Using Reinforcement Learning to Train In-game Non-Player Characters (NPCs)

Ransom Duncan
Computer Science Department
Michigan Technological University

***Abstract*—The video game industry is continuously evolving, driven by technological advancements, offering opportunities for more immersive gaming experiences. This research explores the integration of machine learning in game design, focusing on applications for in-game artificial intelligence (AI). Traditional game AI design relies on predefined algorithms and scripted behaviors. This often results in predictable AI and static game play. Here I investigate how machine learning can create AI systems that dynamically adapt to a player's skill level to enhance engagement and challenge. I first present a literature review to learn about current uses of machine learning in video games. Then I describe a prototype game using the Unity ML framework. I use the creation and success of this prototype to test the feasibility of incorporating AI into video game development.**

I. INTRODUCTION

As the gaming industry continues its pursuit of providing players with increasingly immersive experiences, the integration of cutting-edge technologies becomes not only an opportunity but a necessity. This research project delves into the realm of Machine Learning (ML) and its potential to redefine video game development, with a specific focus on in-game NPCs (non-player characters). In an era where player expectations are more sophisticated than ever, the conventional use of pre-defined algorithms and scripted behaviors for in-game entities has left a discernible void in the dynamic and adaptive nature of gaming experiences.

Machine learning is a branch of artificial intelligence that endows systems with the ability to learn and adapt autonomously [5]. ML holds the potential to revolutionize how NPCs respond to player actions. No longer bound by static algorithms, NPCs could evolve dynamically, providing players with experiences that are not only challenging but uniquely tailored to their skill levels.

The application of machine learning in video game development has the potential to overcome limitations imposed by traditional methodologies. By focusing on adaptability, responsiveness, and a nuanced understanding of player behaviors, machine learning could usher in a new era of gaming that captivates and challenges gamers. This paper describes for a proof-of-concept system for the integration of machine learning in video game development.

To achieve the objectives of this research, a multi-faceted methodology is employed. The initial phase involves a review of existing literature and research on machine learning in game design to elucidate best practices and identify challenges in implementing machine learning in the gaming context. Following this, a practical design approach is taken. This process involves the creation of a simple game prototype. To create this prototype the game development engine Unity will be used. The machine learning framework Unity ML will be used to train in-game entities. These entities will then be implemented into the game prototype. This will allow for the observation and analysis of their adaptive behaviors based on player performance.

II. TRADITIONAL NPCS VS AI NPCS

NPC design in video games relies on algorithms and scripted behaviors for in-game decisions and actions. However, these systems often result in predictable and static gameplay experiences, limiting the potential for player engagement. The crux of the issue lies in the need for in-game NPCs that can dynamically adapt to a player's skill level, providing a more personalized and challenging gaming experience. The goal is to observe the impact of machine learning algorithms on AI behavior, specifically in terms of how it can adapt to the player's skill level.

A. Traditional NPCs

Traditional NPCs have always been a keystone element for many games. These NPCs can be incredibly complex to the point of seeming like AI. Behind the scenes, however, they are all operating on predetermined algorithms and scripted behavior patterns created by the developers. These NPCs, while serving essential roles within games, often lack the adaptability and responsiveness necessary to create truly challenging gameplay. Their actions follow set patterns, leading to scripted interactions that players can learn and predict. This creates artificial limits on in-game difficulty. Experienced players will find these NPCs boring, with predictable patterns that are easy to beat. New players conversely will find the NPC's actions nonsensical and confusing. Causing frustration which may lead them to stop playing the game prematurely.

On the other hand, the emergence of AI has opened-up the possibility of creating AI-driven NPCs, leveraging advanced machine learning techniques. AI NPCs have the potential to be a new shift in gaming AI. This would offer a departure from the static and predictable behaviors of traditional NPCs. These AI-driven entities possess the capacity to learn and evolve based on interactions with players and the game environment.

