Reinforcement Learning for Missile Evasion and Guidance Using Deep Neural Network(DQN)

THE UNIVERSITY OF
SOUTHERN
MISSISSIPPI®

Project

Submitted by:

Anil Katwal

December 12, 2024

Contents

Chapter 1

Introduction

Accurate algorithms are essential for guidance systems for missiles and other modern defense that are supportive of the tracking and interception of a moving target. Hence, this report looks into the utilization of reinforcement learning to create a smart guidance algorithm. Such optimization is possible by interacting with the environment and receiving feedback from it. Beforehand, discrete state and action spaces are dealt with using Q-learning with an established Q-table. Because of limits on scope, a neural network-based Deep Q-Network (DQN) is available that has good performance on a vast range of state-action space. This project describes the evolution from using simple models of missiles to advanced algorithms while using aerodynamic equations to construct the missile flight dynamics. This report is Structured in fallowing order.

- Chapter 2 explains the custom environment designed for missile guidance and evasion.
- Chapter 3 provides over view of DQN, including its mathematical foundations.
- Chapter 4 outlines the implementation details of the simulation and agent.
- Chapter 5 presents the results and analysis.
- Chapter 6 concludes the report with potential directions for future work.

1.1 Environment Description

The simulation environment, referred to as *MissileEnv*, is implemented using OpenAI's Gym framework. It models the interaction between a missile and an interceptor. The environment consists of:

- State Space: Position and velocity information of both the missile and the interceptor.
- Action Space: Discrete actions allowing the missile to adjust its heading (turn left, right, or maintain course).
- Reward System: Encourages hitting the target while avoiding interception.

1.1.1 State Space Representation

The state space is defined as:

$$\text{State} = \begin{bmatrix} x_m & y_m & x_i & y_i \end{bmatrix}$$

where:

- x_m, y_m: Missile position.
- x_i, y_i: Interceptor position.

The state space allows the agent to observe the relative positions of the missile and the interceptor.

1.1.2 Action Space

The action space consists of three discrete actions:

1. No adjustment (continue current trajectory).
2. Adjust heading left by 10 degrees.
3. Adjust heading right by 10 degrees.

1.1.3 Mathematical Equations for Dynamics

The missile's position is updated using the following equations:

$$x_m(t+1) = x_m(t) + v_m \cos(\theta_m)$$

$$y_m(t+1) = y_m(t) + v_m \sin(\theta_m)$$

where:

- v_m: Missile speed.
- θ_m: Missile heading angle.

The interceptor updates its heading to track the missile:

$$\theta_i = \arctan\left(\frac{y_m - y_i}{x_m - x_i}\right)$$

The interceptor's position is updated similarly:

$$x_i(t+1) = x_i(t) + v_i \cos(\theta_i)$$

$$y_i(t+1) = y_i(t) + v_i \sin(\theta_i)$$

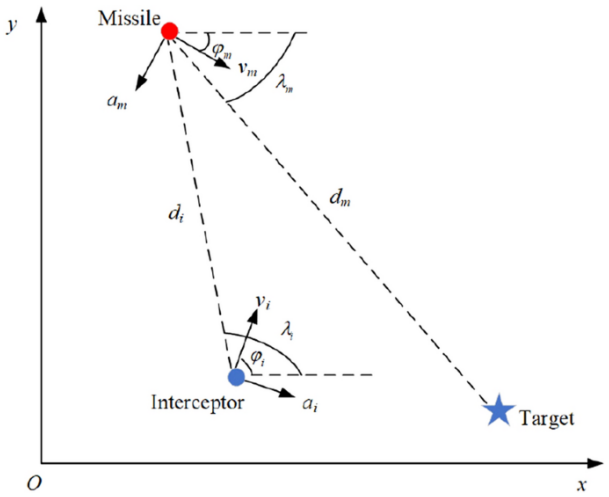


Figure 1.1: Actual trajectory and parameter of missile among these parameter only 4 state space is consider.

1.2 Q-Table Implementation

Q-learning involves maintaining a Q-table that stores the expected reward for taking each action in a given state. The Q-value update rule is:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$$

where:

- α: Learning rate.
- γ: Discount factor.
- r: Reward received.

