

# **Utilizing Dynamic Difficulty Adjustment for improving player experience**

*Kieran Mifsud*

*Owen Sacco*

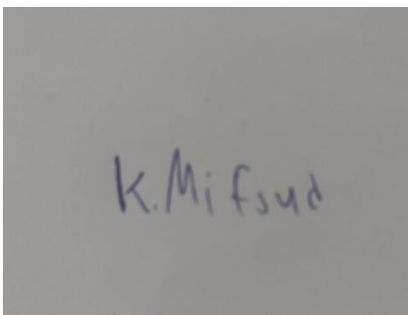
**June 2021**

A dissertation submitted to the MCAST Institute of Information & Communication Technology in partial fulfilment of requirements of the Bachelor of Science (Honours) in Multimedia Software Development.

## Authorship Statement

This dissertation is based on the results of research carried out by myself, is my own composition, and has not been previously presented for any other certified or uncertified qualification. The research was carried out under the supervision of Mr. Owen Sacco.

Signed:

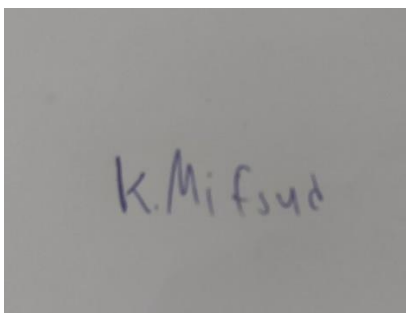
A rectangular box containing a handwritten signature in blue ink that reads "K. Mifsyd".

Date: 8<sup>th</sup> June 2021

## Copyright Statement

In submitting this dissertation to the MCAST Institute of Information and Communication technology I understand that I am giving permission for it to be made available for use in accordance with the regulations of MCAST and the Library and Learning Resource Centre. I accept that my dissertation may be made publicly available at MCAST's discretion.

Signed:

A grey rectangular box containing the handwritten signature 'K. Mifsud' in blue ink.

Date: 8<sup>th</sup> June 2021

## **Acknowledgements**

Firstly, I would like to thank my mentor Mr. Owen Sacco for accepting to be my mentor and assisting me in the completion of this dissertation. I would also like to thank my family and friends for their endless support and encouragement throughout the process of this dissertation.

## Table of Contents

<b>Authorship Statement.....</b>	<b>2</b>
<b>Copyright Statement.....</b>	<b>3</b>
<b>Acknowledgements .....</b>	<b>4</b>
<b>Abstract.....</b>	<b>9</b>
<b>1. Introduction.....</b>	<b>10</b>
1.1 Scope .....	10
1.2 Relevance of research .....	10
1.3 Method of investigation .....	10
<b>2. Literature review .....</b>	<b>12</b>
2.1 Literature review introduction .....	12
2.2 Player experience .....	12
2.3 Dynamic Difficulty adjustment (DDA) .....	13
2.4 Artificial intelligence (AI).....	15
2.5 Machine learning .....	16
2. 6 Machine learning agents (ML-agents) .....	19
2.7 Reinforcement learning .....	26
<b>3. Methodology .....</b>	<b>28</b>
3.1 Methodology Introduction .....	28
3. 2 Tools and Resources .....	28
3.3 Implementation .....	29
3.3.1 Creation of Difficulty level System (Easy and Hard levels) .....	29
3.3.2 Player Car.....	29
3.3.3 ML-Agent implementation.....	30
3.3.4 Track .....	31
3.3.5 Training of Agent.....	32
3. 4 Training of different agents .....	34
3.5 Testing of agent .....	39
3.6 Dynamic Difficulty Adjustment (DDA) implementation into ml-agents.....	39
3.7 Methodology summary .....	41
<b>4 Findings.....</b>	<b>42</b>
4.1 Overview .....	42
4.2 Analysis of implementation and Limitations.....	42
4.3 Research method for data collected .....	43
4.3.1 Data collected from participants during the prototype testing phase .....	43

