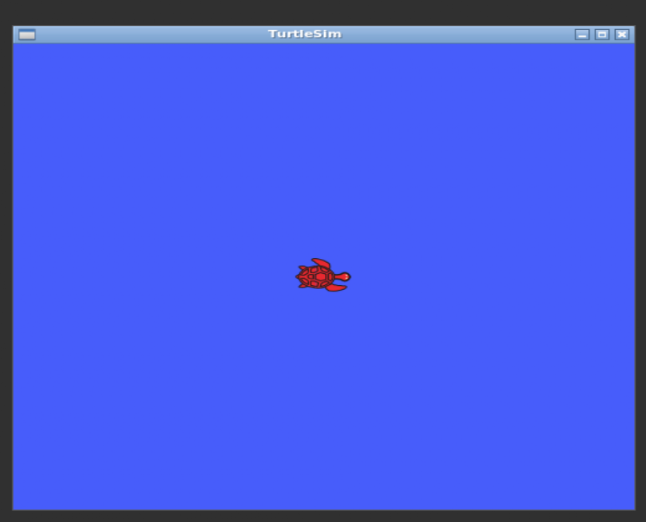
The following document serves as a fina; progress report for the independent study using the Turtlebot3 and ROS. At the time of this report’s creation it is currently the last weeks and I have just presented what I have working.Using the requirements within the Turtlebot3 Independent Study proposal, I will detail how far I have gotten with the project.

Week 1:

Turtlebot Sim runs-

The turtlebot sim is a small simulated version of the turtlebot. To run the turtlebot sim roscore for the core functionality must be run followed by the rosrun command calling the turtlesim node. Attached is a snip from ros.

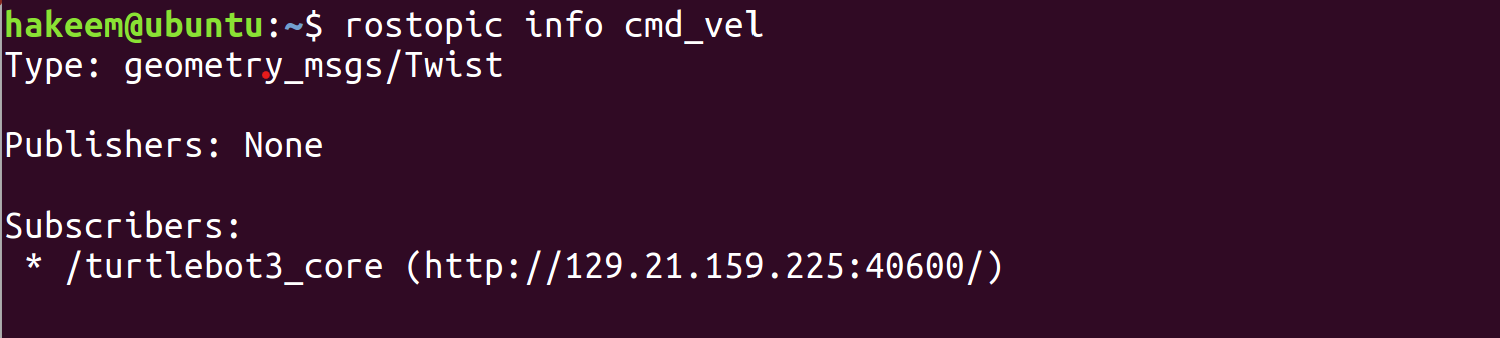


This GUI can be used to test the various code and functions of ROS. Using the teleoperation node command, the small turtle can be moved around and manipulated. Turtlebot sim works as a barebones simulation tool.

