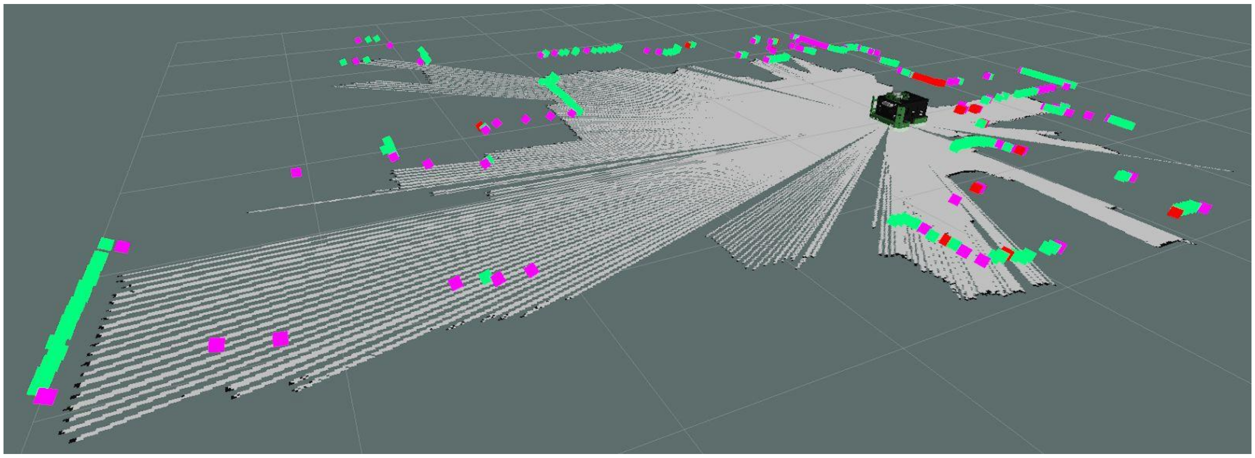
Contents

Lab 3 – Mapping

This lab will investigate 2D localization and mapping in a physical environment. The map will be built by driving the MacBot around and collecting data about the local environment using a laser scanner. Configuration parameters will be set in the launch file.



SLAM

SLAM – Simultaneous Localization and Mapping

SLAM is the process of collecting data about the robots' position in an unknown environment using sensors and leveraging that data to make path planning decisions.

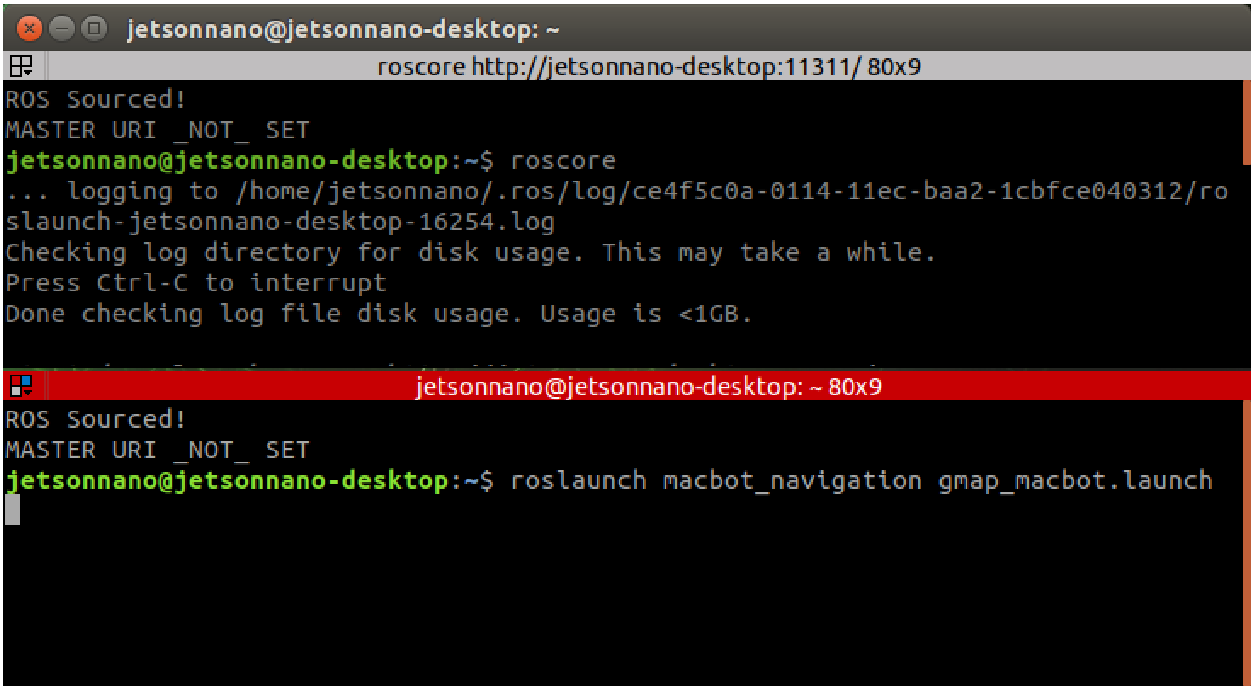
For example, say that a person wishes to walk from the Engineering Technology Building (ETB) to the Dave Bradley Athletic Centre (DBAC) but has no prerequisite knowledge of the campus environment. That person is likely to use a mapping tool like google maps. In this case, the **navigation goal** would be entered as DBAC. Google cloud would then use an **algorithm** to perform **path planning** in order to find the shortest route. In such a case, Google would be getting feedback through the global positioning system (GPS) on the persons smartphone. This data is used in the **localization** of the persons current position relative to its prebuilt map database.

