Value Iteration Project

For this project, we were tasked with designing and implementing an algorithm which would determine the optimal action for our robot to take given some beliefs about the world. To start this lab, we used our project 2 code because we found it to be highly effective at forming beliefs about where in the world the robot is. Once we know where the robot may be, we need to figure out the optimal policy for reaching the goal state. We accomplished this using value iteration.

To start off the value iteration algorithm, we need to create a reward associated with each state. For the purposes of this lab, we set the value of all terminal states to be a large positive number or a large negative number for the goal and death states respectively. All non-terminal states are considered to have a reward of -1. Once we have determined our reward function, we need to iteratively calculate the 'actual' value of a state. We start off with a utility mapping which assigns the same positive/negative numbers to terminal states, but zeros for all other states. Then, as long as we see changes in the utility mapping, we iterate and create a new utility mapping. To iterate the utility mapping, we go state by state and set the new utility value to be the immediate reward of the state added with the expected reward from the best action in that state. I also added a discount factor which weighs immediate rewards more highly than delayed rewards. At the bottom of this report is a visual comparing the value map to the visual for a few of the different worlds.

After running this algorithm and deriving the 'optimal' utilities, it is quite easy to identify the utility-maximizing policy. If we had perfect knowledge about which state we were in, we could simply create a table that would immediately determine the action to take by looking at the state. However, for this lab we also had to deal with uncertainty in the position. As a result of this requirement, I had to be more clever than just storing a single action for each state. Instead, for each state, I stored the expected rewards in a W x H x 5 matrix, where W and H are the width and height of the grid, and 5 because we need to store the expected utility of each of the 5 actions. With this W x H x 5 matrix calculated, we can determine the optimal action given a set of possible states and their associated probability. To do this, we can iterate over each state and create a weighted sum of the states probability and its expected value for each action. This will tell us which action is going to be best, and we choose that one. Below I've given a small example of a world's current probability beliefs and rewards..

-100,.3	100, 0.0	-100, 0.0
-100,0.0	-100, .7	100, 0.0

In this small world, we can see there are two states we could be in, the top left and the bottom middle. If we assume our transition model is deterministic, then we can see that we optimize our outcome by moving to the right. The top left square has a value of 100 for moving to the right, and a value of -100 for all other actions. The bottom middle square has a value of 100 for moving to the right or up, but -100 for down, left, and stay put. By weighting these values by the probability of being in each state, we get values of up, down, left, right, and stay of

40, -100, -100, +100, -100. To get the +100, we should choose to move to the right. This oversimplified example glosses over much of the detail with value iteration, but it does show generally how my algorithm makes decisions given beliefs about being in different states.

Once my algorithm was written, it was able to handle the different worlds pretty well. When it was given full information (1.0, 1.0, known), my algorithm was able to reach the goal in perfect or nearly perfect time. The discount factor along with the implicit -1 reward for all non-terminal states pushes the algorithm to find a short path to route through. In the simple worlds such as mundo_15_15 and mundo_maze, it performed exceptionally well. Time and time again, it would be plopped into the world and quickly identify its location, then start off towards the goal. When it spontaneously became uncertain about its position, potentially due to a bad sensor reading, it was able to still avoid obstacles and reach the goal. I would say the robots' performance in these worlds is comparable to that of a human.