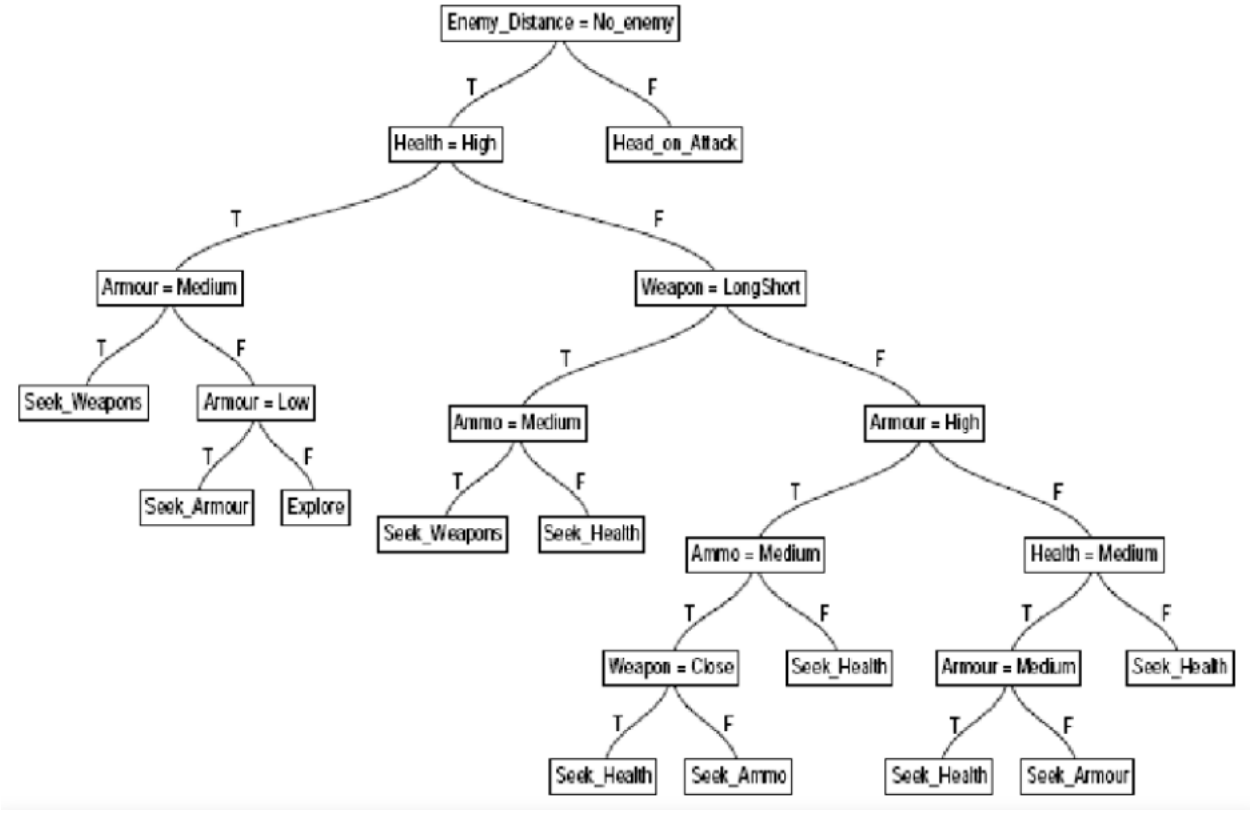

Fig. 1. Example of a traditional NPC decision tree [1].

B. AI NPCs

The hallmark of AI NPCs, particularly those utilizing reinforcement learning, lies in their adaptability and dynamic nature. Unlike traditional NPCs bound by predefined behaviors, AI NPCs can adapt their strategies and responses based on the player's actions. This adaptability introduces an element of unpredictability, making gaming experiences more engaging and challenging. The transition from traditional NPCs to their AI-driven counterparts signifies a shift from static to adaptive gameplay experiences. AI NPCs, empowered by reinforcement learning, continuously learn and evolve, mirroring the complexity and unpredictability of human behavior. This evolution allows for a more personalized and tailored experience for players, where challenges dynamically adjust to their skill levels and playstyles.

The integration of AI-driven NPCs in games heralds a new era where gaming experiences are not only immersive but also responsive and adaptive to individual players. By breaking away from rigidly scripted behaviors, AI NPCs enhance player engagement by offering unique and evolving interactions, contributing to the evolution of gaming landscapes.

III. REINFORCEMENT LEARNING

Machine Learning is "A machine that "learns" data and develops expertise on categorizing or garnering that knowledge over time" [3]. There are three main types of machine learning used to create AI systems. There is supervised learning, where the AI learns from a set of manually labeled data. Unsupervised learning, where the AI learns from a larger unlabeled data. And reinforcement learning, where the AI learns through a system of trial and error [5], [6]. We focus on the application of reinforcement learning as it is the most compatible with the game development.

The core of reinforcement learning is training an agent to make decisions by having that agent interact with an environment. The agent receives feedback in the form of rewards or punishments, allowing it to learn the optimal behavior over time [1]. The goal of reinforcement learning is to find a policy, strategy, or rule that maximizes the cumulative reward the agent receives. The reinforcement learning process typically involves the following steps: the agent makes an observation, the agent selects an action based on its observation, the agent takes its chosen action and then receives either a reward or a punishment, and the agent updates its observation policy based on its reward or punishment. This process is repeated iteratively, allowing the agent to learn how to maximize the reward it receives and create the desired behavior. This cycle is shown in Figure 2.

