|  |
| --- |
| Applicability of dynamic difficulty adjustment with Q-learning in ARPG encounters  Joshua Bridges  MSc Computer Games Technology, 2022-2023 |

|  |
| --- |
| School of Design and Informatics  Abertay University |

# Table of Contents

[Table of Contents i](#_Toc144818990)

[Table of Figures iii](#_Toc144818991)

[Table of Tables iv](#_Toc144818992)

[Acknowledgements v](#_Toc144818993)

[Abstract vi](#_Toc144818994)

[1. Introduction 1](#_Toc144818995)

[1.1 Dynamic Difficulty Adjustment 1](#_Toc144818996)

[1.2 Reinforcement Learning 3](#_Toc144818997)

[1.3 Research Expectations 5](#_Toc144818998)

[1.4 Overview 6](#_Toc144818999)

[2. Literature Review 6](#_Toc144819000)

[2.1 Enjoyment of Games 6](#_Toc144819001)

[2.2 Artificial Intelligence in Games 8](#_Toc144819002)

[2.3 Artificial Intelligence Techniques 9](#_Toc144819003)

[2.3.1 Neurogenetic Approach 9](#_Toc144819004)

[2.3.2 Reinforcement Learning (RL) 10](#_Toc144819005)

[2.3.3 Belgian Artificial Intelligence Algorithm 11](#_Toc144819006)

[2.4 Data Collection in Game Research 11](#_Toc144819007)

[3. Methodology 12](#_Toc144819008)

[3.1 Game Application 12](#_Toc144819009)

[3.1.1 Game Walkthrough 14](#_Toc144819010)

[3.1.2 Player Playstyles 16](#_Toc144819011)

[3.1.3 Enemy Behaviour 17](#_Toc144819012)

[3.1.4 Game Implementation 18](#_Toc144819013)

[3.2 AI Agent 19](#_Toc144819014)

[3.2.1 State Action Table (Q-table) 20](#_Toc144819015)

[3.2.2 Reward Function 22](#_Toc144819016)

[3.3 Data Collection 23](#_Toc144819017)

[3.3.1 Survey 23](#_Toc144819018)

[4. Results 24](#_Toc144819019)

[4.1 Participation Time 25](#_Toc144819020)

[4.2 GEQ Metrics 26](#_Toc144819021)

[5. Discussion 30](#_Toc144819022)

[5.1 Metrics and Data 30](#_Toc144819023)

[5.2 Implementation 32](#_Toc144819024)

[6. Conclusion and Future Work 33](#_Toc144819025)

[References 36](#_Toc144819026)

[Bibliography 40](#_Toc144819027)

[Appendix 41](#_Toc144819028)

[A: Survey 41](#_Toc144819029)

[Pre-Game Question: 41](#_Toc144819030)

[Post-Game Questions: 41](#_Toc144819031)

# Table of Figures

[Figure 1: Flow channel concept proposed by Csikszentmihalyi. (Zohaib, 2018) 7](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818977)

[Figure 2: Heuristics of AI in adhering to principles of difficulty in games. (Sweetser, Johnson and Wyeth, 2012). 8](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818978)

[Figure 3: Initial Menu screen as seen by the player. 14](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818979)

[Figure 4: In-game HUD describes player and enemy health as well as control scheme on the left-hand side. 15](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818980)

[Figure 5: Class diagram of expanded UE5 game instance class. 18](#_Toc144818981)

[Figure 6: Flow diagram of delayed Q-learning implementation. 20](#_Toc144818982)

[Figure 7: Pointer arithmetic to find the one-dimensional matrix representing the action space of the Q-table. 21](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818983)

[Figure 8: Average participant time in-game in seconds for each experimental group 25](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818984)

[Figure 9: Evaluation of Enjoyment related metrics averaged using 0-4 GEQ scale. 26](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818985)

[Figure 10: Evaluation of Flow related metrics averaged using 0-4 GEQ scale. 27](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818986)

[Figure 11: Evaluation of Difficulty related metrics averaged using 0-4 GEQ scale. 28](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818987)

[Figure 12: Evaluation of Frustration related metrics averaged using 0-4 GEQ scale. 29](file:///D:\DissertationArtefact\Dissertation%20Document\Dissertation.docx#_Toc144818988)

# Table of Tables

[Table 1: Character variations from selected playstyle (class) 16](#_Toc144819032)

[Table 2: Enemy behaviours and available actions. 17](#_Toc144819033)

# Acknowledgements

I would like to thank William Kavanagh for being my primary supervisor on this long research journey. His feedback in all stages of the project made it so that the research could be completed on time.

Thank you to Stuart Anderson for being my secondary supervisor and providing instrumental feedback on the live presentation.

Thank you to lecturers Ruth Falconer and Hadi Mehrpouya for starting me on this research and instructing on how to shape the initial ideas into the resulting artefact and results presented in this paper.

Lastly, I’d like to thank my family for supporting me on this adventure. Their emotional support and ability to discuss topics far out of their area of expertise were instrumental to the completion of this research.

# Abstract

Game difficulty is traditionally a challenging aspect for game developers to successfully implement. Traditional approaches using static difficulty are limited. Once players memorize and rehearse the exact same strategy that will consistently work every time, boredom can set in. In more recent years, developers have begun researching implementations of artificial intelligence (AI) to adapt gameplay difficulty to the skillset of any given player creating Dynamic Difficulty Adjustment (DDA).

This research aims to evaluate how a Q-learning AI algorithm can adapt the difficulty of enemy fights to player strategies. The goal is to identify which algorithms, if any, improve player experiences over the short duration that a boss fight normally lasts. These goals will be accomplished by having players fill out a survey to evaluate their experience with a unique game created for this research. The participant gameplay experience will have players engage in an action game enemy encounter. Afterwards, they will reflect on the difficulty and enjoyment of the encounter by filling out select questions adopted from an established questionnaire standard.

These findings indicate that the established game is paramount to the success of the Q-learning algorithm. Inserted into an unrefined product, DDA does little to improve the experience for players. Additionally, the timeframe is paramount to the success of an AI approach to DDA. Q-learning with DDA should primarily be utilized when players can be guaranteed to return to the experience multiple times.

# 1. Introduction

## 1.1 Dynamic Difficulty Adjustment

Easy, normal, hard, nightmare. These terms are a few of the many that have been colloquially used to describe difficulty in single-player games. These descriptors would define the amount of health the player had, the amount of score earned, or how enemies behaved. As the games industry grows internationally, games find themselves limited by the static difficulties only being entertaining to small subsections of the population. In the current world of game development, designers have pushed past the idea of static, preset difficulties in favor of a more adaptive system of challenge, DDA. This technique works to balance the difficulty of games by responding to the player themselves. The game takes in information via player feedback, various scanning equipment, or knowledge about in-game parameters and uses this to adjust the behaviour of enemies and other game mechanics.

DDA provides improvements to key oversights in the traditional difficulty system. First, players are unreliable evaluators for their own skill level. In abstracting the difficulty from the player, the game creates a buffer between what the player thinks they want, and how the game actually performs. Second, the behaviour of the game becomes predictable. A simple flow chart becomes an easy way for players to memorize the entire game and remove any semblance of difficulty. Third, replayability is becoming a much more prominent aspect of modern game releases. The downtime between major series releases is getting longer, resulting in more players returning to their favorite games or turning to multiplayer releases. If there exist four difficulty options, a player could only have four unique playthroughs. Instead, DDA aims to provide a unique experience at every step throughout a game. Generating a theoretical infinite possible difficulties, DDA opens the door to more dynamic interactions between player and game.

