CS 3600 Project 2 Wrapper

CS 3600 - Spring 2022

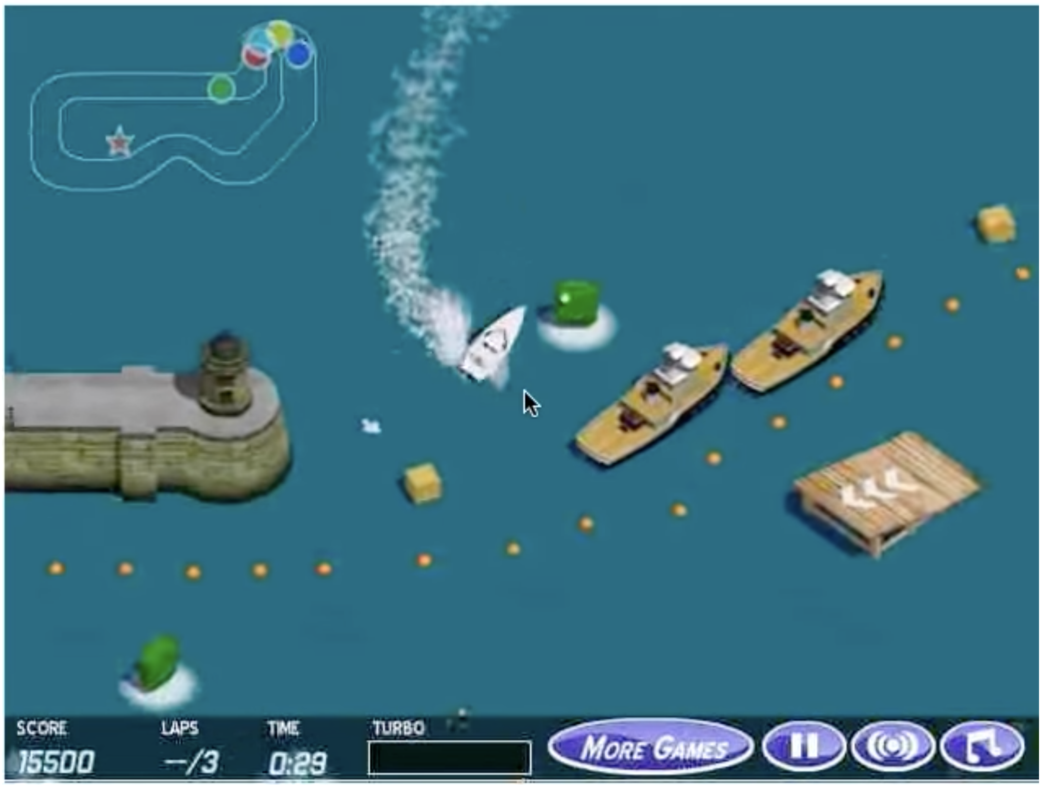
Due March 6th 2022 at 11:59pm EST via Gradescope

Introduction

This Project Wrapper is composed of 4 questions, each worth 1 point. Please limit your responses to a maximum of 200 words. The focus of this assignment is to train your ability to reason through the consequences and ethical implications of computational intelligence, therefore do not focus on getting ”the right answer”, but rather on demonstrating that you are able to consider the impacts of your designs.

Context

Reinforcement learning is a powerful technique for problem-solving in environments with stochastic actions. As with any Markov Decision Process, the reward function dictates what is considered optimal behavior by an agent. Since a reinforcement learning agent is trying to find a policy that maximizes expected future reward, changing when and how much reward the agent gets changes its policy.



However, if the reward function is not specified correctly (meaning rewards are not given for the appropriate actions in the appropriate states) the agent's behavior can differ from what is intended by the AI designer. Consider the boat racing game pictured above. The goal, as understood by people, is to quickly finish the race. Humans have no difficulty playing the game and driving the boat to the end of the course. However, when a reinforcement learning agent learns how to play the game, it never completes the course. In fact, it finds a spot and goes in circles until time runs out. You can see the RL agent in action in this video: https://youtu.be/tlOIHko8ySg.The agent's reward function is the score the player receives while playing the game. Score is given for collecting power-ups and doing tricks, but no points are given to players for completing the course.

Question 1

Watch the video and explain why the agent's policy has learned this circling behavior instead of progressing to the end of the course like we expect from a human player. Explain the behavior in terms of utility and reward.

Answer: The reason we could see the agent exhibiting this circling behavior is because the agent seeks state transitions that maximize utility, which in this case involves the score that the agent is accumulating. As stated in the prompt, there are points given for power-ups and tricks but not for reaching the end of the course, meaning there are only a couple of ways for the agent to receive a reward for executing actions. This reward increases the agent's utility in a state. This is why the agent circles. It is continuing to maximize utility by trying to get as many power-ups and perform as many tricks as possible, meaning the agent is happy to circle and sees no benefit in reaching the end.

Question 2

When humans play, the rules for scoring are the same. Why do humans play differently then, always completing the course? Why don't humans circle in the same spot in the course endlessly if they are receiving the same score feedback as the agent?

Answer: This is because we humans understand the premise of the game and understand that there is some goal we need to achieve. We know that we need to reach the finish line first in order to win, and that the score is just an additional metric. These agents don't have this type of intuition and rely on numbers to both tell it what it should be doing and evaluate its overall performance.

Question 3

The agent's original reward function is:

$$R(s_t, a) = game\_score(s_t) - game\_score(s_{t-1})$$

Describe in terms of utility, reward, and score **two** ways one could modify the reward function to get the agent to behave more like a human player. That is, what do we need to change to make the agent complete the course every single time? Assume the agent has access to state information such as the position and speed of the boat and all rival racers, but we cannot change how the game itself provides scores through the call $game\_score(s_t)$.

Answer: For the first method, I would subtract from the reward function the distance a state is from the current player in first place. This means that states that lead the agent farther away from where the agent would ideally be would have lower utility. This way the agent is likely to progress or transition to states closer to where we want to be. This means that we will always finish because the other competitors will always finish and we are "attracted" to those states. Another method would be to add on reward for being in increasingly high positions within the finishing places. This would mean states that would either maintain or increase finishing position would be preferred over the ones that don't. This ensures that our agent travels a path that at the least follows the other competitors, ensuring that our agent finishes the race.

Question 4

Self-driving cars do not use reinforcement learning for a variety of reasons, including the difficulty of teaching RL agents in the real world, and the dangers of a taxi accidentally learning undesired policies as we saw with the boat game example. Suppose however, that you tried to make a reinforcement learning agent that drove a taxi. The agent is given reward based on how much fare is paid for the ride, including tips given by the passenger. Describe a scenario in which, after the taxi agent has learned a policy, the autonomous car might choose to do an action that puts either the rider, pedestrians, or other drivers in danger.

Answer: Given that the reward is partly based on maximizing the fare to some degree, there is not a whole lot of incentive to go quickly. Let's imagine a scenario where the taxi finds itself in the left lane on a busy highway. Because there is no real incentive to go fast, the taxi could be traveling at a lower speed, especially in comparison to what others would be traveling at in the left lane. This behavior itself is dangerous for the rider and the other drivers because it results in more lane changes for other cars and increases the potential that another car is traveling too far to stop for the taxi.