# Lab 11: Navigation

By Jacky Chen, last modified on 2020.05.17

The objective of this tutorial is to introduce the basic navigation tools currently available on PyRobot.

## Hardware and Software Setup

laptop $ cd ~/sis\_lab\_all\_2020 && git checkout devel-[your\_student\_id]

laptop $ git pull origin master

laptop $ cd ~/sis\_lab\_all\_2020/11-Navigation

[**Lab 11: Navigation**](#_74gram21yqax) **1**

[Hardware and Software Setup](#_chc57ow4x0s1) 1

[Overview](#_x5puhc1p2jjo) 1

[Topics and Activities](#_g0fw0bgwxhdt) 2

[Topic/Activity 1 Base State and Frames of reference](#_aadm982kyolo) 2

[Topic 1.1 Base State](#_rl15slrm10vt) 2

[Topic 1.2 Frames of reference](#_9xf9nsimldmi) 4

[Topic/Activity 2 Basic Control](#_sf5mhl8ztpdm) 4

[Topic 2.2 Velocity control](#_v4sjn3hb2lab) 4

[Topic 2.2 Position control](#_dlktwhuxe8nr) 5

[Reference](#_uwwglharlfb) 6

## Overview

Estimated Time to Finish: 1.5 hours

After completing this tutorial you should

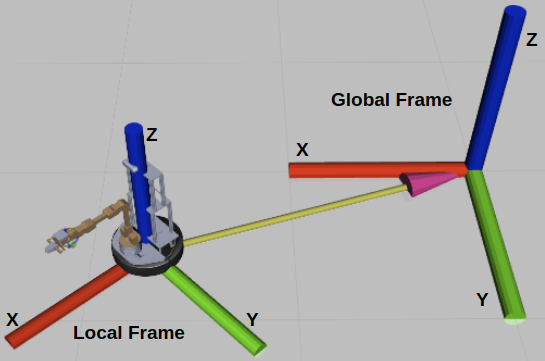
* run through the basic navigation tools currently available on PyRobot.
* control [LoCoBot](http://locobot.org/) by using PyRobot

## Topics and Activities

### Topic/Activity 1 Base State and Frames of reference

Base state is represented by [x, y, yaw], the x-coordinate, y-coordinate of the base\_link and the robot heading yaw. By default this state is estimated via intertial sensors, or wheel encoders.

As shown in the figure below, there are two frames of reference available for the base - Local and Global frames.



Local frame is the frame of reference attached to the base of the robot and moves with it as the robot moves i.e, all the points in this frame are relative to the robot.

Global frame is a stationary frame of reference. It is the initial frame that the robot started at.

#### **Topic 1.1 Base State**

(Estimated Time to Finish: 10 minutes)