DDA related research into potential applications in game design began with the emergence of research in the early 2000’s (Hunicke, 2005). This research was aimed at demonstrating key benefits of DDA. Hunicke (2005) found that even the crudest adjustment algorithm can provide an improved experience to the player. Data began to emerge that indicated players enjoyed this more dynamic experience with games. Furthermore, changes could be made imperceptibly to the player. Participants noticed adjustments when they did not exist or failed to perceive adjustments that had occurred during their experience. Hunicke’s (2005) research focused on the balancing of player economy, what items were accessible to the player at any given time. However, the introduction of this method subsequently created a rise in interest in other applications. Researchers theorized that DDA could be used on more fundamental mechanics of game design.

Research into DDA in games began to spread to cover the vast expanse of domains in this field. Shortly after Hunicke (2005), Hagelback and Johansson (2009) investigated the application of DDA on enemy behaviour in long Real Time Strategy (RTS) games. They set up five different opponents for the players to go up against. These five had a varying level of dynamism. From instances of static to rapidly adapting difficulty, the goal was to provide players with a wide breadth of options. This allowed the study to evaluate the DDA methods in relation to traditional static methods. In evaluation, Hagelback and Johansson found that players preferred a dynamic opponent over a static counterpart. They concluded that it was challenging to construct a static difficulty for the many types of players a game must entertain.

The idea of DDA began to gain further traction as evident in Zohaib (2018). DDA related literature between 2012 and 2017 was three times higher than the amount in 2009. A study by Costa and Luís Magalhães (2017) demonstrated that the DDA philosophy navigates a fine balance in regard to game design. They designed a simple role-playing game to examine if DDA ideas could be used for procedural content generation. The study used a model of the game context to represent how the player was progressing through the game. This model was then used to generate in game resources, in this case, equipment and runes. The results indicated that the designed game was too abstract for player enjoyment. The game was designed towards optimizing probabilities that weren’t directly correlating to any player’s actions. It is important to observe that DDA and game design are inherently intertwined. This means that the game needs to have distinct differences under DDA or PCG for players to be sufficiently engaged. Player engagement is then necessary to gather suitable results and draw any reasonable conclusions from the data.

DDA has been explored in a variety of fields with at least 218 articles (Paraschos and Koulouriotis, 2022) mentioning some aspects related to the topic. Researchers have investigated DDA in genres ranging from Tetris to strategy games. However, that research generally occurred over a sizable timeframe or with significant pre-training. This makes sense as the AI in DDA requires time to train and adapt to the player. Nevertheless, there are plenty of game experiences that occur spontaneously on a small timescale.

## 1.2 Reinforcement Learning

For DDA to successfully evaluate the user experience and come to decisive conclusions, a sophisticated intelligence needs to be making these decisions. Humans are not ideal to do this type of task, therefore AI has risen as the optimal replacement in this scenario. The evolution of AI has coincided in timescale with the emergence of DDA. Where older literature (Hunicke, 2005) used a probabilistic model to make decisions, modern attempts are using much more sophisticated learning methods.

One specific approach is reinforcement learning (RL) (Sutton and Barto, 2018). This work focuses on using a specific approach to RL dubbed Q-learning (Watkins, 1989). RL works through a series of trial and error. The game is represented as a table of states. Each state has a value tied to it, V(s). Navigating the game from state to state, by proceeding along the highest value path, would inevitably result in the agent winning the game. Thus, the RL agent proceeds with a course of action referred to as a policy. After a series of actions has concluded, the agent then reflects. This reflection occurs with a reward function that evaluates how well the agent performed at this task during the individual trial. The reward function is then used to backpropagate through the states that the agent visited. This process iteratively updates V(s) to reflect the expected rewards. In a sense, this makes RL comparable to other unsupervised learning approaches such as genetic algorithms. Using this traditional approach, the reward is generally delayed to after a series of actions or an entire gameplay sequence. This delay makes a traditional approach undesirable for game AI. If an agent is to learn while performing, they should be adapting during the performance as well. Otherwise, the difficulty looks less like a curve and more similar to a staircase if the player even decides to attempt again.

To remedy this situation, Q-learning utilizes an approach later dubbed Temporal Difference Learning (TD-learning) (Tesauro, 1995). TD-learning shortens the temporal distance between action and reward. Instead of progressing to the end of the game to calculate the rewards, the agent looks ahead to the next state. The next state would reflect the one after and, therefore, the one state ahead provides an approximate value of the expected future rewards.

V(s) is not enough to reflect the entirety of the game. If an agent could take multiple possible actions from a given state, it would be necessary to know what each individual action could do. If a state with the winning action could also lead to losing states given other actions occurring, the agent may avoid it entirely. A Q value, Q(s, a), reflects the value of each action in any state. This change increases the requirements of the table, needing another dimension to reflect the actions. However, this vastly increases the likelihood and speed of convergence. Convergence on the optimal policy indicates success of the RL agent.

Consequently, the crux of Q-learning lies in its reward function. The reward provides the incentive for an agent to pick a certain policy over another. A typical reward function for an agent playing a single-player game is as follows: -1 for each action, +100 for entering a winning state. This configuration would learn a policy that approaches the winning state in the fewest number of steps possible. However, with DDA as the goal, the agent needs to learn around the player character. The goals shift from simply finding a winning state, to establishing a balance with the player. A challenge is that finding balance is a very vague goal. Thus, establishing a definition for balance is imperative for the development of the reward function.

This research uses an epsilon-greedy approach to the policy. A greedy policy dictates that during action selection, the agent chooses an action with the maximum Q(s, a). However, pure greedy policies suffer issues with learning the optimal behaviour. If a greedy policy finds a local maximum, it will stop learning and solely focus on the discovered path. Epsilon-greedy is designed to avoid this issue. Instead of choosing the greedy option every time, the agent will randomly select an action epsilon percent of the time. An epsilon of 0.2 means the agent will behave greedily 80% of the time and explore actions randomly the remaining 20%. This exploration allows the agent to avoid local maxima. Furthermore, DDA works to erase monotony in the gameplay experience. A 100% greedy policy would be similarly monotonous to static difficulty options. Epsilon exploration forces the agent to occasionally make sub-optimal decisions and add variety to even the most optimal experience.

## 1.3 Research Expectations

This research aims to explore how successful DDA is on adapting gameplay over a brief period of time. Specifically, over the course of time during a boss fight that a player may encounter in an action role playing game (ARPG). To this end, a game will be designed and created in Unreal Engine 5 with a RL agent set up to manage the DDA. Participants engage with the combat in the application while the agent learns optimal actions to balance the difficulty. Players then fill out a questionnaire that describes their experience in accordance to how much they agree with provided statements. Through the survey, players will provide insight into the AI’s learning behaviour and whether the learning contributed to an improved user experience. The results will provide insight into whether RL can generate substantial DDA without pre-training, and how the experience affects the players enjoyment of the ARPG encounters.

## 1.4 Overview

The following literature review explores the history behind flow and its relevance to games. This review also looks closely at some more relevant instances of prior research into DDA. Specifically, this section reviews the relevance of prior AI implementations to this research. The methodology details the process behind the three main components of this research artefact. The section describes the observed benefits of the design decisions behind the game application and the interconnected AI agent. Data acquisition and the methods behind generating results are also described in this section. The results and discussion describe the results derived from participant data. These results are then interpreted in the context of the research goals. This document concludes with a conclusion discussing the ramifications of the results and potential avenues for future work.

# 2. Literature Review

Ideally, readers of this paper have an understanding of the history of AI in game design, however, for those that do not, a brief review will follow. This review will be an introduction to the three key aspects of this research. First, we will explore the history of researching AI with games. Second, an overview of frequently used AI techniques will establish a background for the selection made in this project. Third, a brief insight into the history of post-game surveys will highlight key elements included in this research’s data collection.

