Lab 5 Report

Matt Hartnett, Blake Novak, Abdallah Esmael

Section 1

This lab was completed by Matt Hartnett, Blake Novak, and Abdallah Esmael.

Section 2

If there are noisy LIDAR readings when the odometry is perfect, then the map will have an overestimation of the number of obstacles. Since the LIDAR is noisy, it produces several different (all but one being erroneous) values for a single distance measurement (when repeated). This means that the map will contain more points that the robot cannot travel to than it should. If the LIDAR is perfect but the pose_theta of the odometry is off, then the robot will be able to see obstacles correctly, but not keep track of them correctly. Since the robot needs to build a map out of the LIDAR readings, an inaccurate pose means that the robot will no longer be able to accurately correlate the readings to a 2D map of the world.

Section 3

It is important to choose the value with which you increment the map since it determines how “important” each sensor value is. In order to choose the right value, you need to consider the amount of noise on the LIDAR reading and the amount of times that a given object will be mapped. The noisier the signal, then the lower the increment should be since each individual measurement is more error prone. If you are going to be mapping each object a relatively small number of times, then the value needs to be larger so that it gets to a reasonable threshold by the time the robot finishes mapping. If the value is too small then the robot has to scan the same object many times to get a reading, but if its too large, then too much noise is introduced and the map is no longer accurate.

Section 4

We chose our threshold value by trial and error, first starting with a large threshold and slowly lowering it until we were satisfied with the level of noise and speed at which we could map. At first we tried a very low threshold which is good for quick responses, however we quickly noticed that with too low a threshold we were still introducing noise into our readings, especially if we spent a long time scanning one room or object. With too large a threshold, it took us much too long to map every room as we had to spend a lot of time pointing the sensors at an object before it recognised it. This was especially apparent for small objects like chair/table legs. By starting with a high threshold and working our way down we were able to come to a good balance between speed and accuracy.

Section 5

A simple next step to implement a way around unforeseen obstacles would be to add a check every time a new sample location crosses our threshold. The check detects if a new object has been added to our virtual representation, and if so then checks to see if the object intersects our waypoint paths. If both a new point is added and it's in the way of our current path, then we would call Dijkstra's/A* to recalculate a new route from our current location.

Section 6

In theory we could use RRT to generate a viable path to our goal, however in practice there are several pitfalls which leave Dijkstra's/A* having a significant advantage when planning a route, including the following. RRT will eventually create a route to our goal, however because the path is randomly sampled instead of checking neighbors the path it takes to get there could be much longer than the shortest path which we want. Another problem is if a goal is not possible to reach. In this case, Dijkstra/A* will systematically check every path and return false if not possible. RRT on the other hand will keep guessing and checking unless we set a timeout or other threshold to cancel the search.

Section 7

Overall, we spent approximately 8 hours programming this lab. Unfortunately we were not able to get our path planning to work. Everything else seemed fine, such as the mapping functions worked great. Our Dijkstra's algorithm for finding the path of waypoints did not work. We think this is because of the data structures we are using when trying to add and pop each node onto the stack, and accessing lists of tuples proved difficult.