<b>4.3.2 Survey.....</b>	<b>47</b>
<b>5 Discussion of Results.....</b>	<b>57</b>
<b>5.1 Discussion concerning game challenge.....</b>	<b>57</b>
<b>5.2 Discussion concerting player experience.....</b>	<b>64</b>
<b>5.3 Observations.....</b>	<b>69</b>
<b>5.4 Conclusion of discussion.....</b>	<b>71</b>
<b>6. Conclusion .....</b>	<b>72</b>
<b>7. References.....</b>	<b>74</b>
<b>8. Appendix.....</b>	<b>76</b>

## Table of Figures

Figure 1 flow channel .....	15
<i>Figure 2 amount close calls and deaths for player 1 .....</i>	<i>17</i>
<i>Figure 3 amount close calls and deaths for player 2.....</i>	<i>18</i>
Figure 4 amount close calls and deaths for player 3 .....	18
Figure 5 hide scenario agent training.....	20
Figure 6 improvement of hide scenario agent training. ....	22
<i>Figure 7 escape scenario agent training .....</i>	<i>24</i>
<i>Figure 8 reinforcement learning cycle.....</i>	<i>25</i>
<i>Figure 9 ML-agent with sensors .....</i>	<i>30</i>
Figure 10 checkpoint that aids the agent.....	30
<i>Figure 11 track used for racing and training. ....</i>	<i>31</i>
Figure 12 learning cycle for the agent .....	33
<i>Figure 13 cumulative reward for Agent 1.....</i>	<i>34</i>
<i>Figure 14 Episode length for Agent 1 .....</i>	<i>34</i>
<i>Figure 15 cumulative reward for Agent 2.....</i>	<i>36</i>
<i>Figure 16 episode length for Agent 2.....</i>	<i>36</i>
<i>Figure 17 Cumulative reward for Agent 3.....</i>	<i>38</i>
<i>Figure 18 episode length for Agent 3.....</i>	<i>38</i>
<i>Figure 19 Track with the 7 colliders used for DDA.....</i>	<i>40</i>
<i>Figure 20 Script used to show which car is ahead. ....</i>	<i>40</i>
<i>Figure 21 Different speeds for the AI agent .....</i>	<i>41</i>
Figure 22 Results of question 1.....	47
Figure 23 results of question 2.....	48
Figure 24 results of question 3.....	48
Figure 25 results of question 4.....	49
Figure 26 results of question 5.....	50
Figure 27 results of question 6.....	50
Figure 28 results of question 7.....	51
Figure 29 results of question 8.....	52
Figure 30 results of question 9.....	52
Figure 31 results of question 10.....	53
Figure 32 results of question 11 .....	54
<i>Figure 33 results of question 12 .....</i>	<i>55</i>

Figure 34 results of question 13.....	56
Figure 35 results of question 6.....	62
Figure 36 results of question 7.....	62
Figure 37 results of question 8.....	63
Figure 38 results of question 3.....	64
Figure 39 results of question 4.....	64
Figure 40 results of question 9.....	65
Figure 41 results of question 10.....	66
Figure 42 results of question 11.....	67
Figure 43 answers of question 12 part 1.....	67
Figure 44 answers of question 12 part 2.....	68
Figure 45 results of question 5.....	68
Figure 46 Hide scenario agent training.....	69
Figure 47 Escape scenario agent training.....	70
Figure 48 the agent training used in this study.....	70

## List of Tables

Table 1 Shows lap times/ difference in time/ Winner.....	44
Table 2 Shows winner of every level.....	46
Table 3 Shows winner of every level.....	58
Table 4 average of time difference on winning times.....	58
Table 5 Shows lap times/ difference in time/ Winner.....	59



## **Abstract**

In racing games, the speed of AI opponents is predetermined based on the level they are in and does not make use of dynamic difficulty adjustment (DDA) to adjust the speed of the opponent during the race. The goal of this study is to create an AI that uses machine learning Agents and trains by use of reinforcement learning, which is then implemented with DDA to try to provide a better gaming challenge and player experience than the traditional difficulty level system (easy and hard levels).