## 2.1 Enjoyment of Games

Player enjoyment is typically represented using Flow (Zohaib, 2018). Adapted from Csikszentmihalyi’s theory of flow to games by Koster (2013), flow derives its meaning from a complex idea of philosophy. “When an important goal is pursued with commitment and focus, and all the varied activities fit together into an unified flow experience, the result is harmony that is brought into consciousness.” (Beck, 1990). This “flow experience” is represented graphically as a harmonious middle channel where player expertise and game difficulty are linearly correlated (Figure 1). To provide a game that reaches and satisfies as large an audience as possible, designers should provide features allowing players to enjoy the flow experience. Games should “Keep the user’s experience within the user’s Flow A diagram of a channel

Description automatically generatedZone” and “Offer adaptive choices; allowing different users to enjoy the Flow in their own way” (Chen, 2007).

Figure 1: Flow channel concept proposed by Csikszentmihalyi. (Zohaib, 2018)

Adapting this idea to games, a model named GameFlow was developed to instruct the intent behind creating a flow experience (Sweetser and Wyeth, 2005). Further research developed more detailed heuristics to highlight specific elements of games (Sweetser, Johnson and Wyeth, 2012). Specifically, eight detailed heuristics were developed to focus how AI should operate whilst adhering to flow principles (Figure 2). These principles highlight the importance of striking a balance between not being overbearing but remaining constantly engaging. While the focus is firmly on strategy games, these heuristics can be abstracted to apply to many different genres of game design. The focus on difficulty in the GameFlow heuristics works well in tandem with DDA. These statements create a target that dynamic difficulty should strive to achieve. Once an AI adheres to the principle and creates the desired flow experience, then it can be considered successful.

## A white text on a white background Description automatically generated2.2 Artificial Intelligence in Games

Figure 2: Heuristics of AI in adhering to principles of difficulty in games. (Sweetser, Johnson and Wyeth, 2012).

The method by which the AI system evaluates the current game state and difficulty can greatly change the applicability of any given DDA technique. Dziedzic and Włodarczyk (2018) identified three key methods:

1. Formal model of gameplay
2. Selected features of the game
3. Direct examination of the player

The formal model allows reusability of DDA algorithms since the entire gameplay is simplified into a decision tree. The use of selected features provides a unique advantage by allowing representation of more complex games than a formal model. A direct examination of the player allows the DDA to more accurately reflect the player’s emotional state at any given time.

In the research undertaken by Altmira et al. (2017), was driven by the need to answer the question “How does game adjustment design that alters the sport equipment statically and dynamically affect game balancing and player engagement in a non-parallel game?”. Their investigation concluded that the frequency of updates can impact the player engagement in these physical games. Reis, Reis and Lau (2020) suggest that by replacing the dynamic aspects with other game artifacts, results can be extrapolated to reflect other games. Furthermore,  Cechanowicz et al. ([2014](https://link.springer.com/article/10.1007/s10726-020-09652-8#ref-CR12)), contemplated other crucial ideas about how balancing affects the player experience.

* Does balancing work?
* Is balancing preferred overall?
* Does balancing improve novice experience?
* Does balancing detract from expert experience?

Their research found that the balancing of competition was received positively. Both novice and expert players enjoyed a more balanced experience in terms of difficulty. For this project’s line of research, these findings are extrapolated to indicate that any player would prefer a more balanced engagement against AI opponents as well.

## 2.3 Artificial Intelligence Techniques

Zohaib ([2018](https://link.springer.com/article/10.1007/s10726-020-09652-8#ref-CR40)) classifies DDA techniques under eight different archetypes: (i) Probabilistic Methods; (ii) Single and Multilayered Perceptrons; (iii) Dynamic Scripting; (iv) Hamlet System; (v) Reinforcement Learning; (vi) Upper Confidence Bound for Trees and Artificial Neural Networks; (vii) Self-Organizing System and Artificial Neural Networks; and (viii) Affective Modelling Using EEG. The primary focus of these techniques being to develop a challenge function to represent difficulty (Zohaib, 2018). This heuristic function allows an intelligent system to interpret any current game state as a numerical value. Game values that reflect the difficulty (health points, time, score, etc.) are often used as the parameters to this challenge function. There exist many different ways to utilize and represent this challenge function in AI specifically. A look into some popular and modern approaches for DDA are listed below.

### 2.3.1 Neurogenetic Approach

Shakhova and Zagarskikh (2019) implemented a genetic algorithm as a supplementary system to a multi-layered neural network (NN). They worked in tandem by using the GA to set the weights for the different perceptrons of the NN, with each configuration being known as a gene. The NN then uses this weight configuration to calculate a value that represents a final decision for the opponent to make. The GA evaluates the success of the current neural network using a grade function. The grade function works as the challenge heuristic in this application. This tells the algorithm how well each gene performs by evaluating any given configuration’s difficulty. The best performing genes are then mutated together to create better and better genes to optimize the neural network. The results indicated that this approach held a couple key advantages over other AI systems.

* Learning is mainly done online during gameplay, however, can also be done offline if there exists game data to train.
* A lack of predefined rules cuts back on initial development costs.
* This approach generated more diverse actions from the AI agent.

### 2.3.2 Reinforcement Learning (RL)

RL is constructed around maximizing the anticipated reward of any action given the current state of the game. Reis, Reis and Lau (2020) investigated the use of reinforcement learning with DDA. This study constructed an agent that learns how to optimize a game's difficulty by gradually adjusting a base copy of the game. This study used offline training to set up the AI. Offline learning is the process of training an AI before the game is running e.g., training on a large dataset.

Ashey Noblega, Paes and Clua (2019) makes use of Deep RL in a 3D fighting game to develop an agent that is able to both adapt to the opponent’s skill and learn the game. They propose the use of a reward function that is constrained by a balance constant. This constant is used to force the agent to not become too difficult so that the player stops having fun. The research concluded that the agent stayed in the balanced state for at least 50% of the game. Ashey Noblega, Paes and Clua (2019) closely reflects the setup for the research conducted in this project. A 3D fighting game has close relation to the behaviour of the 3D enemy fights designed for this research. Furthermore, their application of the balance constant has applications that permeate into this research.

The balance constant was defined as the difference in health between the enemy and the player. By maintaining the difference to be constrained to a maximum, Ashey Noblega, Paes and Clua (2019) developed a system that sufficiently balanced around a health difference. The representation of the balance constant as this difference is only one way to look at it. Ratios, along with other combinations of state values, could present other methods of representing a balance constant. This article does not sufficiently answer the questions in respect to GameFlow. Specifically, does maintaining the balance constant truly provide a better experience for players?

### 2.3.3 Belgian Artificial Intelligence Algorithm

The Improved Belgian Artificial Intelligence (IBAI) algorithm proposed a specific learning approach to ARPGs (Mi and Gao, 2022). A set of player-centric rings defined enemy behaviour at different radii from the player. DDA is applied by the algorithm defining a maximum grid capacity (MGC) and maximum attack capacity (MAC). The values constrain the number of enemies allowed into certain rings as well as the frequency of attacks the enemies can attempt. Fundamentally, this approach used distance from the player to establish a progressing level of engagement. The method manages both ranged and melee enemies adequately and provides an improvement on the established ARPG algorithm (Mi and Gao, 2022). Benefits aside, the fixed domain provides a heavy limitation to the research of IBAI and similar algorithms. Additionally, the research artefact needs to be designed with the study of IBAI in mind. Nevertheless, IBAI developers are confident that the algorithm can be adapted to use towards general intelligence cases, particularly in the field of games (Mi and Gao, 2022).

## 2.4 Data Collection in Game Research

According to Wirth et al. (2007), it was difficult to define a standardized approach to game results due to the inherent subjective nature of the evaluations. The Game Engagement Questionnaire (GEQ) utilized classical and Rasch analyzes to establish a general use case survey that allowed for collection of subjective results for games (Brockmyer et al., 2009). This research identified six relevant terms to establish indicators of involvement: (i) engagement; (ii) immersion; (iii) presence; (iv) flow; (v) psychological absorption; and (vi) dissociation. Initially developed to investigate effects of immersion in violent video games, the comprehensiveness of the GEQ extends its usability to more domains. Modern research in DDA makes use of the GEQ to establish a subjective description of the user experience (Mi Q and Gao T, 2023) (Moon et al., 2022).

