Working with Robot Operating System (ROS)

**Installation**

Every year in May a new version of ROS is released, and on even numbered years, an LTS version is released, which is supported for five years. The only operating system *supporting* the recent releases of ROS is Ubuntu, while other operating systems are considered *experimental*. It is important to cross-check the version of ROS one is interested in using with the version of Ubuntu installed.

At the time that this project took place, ROS Indigo Igloo was the most recent stable LTS distribution. Indigo is primarily targeted to work best with the Ubuntu 14.04 LTS release. As a result, all findings discussed in this report were conducted using Indigo and Ubuntu 14.04, installed using VirtualBox on a MacBook Pro computer. Initially it was attempted to install Ubuntu 14.04 and Indigo on a remote server to allow others easy access to the system. X2Go was installed to display the GUI for the system. However, the GUI for Ubuntu and Indigo was too slow for reasonable functioning on the remote server, and thus the individual MacBook Pro was utilized instead.

**Basics of ROS**

ROS is made up of nodes, or executable files within a ROS package that uses ROS to communicate with other nodes. Additionally, ROS offers its own middleware, which provides facilities such as publishing (sending) and subscribing (receiving) data. The first fifteen Beginner Level tutorials found at the following website were performed and were found beneficial in understanding the basics of ROS: <http://wiki.ros.org/ROS/Tutorials>.

Using the following tutorial, a basic code for controlling the motion of a Turtlebot was developed: <http://web.engr.oregonstate.edu/~smartw/rob514/programming/programming1/index1.html>. The Turtlebot’s motion can either be controlled by running code (either in Python or C++), or using the keyboard after launching the necessary file (roslaunch turtlebot\_teleop keyboard\_teleop.launch).

**Customizing Maps**

An important task within ROS is creating a customized map that the robot can traverse. In order to learn how to do this, the following tutorial was used: <http://wiki.ros.org/turtlebot_gazebo/Tutorials/indigo/Make%20a%20map%20and%20navigate%20with%20it>. Errors may occur during this tutorial if software for more than one robot is installed on the computer. For example, the amcl launch file is included for both the Jackal robot and the Turtlebot. Thus, when running the customized map, it is important to ensure that all launch files are for the same robot.

The tutorial mentioned is designed for use with Gazebo. Another tutorial exists to make a map using Stage instead of Gazebo, but errors were found in this tutorial, and at this time Gazebo appears functionally and graphically superior to Stage.

After designing a customized map, as directed in the tutorial, the process for running the robot in the map has multiple steps. First, the user must launch the robot within its world, and this is done using Gazebo. Next, the user must launch the customized map file using amcl\_demo, as indicated in the above mentioned tutorial. Following this the user must launch Rviz, a 3D visualization tool for ROS. At this point, the user should see their customized map open up in Rviz. The user can then either launch additional code to initiate motion of the robot or launch the necessary file to use the keyboard for motion.

**Looking Forward**

The next step in this ROS project is to develop a module to simulate the infrared sensor for the mobile thermostat.