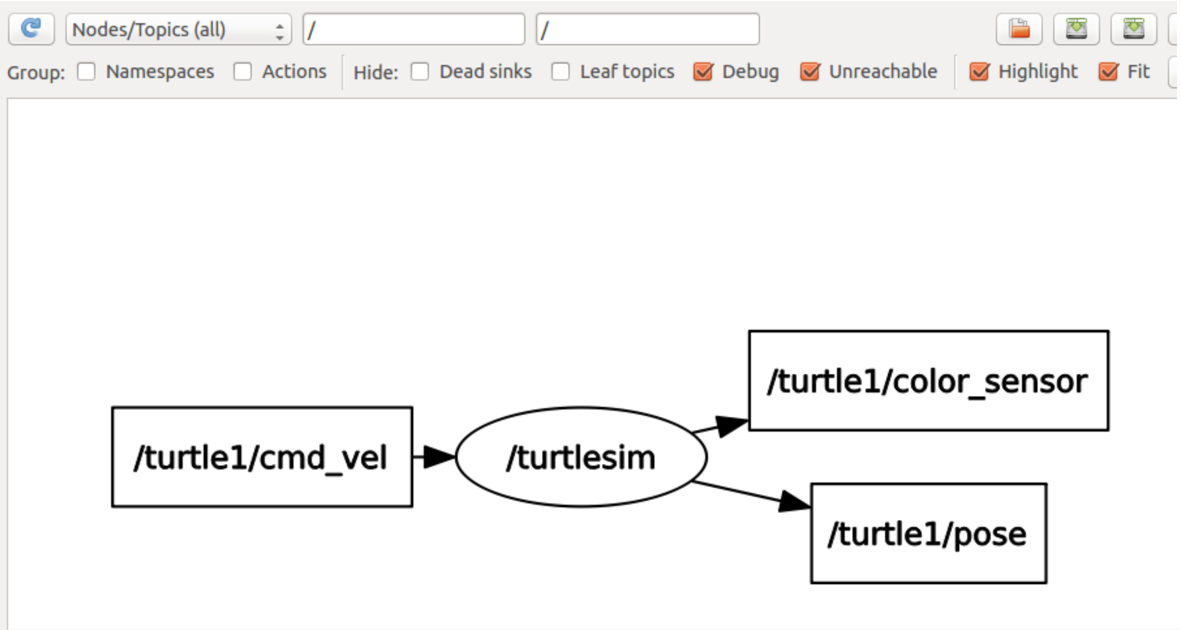

roscore http://me133-VirtualBox:11311/
me133@me133-VirtualBox: ~
TurtleSim
me133@me133-VirtualBox: ~

```
Press Ctrl-C to interrupt
Done c

start
ros_c

SUMMAR
=====

PARAM
 * /r
 * /r

NODES

auto-s
proces
ROS_MA

settin
proces
starte
```

```
me133@me133-VirtualBox:~$ rosrun turtlesim turtle
turtlesim_node     turtle_teleop_key
me133@me133-VirtualBox:~$ rosrun turtlesim turtle
turtlesim_node     turtle_teleop_key
me133@me133-VirtualBox:~$ rosrun turtlesim turtlesim_node
[ INFO] [1509559532.712320662]: Starting turtlesim with node name /turtlesim
[ INFO] [1509559532.717551210]: Spawning turtle [turtle1] at x=[5.544445], y=[5.
544445], theta=[0.000000]
```

```
me133@me133-VirtualBox:~$ rosrun turtlesim turtle
turtlesim_node     turtle_teleop_key
me133@me133-VirtualBox:~$ rosrun turtlesim turtle_teleop_key
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
```

