Project 2 Reinforcement Learning for Robot Wall Following Report

Quinn Vo

***Abstract*— This electronic document is a live template. The various components of your paper [title, text, heads, etc.] are already defined on the style sheet, as illustrated by the portions given in this document.**

I. INTRODUCTION

The goal of this project is to implement reinforcement learning algorithms in order to get a robot to follow a wall. The robot determines its state by using lidar data; the robot is given a set of actions that it can perform. Finally, the robot is given a reward following the execution of the action. Using these things, the robot (given enough time) will eventually be able to accurately follow a wall. For this project, only right-side wall following was implemented to limit the scope. The requirements were that the robot must be able to do the following:

1) Follow a straight wall
2) Turn left 90 degrees at an L-shaped corner
3) Turn 180 degrees at an I-shaped corner
4) Turn right 90 degrees at an L-shaped corner
5) Turn 180 degrees at a U-shaped corner

Due to this extensive list of requirements, the time it took to train this robot took a long time, which ultimately turned out to be successful for both implementations of Q-learning and SARSA.

II. DESIGN

A. Implementation

For both Q-learning and SARSA, everything was pretty much implemented the same way except for how table entries were updated. The termination condition for both algorithms were to terminate once the robot was trapped or stuck for 3 consecutive steps. The other termination condition was if the robot had executed 10000 steps.

The only difference between Q-learning and SARSA is that SARSA knew of the next action that the robot would take and used it to update its table entry. On the other hand, Q-learning updated by selecting the maximum value. This makes Q-learning known as off-policy TD control and SARSA known as on-policy TD control.

B. Division of Sensing

Sensing field of view was divided as follows: the left region of the robot had 2 distances (CLOSE, FAR); the front region of the robot had 4 distances (TOO_CLOSE, CLOSE, MEDIUM, FAR); the right-front region of the robot had 2 distances (CLOSE, FAR); finally, the right region of the robot had 5 distances (TOO_CLOSE, CLOSE, MEDIUM, FAR, TOO_FAR). A visualization of this is shown in Figure 1.

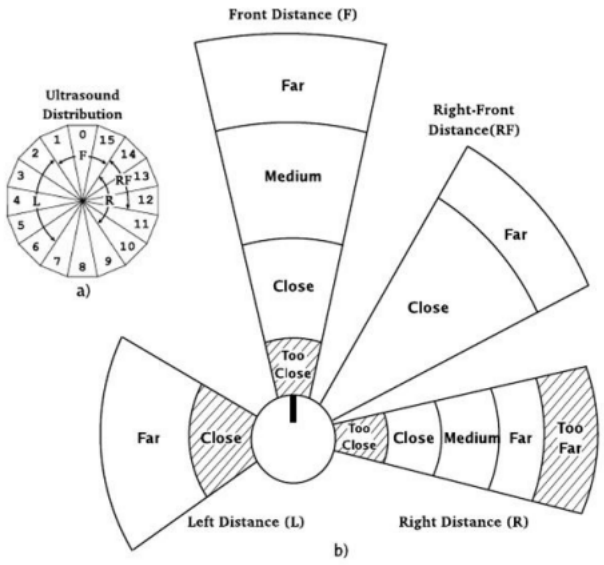

Fig. 1. Division of Sensing Field of View

C. State, Action, and Reward

A combination of each region's distance forms a state. There were 3 actions defined. Action 0 represented forward movement while turning to the left slightly. Action 1 represented strictly forward movement. Action 2 represented forward movement while turning to the right slightly. For the reward function, the robot was rewarded -1 if the right region was TOO_CLOSE, the right region was TOO_FAR, the front region was TOO_CLOSE or the left region was CLOSE. The robot was awarded 0 on any other situation.

D. Definition and Initialization of Q-table

The Q-table had 240 entries. There are 80 combinations of the distances from each region, and there are 3 actions. All entries were set to 0 when the robot is set to learning mode.

E. Model Parameters

Model parameters were kept at the recommended levels. Learning rate was kept at $\alpha = 0.2$; discount factor was kept at $\gamma = 0.8$; epsilon was a function of the episode number. The function for epsilon was $\epsilon = \epsilon_0 * d^n$ where n is episode number. ϵ_0 was set to 0.9, and d was set to 0.985.

