Grid Mapper

By: Lilian Lamb

Introduction:

An important part of SLAM (Simultaneous Localization and Mapping) is being able to map the environment, the robot's trajectory, and potential obstacles. Being able to know the exact positions of a robot or obstacles can be tricky even in a simulator, especially when dealing with lidar. Lidar produces an array of ranges, even when no obstacle is detected. The lidar will report the maximum range when no obstacle is detected but what if there is an obstacle right at range? This project asks students to use a random simulator program and map the environment based on robot trajectory, unknown, free, and occupied cells.

Application:

The main focus of the algorithm is the laser callback function as the majority of the other code was provided. The laser callback function is executed every time the program was provided data from the laser sensor (lidar). The goal of the function was twofold. The first is to collect the range of ranges and angles and calculate the corresponding x and y lidar values. Using these x and y positions, mark every value from the x and y position of the robot to the x and y lidar reading as a free cell. Then after marking the free cells, mark the x and y lidar values as occupied. I will admit that it seems like the program maps more free spaces than it should.

The second goal of the function was maintaining the map, and this is done in several ways. First, since the edge of the lidar is always marked as occupied, the program is set to never override free and robot cells and will always override occupied cells. This can lead to the lack of occupied cells as the positioning is not exact and there can be some overlap. In the program, I chose to prioritize marking free spaces instead of occupied as there were several cases where the program would mark free spaces as occupied when doubling over on marked spaces. This can cause issues where the call obstacle is small; however, after testing, due to the pixel size of the grid mapper, the error of small obstacles is high regardless. The second way the program maintains the map is by ensuring that the robot path is never overwritten. In the program, the area should only be marked as free when the space is marked as occupied or unknown, ensuring that the robot trajectory is clear. Both maintaining the current map and being able to contribute is crucial, so I made sure both are considered in the program.

Figures:

The figures represent the map that the robot traversed and the path that the robot mapped. Due to the random map, the paths do not show a complete map as the robot had a tendency to get stuck.