**(Running on real robot)**

**In locobot open a terminal, type**

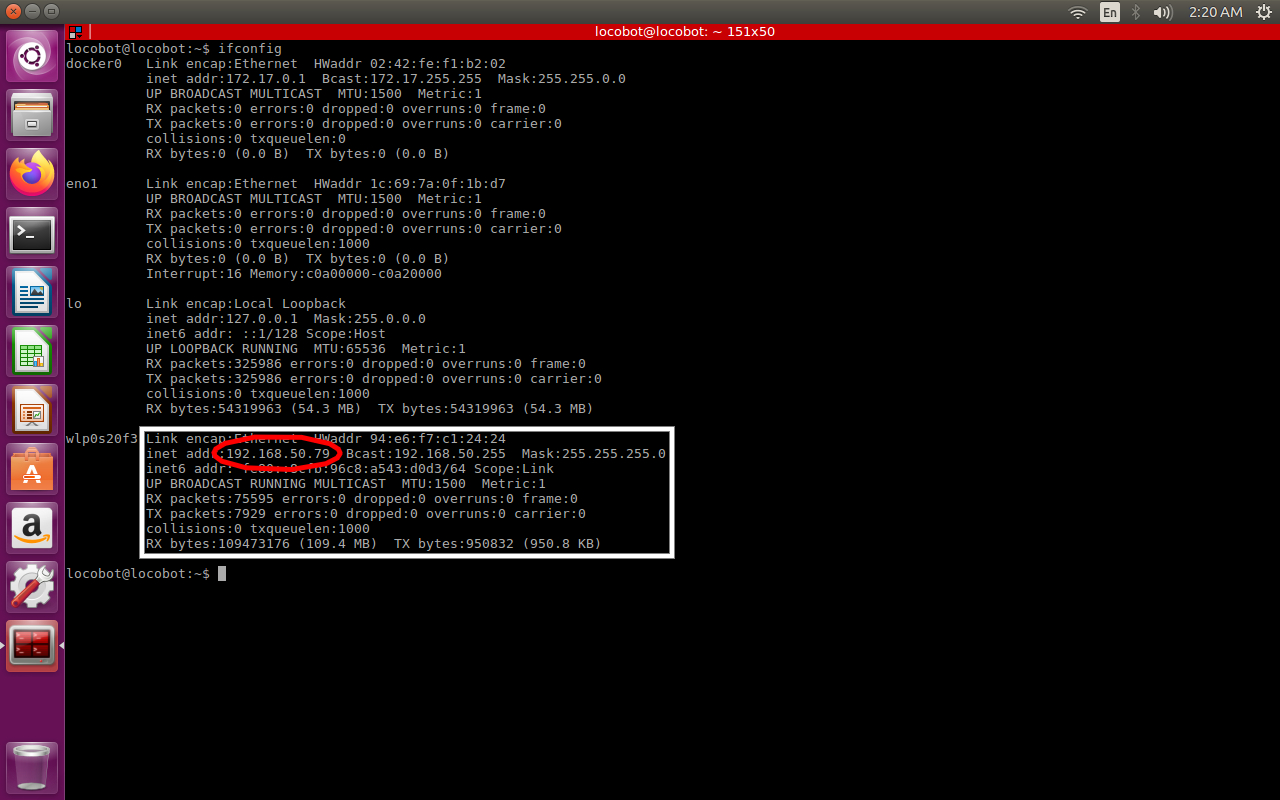
locobot $ **sudo chmod 777 /var/run/docker.sock**

**(password for locobot is “locobot”)**

**Next check locobot’s address**

locobot $ **ifconfig**

**(For example)**

****

locobot $ **cd sis\_lab\_all\_2020/11-Navigation/**

locobot $ **source docker\_run.sh cuda9**

**Setup ROS environment variables**

locobot\_docker $ **export ROS\_MASTER\_URI=http://{locobot’s address}:11311/**

locobot\_docker $ **export ROS\_IP={locobot’s address}**

**(For example)**

locobot\_docker $ **export ROS\_MASTER\_URI=http://192.168.50.79:11311/**

locobot\_docker $ **export ROS\_IP=192.168.50.79**

locobot\_docker $ **roslaunch locobot\_control main.launch use\_base:=true use\_rviz:=false**

