Thomas Pollard
CS – 441 Artificial intelligence, Winter 2021
Programming 3 Report

AI – Programming Assignment 3

Overview

“Programming 3 – Thomas Pollard.py” is my implementation of a Q Learning algorithm that teaches “Robby the Robot” to pick up cans in a 10 x 10 grid. Robby can move in any of the four cardinal directions or pickup a can in his current space. Moving into a wall will yield -5 reward and attempting to pickup a can in an empty square will yield -1 reward. He can also sense if there is a can or wall in his current square or in the squares directly adjacent in the four cardinal directions. His Q-matrix initially starts with all zeros. In each state, he chooses the action that has the highest potential return. There is also a hyper parameter epsilon. If there is a tie in potential return or at probability of epsilon, he will choose a random action. There is also a learning rate hyperparameter eta 0f 0.2 and a discount rate of 0.9. After each step, his Q-matrix is updated based on the reward received from the last action.

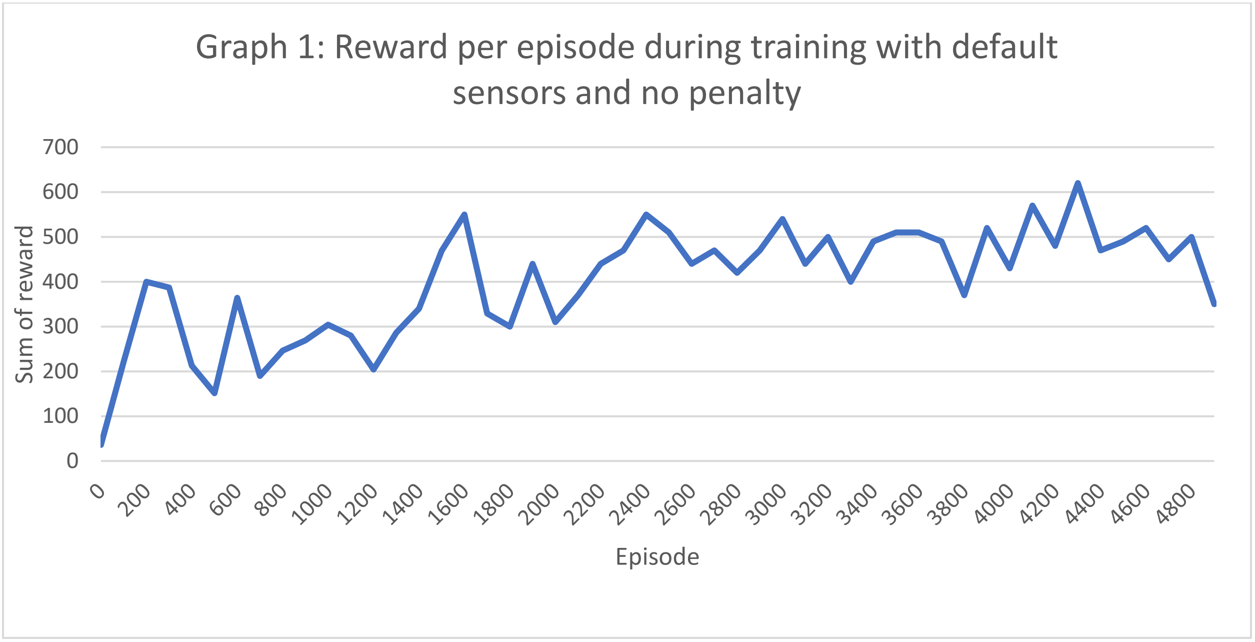
Robby is run for 5000 training episodes of 200 steps each. I plotted a graph of his episode reward for each training episode. After his training episodes, he is run for 5000 test episodes with an epsilon of 0.1 and his score averaged over all episodes. I included the spreadsheet of data as an external file in the submission called “programming3charts.xlsx”. There are 4 experiments I decided to try. For the first part, I showed the results of Robby with the default values as a baseline. For the second part, I gave Robby a penalty of 0.5 for every step to see if that would affect his learning curve or his average test run score. Third, I modified his sensors to give him vision two spaces away in each of the cardinal directions. For the fourth experiment, I doubled the number of training episodes for experiment 3 because there are four more sensors than usual so I hypothesized it would take longer to acquire an optimal solution.