For a robot to position itself in an environment, it needs the ability to detect and recognize recurring landmarks. This is done with sensors. In the example above, GPS and eyesight would have been used. In robotics, since GPS is not reliable in pinpointing location in some environments like indoors or near large buildings, this is typically achieved using LiDAR or camera technology. Odometry data is the estimate of the robot's position and velocity in free space. For humans, this might be trivial, but robots require complex algorithms to produce and reason with odometry data.

Mapping Launch Files

Please launch the following:

roslaunch macbot_navigation gmap_macbot.launch



The ydlidar node is already included in the launch file therefore it is unnecessary to launch the macbot_sensors LiDAR node.

In the gmap_macbot.launch file, the laser_scan_matcher node must also be launched.

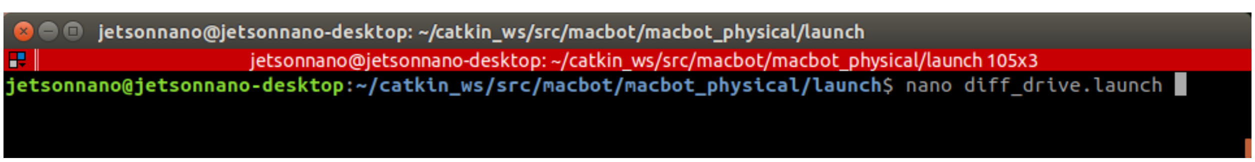
There are parameters for the gmapping node that can be used to tune the mapping node to represent the local environment more accurately. An exercise is to practice tuning these parameters to achieve the best map possible.

Next, please include the following nodes to the diff_drive launch file:

<node pkg = "macbot_physical" name = "macbot_node" type = "macbot_node.py"/>

<node pkg = "macbot_physical" name = "tf_broadcaster" type = "tf_broadcaster.py"/>

<node pkg = "teleop_twist_keyboard" type = "teleop_twist_keyboard.py" name = "teleop_twist_keyboard" output = "screen"/>



```
jetsonnano@jetsonnano-desktop: ~/catkin_ws/src/macbot/macbot_physical/launch
jetsonnano@jetsonnano-desktop: ~/catkin_ws/src/macbot/macbot_physical/launch 105x39
  GNU nano 2.9.3                              diff_drive.launch

<?xml version="1.0" ?>
<!-- Author : Eech Hsiao -->
<!-- hsiaoy4@mcmaster.ca -->

<launch>

    <!-- Loading in URDF.XACRO file -->
    <param name = "robot_description" command = "$(find xacro)/xacro --inorder '$(find macbot_descriptio$

    <!-- We need robot_state_publisher to publish static transforms of the robot from URDF -->
    <node name = "robot_state_publisher" pkg = "robot_state_publisher" type = "robot_state_publisher"/>

    <!-- Launching our own macbot_node.py -->
    <node pkg = "macbot_physical" name = "macbot_node" type = "macbot_node.py"/>

    <!-- Launching our own tf_broadcaster.py -->
    <node pkg = "macbot_physical" name = "tf_broadcaster" type = "tf_broadcaster.py"/>

    <!-- Differential Drive Controller -->
    <arg name = "ticks_per_meter"       value = "5593" />
    <arg name = "wheel_separation"      value = "0.275" />
    <node name = "controller" pkg = "diff_drive" type = "diff_drive_controller" output = "screen">
        <rosparam subst_value = "true">
            ticks_per_meter: $(arg ticks_per_meter)
            wheel_separation: $(arg wheel_separation)
            max_motor_speed: 60
            timeout: 0.1
            rate: 5
        </rosparam>
    </node>

    <!-- teleop_twist_keyboard drive with keyboard for cmd_vel -->
    <node pkg = "teleop_twist_keyboard" type = "teleop_twist_keyboard.py" name = "teleop_twist_keyboard"$

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     M-U Undo
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line  M-E Redo
```

This allows for the launching of both the macbot_node and tf_broadcaster without needing to run additional launch commands. The teleop_twist_keyboard allows for the remote control of the macbot.

Next, please launch **diff_drive.launch**.

```
jetsonnano@jetsonnano-desktop: ~/catkin_ws/src/macbot/macbot_physical/launch
jetsonnano@jetsonnano-desktop: ~/catkin_ws/src/macbot/macbot_physical/launch 105x19
jetsonnano@jetsonnano-desktop:~/catkin_ws/src/macbot/macbot_physical/launch$ clear
jetsonnano@jetsonnano-desktop:~/catkin_ws/src/macbot/macbot_physical/launch$ roslaunch diff_drive.launch
```