Group 11

Kacper Puczydlowski

Jacob Wennersten

Lab 4 Report

Implementation

This code is built on Kacper's lab 3 A* code and Jacob's lab 2 drive code. A significant amount of work went into properly reading, expanding, and integrating the map provided by gMapping. The default grid size was used mainly in order to simplify the code, as trying to increase the cell size only resulted in more problems. Whenever a map is received all obstacles and walls are expanded so that the robot can be treated as a point. Additionally the map callback function checks the current A* path against the new map to see if an obstacle is now blocking the path. If so then the robot is stopped and A* is recalculated with the new map data. Movement is purely linear between grid waypoints, but the A* algorithm operates in 8 directions so diagonals are used. The map call back function checks for updated data from gmapping. Upon detection of an obstacle, the A-star algorithm will recompute the optimal path given the current and goal poses.

Results

One of the main issues encountered is that gMapping only refreshes the map after a certain amount of movement or rotation. Because of this, obstacles placed immediately in front of the robot wouldn’t be detected in time. Additionally, scan information did not properly register grid cell information directly in front of the sensor. That and other issues with blocking were likely due to a lack of understanding of how rospy handles subscribers and threading.

The final code is fairly reliable as long as gMapping sends the anticipated sensor data to the occupancy checking algorithm. Since the drive code references the reported current position of the robot at each waypoint it can correct for small map shifts but it shifts too far or inside of an obstacle then the navigation can break down. Expansions will include frontier exploration for SLAM, reading data from a local cost map, and using the built-in navigation stack for executing trajectories between waypoints.