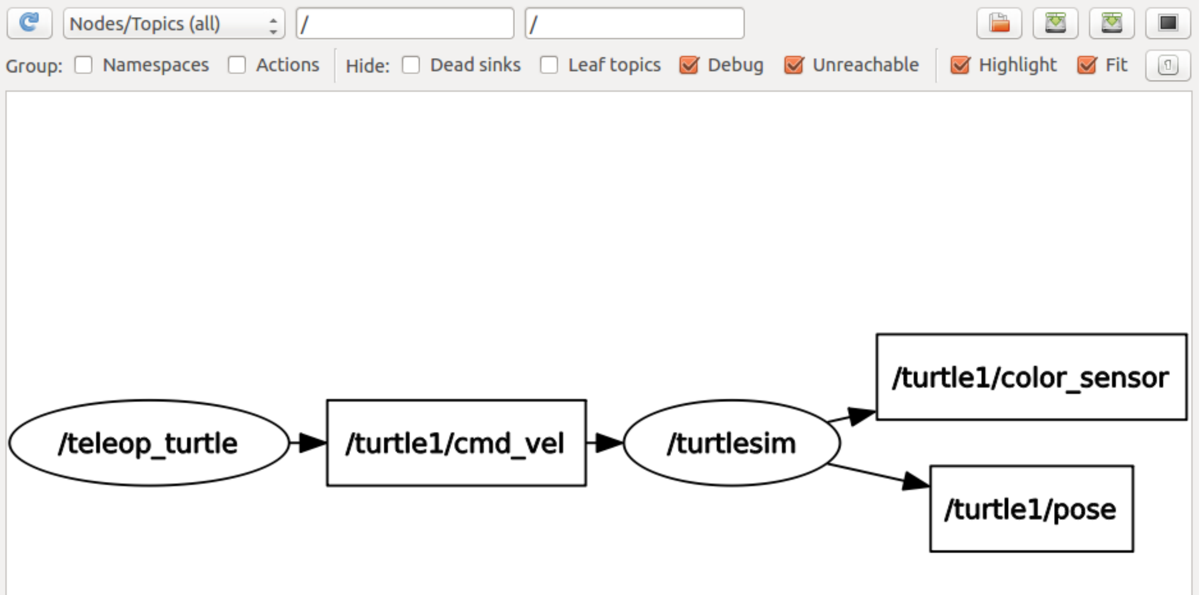
- After turtlesim_node



Nodes/Topics (all) / /

Group: □ Namespaces □ Actions Hide: □ Dead sinks □ Leaf topics ☑ Debug ☑ Unreachable ☑ Highlight ☑ Fit

- After turtlesim_node and turtle_teleop_key



Nodes/Topics (all) / /

Group: □ Namespaces □ Actions Hide: □ Dead sinks □ Leaf topics ☑ Debug ☑ Unreachable ☑ Highlight ☑ Fit

Introduction to ROS

2D Turtle Simulation

3D Turtlebot Simulation

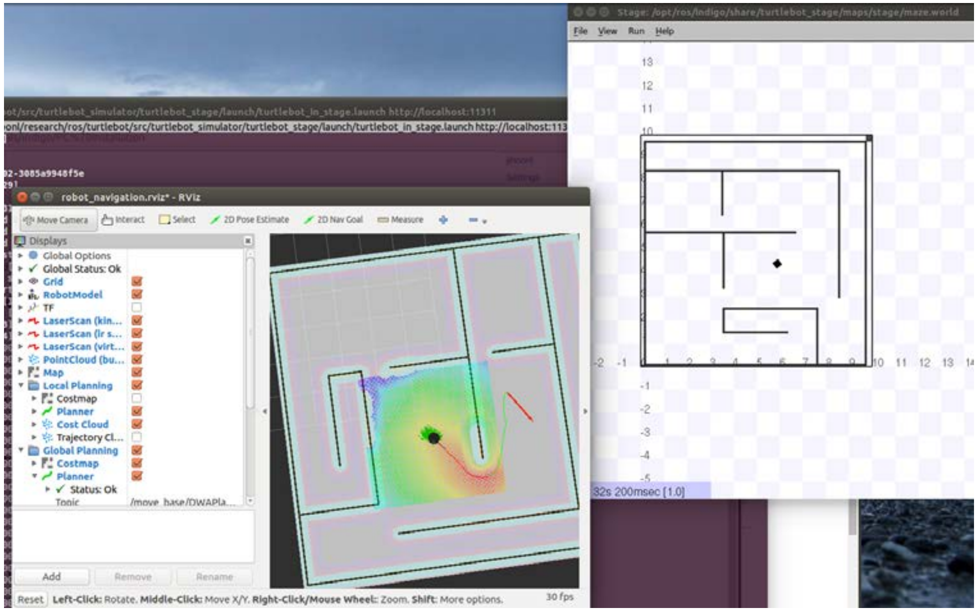
Real Turtlebot Demo

3D Turtlebot Simulation

- We have limited hardware and it can break: we will use the simulation to test our algorithms
- It includes dynamics, sensors and actuators models
- It uses Gazebo, a simulation environment built-in ROS