1.2.1 Challenges of Q-Table

As the state space becomes continuous or high-dimensional, storing and updating the Q-table becomes computationally infeasible. This limitation motivates the use of function approximators like neural networks.

1.3 Transition to Neural Networks

Deep Q-Networks (DQN) address the scalability challenge by approximating the Q-value function using a neural network:

$$Q(s, a; \theta) \approx Q(s, a)$$

where θ represents the network weights.

The DQN uses experience replay and a target network to stabilize training. The loss function is:

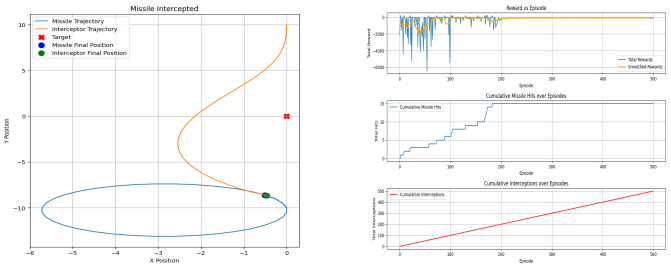
$$L(\theta) = \mathbb{E}_{(s,a,r,s')\sim D}\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta') - Q(s, a; \theta)\right)^2\right]$$

1.4 Results and Analysis

1.4.1 Performance Metrics

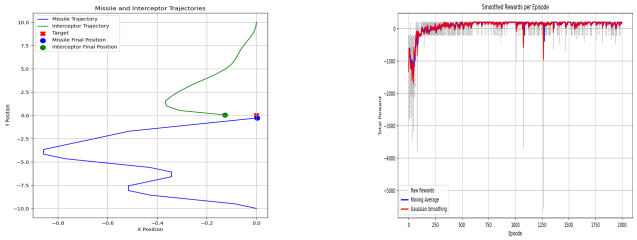
Performance is evaluated based on:

- Total rewards per episode.
- Successful hits on the target.
- Avoidance of interception.



(a) (b)

Figure 1.2: Interceptor velocity and missile velocity are the same at 500 episodes.



(a) (b)

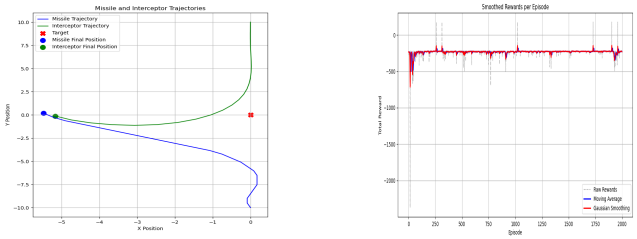
Figure 1.3: Missile velocity and interceptor velocity are the same, but episodes have increased(episodes=2000).

1.4.2 Trajectory Visualization

Figure 1.2 shows the missile and interceptor trajectories for the 500 episode. The interceptor successfully intercept the target.

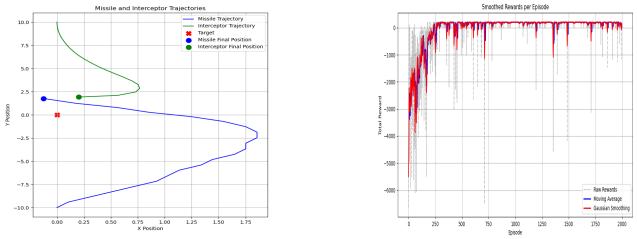
We examined how the plots for both the trajectory and reward change for various scenarios.

In the Figure 1.2 trajectory, it is shown that the missile was successfully intercepted by the interceptor. The limited amount of episodes meant that the interceptor wasn't overly punished, and therefore the missile also did not have enough opportunities to learn more about the surroundings and instead focused more on exploitation rather than exploration.