Denisova and Cairns (2019) addressed the common concern of DDA awareness in their research. This study showed that the more information provided to the player about the adjustment, the more the player felt immersed in the game. Since immersion is arguably the most relevant feature of game design (Cairns, Cox and Nordin, 2014), this research holds that awareness of DDA would not negatively impact a player’s enjoyment. Therefore, the research states that players could, and should be informed in the case that DDA research will be performed.

# 3. Methodology

## 3.1 Game Application

A three-stage, boss rush, action role playing game (ARPG) was developed for this research. In this game, the player is tasked with lowering the health of three consecutive enemies to zero. In summary, three distinct playstyles are provided that the player can choose from. Each playstyle brings a different arsenal of tools the player can employ to accomplish this objective. Similarly, each enemy introduces a different playstyle for the player to experience. The first enemy was designed to be a close quarters brawler. The second enemy was created as a long-distance sharpshooter. The third enemy was created as a hybrid with the capability to fight in close quarters as well as zone out players with ranged capabilities. In order to stay within scope of the research and not overly complicate the AI, each enemy has two attacks that they can utilize against the player. In addition, the enemy can choose to close the distance between itself and the player to culminate in three possible actions for each enemy at any moment.

As the primary focus of this research was gathering data, the development of the game artefact focused predominantly on functionality with little attention given to aesthetic aspects or presentation. The goal was to get players in and out as rapidly as possible while providing the experience of undergoing a challenging encounter. Since the research was focused on player enjoyment and experience with the addition of a RL agent, the base game had to be, at least, enjoyable to provide adequate data. This meant that a lot of development was focused on honing elements of the gameplay to create a difficult but balanced experience that participants could experience without feeling cheated or overburdened.

Due to the comparative nature of the research, it was necessary to establish a baseline, or control. To this end, three experimental groups were established to allow comparative evaluation of the participant data. Each of the three experimental groups provides different learning models for each participant to encounter.

1. The control group (Group 0) plays against enemies with traditional state-based logic.
2. Experimental Group 1 faces enemies learning using a reinforcement learning agent with a learning rate of 0.1.
3. Experimental Group 2 faces enemies learning using a reinforcement learning agent with a higher learning rate of 0.2.

Throughout the duration of the encounter, the experience for group 1 and group 2 will change as the enemy adapts to the player’s behaviour. In contrast, group 0 will be engaging enemies with pre-defined logic that will remain static throughout the entire experience. This dichotomy was necessary to establish a baseline control for the qualitative feedback that this research aimed to gather.

### 3.1.1 Game Walkthrough

The player begins the game at a simple menu screen shown in Figure 3. This menu allows the player to pick from one of three playstyles as well as provides a numerical value that the participant is instructed to save and record in the post-game survey. This number is used to categorize the data from any participant into one of the three experimental groups. The control group is therefore also categorized as experimental group zero for the sake of consistent numeration of results.

Figure 3: Initial Menu screen as seen by the player.

After the participant selects a playstyle, the application places the character into an open arena where the main gameplay loop begins. A new heads-up display (HUD), seen in Figure 4, provides the player with information necessary to complete the game.

The control overlay details how the user can interact with the encounter. Movement and camera controls are bound to frequently used buttons that allow the player to easily get accustomed to the key bindings. Additionally, with all of the key bindings paired to buttons on the left-hand side of a traditional keyboard, the remaining input options for primary and secondary attack are bound to mouse input. This setup allowed the player to utilize all essential mechanics with only a limited range of movements required.

Figure 4: In-game HUD describes player and enemy health as well as control scheme on the left-hand side.

Also visible on the HUD are the health totals for the player and the current enemy. The player can see their total health broken up into segments, so when they get hit, a segment disappears. These segments can be restored a limited number of times in accordance with the provided “heals” counter adjacent to their health bar. The enemy’s health is shown in the lower central section of the screen along with a name tag to allow the player to easily identify which health total corresponds to whom. This HUD encompasses all the necessary information the player needs to experience the game while not obscuring the focal area in the center of the screen.

From this screen, it is a simple process of engaging the enemy until either the player’s health or the enemy’s health reaches zero. If the player’s health reaches zero, then they are returned to the playstyle selection screen to potentially try again. If the opponent’s health reaches zero, then the next enemy of the encounter will spawn at the location of the beaten foe. If the player managed to reduce the health of all three enemies to zero, they are provided a celebratory screen pronouncing them the winner and providing a button to return to the playstyle selection menu. The player will remain in the same experimental group as long as the application remains running. Closing the application and relaunching, would permit a player to randomize into a different group.

### 3.1.2 Player Playstyles

Three different playstyles were offered to each player: melee, mage, and archer. Each playstyle contributes to a fundamentally different approach to the game experience. These playstyles had three main spaces for variance: attack behaviour, damage, and movement speed. Three attack behaviours were developed for the player as well as the enemies: melee, projectile, and area of effect (AOE). The maximum health was restricted from fluctuating between playstyles to adhere to the constraints of the AI agent. The dimensions needed to stay consistent to facilitate the AI learning. Therefore, the player’s health was fixed to six. The three playstyles and their differences are displayed in Table 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | Speed | Light Attack (Damage) | Heavy Attack (Damage) | Notes |
| Melee | 100% Base | Single Sword Swing (3) | Parry (0) | Parry stuns opponent for 6s |
| Mage | 100% Base | Slow Homing Projectile (1) | Three Homing Projectiles (1/per) / Explosive AOE (2) | Mage can toggle between two variant Heavy Attacks |
| Archer | 150% Base | Quick Bow Shot (1) | Charged Bow Shot (3) | Aim is based on character orientation unless camera lock is enabled. |

Table 1: Character variations from selected playstyle (class)

Melee characters must get close in order to engage the enemies with their sword. This comes with the inherent risk of getting damaged by one of the enemy’s attacks. Mage players have a mix of both short and long range and are best described as utility. They have three potential actions to choose from rather than the two provided to the other classes which can be toggled by pressing TAB. Archers excel at a simple strategy of attacking the enemy from range. Their actions are all fast and require aiming or camera lock to prove effective.

### 3.1.3 Enemy Behaviour

Three enemies were designed for this application with the goal to introduce a unique experience at each stage of the encounter. Since the enemies shared the same AI model, the quantity of actions they have available must be the same. There are three possible actions for each enemy: move, light attack, and heavy attack. Movement acts the same for each enemy, with the enemy approaching the player until it gets into close quarters. Traditionally, the ranged enemy would move away from the player. However, for the sake of continuity as well as pressuring the player, all enemies adhered to the same movement behaviour. The remaining attack actions vary, with each enemy having two unique attacks that they can use. Descriptions of the different attacks implemented for each enemy are described in detail in Table 2.

|  |  |  |  |
| --- | --- | --- | --- |
| Order in Sequence | Target Range | Attack #1 | Attack #2 |
| Enemy #1 | Close-Quarters | Fast Single Swing | Four Swing Sequence |
| Enemy #2 | Ranged | High-Speed, Linear Projectile | Slow, Homing, Proximity-Triggered Explosive |
| Enemy #3 | Hybrid  (Close-Quarters / Ranged) | Two Rapid Melee Swings | Single Melee Swing producing Slow, Damaging AOE |

Table 2: Enemy behaviours and available actions.

