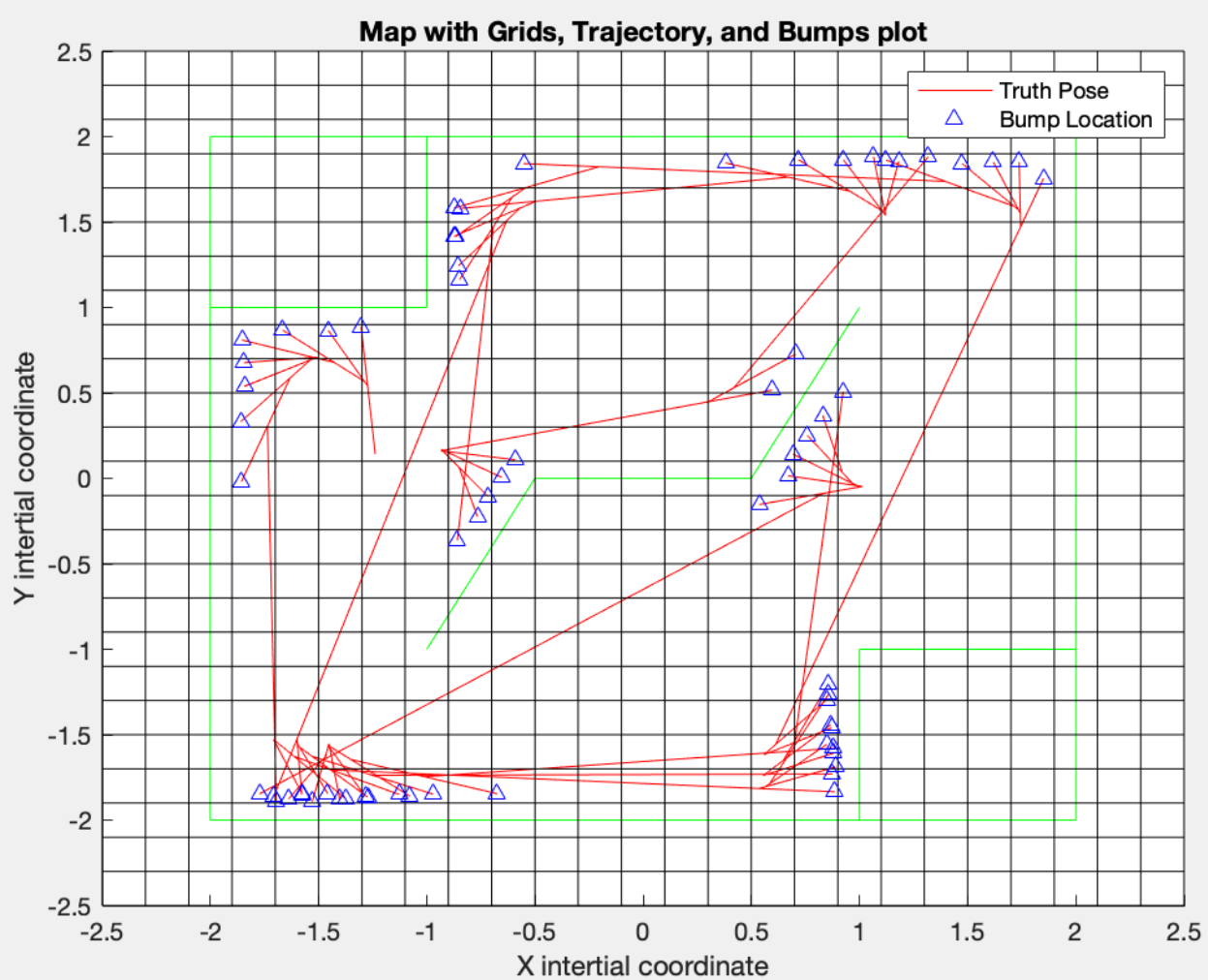
Jonathan Nusantara
AMR
HW5

Mapping – Bump Sensor

1.



2. The inputs are:

- robotPose: truthPose of robot (x,y,theta) across the entire time t
- bump: The 3 bump sensor measurements throughout the entire time [rightbump leftbump frontbump]
- L0: The prior initialization value of logOdds for each cell in the grid.
- numCells: the number of cells in X and Y direction in the grid [X Y]
- boundary: vector representing the boundary of the grids [Xstart Xend Ystart Yend]

The outputs are:

- logOddsGrid: The final log odds matrix representing the occupancy grid.

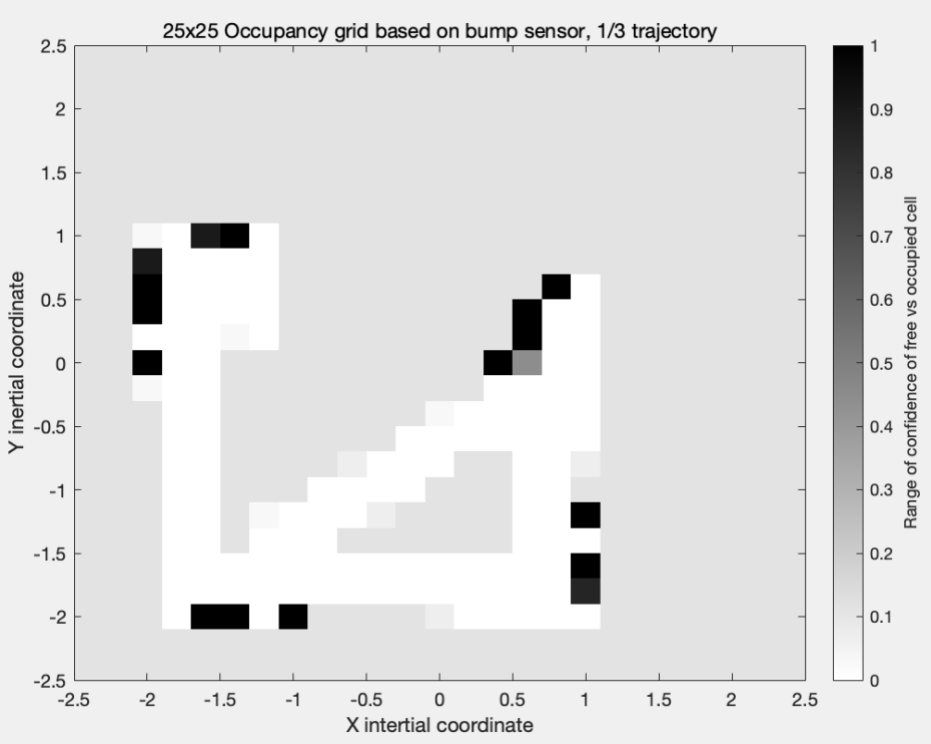
3. In the log odds calculation for each cell at every timestep, we are adding the value of the inverse sensor model calculation, given by the formula:

$$\log \frac{P(m_i = 1 \mid z_{1:t}, x_{1:t})}{1 - P(m_i = 1 \mid z_{1:t}, x_{1:t})}$$