3D Turtlebot Simulation

- Close all previous terminals and execute
  - $ roslaunch turtlebot_stage turtlebot_in_stage.launch
  - $ roslaunch turtlebot_teleop keyboard_teleop.launch
  - $ rqt_graph



```
ot/src/turtlebot_simulator/turtlebot_stage/launch/turtlebot_in_stage.launch http://localhost:11311
onl/research/ros/turtlebot/src/turtlebot_simulator/turtlebot_stage/launch/turtlebot_in_stage.launch http://localhost:113
02-3085a9948f5e
291
```

robot_navigation.rviz* - RViz

Displays

- Global Options
- ✓ Global Status: Ok
- Grid ☑
- RobotModel ☑
- TF □
- LaserScan (kin... ☑
- LaserScan (ir s... ☑
- LaserScan (virt... ☑
- PointCloud (bu... ☑
- Map ☑
- Local Planning ☑
  - Costmap □
  - Planner ☑
  - Cost Cloud ☑
  - Trajectory Cl... □
- Global Planning ☑
  - Costmap ☑
  - Planner ☑
    - ✓ Status: Ok
    - Topic /move_base/DWAPla...

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:** Zoom. **Shift:** More options.

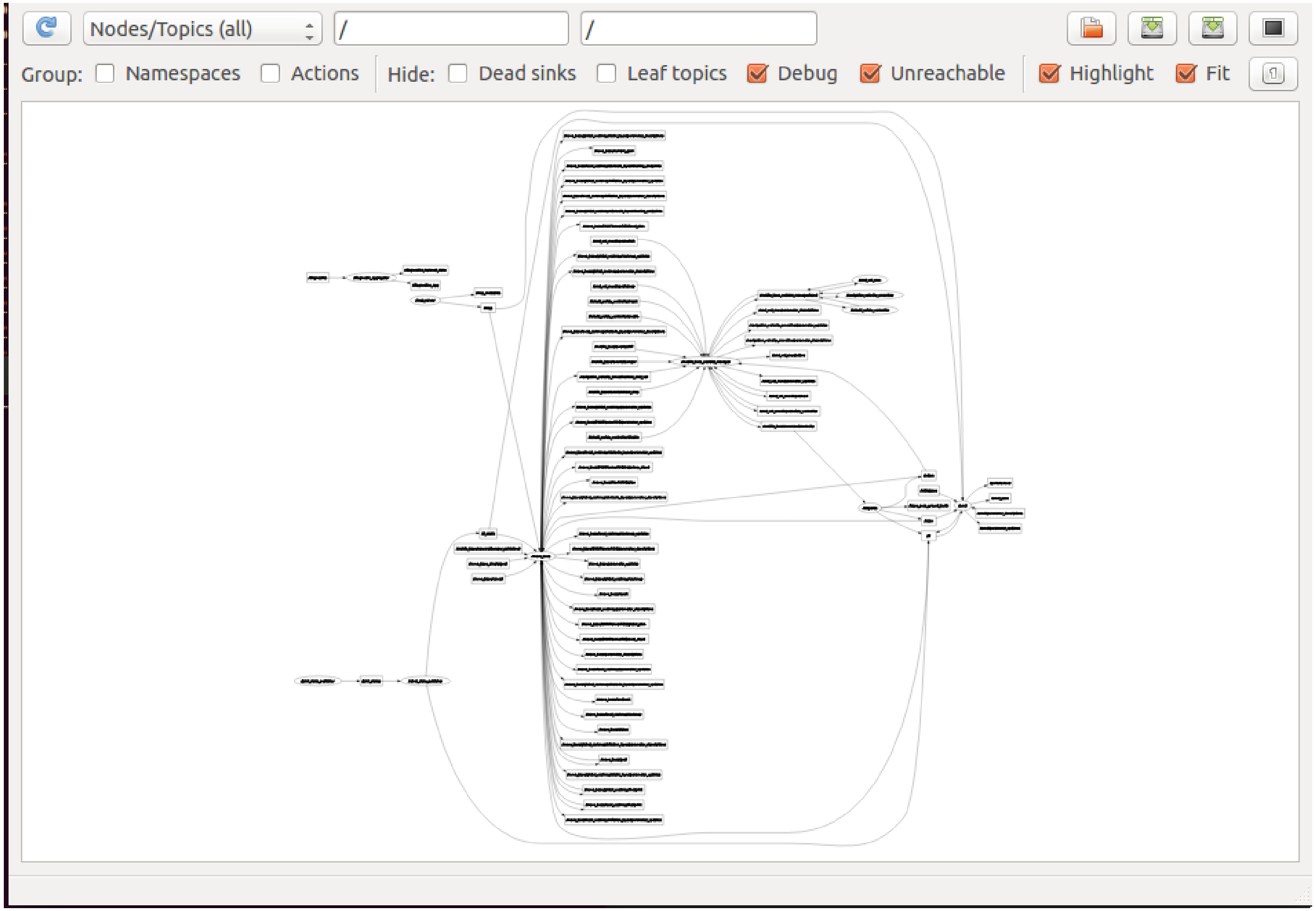
30 fps

Stage: /opt/ros/indigo/share/turtlebot_stage/maps/stage/maze.world

32s 200msec [1.0]

Caltech

3D Turtlebot Simulation Graph



Nodes/Topics (all) / /

Group: ☐ Namespaces ☐ Actions

Hide: ☐ Dead sinks ☐ Leaf topics ☑ Debug ☑ Unreachable

☑ Highlight ☑ Fit

3D Turtlebot Run a Script

- Open Firefox and go to the class webpage
- Download ‘Python script for lab 1’ and save it