### 3.1.4 Game Implementation

The game was developed using Unreal Engine 5.1 (UE5) (Epic Studios 2023) to allow for rapid prototyping and development. UE5 provides several free, high-quality assets which were used to develop the game efficiently. The project was created as a hybrid project of Unreal blueprints and C++ code. The blueprint system is a simpler visual scripting method that facilitates rapid implementation of pre-defined behaviour. This system was used to define basic game mechanics and establish all the elements players see in the game.

C++ was used to create the enemy AI. For the AI to be accessible to the game at all times, as well as persist between levels, the behaviour was wrapped in a Game Instance class. This allowed the AI behaviour to be defined at a higher level of visibility than all other elements of the game. This Game Instance was created in Visual Studio 2022 as a data structure encompassing everything the agent would need to perform Q-learning. Figure 5 contains a diagram of the new Game Instance class.

A diagram of a game

Description automatically generated

Figure 5: Class diagram of expanded UE5 game instance class.

The C++ class was visible to Unreal blueprints in three select situations:

1. State parameters and AI hyperparameters
2. Function to allow enemy AI to select actions.
3. Functions to update and clear the Q-table.

Limiting the crossover to this degree meant that the math and logic behind the Q-learning was isolated to the code, while the game related elements, in particular, state values, remained updated through the integrated blueprints. This system allowed the separate aspects of this project to be iterated on in isolation while still coordinating the necessary information to the other respective module.

Supplementary functionality was made public to support debugging and testing in UE5. Reward calculations were made publicly accessible to enable in-engine visual feedback on the exact values of each update. This further enabled visualization of all states as a reward value not simply the ones triggering updates. This process allowed desired behaviour to be confirmed in all potential states and allowed for more consistent adjustments to the reward function.

Quantization functions were implemented to constrain continuous float values of time and distance to discrete integers. This mechanism enabled floats to be interpreted as dimensions for the instantiation and accessing of array dimensions. Distance was quantized into three discrete values:

* Near [0 - 200]
* Mid (200 - 400]
* Far (400 - Infinity)

The distances are measured in UE5 units where each unit is the equivalent of one centimeter. Time was also quantized. However, its functionality was deprecated since the value was removed from reward calculations.

## 3.2 AI Agent

Learning was performed by function calls to trigger updates to the Q-table. After the AI chooses an action, a snapshot of the current state is saved to a queue. Later, once the action is completed, the game calls the update function of the Q-table. This function uses the current state as the result of the action and pops the old state from the queue. These states are then used to evaluate the new Q value for the chosen action using Equation 1.

) [1]

is the new value for the Q-table being calculated is the value currently at that index. is the term describing the highest possible value in the new state across all potential actions. R is the term for the calculated reward. and are hyperparameters describing the learning rate and discount factor respectively. The three hyperparameters for Q-learning: γ, α, and ε were set to initialize at the start of the game. The values for γ and ε were identical across the two AI models. γ was set to the standard 0.9 value to ensure future evaluations were influenced by past rewards. ε was set to 0.3 making the model explore 30% of the time. Lastly, α, was the hyperparameter selected to be variable across experimental groups. Group 1 would encounter a model with α set to 0.1. Whereas group 2 would encounter a model with α at 0.2. Figure 6 details the process used in this research to update with Q-learning.

A diagram of a flowchart

Description automatically generated

Figure 6: Flow diagram of delayed Q-learning implementation.

### 3.2.1 State Action Table (Q-table)

The state space was defined by what should be visible to the enemy and influence the decision making. According to Zohaib (2018), the state and reward functions should build towards a challenge function. Describing the state with a challenge heuristic would allow state parameters (e.g., health, distance, resources, etc.) to be condensed into the single value. This approach would reduce the dimensions of the matrix. Therefore, representing the state this way could reduce memory overhead and time complexity. However, there were issues anticipated with this approach. Two distinct states could overlap into a single challenge value if the function was not set up properly. If a set up to avoid this issue was implemented, there would be little to no reduction in the dimensionality of the Q-table. To avoid this issue entirely, the challenge heuristic was isolated to the reward function. The Q-table was then created as a multi-dimensional array.

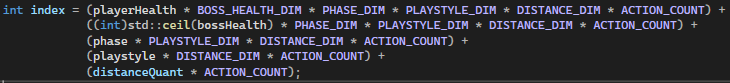
C++ limits the number of dimensions you can define a matrix using the standard array notation A[i] [j] [k]. Consequently, the Q-table is defined using pointers. A game instance class in Unreal initializes when the application launches. The setup of the Q-table is mapped to this initialization. A call to the malloc function allocates the required space in memory for the Q-table when the application is launched. The values are then defaulted to zero for the content of the table. As the pointer is not defined as a multi-dimensional array on initialization, pointer arithmetic is used to access the respective elements of the table. An IndexTable function wraps the pointer arithmetic and returns a one-dimensional matrix that can then be indexed by the action to get the required value. This function uses the state values to calculate the location of the necessary pointer as seen in Figure 7.

Figure 7: Pointer arithmetic to find the one-dimensional matrix representing the action space of the Q-table.

Five parameters were designated to represent the game to the AI.

* Enemy Health. [0 - 5]
* Player Health. [0 - 5]
* Distance from Player to Enemy. [0 - 2]
* Current Phase of the Encounter. [0 - 2]
* Player Playstyle. [0 - 2]

Enemy and player health were set up to have the same scale. Player health was represented as zero through five inclusive, one for each section of health in game. The enemy’s health was scaled from 30 down to six by using Equation 2 and rounding up the result. Hb was then used to index the Q-table and for calculating the reward function.

[2]

A sixth dimension representing the action space was defined as zero through two representing the selection of three actions the enemy may choose from. This representation culminated in a Q-table with 972 state-action pairs to evaluate.

### 3.2.2 Reward Function

The goal for this AI is to maximize player enjoyment. Consequently, the rewards cannot be defined objectively. For defining the reward function this means the application does not have an explicit objective for the AI. This research worked with the assumption that player enjoyment peaks when the stakes are high, and the player pulls off a close victory. In terms of state values, this is reflected in low player health and zero enemy health. However, this alone would not train rapidly enough for this application. From the works of Ashey Noblega, Paes and Clua (2019) a balance constant was adopted. If the AI managed to adhere to this balance constant, it would be creating an engaging experience for the player. Their research used a difference in health. This research, instead, utilizes a ratio between the player and enemy’s health. This would define a balance that re-evaluates across all values instead of simply negatively rewarding behaviour that fell outside of a pre-defined range. Additionally, if either the player or an enemy’s health was at zero, the constant would either fail to evaluate or equal zero. Both of these situations are considered failing states. To avoid this behaviour a hard cutoff is performed if either entity’s health falls to zero. The balance constant is assigned a static value of negative one. This allows the equation to avoid dividing by zero errors while still incorporating the health ratio for all other situations. In total, this reward reinforces a behaviour where the enemy desires not to get hit but also not to completely defeat the player.

Distance was isolated as a factor that can teach the AI to converge on an optimal policy for each enemy faster. In turn this would help the agent converge on an optimal policy faster than just the health in isolation. Thus, the enemies were rewarded slightly for staying at a distance appropriate to their gameplay behaviour.

[3]

The resulting reward function described by Equation 3 demonstrates how these values shift the reward with Hp representing the player’s health and Hb representing the enemy’s health. The distance modifier was appended to incentivize certain enemies to behave in different proximity to the player. If the enemy is not in the distance suited for it, a negative penalty is added to all actions chosen.

## 3.3 Data Collection

Data was collected through an anonymized external survey given to participants on completion of a blind test. The blind test involved randomly placing the participants into one of three experimental groups and playing the game artefact for 10 to 15 minutes without observation.