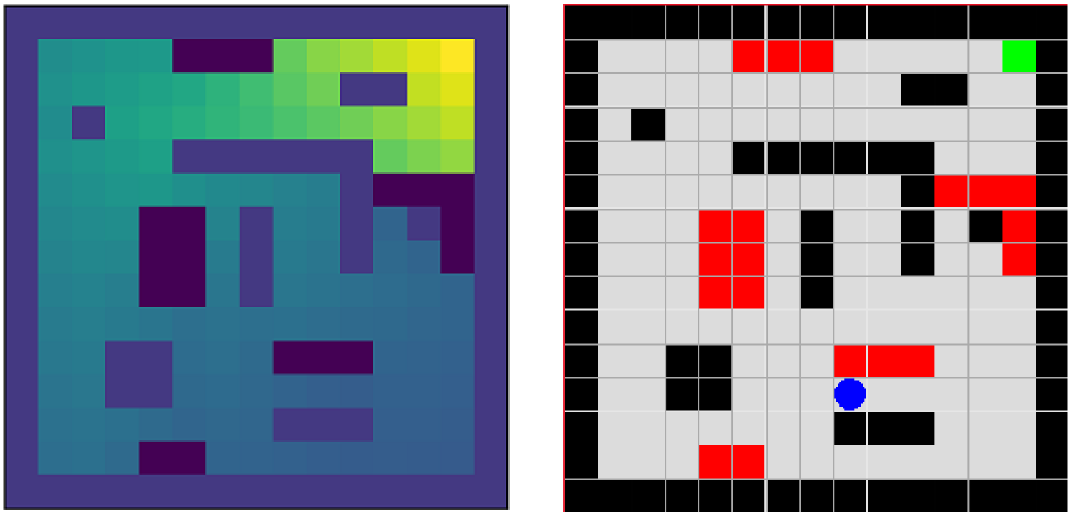
In the larger maze2 world, the robot struggled a bit more with moving towards the exit. It would frequently get caught in a small loop in the left side where it wouldn't go to the right spot immediately. I believe this behavior comes from the weakened gradient from the extra long path. As the path to the goal gets longer, the signal of the goal's reward has more and more noise competing with its subtle yet seductive call. This type of looping is shown in the maze_2 video. After a few extra moves of going in circles, the robot is usually able to get out of the rut and continue towards the goal state. In the video, you can see it get stuck in a few places, then cower from the death squares, then finally start moving towards the goal when its low accuracy transition model throws it down the stairway. Overall, the robots' performance in this world was noticeably worse than that of a human due to its rough behavior in a few particular places.

I found that the robots survival and efficiency rates were directly tied to the accuracy of the transition/sensor models. When I decreased the accuracy of the transition model, the robot would act more cautiously, decreasing efficiency, and it would also throw itself down the stairs more frequently, decreasing survival rates. In the limit, having complete information led to a very efficient and safe robot, while having little to no information led to poor performance in both survival and efficiency. This is a great application of the saying 'garbage in, garbage out.' Only when we provide the robot with the information it needs to make good choices, is it able to achieve its goals. I think there's a point to be made here about education, but I'm not exactly sure.

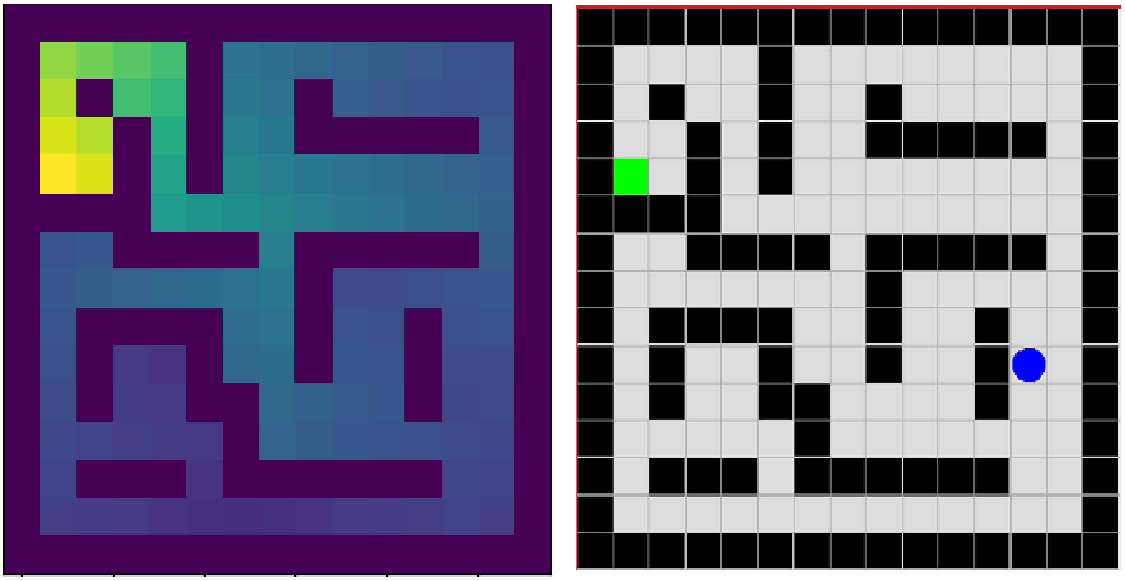
Video Links;
maze certain https://youtu.be/8AIvoQ8Bfno
Maze uncertain https://youtu.be/kXy-8P9nsqE

Maze 2 certain https://youtu.be/ngW0HfccKEM
Maze 2 uncertain https://youtu.be/ZKn4gv7nyjs

mundo_15_15.txt



mundo_maze.txt



mundo_maze2.txt