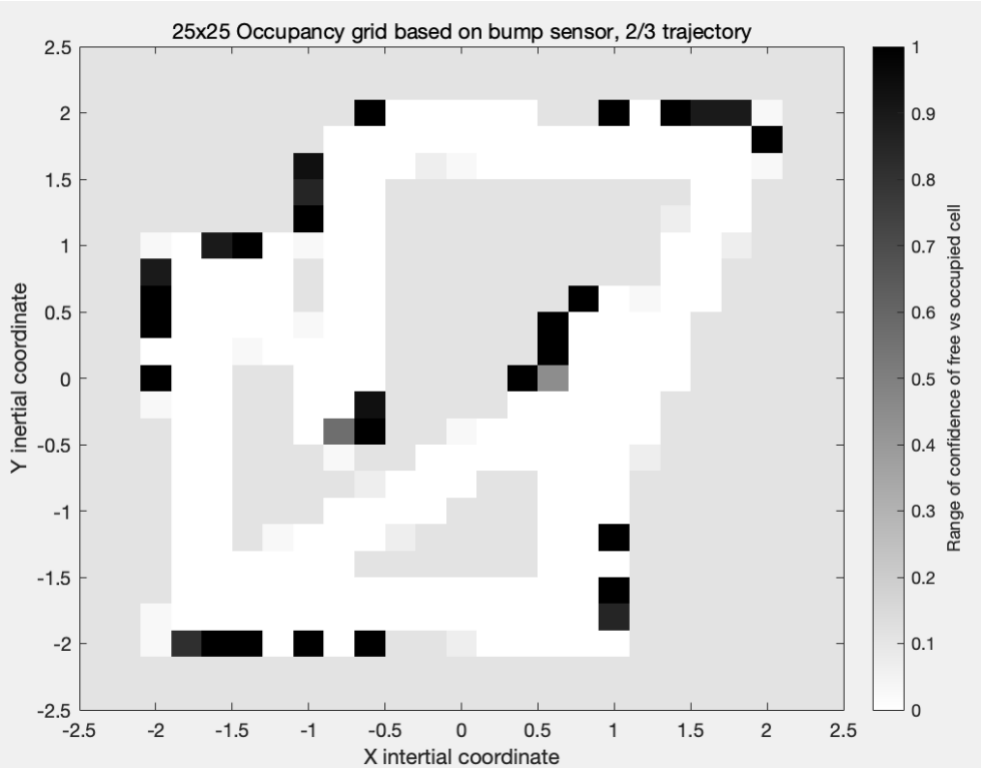
This is based on how likely is a cell occupied, divided by how likely is a cell free, given the location and bump information. At every timestep, we are able to get information from 4 different locations: the current robot truthPose and the 3 bump sensor locations. The right and left bump sensor is assumed to be 45deg from the front facing bump sensor.

Then, we define the probability of a cell being occupied based on that information. If a bump sensor is triggered, we assumed a 0.99999 probability that the cell at the bump sensor location is occupied, and so the inverse sensor model will be 5. The probability shows that we are very confident that the space is occupied, as bump sensor is triggered. On the other hand, at the location of the robot truthPose and at locations of bump sensor is not triggered, we can learn that the cel is probably free. We assumed the probability of occupied to be 0.3, and the inverse sensor model = -0.368. The 0.3 chance that the space is occupied shows that we are not too confident that the space is free despite passing that cell, because the body of the could have only passed part of the cell, and other part of the cell may actually be occupied.

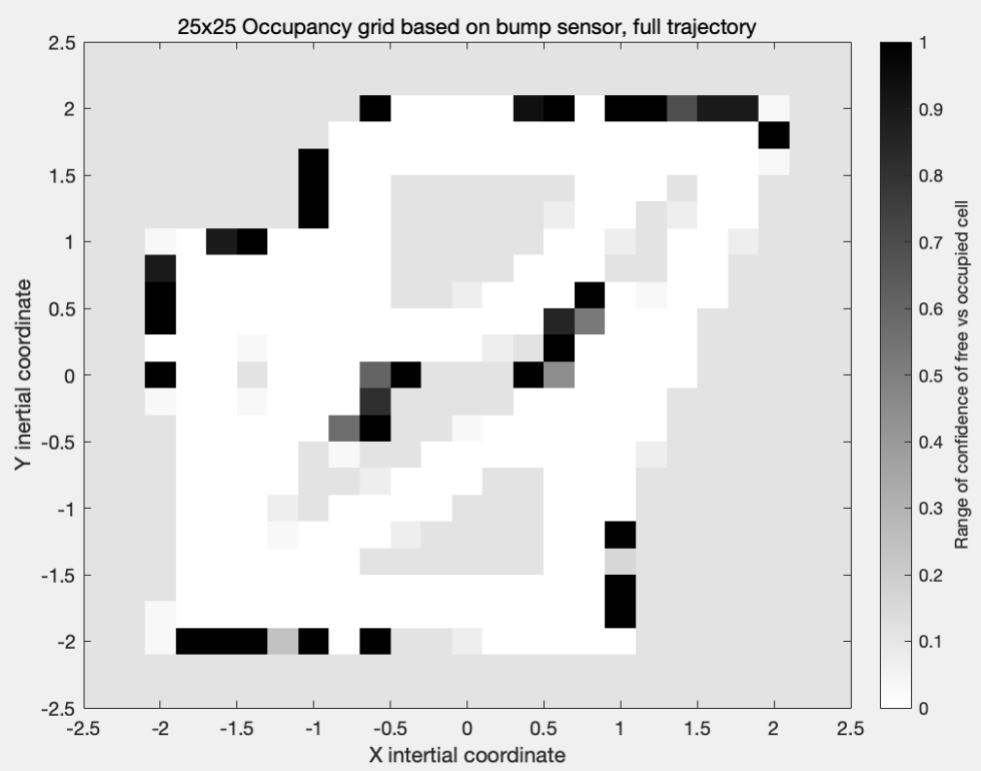
4. Plot at 1/3 of entire time done:



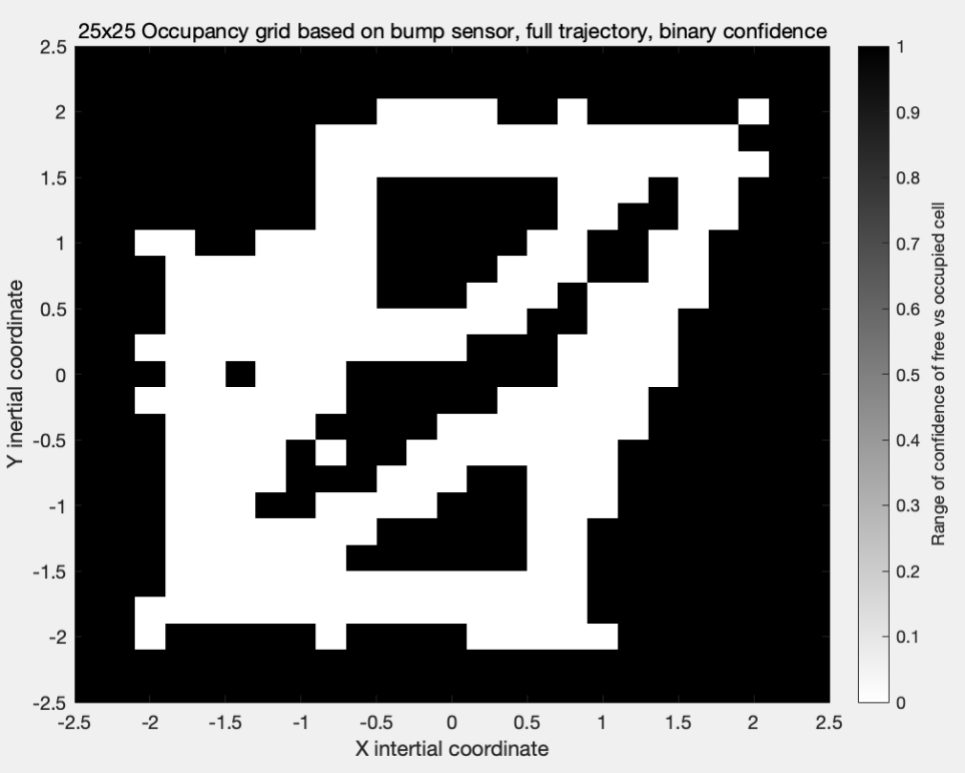
Plot at 2/3 of entire time done:



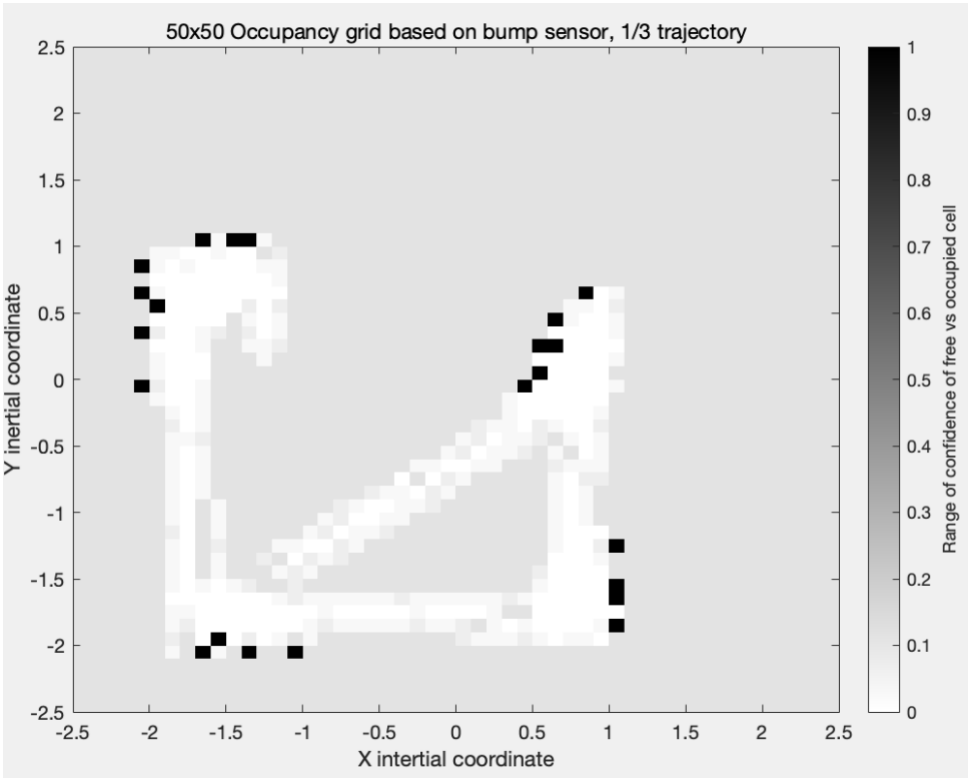
Plot after entire time done:



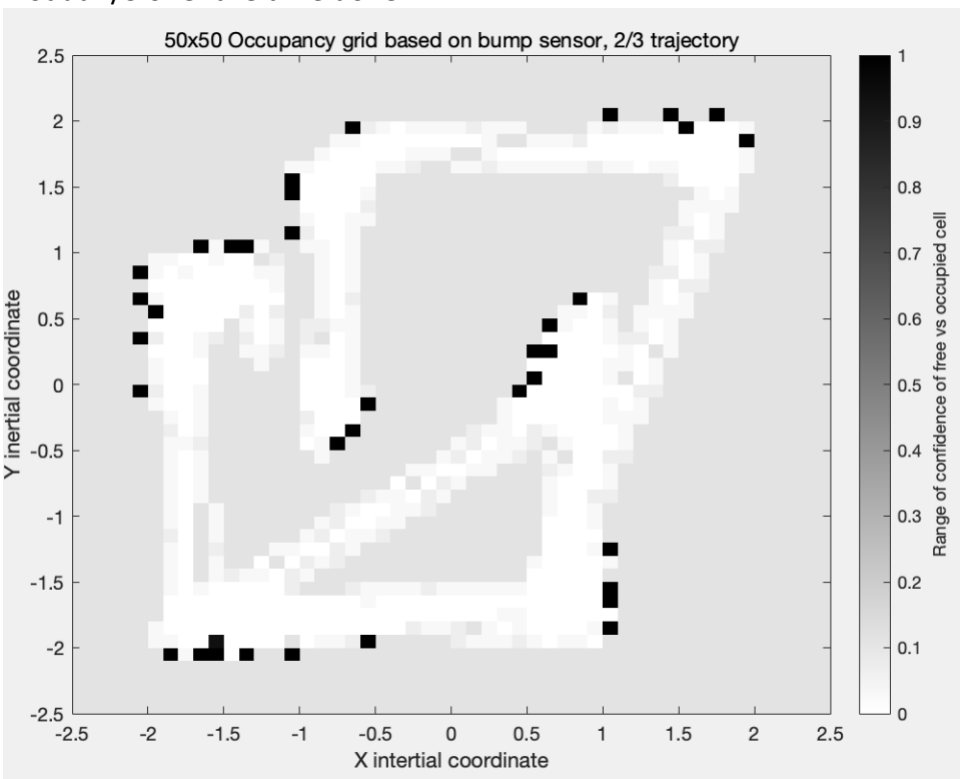
5. It would be better to wrongly assume that a space is occupied rather than to wrongly assume that a space is free. In order to decide if each cell is occupied or free, we would have to perform threshold on the log odds for each cell to get a binary result for each cell. The threshold value that I have chosen is 0.5. If the logodds is above or equal to -0.5, then the cell is occupied. If it is lower than -0.5, then it is unoccupied. The reason that we chose -0.5 is because the initial (prior) logodds for each cell is 0 and the logodds of each cell will be subtracted by 0.368 if the robot pose (or bumper location) is at that particular cell. So, we would only assume or belief that the cell is unoccupied the robot is at that cell for at least 2 timesteps and that the bump sensor is never triggered at that location. The threshold is not at 0 because it would not be wise to assume that the cell is unoccupied just because the robot was there for one timestep, as the cell may be way bigger than the robot, and there may actually be an obstacle at that same cell. After the change as mentioned above, the plot is as below:



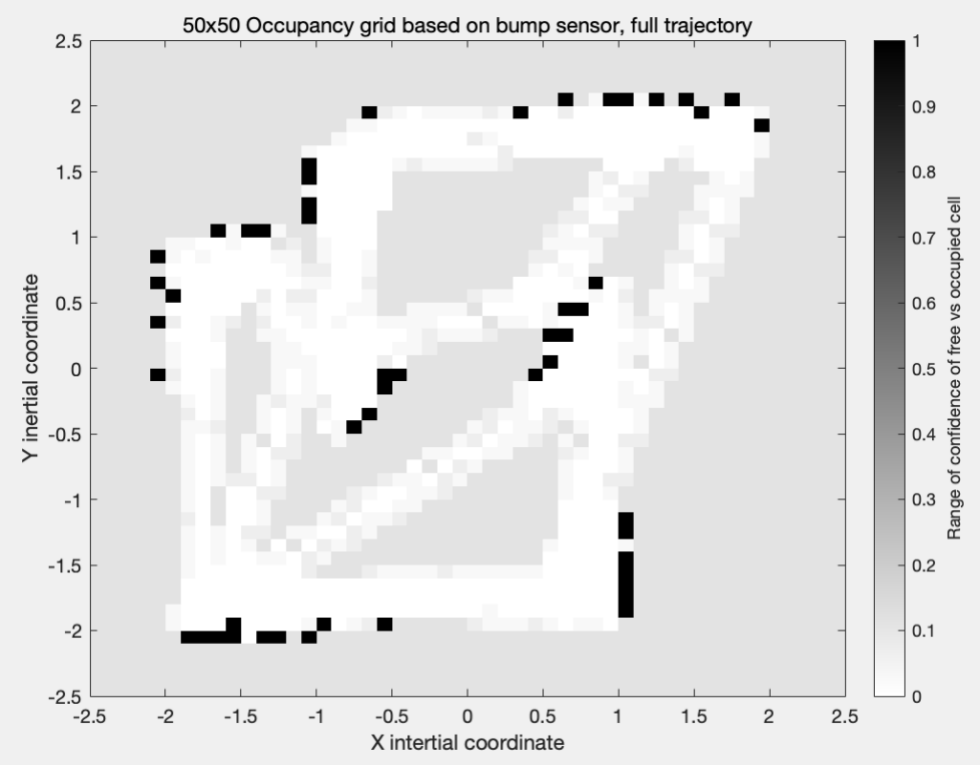
6. Plot at 1/3 of entire time done:



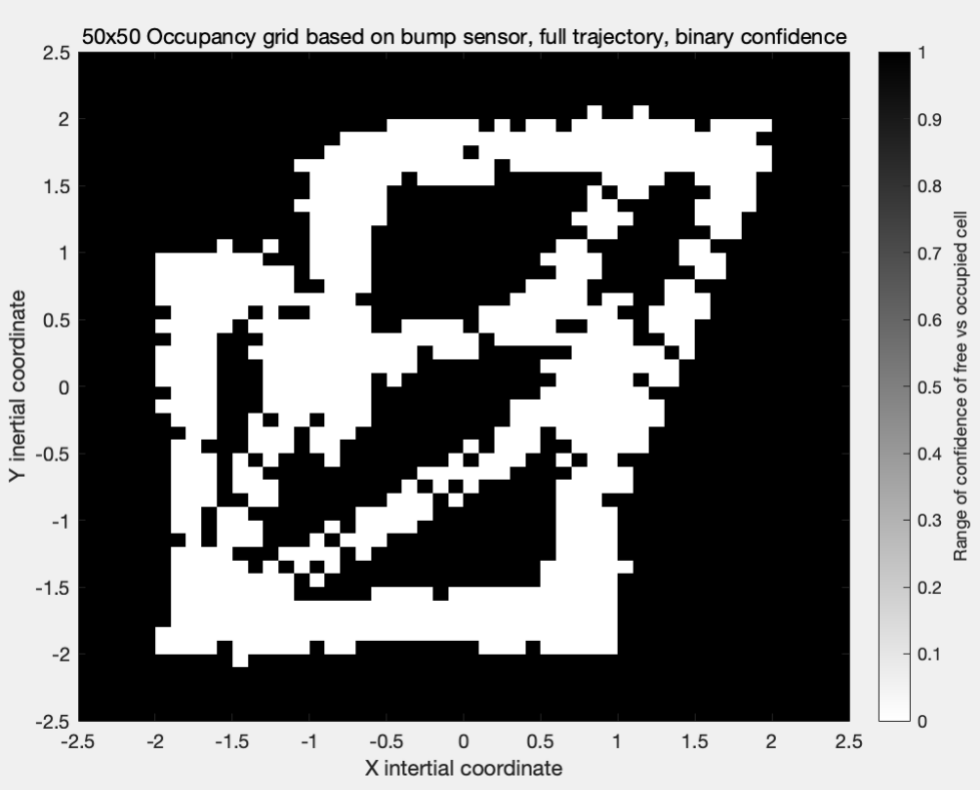
Plot at 2/3 of entire time done:



Plot after entire time done:



Plot with binary confidence in each cell:



One thing that we can see is that with the smaller cell size, the information we learn from the location of robot or bump sensor would cover less area at each timestep. When the bump sensor is triggered, we learn about a smaller occupied area. Looking at the final non-binary occupancy grid, we have much more “gray” uncertain areas with a 50x50 grid, and so when we take a look at the binary 50x50 occupancy grid, there is less “white” free space in the map. This is actually good, because in the 50x50 grid we have a more accurate mapping of the spaces in which our robot has passed through and verified to be free, when compared to a 25x25, in which we may make a wrong occupancy assumption of a cell, based on the small area of the cell that our robot has covered.

7. The difference is that the result of the occupancy grid form the real bump sensor data may be less similar to the real map when compared the result from simulated. Firstly, there may be some delay from the bump sensor. Our program does not perform timestep interpolation and only matched the bumpSensor and truthPose based on the order they are recorded. So, location of detected bumps (or free space) may actually be wrong when delays are significant. Additionally, in real life, it takes some amount of force for the bump to be “pushed” against. If a robot merely touches the wall, the bump sensor will not be triggered. This would result to a false information, which would not happen in a simulation. Assuming that using real bump sensor data also means using real truthPose data, there will be cases where truthPose will not be recorded as the localization camera could not detect the robot. This will again result to a loss of information and poorer occupancy grid for using the real bump sensor data.

Mapping – Depth Sensor

1. The inputs are:
    - robotPose: truthPose of robot (x,y,theta) across the entire time t
    - depth: The 9 depth sensor measurements throughout the entire time
    - L0: The prior initialization value of logOdds for each cell in the grid.
    - numCells: the number of cells in X and Y direction in the grid [X Y]
    - boundary: vector representing the boundary of the grids [Xstart Xend Ystart Yend]

The outputs are:

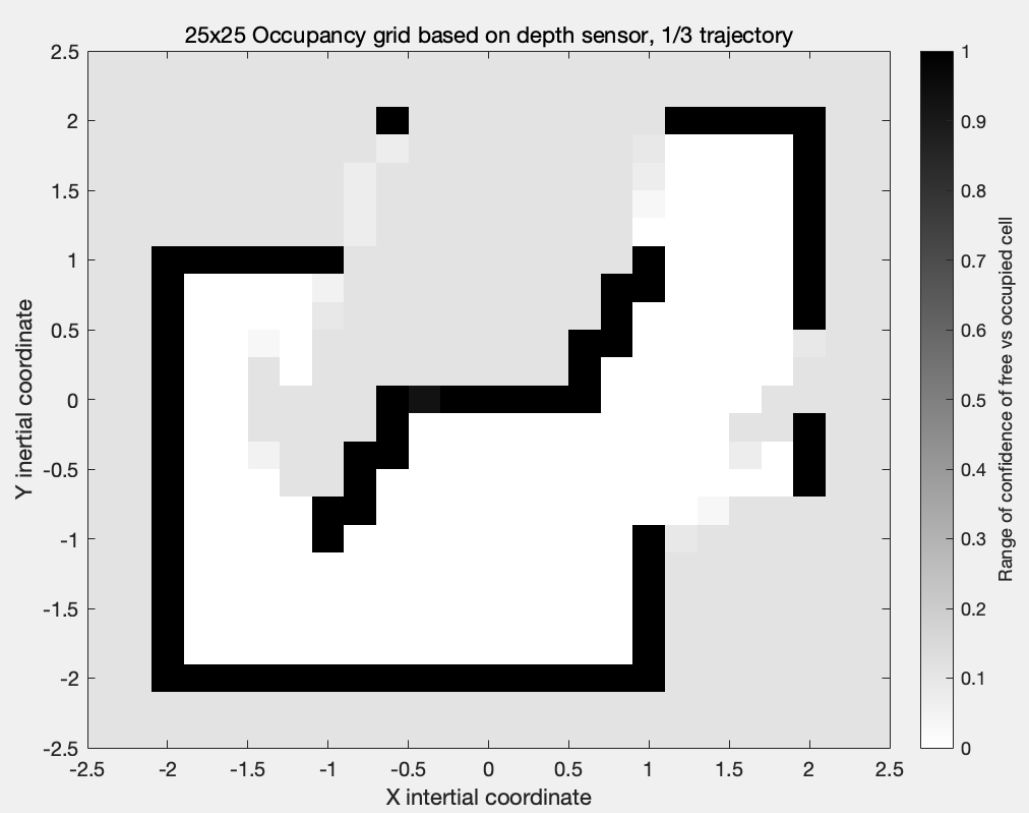
- logOddsGrid: The final log odds matrix representing the occupancy grid.

2. We are using a similar inverse sensor model to the bump sensor problem, however we do have much more information at every timestep. Firstly, we have the truthPose of robot. We define a 0.3 probability that the cell of the truthPose to be occupied, and the inverse sensor model = -0.368. We assumed that there is still a 0.3 chance that the cell is not free as the robot will not cover the entire cell.