An AI agent was trained to drive around the track and try to achieve the best possible lap time using reinforcement learning. Participants had to play the easy, hard, and DDA levels and try to beat and achieve their best possible time. From the prototype, the lap times of the participants and the AI were collected, then the participants answered the survey and the data from the survey was also used to achieve the goal of this study.

# **1. Introduction**

## **1.1 Scope**

The main purpose of this study is to see if Dynamic difficulty adjustment (DDA) can be implemented with ML-agents that will use reinforcement learning to train and then will be compared to a Difficulty level system which will have a non-adaptive AI with easy and hard levels to see which system provides the best gaming challenge and player experience. To evaluate which system is the most optimal for the gaming challenge and player experience appropriate data will be collected from each participant while they are playing the prototype. Additionally, a survey will be used to gather more needed data to see which system will be the most optimal and most preferred.

## **1.2 Relevance of research**

If the ideal outcomes are achieved, this study will help prove which system between the dynamic difficulty adjustment implemented with ML-agents and the traditional difficulty system gives the best gaming challenge and player experience.

## **1.3 Method of investigation**

In order to arrive at a suitable conclusion and answer the research questions, numerous key literature sources will be researched in the hope of learning more about how a better player experience could be achieved and evaluated, learning more about what machine learning is, how it works and how aspects of machine learning such as the ML- agents and reinforcement learning could be used together to create an AI. In addition, papers on how dynamic difficulty adjustment could possibly be implemented and cases of how it has been used with ML -agents using reinforcement learning.

Using the information obtained from this study, a game will be created consisting of a racetrack and a player car and an AI car. An easy and hard AI will be created for their respective levels where those AI's will be non-adaptive and scripted. For the DDA, which will be integrated into the ML agent system, an AI agent will be trained by use of reinforcement learning. Once the training of the AI agent is complete, the participants will have to compete against all three AI's and try to beat them and achieve their best lap times. Data will be collected from each participant's game and then a survey will be given out to the participants to collect data on their experience with the game to come to a conclusion on which system is best and most preferred.

## **2. Literature review**

### **2.1 Literature review introduction**

This chapter discusses work on player experience, dynamic difficulty adjustment (DDA), machine learning, reinforcement learning, ML- agents, and artificial intelligence is reviewed. This research will be used to assist in the creation of the prototype, as key features will be explored to assist in the execution of the thesis.

### **2.2 Player experience**

In Human-Computer Interaction (HCI) there are multiple research contributions which are similar to the methodology for calculating player experience in-game. The idea of player experience is often replaced by other concepts such as flow, fulfilment, engagement, playability, and pleasure. Player experience does not depend on a single or specific emotion, but is based on a wide variety of emotions that together are responsible for the player's experience. For example, when a player plays a game, they are faced with a variety of emotions such as fear, excitement, anger, relief, and more. Together, the different emotions make up the elements of the gaming experience. **(Khong et al., 2011)**

Player behavioural modelling is a research area in game development that is obtaining popularity from game developers. It focuses on making models of player behaviours and making use of the models in-game. Taking into consideration the increased complication of brand-new video games, player modelling is extremely important to specifically define and adapt player experience. Typically, a layer model is a summarized representation of a player in a game environment. **(Lankveld et al., 2012)**

In this paper, a neural network model was used to automatically create new levels to enhance the player's experience. The game used for this study was modified version of Infinite Mario bros. To make this model happen three types of data were collected to train the model on which

are: Controllable features of the game, which are the parameters used to generate the level and to decide the difficulty and type of the level. Gameplay characteristics, which indicated how well the player is doing in the game depending of factors like amount of time taken and deaths. Players experience playing the game, this is found out by means of a survey where the participant choose which version of the game had the most emotional state which in this test where fun, challenge and frustration. **(Pederson et al., 2009)**