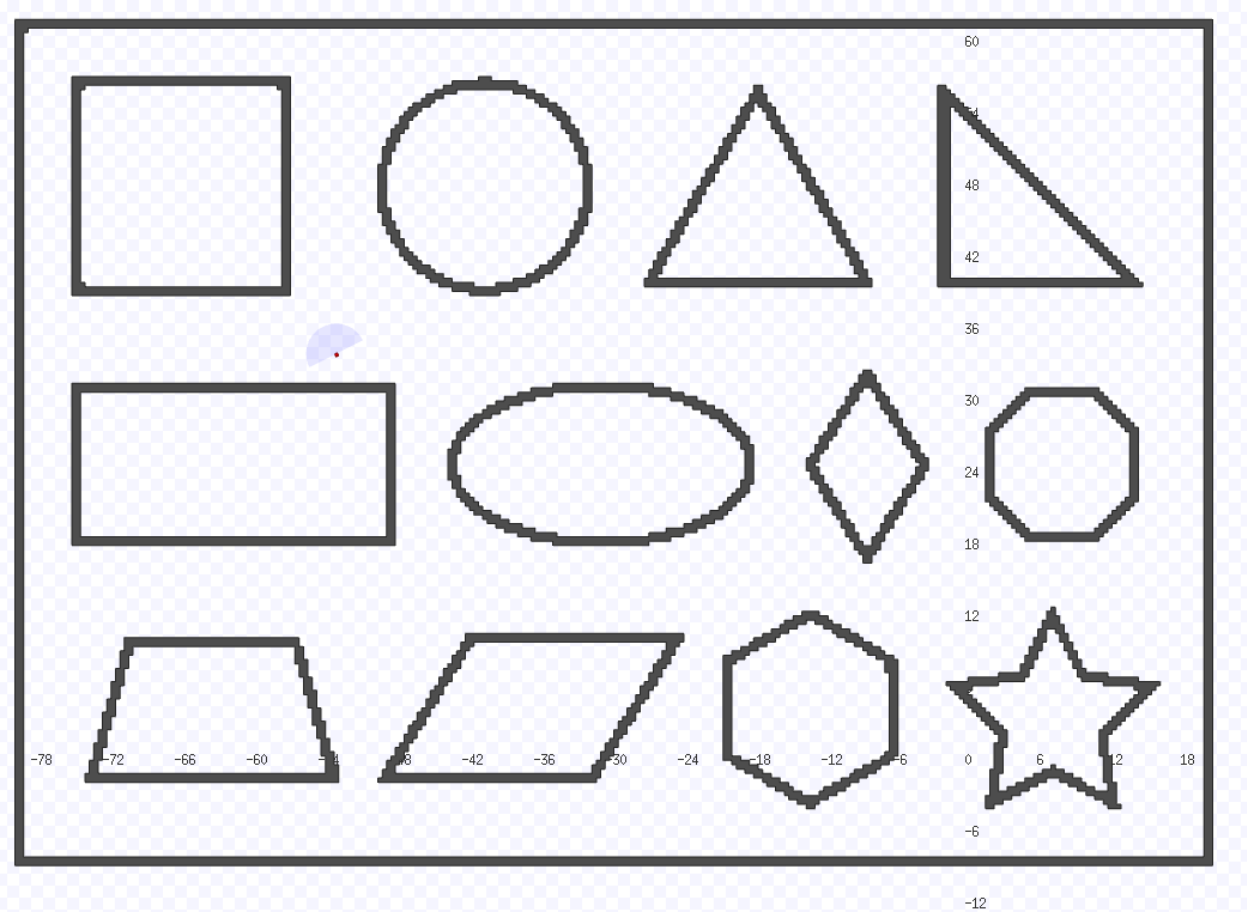

Figure 1: The world the robot traversed.

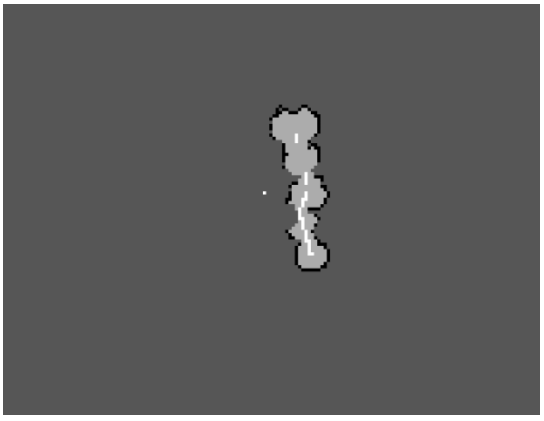
Figure 2: Robot starting traversal.

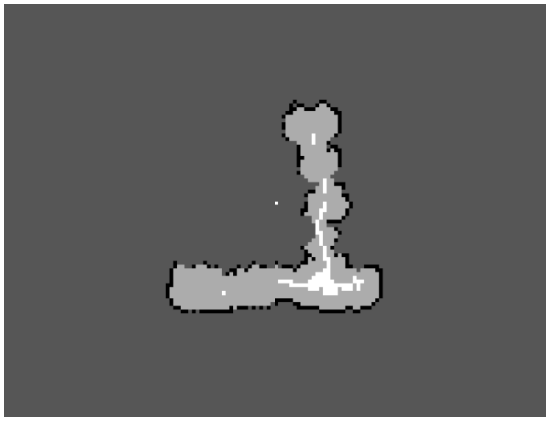
Figure 3: Map after robot got stuck on bottom

Figure 4: Robot path and obstacles without mapping free squares.

Conclusion:

Being able to map to an environment and maintain position within that environment is a crucial and important aspect to robotics. Assignment 2 exemplifies this point and asks students to create a program that maps a given stage world based on data from a lidar in a random walk. The goal of the grid mapper program is to map all of the cells in a given environment (free, robot, and occupied) and maintain the robot's trajectory. The program is not perfect but fulfills its goal at the bare minimum.