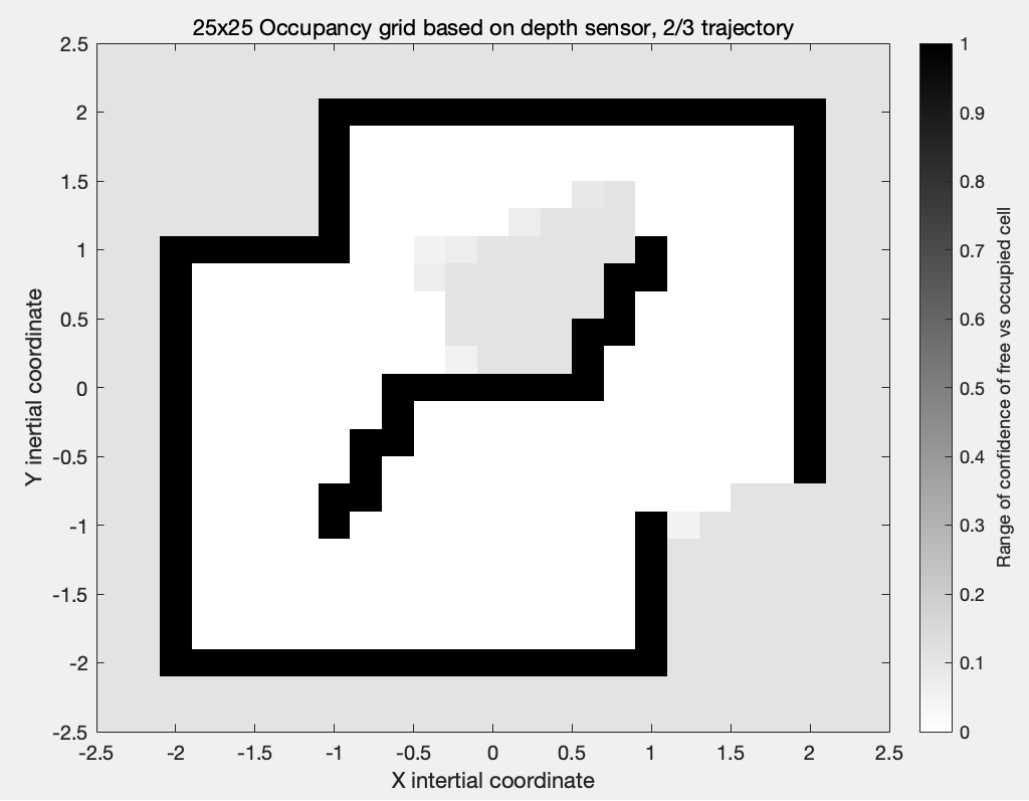
Secondly, we have the detected obstacle's location based on depth sensor. We ignored the 0 or 10 measurement, as they won't be accurate due to sensor's limitation. At the location where the obstacle is detected, we define a 0.99999 probability that the cell is occupied, and so the inverse sensor model will be 5. The probability shows that we are very confident that the space is occupied.

Lastly, we have the information that the cells covered by the the depth sensor rays (cells between depth sensor to detected obstacle) may actually be free. In order to find the location of each connected cell covered by the rays, we calculated find the coordinates based on an increment of the smaller between the cell width and height. At each of this cell, we define a 0.4 confidence that the cell is occupied, and so the inverse sensor model = -0.176. We have a low confidence that the cell is free because the depth sensor rays is so thin that it may have narrowly missed the obstacle on that cell.

3. Plot at 1/3 of entire time done:

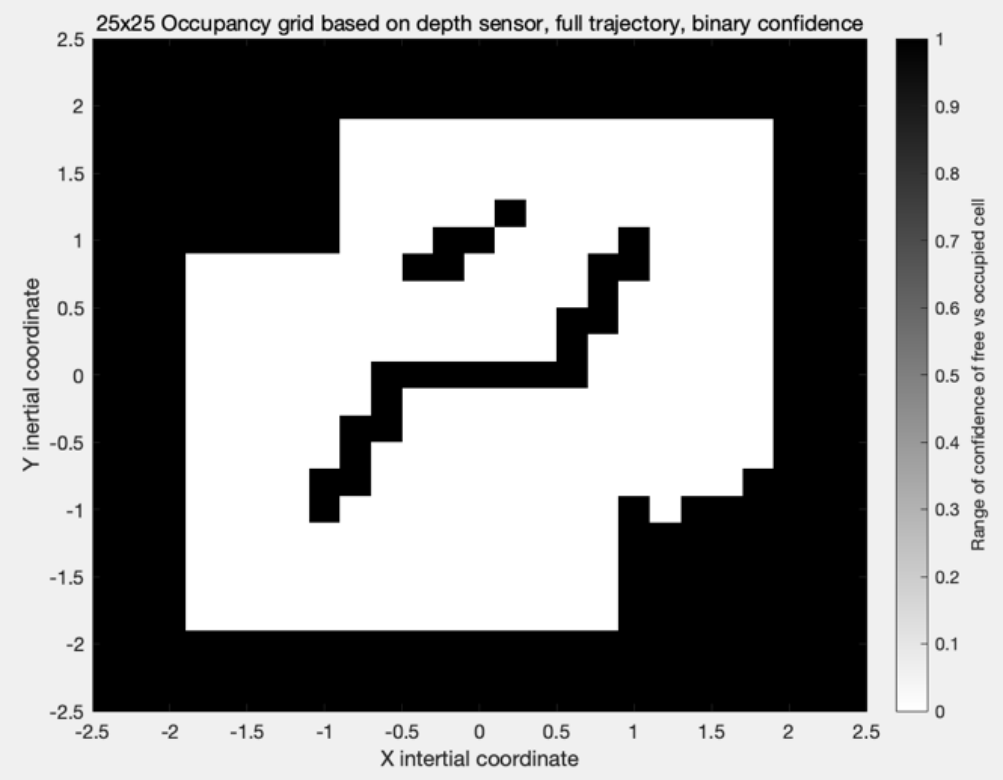


Plot at 2/3 of entire time done:

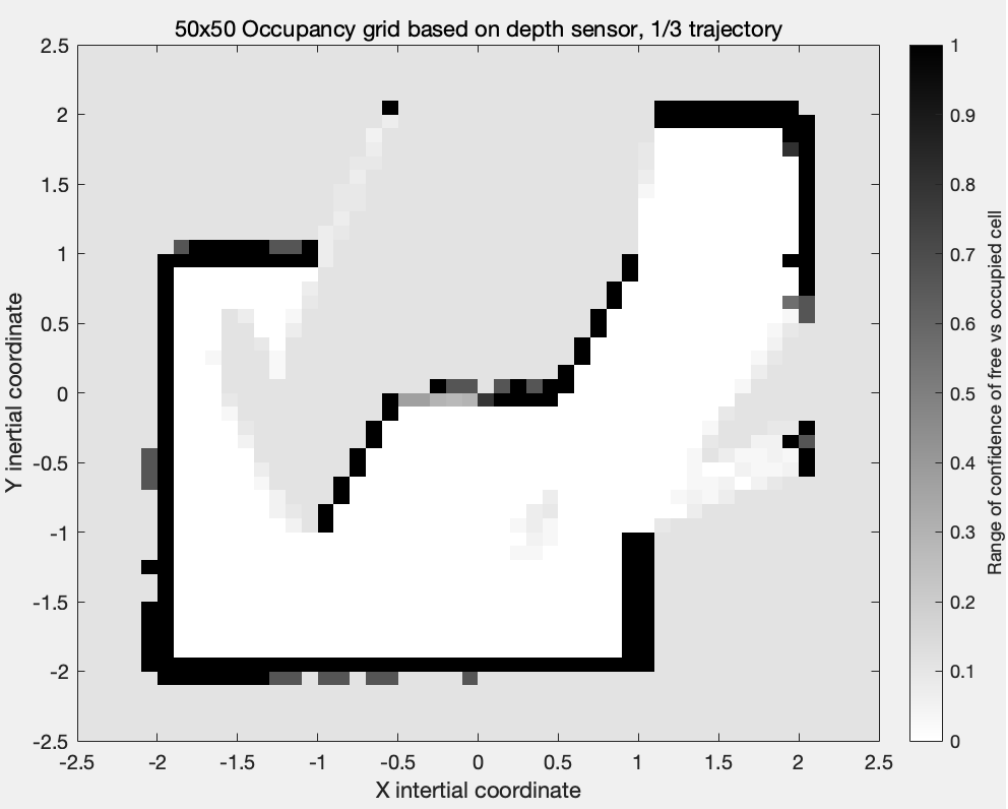


Plot after entire time done:

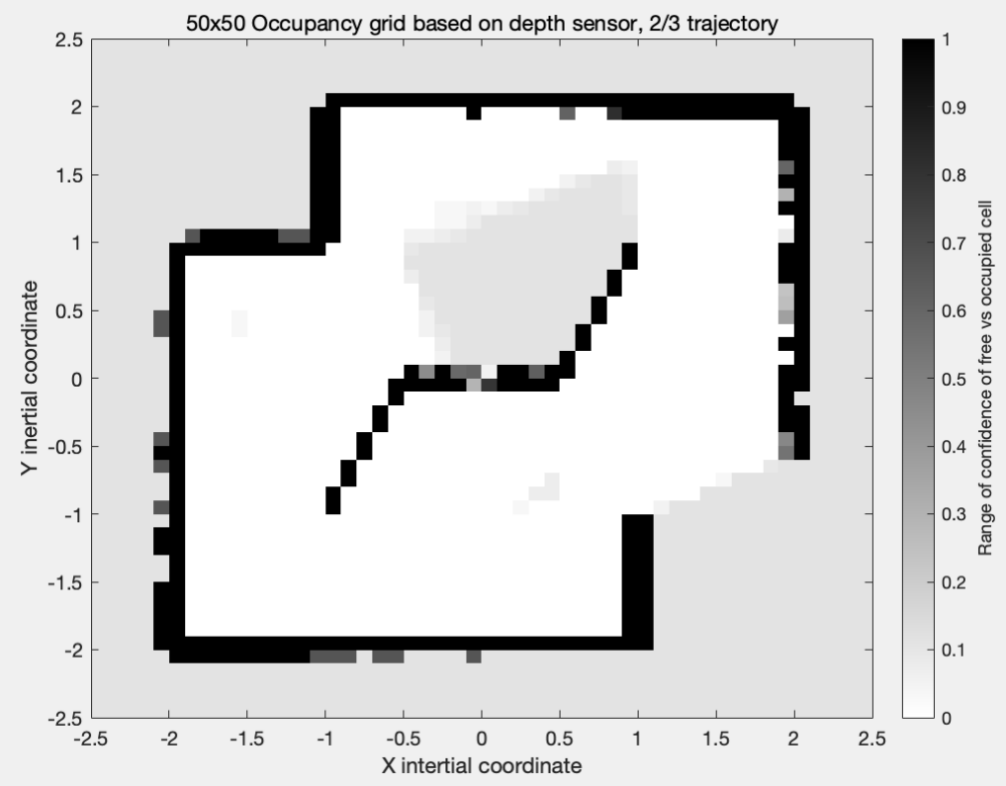
Plot with binary confidence in each cell (not asked, but decided to plot it):



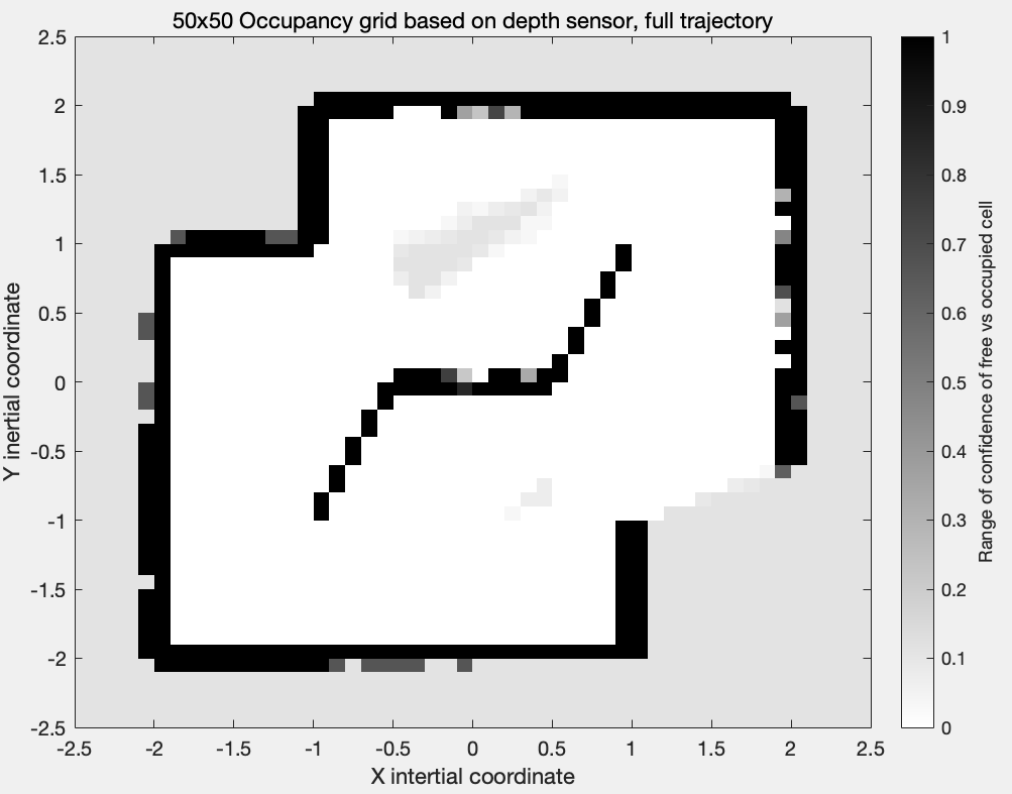
4. Plot at 1/3 of entire time done:



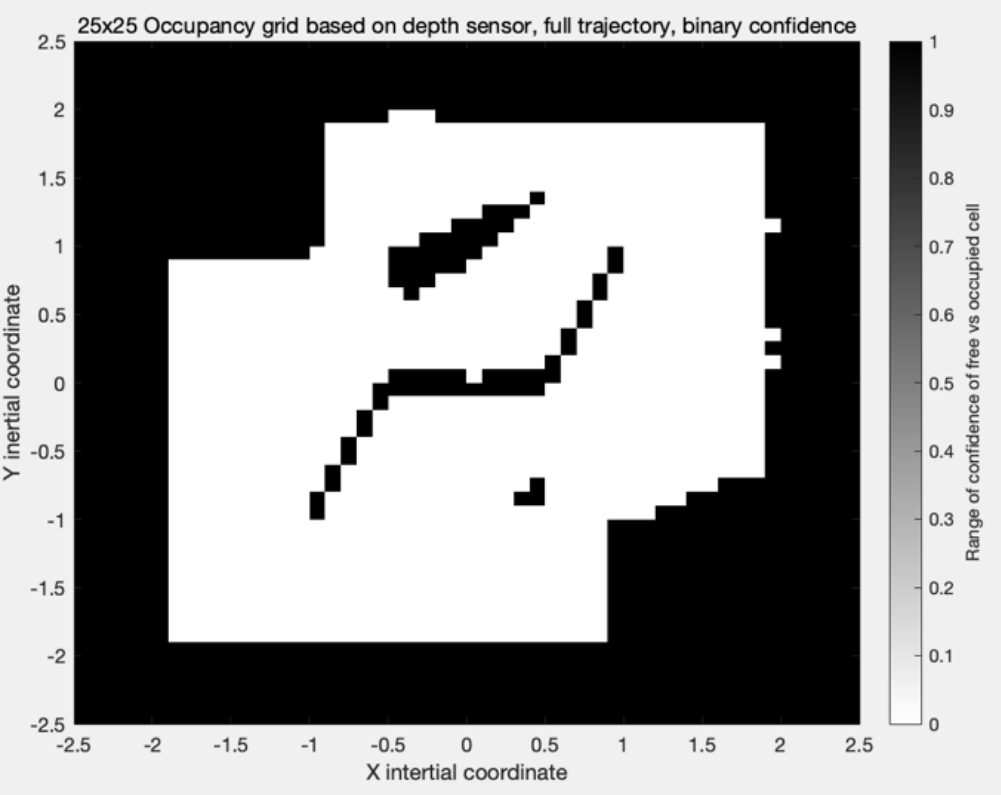
Plot at 2/3 of entire time done:



Plot after entire time done:



Plot with binary confidence in each cell (not asked, but decided to plot it):



The first thing that is obvious is that the 50x50 results to a smaller cell dimension. The 50x50 grid allows the occupied cell to more accurately represent the actual obstacle, being refined and thin. This is something that the 25x25 grid cannot do because the size of each cell is pretty big. As a result, we can see that there is a bit more free space in the 50x50 grid. I also noticed that there is an obstacle/occupied cell at (0.5,-1) in the 50x50 grid, which does not exist in 25x25 grid. This is because the cell in 25x25 would cover a larger area. That cell may receive more confidence from the log odds for the cell to be assumed as free.

5.
Pros:
1.A finer grid resolution would provide a more accurate representation of the actual map. The location of the obstacle will be more accurate, in which the occupied cell will only cover the actual obstacle itself and not the free space surrounding the obstacle.
2.Similar to the previous point, we may be able to detect free space or obstacle that is previously not detected with a coarser grid. A coarser grid would have a bigger cell dimension, in which the log odds would be form a larger area. This would lower the accuracy of the occupied or free assumption.
Cons:
1.In order to provide an accurate map, we would need more sensor data from the robot, because we have more cells to perform log odds update. This is because each sensor data (i.e truthPose) would now provide information for a smaller dimension of space (as each cell has a smaller dimension).
2.With a finer grid, we would need more computation power to perform the log odds calculation. Based on our experiment, we may not feel much difference between 25x25 or 50x50. But if this grid is to be significantly increased, then the program will take slower to complete.

6. The occupancy grid map provided by the depth sensor is better compared to the bump sensor because we are able to gain much more information from the depth sensor at every timestep compared to the bump sensor. With bump sensor measurements, we can only learn about the cell in which our 3 bump sensors or robot truthpose is located at every timestep. So, it would be extremely difficult to get information from every cell on the grid. As we can see from the final non-binary occupancy grid from bump sensor, we only have a few “black” cells that we learned when our bump sensor is triggered. The “white” cells are only cells that the robot has passed through. The “gray” uncertain areas covers most of the map, which are sections that the robot did not touch.
On the other hand, the depth sensor is able to get a lot of information at every timestep. With each of the 9 directions, we are able to learn the about the location of occupied cell as well as the free cell from the trajectory of each depth sensor rays. This amount of information per timestep allows us to cover most of the map, despite having the same trajectory to using the bump sensor.

7. The result of occupancy grid from using real depth measurement will be less similar to real map compared to using simulated depth measurement. Firstly, there are time delays from the depth sensor. The program that I currently have would match the depth sensor measurement to the truthPose based on the order they are recorded, and not by the time stamps. If this is not fixed, then there will be a lot of errors in the occupancy grid. Secondly, there are sensor noise in a real depth sensor. This sensor noise could be fatal, considering the scenario where the depth sensor should have detected an obstacle in a cell, but just missed it due to the noise. That cell will then be treated as “free”, when it should have been empty. There may be plenty of this occurrences in the occupancy grid, depending on how much noise exist. Due to these reasons, the occupancy grid from the real depth measurement may be slightly poorer.