Harjot Mangat

EECS269 – Reinforcement Learning

Final Project Report

For this project, I implemented a Tabular Dyna-Q algorithm to interact with the racetrack environment. I used this algorithm because of the combination of planning and experience to help speed up the time to find the solution. By default, the program runs for 500 episodes, has an alpha of 0.5, discounting of 0.9, and an epsilon of 0.05. I followed the pseudocode in the book for the implementation of the algorithm. The action is selected epsilon-greedily with respect to the max Q value for that state. Then, observe the reward and next state from the selected state and action, and update the Q values with the Q-learning update formula. For the model(s,a) and planning loop, I used the 'sample_new_state' method to query the reward and next state of a randomly selected state and action from a list of visited states. I know that the method is not completely accurate, but I used it for the simplicity, and it is good enough as an approximation. The default max number of planning steps in my program is 25, the planning loop begins with the minimum(length(observed states),25). This means that for the early episodes, the program only uses a few planning steps, as the number of states visited increases, the planning steps increase until they reach the max number of planning steps 25. There is a graph of rewards per episode printed at the end of the program. This graph is saved as 'Dyna-Q_racetrack.png' to the directory of the program.