Thomas Pollard
CS – 441 Artificial intelligence, Winter 2021
Programming 3 Report

Part 1 – Control Robby

Attributes:

- Training episodes: 5000
- Steps per episode: 20s0
- Eta: 0.2
- Eta decay: -0.0025 per 50 episodes
- Discount rate: 0.9
- Uniform penalty for each action: 0



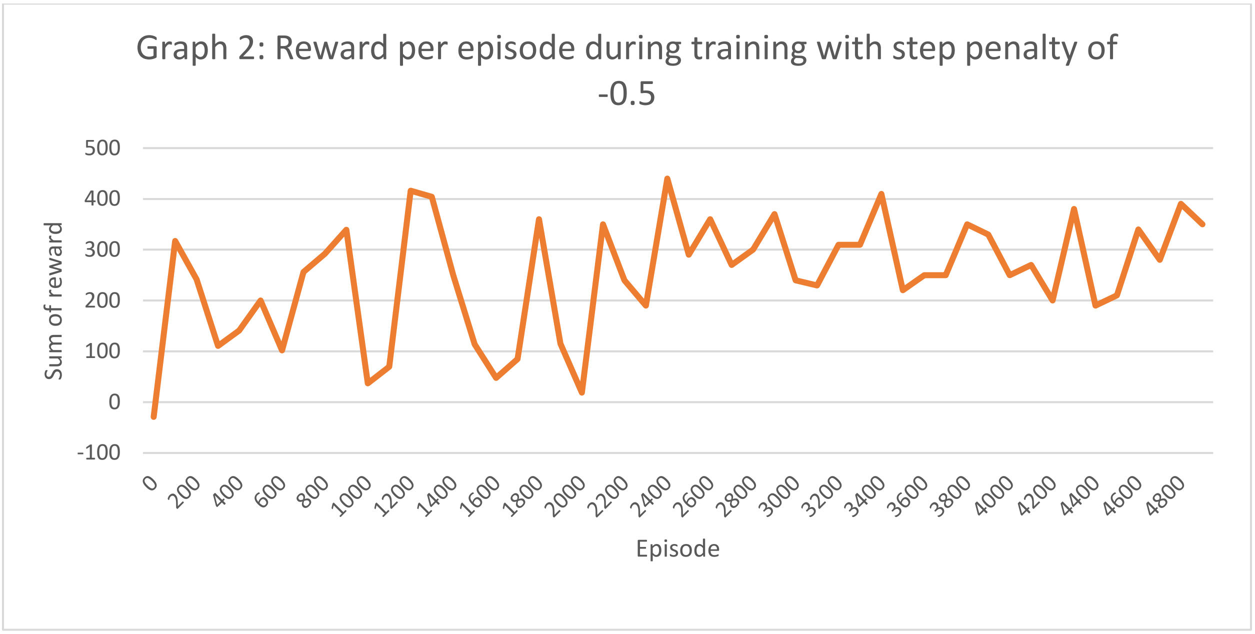
Graph 1 shows the plots of the rewards at every 100 training episodes for part 1 and the data is in page "control" of the spreadsheet. The 5000 test episodes had an average sum of rewards per test episode was 438.1458 and the standard deviation of sum of rewards per test episode was 49.71932967.