In the context of in-game AI, reinforcement learning empowers NPCs to learn from their experiences within the game world. These NPCs, acting as agents, make decisions and take actions based on their observations of the environment.

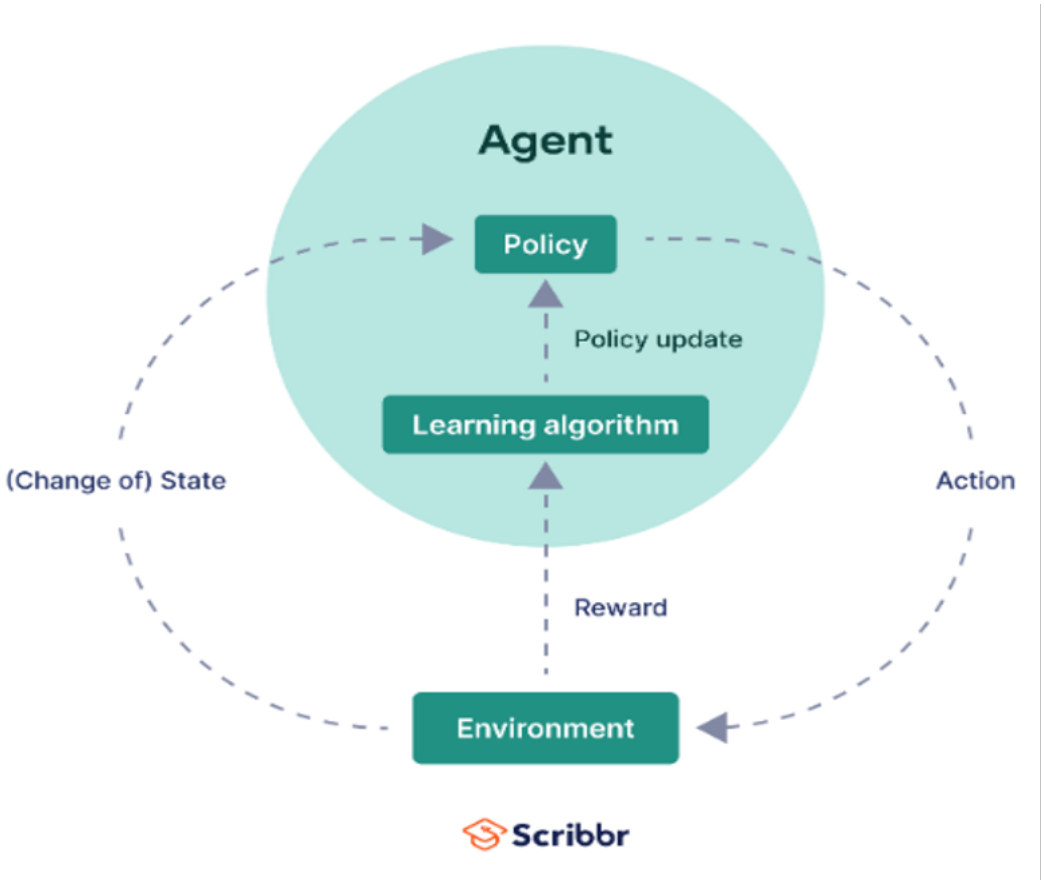

Fig. 2. General framework of reinforcement learning [4].

When an NPC's actions contribute positively to its objectives, it receives rewards. Conversely, detrimental actions result in penalties. Over successive iterations, NPCs adjust their decision-making processes to maximize rewards, evolving their behavior to align with desired gameplay dynamics. The cyclic nature of reinforcement learning involves the NPC observing the game environment, selecting actions, receiving feedback in the form of rewards or penalties, and subsequently updating its strategies. This iterative learning loop enables NPCs to dynamically adapt to player behaviors, leading to enhanced unpredictability and tailored challenges within the game.

Because video games are inherently an interactive environment, reinforcement learning is a tailor-made tool for creating adaptive and responsive AI-driven NPCs. It allows NPCs to evolve beyond static behaviors, providing players with dynamic and engaging experiences that evolve and adapt alongside their actions. When compared to traditional methods of NPC creation, reinforcement learning does not take more work to create, as long as a preexisting framework is being used.