### 3.3.1 Survey

The survey was created by pulling elements from the Games Experience Questionnaire (GEQ) related to post-game examination (Brockmyer et al. 2009). The comprehensive nature of the full questionnaire extended beyond the complexity of this research. Therefore, select questions were chosen based on their relation to three criteria: difficulty, enjoyment, and immersion. The selected questions located in Appendix A. These questions are designed to be answered on a five-point scale describing how much the player agrees with the statement in question. The answers ranged from zero, not at all, to four, extremely. Select questions were then modified for further clarity in regard to this research. Statements such as “I felt pressured” were clarified to “I felt pressured by the enemy during the encounter”. These changes are also highlighted fully in Appendix A. Asking the participants to evaluate their experience in each experimental group numerically was expected to indicate that one configuration is more enjoyable than the other two. The survey was delivered through Google Forms.

# 4. Results

Participants were invited to this research by reaching out through multiple Discord servers. Links were provided in the Abertay Game Development Society server, a server for all master’s students, as well as other private channels. In total, 11 participants were involved, and they provided feedback through a Google Forms survey. The three experimental groups were mostly evenly balanced with four participants in both group 0 and group 2 and three participants in group 1. The artefact was only built for Windows, so participants were limited to only participating on those devices.

Personal data was deemed inappropriate for this research. Feedback was entirely anonymized and no demographic information was collected on the participants. As such, the results from this research are entirely based on the answers to the modified GEQ questions, as well as self-reported information on the time spent performing the experiment. Results were averaged inside each experimental group and represented in column graphs to compare the differences between each group.

## 4.1 Participation Time

A graph with blue bars

Description automatically generatedParticipants were asked to set aside 10 to 15 minutes to repeatedly play the game. As RL learning inherently behaves uniquely over different timespans. Players were asked to record the time spent in application with a stopwatch or similar application. The results are divided into the experimental group each participant belonged to and displayed in Figure 8.

Figure 8: Average participant time in-game in seconds for each experimental group

The group with the most reported time in-game was the control group. The experimental groups saw a steep decline with group 2 participants recording an average of almost three minutes. Group 0 was the only group to approach the average desired playtime of 10 to 15 minutes. Group 1 participated on average for seven minutes. Group 2 barely approached three minutes on average for playtime.

Lack of adequate training time indicates that the AI would have significantly less time to adapt to player behaviour. The results for participation duration will greatly shape the way that the GEQ metrics must be interpreted.

## 4.2 GEQ Metrics

A graph of different colored bars

Description automatically generatedA key measure of this research was the improvement of enjoyment in participants. Enjoyment was measured using five metrics from the survey: “It felt like a victory”, “I felt satisfied”, “I felt powerful”, “I felt proud”, and “I felt bad”. Figure 9 displays the average values for these five metrics across the three control groups.

Figure 9: Evaluation of Enjoyment related metrics averaged using 0-4 GEQ scale.

Group 1 displayed the most enjoyment across the board with average metrics evaluating around the median value for this scale of two. In contrast to the time spent in application, the control group recorded less enjoyment than group one. The metrics for all groups fall around or below the median value for the scale. In terms of the application, this reflects poorly on the game design as well as the agent behaviour. It is worth noting that the improvement from group 0 to group 1 does indicate the AI increasing the enjoyment from the base control group. Group 2 provides poor metrics on all fronts. In the one negative metric, “I felt bad”, group 2 is the only group to provide an average value above zero.

A graph of different colored bars

Description automatically generatedAn element of enjoyment is the flow experience. Three questions regarding flow from the GEQ: “I was deeply concentrated in the game”, “I was fully occupied with the game during my participation”, and “I felt pressured by the enemy during the encounter” along with one additional question, “I felt frustrated by the controls/mechanics” were selected to evaluate the participants flow experience. Figure 10 displays averaged metrics for comparison.

Figure 10: Evaluation of Flow related metrics averaged using 0-4 GEQ scale.

Similar to enjoyment, these metrics capped near the median value of two. None of the averages pushing near values of three or four indicates that the application was not sufficient for creating a flow experience in players. Low metrics for frustration with regards to mechanics indicate that this was not a substantial contributing factor to low flow. The green metric indicating enemy pressure during the encounter shows a significant drop when participants were not in the control group. This metric overlaps somewhat with evaluations pertaining to challenge seen in Figure 11.

A graph with different colored bars

Description automatically generatedDifficulty, being an essential part of DDA, was evaluated using four metrics: “I thought it was hard”, “I felt appropriately challenged by the encounter”, “I had to put a lot of effort into it”, and “I felt frustrated by the difficulty”. The results for these metrics were significantly low across the board. Notably frustration with the difficulty increased from the control group to group 1. Then, another small increase was noted from group 1 to group 2. Outside of frustration, group 2 showed no other indications of experiencing difficulty. The three other metrics average from zero to 0.5 when pertaining to group 2. Group 1 had the sole standout metric regarding difficulty. A value of 1.7 for “I thought it was hard” was the single highest average metric in terms of difficulty. The control group observed low metrics for all questions pertaining to difficulty. Participants indicated that the base game with the control group was not difficult.

Figure 11: Evaluation of Difficulty related metrics averaged using 0-4 GEQ scale.

A graph with red and blue squares

Description automatically generatedThe remaining metrics pertained to the perceived frustration players experienced in the experiment. Figure 12 displays each group’s average for the two metrics: “I felt frustrated by the controls/mechanics” and “I felt frustrated by the difficulty”.

Figure 12: Evaluation of Frustration related metrics averaged using 0-4 GEQ scale.

Isolation of these two metrics allowed observation of where potential frustration may originate. Frustration with the intrinsic gameplay experience or frustration with the adapted difficulty may both contribute to the lack of a flow experience. The results for these metrics were low in both regards. Despite a slight increase in difficulty-related frustration from the control group to group 1 and 2, none of the groups recorded values indicating significant frustration.

# 5. Discussion

## 5.1 Metrics and Data

With a small sample size of 11 participants, it becomes difficult to draw significant conclusions from the data. Nevertheless, small trends can be observed in the aggregated metrics.

Participants did not spend as significant of a period of time in the application as expected. This was theorized to stem from a frustration with the game mechanics not being as complex as players might expect. However, this ended up not being the case. Participants unexpectedly provided extremely low metrics in terms of frustration. Low enjoyment and low difficulty metrics both indicate that group 2 suffered from an overly simple experience. Despite not recording the success rate of participants, the data for group 2 indicates that their time spent in game was simply too easy. Low difficulty metrics as well as low pressure from the enemy indicates that group 2 participants did not feel engaged by the experience. Higher metrics for concentration imply that they were focused throughout the encounter, however there is no indication that participants struggled in group 2. As a result, their low time in application appears to be a consequence of them beating the game and not feeling any desire to return.

Time spent in the application directly correlates to time that the AI agent spent learning. A low time in-game for participant groups would indicate that the AI was not able to spend much time learning before participants disengaged with the application. Whether or not the player defeated all of the enemies in the encounter was not recorded. This information was thought to be irrelevant as the player would return regardless and continue in the experiment. These results indicate that was not the case. Knowing the reason behind players closing the application early may have provided key insight if it was difficulty related or not. Without that information, the only observable takeaway is that the AI agents provided significantly less retention than the control group.

The flow metric, “I felt pressured by the enemy during the encounter”, highlights an issue with the RL agent. A higher average for the control group than the experimental groups indicate that the AI learned to behave more passively. The distance modifier of the reward function was intended to incentivize the agent to stay within the optimal range for each enemy. This behaviour would allow each enemy’s attacks to be threatening to the player. However, the data contradicts that assumption. As the rate of learning increased, the AI was behaving in a less threatening manner to the player. This result is also substantiated by the results of the difficulty metrics. Group 2 reported significantly lower difficulty metrics than the other groups. This could be the result of two factors. The reward function could have a fundamental flaw that incentivizes unintended behaviour, or the higher learning rate of group 2 is overlearning beyond the optimal policy. The improved difficulty reported by group 1 indicates a higher likelihood the latter conclusion is correct. The epsilon value of 0.3 does not appear to have been significant enough to explore back into the optimal policy. Instead of treating 0.1 as the lower bounds for learning rate, agent behaviour may be improved by using 0.1 as the upper bounds. These results could also be attributed to the significantly lower time spent in-game of the group 2 participants. The less time the RL agent has to learn, the less likely it becomes that the agent discovers the optimal policy.