Thomas Pollard
CS – 441 Artificial intelligence, Winter 2021
Programming 3 Report

Part 2 – Robby with uniform action penalty

Attributes:

- Training episodes: 5000
- Steps per episode: 200
- Eta: 0.2
- Eta decay: -0.0025 per 50 episodes
- Discount rate: 0.9
- Uniform penalty for each action: -0.5



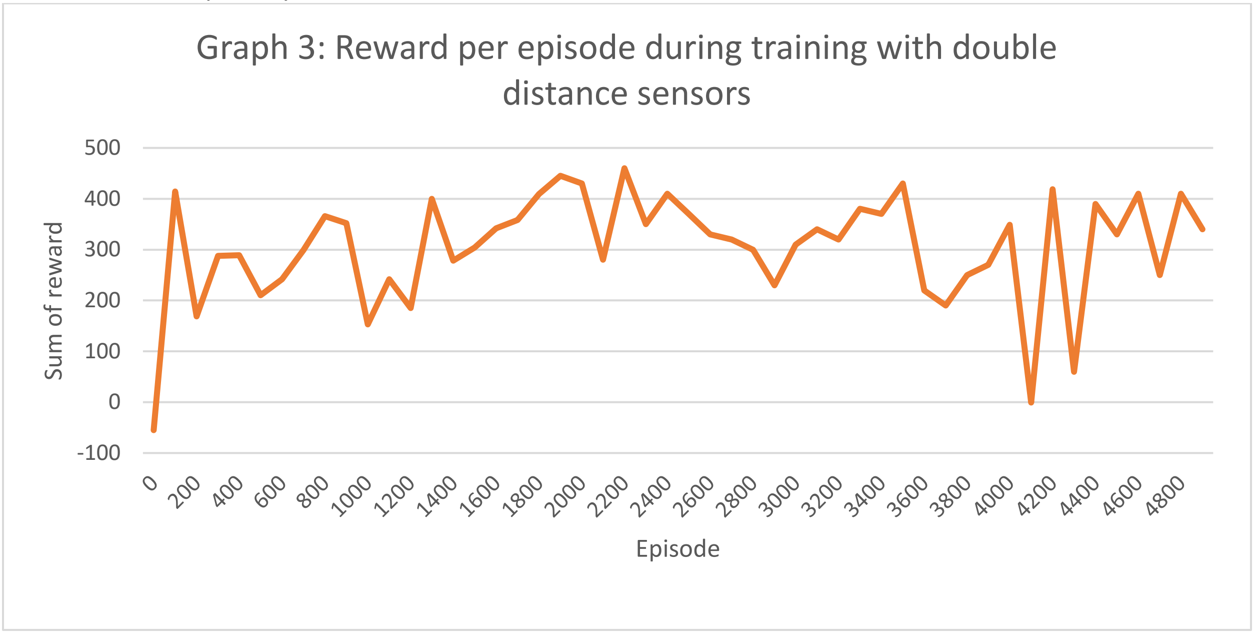
Graph 2 shows the plots of the rewards at every 100 training episodes for part 2 and the data is in page "With penalty" of the spreadsheet. The 5000 test episodes had an average sum of rewards per test episode was 257.3808 and the standard deviation of sum of rewards per test episode was 87.510. Because the penalty gave Robby -100 points each episode regardless of actual performance, I am adding 100 to the average to give it a fair comparison to part 1. The adjusted average was 357.3808.

Thomas Pollard
CS – 441 Artificial intelligence, Winter 2021
Programming 3 Report

Part 3 – Robby with double distance sensors and uniform action penalty

Attributes:

- Training episodes: 5000
- Steps per episode: 200
- Eta: 0.2
- Eta decay: -0.0025 per 50 episodes
- Discount rate: 0.9
- Uniform penalty for each action: -0.5