- Cancel the turtlebot_teleop node (ctrl + C) and then run the commands:
  - $ cd Downloads
  - $ python me133a_lab1.py

- Open the file with the command:
  - $ gedit me133a_lab1.py

Introduction to ROS

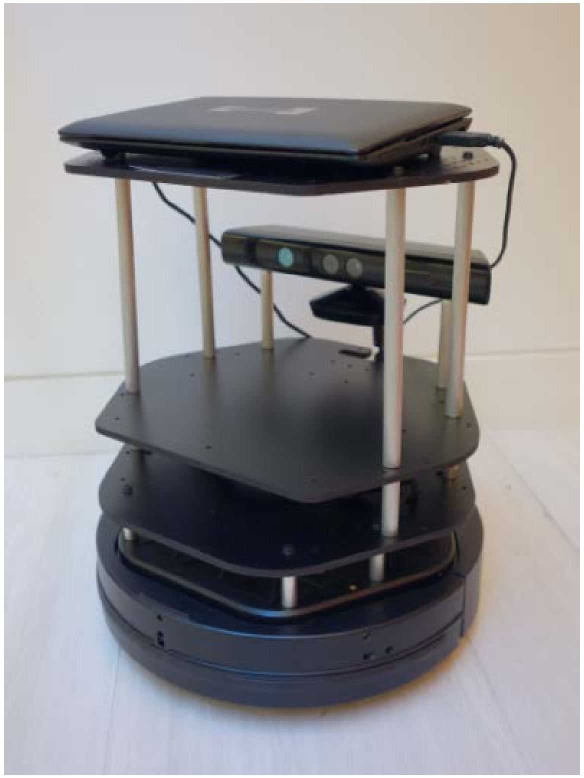
2D Turtle Simulation

3D Turtlebot Simulation

Real Turtlebot Demo

Turtlebot

- Kobuki Base: it has 2 motors with wheel encoders
- Sensors:
  - Gyroscopes
  - Wheel encoders
  - Hokugi 2D lidar
  - Kinect rgb-d camera



Real Turtlebot Demo

- We will use turtlebot for future labs

- To start it:
  - Turn on the turtlebot laptop and log in
  - Turn on the base
  - Run $ roslaunch turtlebot_bringup minimal.launch
  - Run whatever scripts you need

That is just the beginning..

- Things we haven’t covered:
  - How to record and play data using bags
  - How to write your own programs to publish and subscribe topics
  - How to create your own packages
  - How to create your own messages
  - Undestanding of ROS transforms (tf)
  - How to create your own rqt plugin

Extra: using Bag files

- Bag file: ROS format to store data
- Binary.
- Record:
  - $ rosbag record <newBagName.bag>   <topicsToRecord>
  - Use option “-a” to record all topics (warning: recording video takes a lot of space!)
- Analyze
  - $ rosbag info <existingBagName.bag>
- Play
  - $ rosbag play <existingBagName.bag>