Group 2 was the only group that evaluated the “I felt bad” metric with an average above zero. The enjoyment metrics for group 2 were significantly below the reported averages for the control group and group 1. Group 1 reported the best metrics in areas of enjoyment and difficulty. These results indicate that the RL agent in group 1 did provide a more enjoyable experience to the players. Even with the improved metrics for group 1, the players still recorded less average time in-game than desired. While more enjoyable than the control group, the addition of the RL agent to group one still did not create more replayability. Small sample size may have been a contributing factor to this surprising result. The constraints of the game design could have also been a limiting factor. Enemies were limited to only three possible actions. In the control group, it was likely that each participant would experience all the possible actions. However, with the RL agent, there was no guarantee that the player would experience all the game could offer. The participants in group 1 and group 2 likely grew tired of a repetitive policy and decided to submit their results early. More concrete information regarding repetition could have provided more insight into these conclusions.

Supplementary questions to the GEQ would have contributed to additional clarity in the results. As previously discussed, asking participants if they cleared the challenge would provide additional insight into difficulty and enjoyment related metrics. While the completion can be extrapolated from the low average of difficulty metrics, a value more grounded in data would be substantial in reinforcing these conclusions. Groups could also be further subdivided by way of player class. This division was not feasible here due to the small sample size. However, isolating data between classes would assist balancing both the AI and the base game. Data on classes in the control group would help isolate key issues with the difficulty balancing of the base game. The difference between the class data in the experimental groups and the control group would then provide further insight into the behaviour of the AI.

Monotony was a feature this experiment wanted to avoid. Zero insight with the selected questionnaire limited the number of conclusions that could be drawn regarding repetition. A simple question regarding “Did the experience change between attempts?” could have provided more information on the experience of participants.

## 5.2 Implementation

Initially the AI for this research was designed with a neuroevolutionary approach in mind (Shakhova and Zagarskikh, 2019). Issues arose with this approach and the time scale of the application. Genetic algorithms must test the viability of each gene in each generation to properly evaluate the cost. If each gene is tested for one attempt in the application, the generations would scale to a time requirement that is greatly out of scope for this research. The requirement of avoiding pre-training restricted this from being a viable approach to the AI.

The hyperparameters configured for Q-learning were settled based on the desired behaviour of the model. ε was set to 0.3 for this research. This is substantially higher compared to the norm of 0.01 to 0.1, however, it adhered more towards the goals of the research. Exploration breaks the predictability of the enemy’s behaviour. Additionally, it helps the AI learn more of the Q-table by avoiding local maxima the policy may get stuck in. α, the learning rate, was the variant hyperparameter across the two RL configurations. Alpha fluctuates largely based on the desired intent of the model. Lower values of α allow the learning process to dampen oscillations that can occur in the Q-values. Given that α typically exists between zero and one, the values used in this experiment were selected from the more conservative lower end. However, 0.1 and 0.2 are still very distinct, especially in rapid training and the large step sizes that occur at the beginning of Q-learning. Smaller differences could lead to more rapid identification of an ideal α value. More models would be important for exploration into other configurations of hyperparameters, however these two were selected for the scope of this research.

# 6. Conclusion and Future Work

This goal of this research was to determine if the tested models of Q-learning could improve a user’s enjoyment of ARPG enemy encounters using DDA. This research gathered data from 11 participants to evaluate their sense of difficulty, enjoyment, flow, and frustration when experiencing consecutive enemy engagements. Their feedback indicated that no, this approach did not work for improving their overall experience. While a group with the AI did record some slightly higher average values for difficulty and enjoyment, the data does not provide significant reinforcement of group 1 outperforming the control group. With participants spending less time than anticipated in the game, it becomes difficult to accredit the good results to the AI. The margin is small enough to where these results could be better simply because participant experiences were blind and random.

Significant feedback pointed to the game artefact being a source of failure. The game itself did not provide a significant enough challenge for DDA to provide an improvement in the allocated timespan. Open ended feedback from one participant noted that “Classes are very imbalanced and bosses need more than 2 attacks”. This follows a trend recognized by discussion where the AI may be learning, but it is not learning significant traits. Flexibility and variety were key elements that the AI was not able to learn simply due to a lack of available features. The participant’s initial point also indicates flaws with the initial game artefact. Not enough testing was able to be performed for the game to be balanced in isolation. A more personalized AI does not appear comprehensive enough to transform an incomplete game into one that is suddenly enjoyable to the players.

It is too drastic to conclude that Q-learning is flawed for this approach simply because this implementation is flawed. Constructing a standalone game artefact and the Q-learning models was too wide of an approach for a single project. Mentioned in the discussion, improvements to the comprehensiveness of the survey could provide more key insight into where players struggled, and whether RL was a factor in the difficulty.

Future work can improve substantially on this area in both the game design and Q-learning approach. Refinement and testing on the balance of player classes can ensure that the experience is not excessively favoured towards one approach. Furthermore, the reward function provides a plethora of flexibility for further experimentation. Two approaches could provide better convergence than the one in this research. A more complex reward function utilizing additional state values such as cooldowns, hit frequency, overlapping actions, etc., or a much simpler reward scheme designating key target states with positive rewards. The complex reward function would allow the AI to provide dynamic difficulty while the encounter is underway. However, if the game’s design can guarantee repetition, a simpler reward scheme could target the desired states with more guarantee. For example, designate all states where the player has one health and the enemy has zero as success states with a positive reward. This has little value to Q-learning until it reaches this state at least once and can propagate the positive value back through the table. A less complex approach could provide a unique perspective on Q-learning in this domain not yet observed through this research.

Flexibility of the Q-learning model can also be further improved. In this research, the enemies are differentiated by a state value. However, if enemy behaviour overlapped, the pursuit of an overarching RL model could prove beneficial.

In conclusion, this research does not provide a successful model for DDA in ARPG encounters. However, it does raise key questions about how and where future work in this field can improve the complexity of game design.

# References

Altimira, D., Mueller, F. ‘Floyd’, Clarke, J., Lee, G., Billinghurst, M. and Bartneck, C. (2017). Enhancing player engagement through game balancing in digitally augmented physical games. *International Journal of Human-Computer Studies*, 103(C), pp.35–47. doi:https://doi.org/10.1016/j.ijhcs.2017.02.004.

Ashey Noblega, Paes, A. and Clua, E. (2019). Towards Adaptive Deep Reinforcement Game Balancing. *Proceedings of the 11th International Conference on Agents and Artificial Intelligence* , 2. doi:https://doi.org/10.5220/0007395406930700.

Beck, L.A. (1992). Csikszentmihalyi, Mihaly. (1990). Flow: The Psychology of Optimal Experience. *Journal of Leisure Research*, 24(1), pp.93–94. doi:https://doi.org/10.1080/00222216.1992.11969876.

Brockmyer, et al. (2009). The development of the Game Engagement Questionnaire: A measure of engagement in video game-playing. Journal of Experimental Social Psychology, 45(4), pp.624–634. doi:https://doi.org/10.1016/j.jesp.2009.02.016.

Cairns, P., Cox, A. and Nordin, A.I. (2014) “Immersion in Digital Games: Review of Gaming Experience Research,” *Handbook of Digital Games*, pp. 337–361. Available at: <https://doi.org/10.1002/9781118796443.ch12>.

Cechanowicz, J., Gutwin, C., Bateman, S., Mandryk, R.L. and Stavness, I. (2014). Improving player balancing in racing games. *Annual Symposium on Computer-Human Interaction in Play*. doi:https://doi.org/10.1145/2658537.2658701.