In this study, the artificial neural network (ANN) and the fuzzy-NN where the two AI techniques used to model player satisfaction in real-time and to Quantitatively evaluate how the qualitative challenging and curious factors contribute to human entertainment in video games. The two AI techniques are trained on gameplay through artificial evolution data to appropriate the function among the examined entertainment factors and satisfaction of the player. The test-bed is a modified version of Pac man is used which is a prey/predator type of game. From the results it shows that the ANN offers a more precise model for player satisfaction in prey/predators games compared to previous models designed for this game genre. **(Yannakakis et al., 2006)**

### **2.3 Dynamic Difficulty adjustment (DDA)**

In this work DDA is explored to see how it can affect the player progress by manipulating the supply and demand of several items. Also, it discusses on how these alterations affect the players experience of enjoyment or frustration and their perception of the game's difficulty. Balance is extremely important in games and for a DDA system to be successful it is a must to maintain the games internal balance and feedback mechanism. If these two factors are left unintended there can be consequences that deny the player of achieving a goal. An example of this is the "rubber band" adjustment AI feature, where the game cheats by giving a boost to the

losing car in the final moments in a race. Players can exploit this in order to win. (**Hunicke, 2005**)

DDA is a proposed answer for a challenging game for every single player that plays the same game since most players get either bored or frustrated. Since every player has a different skill level at the beginning of each game and will sharpen their skills at a different rate. Nevertheless, most DDA techniques are mainly based on basic parameter tweaking and designer intuition which could not reflect actual play patterns and is not adaptable to a lot of games because, they are connected to features that are more demanding to adjust dynamically, such as level design. To counter these challenges (**Smith et al., 2010**) presented Polymorph. Polymorph uses techniques from level generation and machine learning to know the player's skill and game component difficulty, by creating a 2D platform game with a constant suitable challenge. This is believed to create a unique play experience because the changes being made are both structural and personalized.

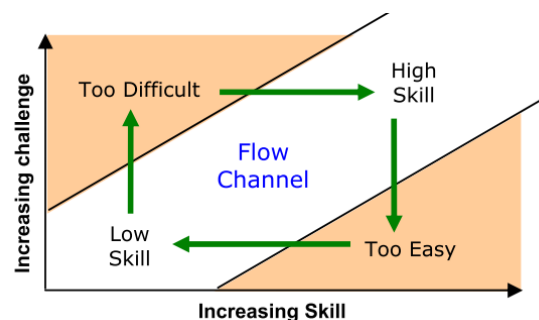
In this study, it was proposed to implement DDA mechanics into a Tetris game to see if it would improve the player experience (PX) by way of a controlled experiment where the chosen participants play Tetris without any DDA mechanics. Additionally, the aim of this study is to increase the understanding of the Dynamic Difficulty Adjustment (DDA) mechanics if there is a significant improvement through the comparison of the two implementations of Dynamic Difficulty Adjustment (DDA). The first DDA is rDDA which in other words means ramping. Which in this case automatically increases the fall rate of the Tetris pieces based on the player's performance throughout the game. The other DDA used in this experiment is pDDA which means that the player can decide when he/she would like to increase or decrease the game's difficulty during the game. The results show that both DDA mechanics achieved a significantly better player experience (PX) in the area of action awareness merging, transformation of time,

concentration at the task at hand, and challenge-skill balance, this shows that Dynamic Difficulty Adjustment helps the players achieve a state of flow. pDDA also supports a sense of control but results in Loss of self-consciousness and transformation of time when it is compared to rDDA. (Ang, 2017)

## 2.4 Artificial intelligence (AI)

Although DDA has great advantages, it does not come affordably. Inevitably, DDA projects ends up removing power from the developer's and placed in the hands of the code, which has clear downsides for example when a game has been published and the developers need to maintain the game. This is the majority of the reason why only a couple commercial developers have produced, and even less have been released, DDA systems into their games. (Chapman, Hunnicke, 2004)

A popular method to keep the player engagement into the game is by use of the flow channel which was developed by M. Csikszentmihalyi, where the goal is to keep the player in the flow channel and distant from the too easy and too difficult regions.



