Targeting and Obstruction Avoidance using RL

Matthew Herndon, Raymond Haynes, and Stuart Edris

September 2020

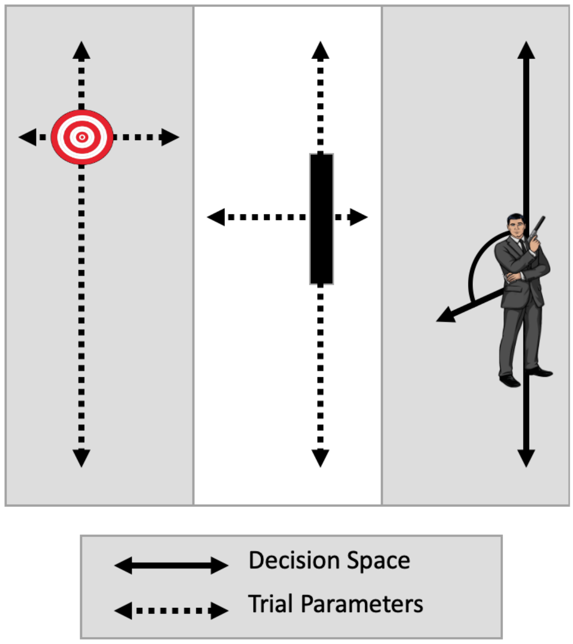
1 Introduction

For our **short project**, we will develop reinforcement learning models to automate a simple game in which the player must attempt to hit a target object placed within an environment wherein a barrier object may or may not be blocking the target.

2 Domain

The simulation environment will be be a square, two dimensional grid with N subdivisions. This region will be divided into thirds to define three distinct regions which are populated with an object randomly. In the leftmost third, a target enclosing S_T grid positions is placed at a random position. Similarly, in the middle third of the region zero or more obstruction(s) are placed. In the rightmost third, a player armed with a gun is placed randomly along a vertical line with a random aiming direction within 90°and 270°with respect to the horizontal.

For a period of time T_0, the player is tasked with adjusting their aim and position. At each time step t_n, the player may take one of several actions—they may move either up or down, or they may adjust aim in the positive or negative direction. All of these actions are discretized such that the agent may only move in steps. Once this is completed, a projectile is fired along a straight-line path from the player's position and the trial is scored. The goal of the game is for the player to hit as close to the center of the target as possible, with higher accuracy (closer to the target's center) resulting in higher scores. If the projectile misses the target completely, no reward is granted.



3 Hypotheses

In this project, reinforcement learning will be used to generate models capable of automating the player from the game described above. We propose the following hypotheses to evaluate the effectiveness of the trained models, where 'hit rate' is defined as the number of trials out of N=20 in which the projectile intersects with the target:

1. A model trained using Q-Learning will achieve an average target hit rate at or exceeding 90% without an obstacle. That is, it can learn to aim effectively given a random placement of the target.
2. The model trained using Q-Learning will achieve an average target hit rate at or exceeding 75% with an randomly placed obstacle(s). That is, the model can learn to move around an obstacle if necessary to hit the target.
3. A model trained using TD Learning will achieve an average target hit rate at or exceeding 90% without an obstacle.

4. The model trained using TD Learning will achieve an average target hit rate at or exceeding 75% with an randomly placed obstacle(s).

5. Models trained using Q-Learning will achieve a higher average hit rate than models trained using TD Learning.[1]

6. It is possible for a model trained to 1 obstacle to be applied to other environments (e.g., 0 obstacles, 2 obstacles, 3 obstacles, etc.) while retaining a decent (>50%) average target hit rate.

4 Work Breakdown

Given that the code work for the game environment does not exist (to the authors' knowledge), the first week is planned as a joint work between all members to create the environment. Upon completion, Stuart will work on the Q-Learning model and attempt to adapt to cases with and without an obstruction. Matthew will develop the TD Learning model and will also attempt to adapt it to both cases (with and without an obstruction). In all four cases, the models will be run until the desired accuracy is hit, or until a set number of iterations, in the event of time constraints. In each case, the models will be run multiple times each to determine the average accuracy, standard deviation, computation time or iterations required to reach the desired accuracy. These values will be compared at the end to determine the better performing model for this game. Raymond will train a Q-Learning model to 1 obstacle environment and apply that model to other environments with 0, 2, or more obstacles to determine applicability of the model.

Task	Matthew Herndon	Stuart Edris	Raymond Haynes
Simulation Environment	✓	✓	✓
Implement Q-Learning Algorithm		✓	
Implement TD Learning Algorithm	✓		
Model Versatility Test			✓
Analysis and Reporting	✓	✓	✓

Table 1: Task designation

[1]This prediction was chosen arbitrarily due to a lack of knowledge—we plan to find the most effective algorithm by testing this hypothesis.