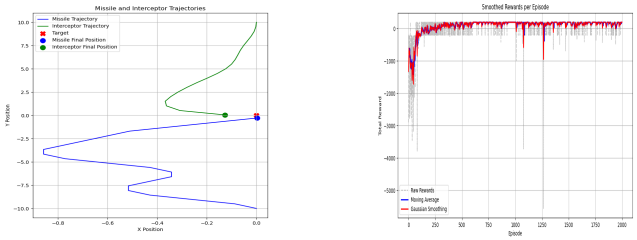
(a) (b)

Figure 1.4: Interceptor velocity is higher, but the interceptor is highly penalized(episodes=2000).



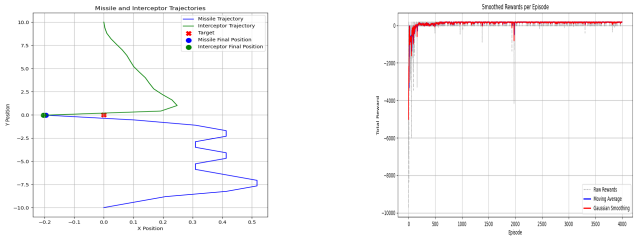
(a) (b)

Figure 1.5: Intercepter velocity is higher but interceptor is highly penalized(episodes=2000).



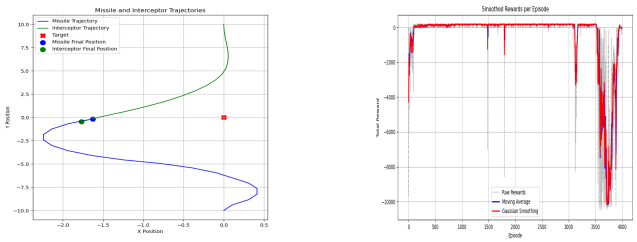
(a) (b)

Figure 1.6: Missile velocity and interceptor velocity are the same, but episodes have increased(episodes=4000).



(a) (b)

Figure 1.7: Missile velocity and intercepter velocity same at (episodes=4000).



(a) (b)

Figure 1.8: Missile velocity and intercepter velocity same at (episodes=4000).

In the Figure 1.3, along with the increase in the number of epochs, missile hits also increased as well as interceptions by the interceptor. This trajectory highlights the fact that the missiles' intercepting capabilities increased with passing time as well as its intelligence.

In the Figure 1.4 there is an increased velocity of the missile interceptor represented. The advantage was made; however, it is apparent that even with this increased speed the interceptor was not able to intercept the missile' although it was still able to divert a direct hit to its target. With the increase in episodes, there was also indication that the missile was learning more about avoiding interceptions. Hitting the target with great velocity turned out to be a success for the missile, however only when the conditions were favorable and didn't include factors such as poor weather or restricted visibility which would make interception hard for the interceptor.

Lastly, in Figure 1.6 to Figure 1.8 depicts the moments where the epochs moved up to 4000, Interceptor able to intercept the missile and safe target. But, to make successful defense mechanism still need to consider several factor such as movement of target, climate condition, land trophography.

1.5 Conclusion

This research shows reinforcement learning (RL) in the context of missile guidance, illustrating how Q-learning allows one to work with discrete state-action spaces and that the learning efficiency is increased with the application of Deep Q-Networks (DQN) in more continuous environments. A more recent RL paradigm, Proximal Policy Optimization (PPO), for instance, can do more than that for complex decision making. In particular, it is possible to use multi-agent systems with advanced rewarding systems taking into account environmental conditions like visibility, landscape and weather which would greatly enhance the model. Starting by considering the locations of missiles and interceptors, our methodology is such that it creates an initial architecture, which incorporates target dynamics and mixed action spaces, avoiding the constraints of purely discrete schemes. This multi-layered approach makes it easy to enhance the algorithms and provides a roadmap for effective and flexible solutions for interception tasks with varying complexity.

References

1. Yan, M., Yang, R., Zhang, Y., Yue, L., & Hu, D. (2024). A Hierarchical Reinforcement Learning Method for Missile Evasion and Guidance. *Journal Name*, X(Y), Z-Z.

2. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.

3. Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529-533.

4. OpenAI Gym Documentation. Available at: https://www.gymlibrary.dev/

5. LibGuides: Citation Help. Available at: https://libguides.lib.usm.edu/citation_help