III. EXPERIMENT

A. Experimental Results

As shown in the Figure 2, as the robot acculumates a negative reward as the number of episodes increases. Because the reward function was poorly defined, it is not possible to see convergence, because the accumulated reward will always decrease. The reward function only returned a 0 or a -1. However, these quantitative results confirm that SARSA is indeed the better algorithm for this type of problem. SARSA allowed the accumulated reward to decrease at a slower rate than Q-learning.

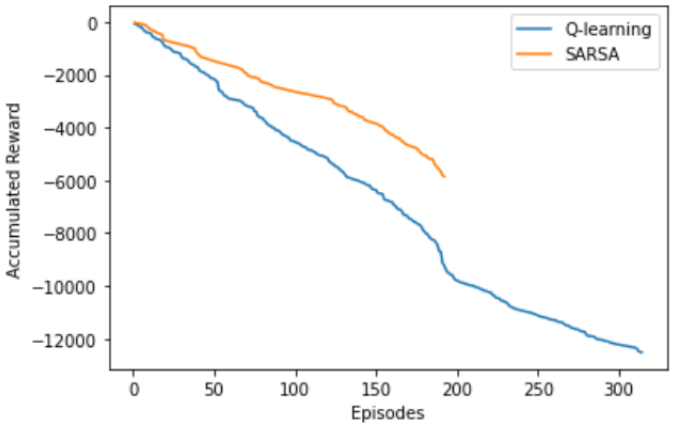

Fig. 2. Number of Episodes Vs Accumulated Reward

B. Qualitative Results

Using both Q-learning and SARSA algorithms, the robot was successfully able to address all five situations as required. Snapshots of each of the 5 situations are shown in the figures 3, 4, 5, 6, and 7.

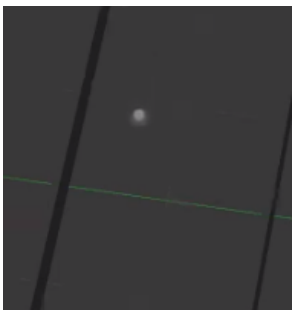

Fig. 3. Robot Following a Straight Wall

Fig. 4. Turn Left 90 degrees at an L-shaped Corner

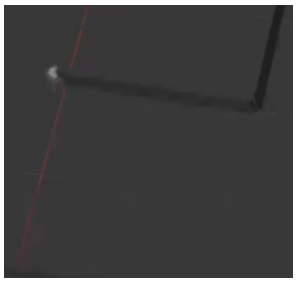

Fig. 5. Turn 180 Degrees at an I-shaped Corner

Fig. 6. Turn Right 90 degrees at an L-shaped Corner

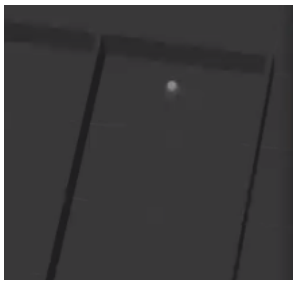

Fig. 7. Robot Turning 180 Degrees at a U-shaped Corner

C. Comparison of Q-learning and SARSA

Using SARSA as the algorithm for reinforcement learning, it was considerably faster than Q-learning. SARSA took approximately 192 episodes in order to get reliable results. On the other hand, Q-learning had to be trained for 314 episodes. Even with 314 episodes, it would sometimes veer

off course. Q-learning had to be trained for about 8 hours; SARSA took only about half that time (4 hours). This is likely because SARSA does not allow much room for exploration, allowing table values to converge much faster.

IV. CONCLUSIONS

The qualitative results indicate that the reinforcement learning algorithms for Q-learning and SARSA were very successful. In future work, the reward function should be defined in a way to easily see convergence in the quantitative results. Having a reward function that returns 0 or -1 is not helpful to visualize the progress of the robot's learning. This is because the robot's total accumulated reward would only decrease and it looks like the robot is not learning. In reality, the robot performed very well. Finally, future work could involve defining more actions that the robot can perform such as backwards movement and turning.