IV. UNITY ML

Unity Machine Learning (ML) is a groundbreaking framework within the realm of game development. It serves as a leading example for the integration of advanced machine learning capabilities into games. This platform equips game developers with the tools necessary to implement and train AI agents, including NPCs. The utilization of Unity ML revolves around its generation of trainable agents. These agents allow developers to harness a variety of machine learning algorithms, notably reinforcement learning, to create NPCs with learning capabilities [8]. These NPCs can then be placed seamlessly into the finished game environment with all of their trained behaviors intact. By integrating Unity ML, game developers can introduce AI-driven entities into their game prototypes, fostering adaptive behaviors that respond dynamically to player interactions. A basic implementation of this framework is shown in Figure 3.

The framework provided by Unity ML streamlines the integration of machine learning into game development, offering a user-friendly interface and specialized libraries tailored to the needs of game developers [7]. This accessibility enables the creation of AI-driven NPCs that evolve, learn, and dynamically adjust their behaviors based on player engagement. Through Unity ML, the synergy between game development and cutting-edge AI integration becomes more attainable. Empowering developers to create gaming environments where NPCs evolve and adapt. This leads to richer, more engaging player experiences. The framework encapsulates a potential

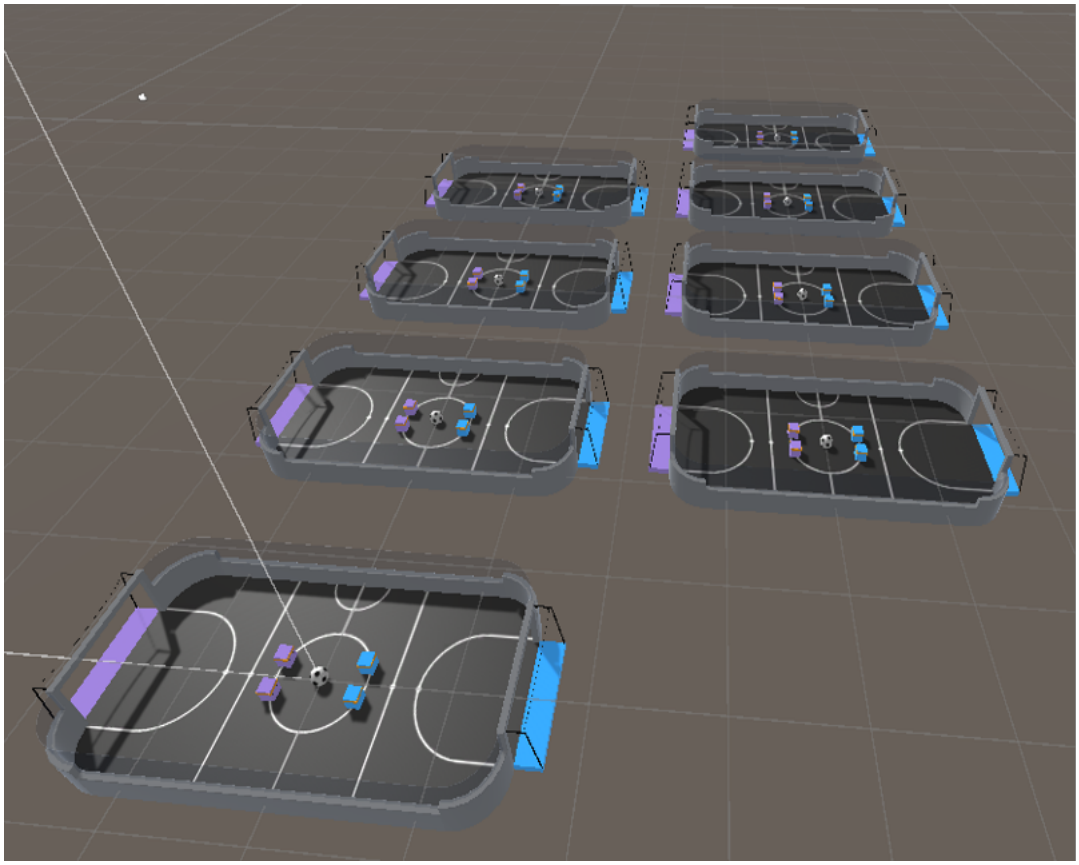

Fig. 3. Basic example of AI-trained NPCs playing soccer [7].

evolution of the gaming landscape. Where AI-driven NPCs adapt to player actions, enhancing the overall dynamism and depth of gameplay encounters.

V. DEVELOPMENT

In order to gain a better understanding of the technology, I decided to develop a simple version of the classic game pong and train an AI system to play it. To keep things simple, I followed the example provided in the Pellet Grabber tutorial series by the YouTube channel Jason Builds [2], creating an environment with few moving parts that would allow the AI agent to train with as few variables as possible.

