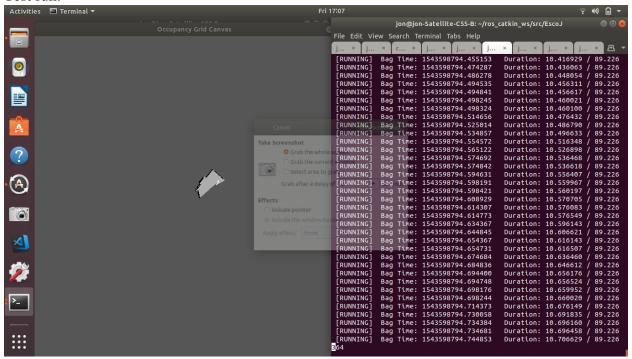
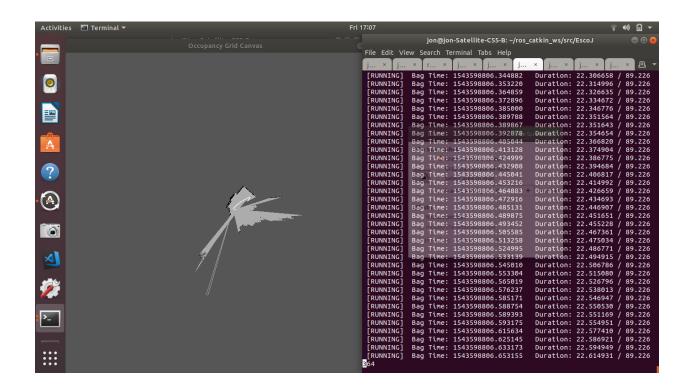
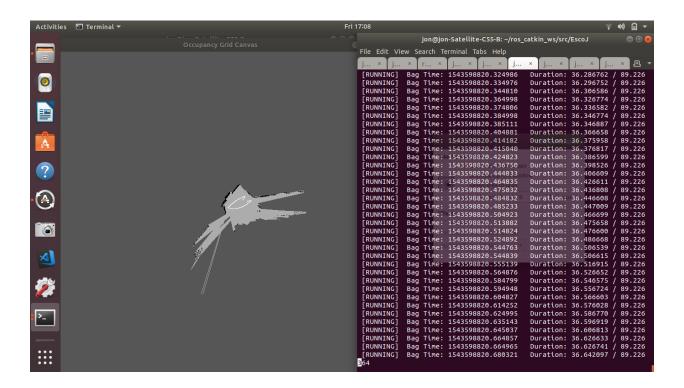
CSCE 574 Section 1 Fall 2018 – Homework 4 – John Esco

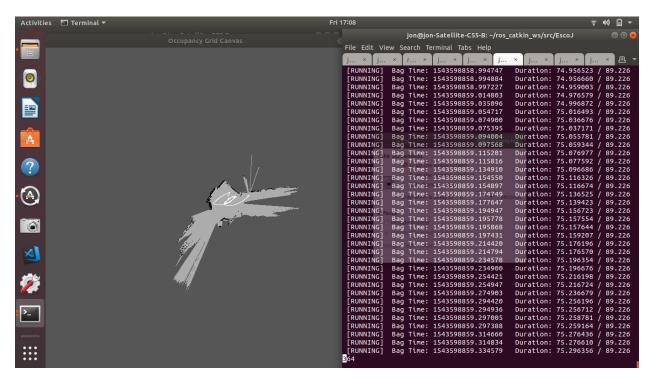
My original occupancy grid for assignment 2 did not function correctly. The first step for this assignment was to make the occupancy grid functional. The main issue seemed to be converting the pose into stage coordinates. After updating the math logic and fixing my line drawing algorithm, the occupancy grid began to work as expected. The logic to implement an occupancy grid starts with parsing the laser scan data. A for loop was then used to increment through the array or range values. I then calculated the x and y distance of each range value and adjusted negative angles by 2*pi. These values are relative to the robot's coordinate system. They are then converted to stage coordinates and used as parameters for the line drawing algorithm. Another issue arose here. The original line drawing algorithm did not plot the points correctly. After updating the placement of the plot() function calls, the occupancy grid plotted lines correctly. The plot function would plot over the CELL_ROBOT path. A quick fix was to add "if ((int)(canvas.at<char>(x,y)) != CELL ROBOT)" as a condition to plotting a new pixel. There was an issue with running the random walk and grid mapper on the Turtlebot at the same time. Occasionally, it would work but usually the terminal would become unresponsive and the raspberry pi would time out. The remedy to this was to have the Turtlebot random walk in the corridor outside the lab and record the data into a bag file. The bag file was then used to run grid_mapper on a local machine. Using measurements of the corridor and a scaling factor, the occupancy grid was set to be ~5cm in resolution.

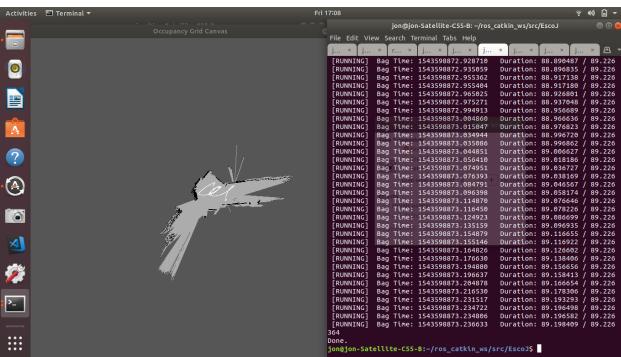
Test run:











The area being mapped is the end of the corridor just outside of the lab and to the right of Amoco Hall. This is a diagram with dimensions of the mapped region.