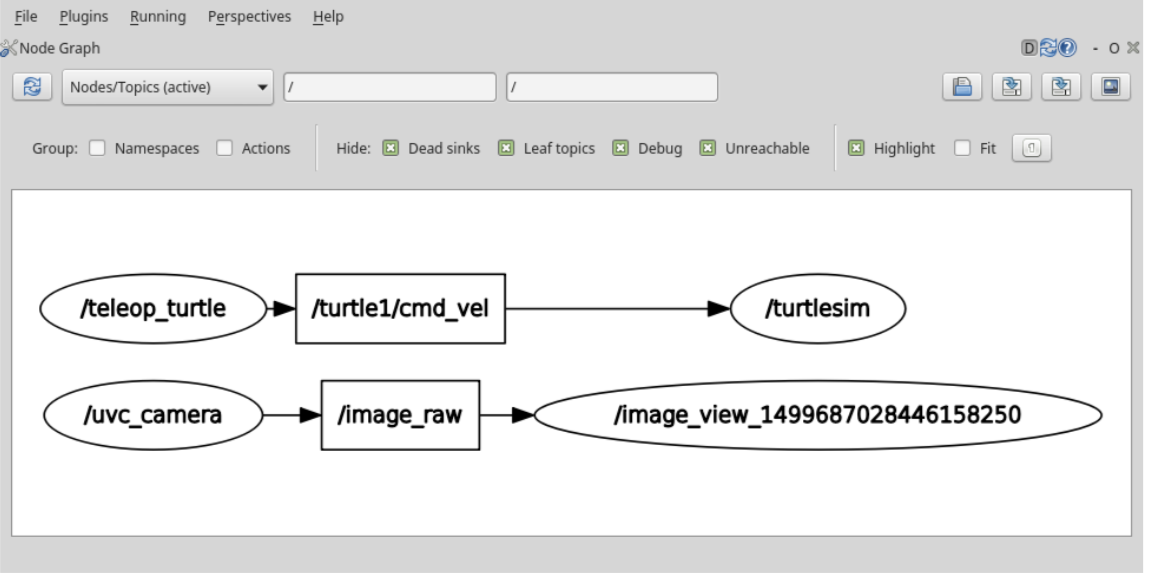
Week 3-4 Demonstrate ability to display key variables and set parameters on computer (grade item) :

There are a variety of ways to display key variables, but first the term “key variables” should be well defined.

In terms of ROS “topics” and “nodes” would be the closest to the definition of key variables. Topics can be compared to pipelines while nodes use topics to publish information for other nodes to communicate. Using the rostopic info command, we can find out what topics are available as well as what nodes are publishing this information.



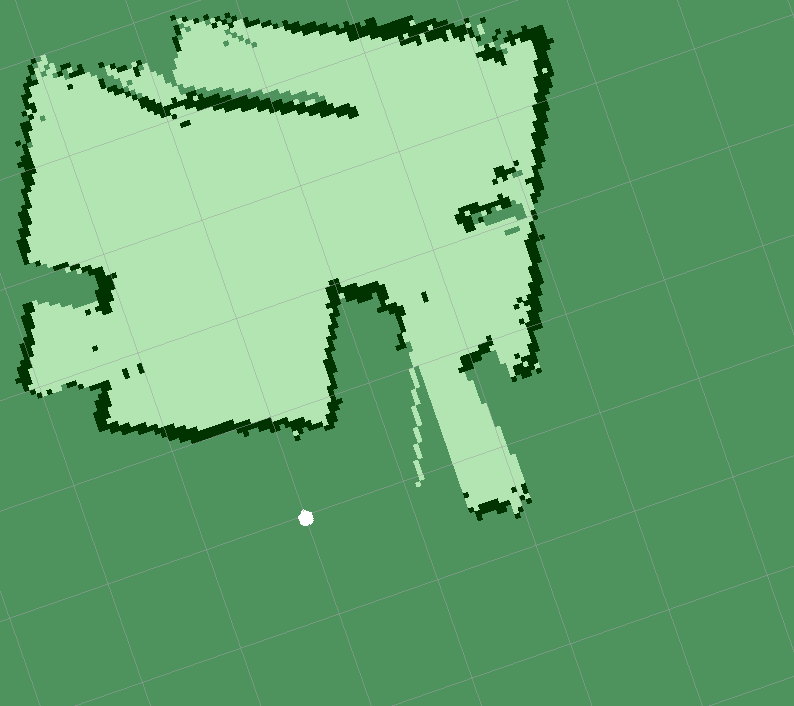
Using a GUI we can find do the same. Doing the rqt command and looking for active nodes and topics will yield a graph with the topics and which nodes are published to them.:

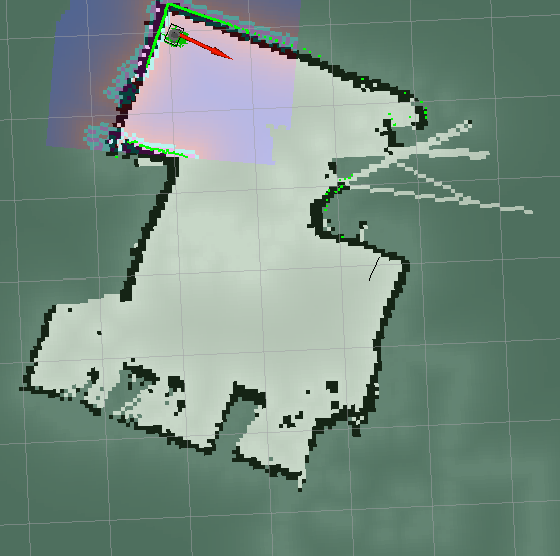


Week 5-6:Display user inputted map to Rviz and set designated colors

In order to use the navigation nodes for the bot, a user map must be created. The space for the bot should ample enough that the bot has space to “breathe”. Too close and the bot will stutter around for a while without actually moving. Once a map is fully created, the user can do things like create navigation goals and pose estimates. A pose estimate is basically the direction the bot and it’s components are facing.

For areas that have large amounts of open space, due to the size of the bot, it will show up on the map as empty space. The walls or seemingly immovable objects will show up as hard black lines.





Above is a screenshot from the .yalm file the map is in. The dark space some spaces are patch and can’t be used. However a large amount of space is available for mapping and navigation purposes. Background colors can be changed as well as the colors for the costmap and other functions. The second picture is of an updated map using the costmpa functionality with colors changed.

Week 7-8:

The python code has been started and is being edited with the various functionalities. At this time 3/27/19 it is not fully operational. The bot can move forward using the python code but cannot yet use the voice recognition function. Moving towards a goal is also possible but would require the use of the generated map.

Moving forward I would like to change the capability from using a phone to using a microphone to respond to voice commands as well some kind of feedback that let’s the user know that the command was received. As well as find a way to save the bot’s current position as goals. If that’s not possible hardcoding the goals will have to suffice.

Further Weeks:

Python Speech recognition hit a huge wall for the weeks following week 8. IN order to still include the speech recognition aspect, a new service was included. Rosbridge, serves as an intermediary between the host computer, an html file and the turtlebot.

Weeks 9-10:

I began learning to use rosbridge and it’s functions. First I tested the creation of the webserver and how to write and subscribe to topics using rosbridge. I found that running the server before running roscore makes the host computer a bit upset (slowing down processes). Also while the webserver is running, running any teleoperation nodes sometimes causes the system to stall.

Weeks 11-12:

These weeks were spent writing the html file and its ability to write to topics. Learning how to write JavaScript for this wasn’t terribly hard, but it was still a bit of a challenge. Connecting the webserver to the code I wrote had some issues. If any of the code is slightly off, the code will still run but some functions will not. Speech Recognition was one of the functions that constantly broke down for my code. I began using Annyang a small javascript library to run speech recognition for this project. Annyang runs well on it’s own(not using my code) and within my project has minimal bugs. One such bug was a Chrome authentication issue where Annyang is consistently asking for permission to use the microphone. No workaround has been found.

Weeks 12,13,14-

By this point the movement functions work fine on testing simulations. These weeks were spent trying to get the Pose estimation and goal movement working. Unfortunately, these things did not work even in simulation. I tried to run pose as a subscribe-able topic and have the html file constantly update using this value. However, any code calling the pose estimation functions did not run well with (Annyang) the speech recognition. Therefore, the independent study has ended with just being able to move the bot using the voice recognition library. Commands like “Go forward, go backward” etc, allow movement in those directions using annyang.