**Open another terminal, type**

locobot $ **source docker\_run.sh same**

locobot\_docker $ **source ~/pyenv\_pyrobot\_python2/bin/activate**

locobot\_docker $ **cd 11-Navigation/script/**

locobot\_docker $ **python base\_state.py**

#### **Topic 1.2 Frames of reference**

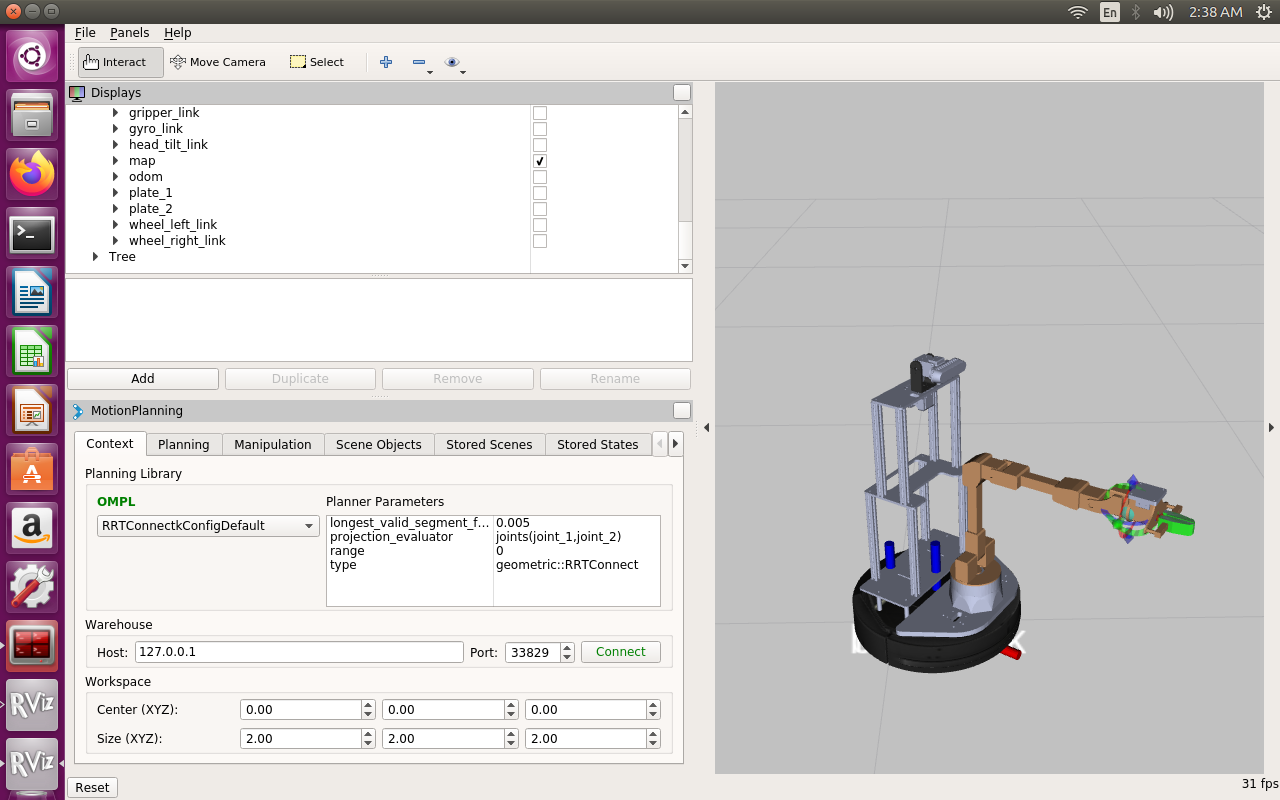
(Estimated Time to Finish: 10 minutes)

**(Running on real robot)**

**In the first terminal, type**

locobot\_docker $ **roslaunch locobot\_control main.launch use\_base:=true**

**Push the locobot, you can see the locobot moving in Rviz.**

****

### Topic/Activity 2 Basic Control

In this section, we will talk about two modes of controlling the LoCoBot base- Velocity control and Position control. One should note the same tutorials also apply to LoCoBot-Lite with very minimal change to the code.

#### **Topic 2.2 Velocity control**

(Estimated Time to Finish: 20 minutes)

This mode of control allows us to command the base with a particular linear and angular velocity for a specified amount of time.

**(Running on real robot)**

**In the first terminal, type**

locobot\_docker $ **roslaunch locobot\_control main.launch use\_base:=true use\_rviz:=false**

**Open another terminal, type**

locobot $ **source docker\_run.sh same**

**Setup ROS environment variables**

locobot\_docker $ **export ROS\_MASTER\_URI=http://{locobot’s address}:11311/**

locobot\_docker $ **export ROS\_IP={locobot’s address}**

locobot\_docker $ **source ~/pyenv\_pyrobot\_python2/bin/activate**

locobot\_docker $ **cd 11-Navigation/script/**

locobot\_docker $ **python velocity\_control.py # for moving**

#### **Topic 2.2 Position control**

(Estimated Time to Finish: 20 minutes)

This mode of control allows us to command the base to go to a specified target (of the form [x, y, yaw]) in the environment.

We currently support three different base-controllers for position control: ILQR, Proportional and Movebase.

**(Running on real robot)**

**In the first terminal, type**

locobot\_docker $ **roslaunch locobot\_control main.launch use\_base:=true use\_rviz:=false**

**Open another terminal, type**

locobot $ **source docker\_run.sh same**

**Setup ROS environment variables**

locobot\_docker $ **export ROS\_MASTER\_URI=http://{locobot’s address}:11311/**

locobot\_docker $ **export ROS\_IP={locobot’s address}**

locobot\_docker $ **source ~/pyenv\_pyrobot\_python2/bin/activate**

locobot\_docker $ **cd 11-Navigation/script/**

locobot\_docker $ **python position\_control.py**

**Disscussion:**

1. **When you run Topic 1.1,you can see the base state of the locobot. If the terminal display the result of the base state is [0.5, 0.3, 0.4]. What are 0.5, 0.3, 0.4 mean?**
2. **After running Topic 1, can you tell us what method is the most likely to do odometry ?**

## Reference

Pyrobot official tutorial: <https://pyrobot.org/>