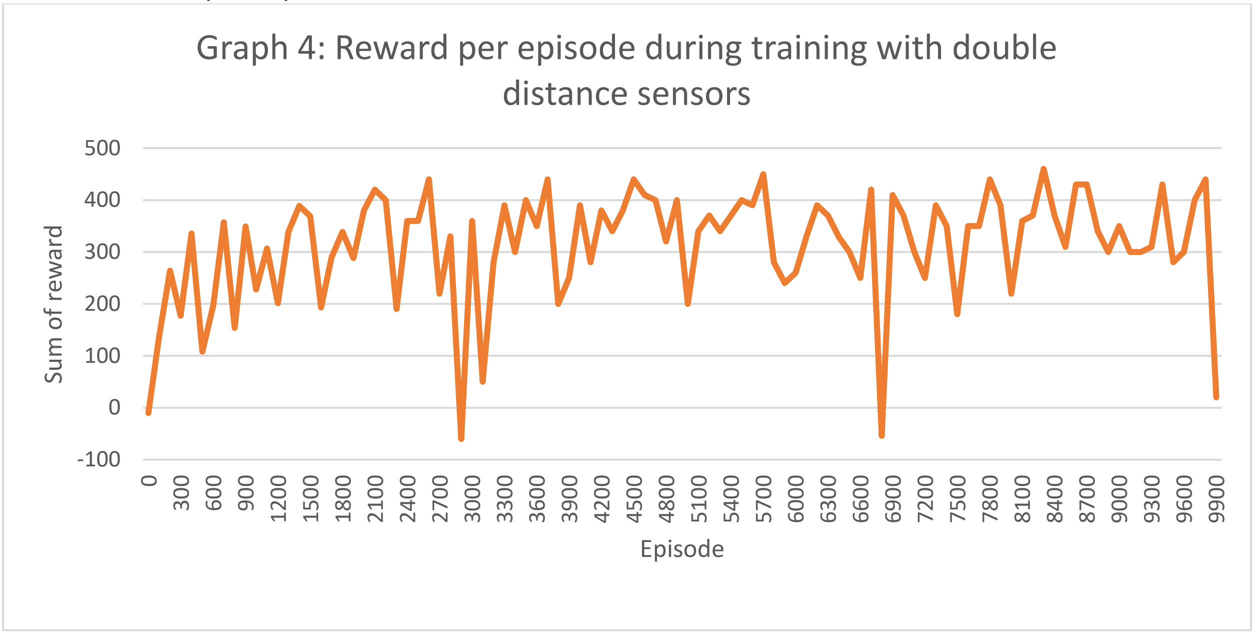
Graph 3 shows the plots of the rewards at every 100 training episodes for part 3 and the data is in page "Double sensors" of the spreadsheet. The 5000 test episodes had an average sum of rewards per test episode was 362.684 and the standard deviation of sum of rewards per test episode was 61.022. I included the uniform action penalty because without it, the increased observation space was too large to randomly explore and I was seeing a really long time before the strategy converged. Because the penalty gave Robby -100 points each episode regardless of actual performance, I am adding 100 to the average to give it a fair comparison to part 1 and 2. The adjusted average was 462.684.

Thomas Pollard
CS – 441 Artificial intelligence, Winter 2021
Programming 3 Report

Part 4 – Robby with double distance sensors, uniform action penalty and double training time

Attributes:

- Training episodes: 10000
- Steps per episode: 200
- Eta: 0.2
- Eta decay: -0.0025 per 50 episodes
- Discount rate: 0.9
- Uniform penalty for each action: -0.5



Graph 4 shows the plots of the rewards at every 100 training episodes for part 4 and the data is in page "Double sensors extra time" of the spreadsheet. The 10000 test episodes had an average sum of rewards per test episode was 333.6788 and the standard deviation of sum of rewards per test episode was 61.022. I included the uniform action penalty because without it, the increased observation space was too large to randomly explore and I was seeing a really long time before the strategy converged. Because the penalty gave Robby -100 points each episode regardless of actual performance, I am adding 100 to the average to give it a fair comparison to part 1 and 2. The adjusted average was 433.6788.

Thomas Pollard
CS – 441 Artificial intelligence, Winter 2021
Programming 3 Report

Conclusion

Table 1: This table shows the results of all four parts of the

Part	Average test episode score	Average test episode std dev
1	438.1458	49.7193
2	357.3808	87.5108
3	462.684	61.0218
4	433.6788	67.3873

As table 1 shows, experiment 3 had the highest average score per test episode. The control Robby had the second highest, which surprised me. Adding a penalty did not always have a strictly positive effect on score, but it did seem to make the algorithm start off in the right direction a little sooner. Also, training for longer did not have the intended effect of increasing score, but instead lowered it by a little. Adding sensors did seem to have a positive effect, but it made many more mistakes and I think that is because the observation state is so much larger and it will take even more training episodes to reach an optimal strategy.