**Using the System (04/25/19)**

The following guide is representative of the system as of the date above. Method of start is likely to change. ALL OF THESE COMMANDS MUST BE RUN IN A LINUX TERMINAL.

The robot is run on ROS, Robotic Operating System. ROS supplies a modular system that does numerous things, but primarily allows different ‘nodes’ to operate on different computers, publishing and subscribing to different topics. In our case, that is two Raspberry Pis. The Pi’s are named *d2.local* (192.168.1.43) and *r2.local* (192.168.1.42.). While it is possible to connect to these addresses via VNC viewer, it is recommended to use the *ssh* command like shown below. You will need two terminal windows for accessing each Pi. (Note: Don’t use the local addresses for connecting to the Pis, use the hostnames *d2.local* and *r2.local*)

*ssh ubuntu@r2.local*

This will allow you to access the terminal for the system without bogging it down, as it is typically sensitive to graphical programs being run on it. Note that both Pis must be connected to the same network to use them independently, and AU\_WiFi does not work for that. Using a Wi-Fi hotspot is recommended. If only one is able to be connected, ssh into the other via the first one over its local connection (ethernet).

Now that you are connected, you can begin to run the program. All of the ROS launch files are started via shell files, located in *ieee-2019-electrical-software/launches* on both Pis*.*

*cd ~/ieee-2019-electrical-software/launches*

* To launch the required components for the ROS navigation stack, on d2.local, launch *./ftc.sh*.
* To launch AMCL, the move\_base node, and the main algorithm node, launch *./amcl.sh* on r2.local.

This will boot up the system and begin running whatever is in main\_node. If you want to stop the system, it’s recommended to use the *kill.py* file also located in the */launches* file. This way starting new nodes will not overwrite each other and create undesirable bugs.

With the system running, there are several tools ROS provides to allow you to see exactly what is going on in the system. It is recommended to run these on a separate computer or Pi with ROS installed that is connected to the network that the robot is on. This alleviates more computing power from the Pis. To ensure that your ROS computer is getting data from the robot, run the following commands:

*cd ~/ieee-2019-electrical-software/*

*source ros.sh*

* *rviz* – A visual GUI showing the map, current path, current location, and other useful information
  + By default, if the system is connected properly, RViz will boot up with the map server already loaded in from the roscore you connected to with *source ros.sh*. Allowing you to see what map is currently loaded in
  + To see other topics not listed by default, on the left pane there is a button that says “Add”, which will allow you to add topics to your view panel. In that window, the “By topic” tab will give you the current topics rviz can see that you can subscribe to. (*ie. amcl\_pose,* current robot position)
* *rqt\_graph*
  + Application gives a visual representation of all the nodes the roscore can see and what topics they are publishing and subscribing to.
* *rostopic echo topic*
  + A simple terminal output of the topic you choose
* *rosnode list node*
  + Lists the nodes currently running under the roscore

There are numerous commands that ROS also includes that are a bit too in depth to list here, however more information and tutorials on ROS can be found at <http://wiki.ros.org/>.