Denisova, A. and Cairns, P. (2019) “Player experience and deceptive expectations of difficulty adaptation in Digital Games,” *Entertainment Computing*, 29, pp. 56–68. doi:<https://doi.org/10.1016/j.entcom.2018.12.001>.

Dziedzic, D. and Włodarczyk, W. (2018) “Approaches to measuring the difficulty of games in Dynamic Difficulty Adjustment Systems,” *International Journal of Human–Computer Interaction*, 34(8), pp. 707–715. doi:<https://doi.org/10.1080/10447318.2018.1461764>.

Hagelback, J. and Johansson, S.J. (2009) “Measuring player experience on runtime dynamic difficulty scaling in an RTS game,” *2009 IEEE Symposium on Computational Intelligence and Games*. doi:<https://doi.org/10.1109/cig.2009.5286494>.

Hunicke, R. (2005) “The case for Dynamic Difficulty Adjustment in games,” *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology* [Preprint]. Available at: <https://doi.org/10.1145/1178477.1178573>.

Costa, M. and Luís Magalhães (2017). Adaptive content generation for games. *2017 24o Encontro Português de Computação Gráfica e Interação*. doi:https://doi.org/10.1109/epcgi.2017.8124303.

Chen, J. (2007). Flow in games (and everything else). *Communications of the ACM*, 50(4), p.31. doi:https://doi.org/10.1145/1232743.1232769.

Koster, R. (2013). *Theory of Fun for Game Design*. ‘O’Reilly Media, Inc.’

Mi, Q. and Gao, T. (2022). Improved Belgian AI Algorithm for Dynamic Management in Action Role-Playing Games. *Applied Sciences*, 12(22), p.11860. doi:https://doi.org/10.3390/app122211860.

Mi, Q. and Gao, T. (2023). General Dynamic Difficulty Adjustment System for Major Game Genres. *Lecture notes on data engineering and communications technologies*, 177, pp.189–200. doi:https://doi.org/10.1007/978-3-031-35836-4\_21.

Moon, J., Choi, Y., Park, T., Choi, J., Hong, J.-H. and Kim, K.-J. (2022). Diversifying dynamic difficulty adjustment agent by integrating player state models into Monte-Carlo tree search. *Expert Systems with Applications*, 205, p.117677. doi:https://doi.org/10.1016/j.eswa.2022.117677.

Paraschos, P.D. and Koulouriotis, D.E. (2022) “Game difficulty adaptation and experience personalization: A literature review,” *International Journal of Human–Computer Interaction*, 39(1), pp. 1–22. doi:<https://doi.org/10.1080/10447318.2021.2020008>.

Reis, S., Luís Paulo Reis and Lau, N. (2019). Automatic Generation of a Sub-optimal Agent Population with Learning. *Advances in intelligent systems and computing*. doi:https://doi.org/10.1007/978-3-030-16184-2\_7.

Reis, S., Reis, L.P. and Lau, N. (2020). Game Adaptation by Using Reinforcement Learning Over Meta Games. *Group Decision and Negotiation*. doi:https://doi.org/10.1007/s10726-020-09652-8.

Shakhova, M. and Zagarskikh, A. (2019). Dynamic Difficulty Adjustment with a simplification ability using neuroevolution. *Procedia Computer Science*, 156, pp.395–403. doi:https://doi.org/10.1016/j.procs.2019.08.219.

Sutton, R.S. and Barto, A. (2018). *Reinforcement learning : an introduction*. Cambridge, Ma ; Lodon: The Mit Press.

Sweetser, P. and Wyeth, P. (2005). GameFlow: A Model for Evaluating Player Enjoyment in Games. *Computers in Entertainment*, 3(3), p.3. doi:https://doi.org/10.1145/1077246.1077253.

Sweetser, P., Johnson, D. and Wyeth, P. (2012). Revisiting the GameFlow model with detailed heuristics. *Journal of Creative Technologies*, [online] 2012(3), pp.1–16. Available at: https://eprints.qut.edu.au/58216/ [Accessed 4 Sep. 2023].

Tesauro, G. (1995). Temporal Difference Learning and TD-Gammon. *ICGA Journal*, [online] 18(2), pp.88–88. doi:https://doi.org/10.3233/icg-1995-18207.

Watkins, C. J. C. H. (1989), 'Learning from Delayed Rewards', PhD thesis, King's College, Oxford.

Wirth, W., Hartmann, T., Böcking, S., Vorderer, P., Klimmt, C., Schramm, H., Saari, T., Laarni, J., Ravaja, N., Gouveia, F.R., Biocca, F., Sacau, A., Jäncke, L., Baumgartner, T. and Jäncke, P. (2007). A Process Model of the Formation of Spatial Presence Experiences. *Media Psychology*, 9(3), pp.493–525. doi:https://doi.org/10.1080/15213260701283079.

Zohaib, M. (2018) “Dynamic Difficulty Adjustment (DDA) in Computer Games: A Review,” *Advances in Human-Computer Interaction*, 2018, pp. 1–12. doi:<https://doi.org/10.1155/2018/5681652>.

# Bibliography

Adobe (2019). *Mixamo*. [online] Mixamo.com. Available at: https://www.mixamo.com/.

Unreal Engine. (n.d.). *Infinity Blade: Effects in Epic Content - UE Marketplace*. [online] Available at: https://www.unrealengine.com/marketplace/en-US/item/38a3355872ef41bab5d2ba5b21c29eba.

Unreal Engine. (n.d.). *Infinity Blade: Weapons in Epic Content - UE Marketplace*. [online] Available at: https://www.unrealengine.com/marketplace/en-US/item/94896689fea14a308a77640f26978304 [Accessed 4 Sep. 2023].

Unreal Engine. (n.d.). *Paragon: Grux in Epic Content - UE Marketplace*. [online] Available at: https://www.unrealengine.com/marketplace/en-US/product/paragon-grux.

Unreal Engine. (n.d.). *Paragon: Revenant in Epic Content - UE Marketplace*. [online] Available at: https://www.unrealengine.com/marketplace/en-US/product/paragon-revenant [Accessed 4 Sep. 2023].

Unreal Engine. (n.d.). *Paragon: Sevarog in Epic Content - UE Marketplace*. [online] Available at: https://www.unrealengine.com/marketplace/en-US/product/paragon-sevarog [Accessed 4 Sep. 2023].

# Appendix

## A: Survey

Participants anonymously completed the following questions in a Google Forms survey.

### Pre-Game Question:

Please record the number located on the left side of the Main Menu in this section.

What is the Number?

* 0
* 1
* 2

### Post-Game Questions:

How long did you spend in the application?

Please provide duration in seconds (rounded down if you have decimals).

Please indicate how you felt while playing the game for each of the items, on the following scale:

not at all          slightly          moderately          fairly          extremely

0             1                    2        3         4

1. I thought it was hard.
2. I felt pressured by the enemy during the encounter.
   1. (old) I felt pressured.
3. I felt appropriately challenged by the encounter.
   1. (old) I felt challenged.
4. I felt time pressure to complete the game quickly.
   1. (old) I felt time pressure.
5. I had to put a lot of effort into it.
6. I felt frustrated by the difficulty.
   1. (old) I felt frustrated.
7. I felt frustrated by the controls/mechanics.
   1. (old) I felt frustrated.
8. I was fully occupied with the game during my participation.
   1. (old) I was fully occupied with the game.
9. I was deeply concentrated in the game.
10. It felt like a victory.
11. I felt satisfied.
12. I felt powerful.
13. I felt proud.
14. I felt bad.
15. I found it a waste of time.
16. I felt that I could have done more useful things in-game.
    1. (old) I felt that I could have done more useful things.

If you have any other feedback, please provide it in the section below.