Initially, I rewarded the agent for successfully deflecting the ball. However, this approach produced an agent that was nearly unbeatable for a human player. To address this issue, I developed a new system that incentivized the agent to keep the score as close to tied as possible. My first attempt at this system punished the agent for being behind in the score and gave a decreasing reward for being ahead. However, this approach led to the agent prioritizing a tied score over hitting the ball. By reducing how frequently the agent received this secondary reward, I was able to create a reward system that worked more effectively.

The actual training process was the next step. When I first started the AI was behaving erratically and was not noticeably increasing in skill. After extensive research, I determined that this was due to 2 factors. The first factor was simply the amount of training I was doing. I was not training the agents for long enough to produce meaningful results. The second issue was slightly more obscure. When training, the MLagents library default is to run the training at a 20:1 timescale. However, running at this timescale can result in errors in Unity's physics calculations. So to fix this issue I ran the training at a 1:1 timescale, and to make up for the increased training time I duplicated my environment and ran the training with many agents simultaneously.

The final stage of development was testing. Once I had a working prototype I asked peers of mine to playtest the game and answer questions about their experience. The results were generally in line with what I expected. The simple system I had created struggled to pose a real challenge to a human opponent However, the potential for a more developed version of the AI could feasibly accomplish the original goal of providing a challenge to players without simply overpowering them.

VI. CONCLUSION

This technology is not so powerful that it should be used universally across the industry. Its potential for success is limited by the genre of the game that is in development. A narrative-led game, whose goal is to tell a story, may still benefit from more traditional NPCs. For this type of game, the developers can tailor the NPCs' actions and behaviors to fit the

story and setting they are looking to create. Implementing less predictable AI NPCs may be detrimental to the immersion of the player, as well as taking up development time that could be used elsewhere.

For more competitive, rules-based games, the potential for this technology is immense. Sports, shooter, and fighting games are all examples of genres that could benefit greatly from the use of AI NPCs. These NPCs could be implemented as a training tool for players looking to practice. Or they could be used to provide a greatly enhanced single-player experience. These are things that fans of these game genres have been asking developers to create for years. Implementing AI into the development of these games could help to push these games into a new era.

AI has the potential to reinvent how game developers create NPCs. Reinforcement learning techniques can be used to train in-game agents that can learn from and react to player actions. This creates an immersive and challenging experience that traditional NPCs cannot replicate. Open-source frameworks like Unity ML can expedite the implementation of this technology while providing robust and easy-to-use tools for developers.

REFERENCES

[1] BAJAJ, P. Reinforcement Learning, 18 Apr, 2023. Geeks for Geeks., Last accessed March, 2024., https://www.geeksforgeeks.org/what-is-reinforcement-learning/.

[2] JASON BUILDS. How To Use MACHINE LEARNING In Unity MLAgents Setup & Basic Environment - Pellet Grabber Tutorial #1. https://www.youtube.com/watch?v=D0jTowlMROc, October 2023. Jason Builds, Last accessed March, 2024.

[3] LINJA, A. *Explicit Rule Learning: A Cognitive Tutorial Method to Train Users of Artificial Intelligence/Machine Learning Systems*. PhD thesis, Michigan Technological University, 2023.

[4] NIKOLOPOULOU, K. Easy Introduction to Reinforcement Learning, August 15, 2023. Scribbr., Last accessed March, 2024., https://www.scribbr.com/ai-tools/reinforcement-learning/.

[5] POTENTIA ANALYTICS, I. What Is Machine Learning: Definition, Types, Applications And Examples. https://www.potentiaco.com/what-is-machine-learning-definition-types-applications-and-examples/. Potentia Analytics, Inc., Last accessed March, 2024.

[6] TORRADO, R. R., BONTRAGER, P., TOGELIUS, J., LIU, J., AND PEREZ-LIEBANA, D. Deep Reinforcement Learning for General Video Game AI. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)* (2018), IEEE, pp. 1–8.

[7] UNITY TECHNOLOGIES. Unity Machine Learning Agents Train and embed intelligent agents by leveraging state-of-the-art deep learning technology. https://unity.com/products/machine-learning-agents, 2024. Unity Technologies, Last accessed March, 2024.

[8] ZHANG, J. An introduction to machine learning with Unity ML-Agents, August 27, 2021. Coder One., Last accessed March, 2024., https://www.gocoder.one/blog/introduction-to-unity-ml-agents/.