What is ROS?

- It is an open source meta operating system
- It provides hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management.
- also provides tools and libraries for obtaining, building, writing, and running code across multiple computers.

Reference- http://wiki.ros.org/ROS/Introduction

How to install ROS?

- Follow the link

http://wiki.ros.org/kinetic/Installation/Ubuntu

- Note-

If you have multiple ROS distributions installed on the system, use following command to change the environment of your current shell

```
source /opt/ros/(distro)/setup.bash
```

Replace (distro) with the distribution you want to use in the shell ex. Lunar, indigo

Learning resources for ROS

- Follow the official tutorials

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

- Books

1) Learning ROS for Robotics Programming.
   by_Aaron.Martinez, Enrique Fernández
2) Programming Robots with ROS: A Practical Introduction to the Robot Operating System

Project 3 Overview

- Implementation of A* algorithm on a mobile robot in simulation/ real robot
- Consider the differential drive constrains
- Navigate turtlebot in the RRL lab’s virtual environment
- Practical implementation of a real robot (Optional)
- Note: If you are planning to implement on real robot then the team that you form should have one member who knows ROS well.

Turtlebot simulator setup

Installing turtlebot

- sudo apt-get update
- sudo apt-get install ros-kinetic-turtlebot ros-kinetic-turtlebot-apps ros-kinetic-turtlebot-interactions ros-kinetic-turtlebot-simulator ros-kinetic-kobuki-ftdi ros-kinetic-ar-track-alvar-msgs
- source /opt/ros/kinetic/setup.bash

Launching the simulator

To launch turtlebot_world

- roslaunch turtlebot_gazebo turtlebot_world.launch

What you need to do for Project 3?

- Generate velocities using A* algorithm
- Load the RRL lab’s environment
- Make a script which reads velocity output of your A* implementation and publish those velocities on /cmd_vel/input/navi topic

Connecting to a real turtlebot

- Follow the link

http://wiki.ros.org/Robots/TurtleBot/Network%20Setup

Project 4 Overview

- Pick and place of an object avoiding the obstacles on the table using Baxter in simulated/ real environment
- May use Gazebo/ Vrep for simulation
- Practical implementation is optional and must be done through ROS

Baxter simulator setup

Make a catkin workspace for baxter packages

- mkdir -p ros_ws/src
- cd ~/ros_ws
- catkin_make

Baxter simulator setup

- Follow the link

  http://sdk.rethinkrobotics.com/wiki/Simulator_Installation
- make change in baxter.sh file

  go to ~/ros_ws/src/baxter and open baxter.sh in any editor

  edit line 30, insert “kinetic” in the ROS version
- Note- The official site recommend ROS indigo for the simulator installation, but you can install it on kinetic. For the implementation on real robot ROS indigo will be used.

Moveit

- MoveIt is a ROS framework that allows you to perform motion planning with a specific robot.
- It uses OMPL (Open Motion Planning Library)
- OMPL includes the implementation of various randomized motion planners.

Moveit setup

Execute following command

- sudo apt-get install ros-kinetic-moveit

To install other packages for moveit-Rviz interface

- cd ~/ros_ws/src
- git clone https://github.com/ros-planning/moveit_robots.git
- cd ..
- catkin_make

Launching the simulator for Baxter

Launching baxter simulator

- cd ~/ros_ws
- ./baxter.sh sim
- roslaunch baxter_gazebo baxter_world.launch

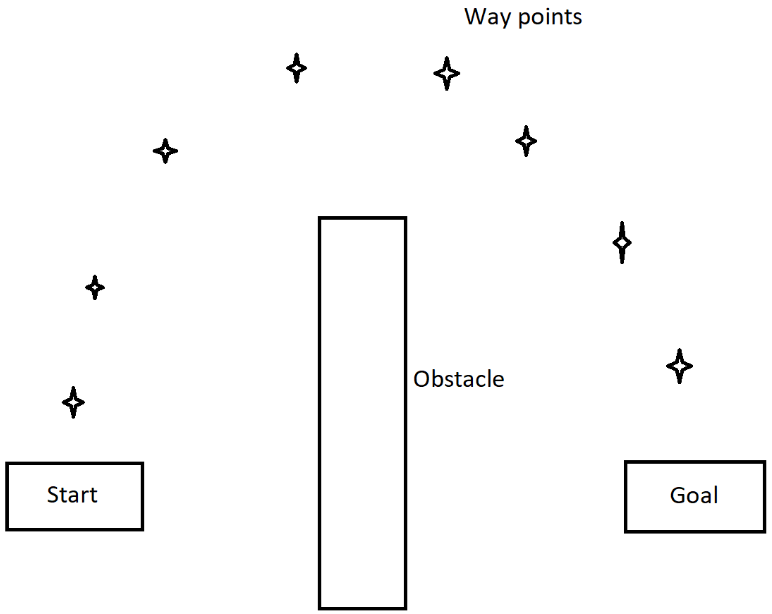
Executing demo pick and place node

- Open a new terminal
- cd ~/ros_ws
- source devel/setup.bash
- rosrun baxter_sim_examples ik_pick_and_place_demo.py

What you need to do for Project 4?

- Edit ik_pick_and_place_demo.py script available in ~/ros_ws/src/baxter_simulator/baxter_sim_examples
- Add way points for the path that you defined
- How will you get way points?
- You will be using Moveit-Rviz interface to get the waypoints
- Rviz is a visualization tool that comes with ROS. It helps you to use Moveit graphically.

What you need to do for Project 4?



Launching Moveit-Rviz

Before launching Moveit-Rviz make sure following things are running

- Baxter simulator

Robot is enabled, it can be done using following command (Run in a new terminal)

- Open a new terminal
- cd ros_ws
- source devel/setup.bash
- rosrun baxter_tools enable_robot.py -e

Launch Moveit-Rviz (Continue..)

Joint trajectory controller is running, use following command (execute in a new terminal window)

- cd ros_ws
- source devel/setup.bash
- rosrun baxter_interface joint_trajectory_action_server.py

Execute following command to launch Moveit-Rviz

- cd ros_ws
- source devel/setup.bash
- roslaunch baxter_moveit_config demo_baxter.launch right_electric_gripper:=true left_electric_gripper:=true
- Note- Path for the obstacle file ros_ws/src/moveit_robots/baxter/baxter_moveit_config/baxter_scenes

Practical implementation on Baxter

- Go through the tutorial on the following link before coming to the RRL lab for the practical implementation

http://sdk.rethinkrobotics.com/wiki/Hello_Baxter

Thank You