*Figure 1 flow channel*

## 2.5 Machine learning

In machine learning there are two main ways to improve opponent intelligence in virtual computer games, which are on-line training and off-line training. The FPS Quake is an example of an on-line training application of machine learning, where the bot uses machine learning techniques to learn its environment and to choose short-term and long-term goals. However, artificial bots cannot learn from their errors or produce brand new tactics to fix this ineffective behaviour. These bots do not adapt to the human player's tactics but, adapt to the environment they are in. Off-line learning in games is mainly used for two reasons which are: to improve their intelligence of what opponents might do by putting them against other scripted bots so they can train and learn more situations. The other reason is to shield the opponents from any hidden tactic from the player by exploiting any “holes” in the opponent’s controller script. **(Postma et al., 2002)**

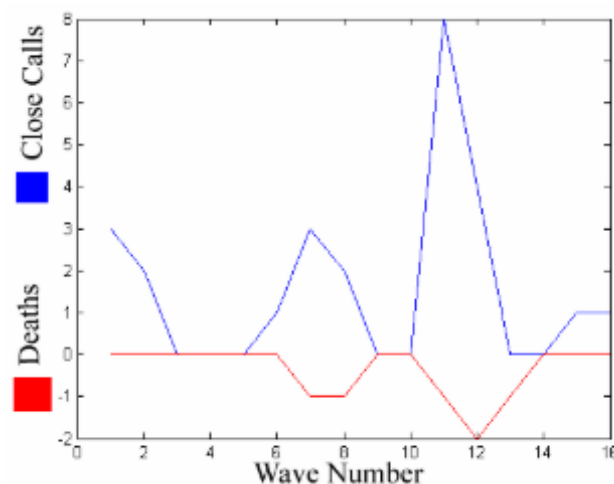
In this game of checkers two machine learning approaches were discussed. The first method which is called Neural Net approach, this works by induced learning and with a reward and punishment procedure. The second method is to have a highly organized network which is only capable of learning specific things. Since the first method works by repeating its actions to see the best result it, takes more time to be ready than the second method which makes it less efficient. On the other hand, the second needs to be reprogrammed for each application. By use of Rote learning the first method started by copying its opponent but it struggled in the middle part of the game, but it quickly learned to avoid traps and go for the win. The method with generalization never learned to play the game in a normal way and it always started the game bad. But, unlike the first method it was good in the middle game and with a piece advantage it mostly beats its opponent in a short period. **(Samuel, 1959)**

A 2-D space shooter game called office wing uses online reinforcement learning in attempt to learn the parameters with generating enemies for the player with the aims to increase the



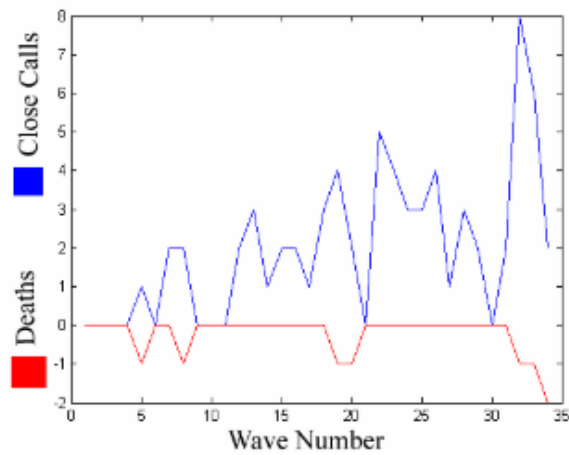
players enjoyment of the game. This works by having the level building agent follow the players recent actions and adapt based on the user's style of play. Every time the players skill increases the, "function" is changing to keep the player challenged. The game works by having a player controlling the plane and has enemies coming at him, the aim is to either eliminate or avoid the enemies to pass the level. The reward model for the level building agent to give the most player enjoyment is between the number of "close calls" and player death. The goal is to have a high number of "close calls" and a low number of deaths. (Cooper, 2005)

The plots below show the amount of close calls and deaths for each player. The first player got to wave 16 with the players deaths both times when there was a high number of close calls which made the game more challenging. After each death the difficulty was reduced.



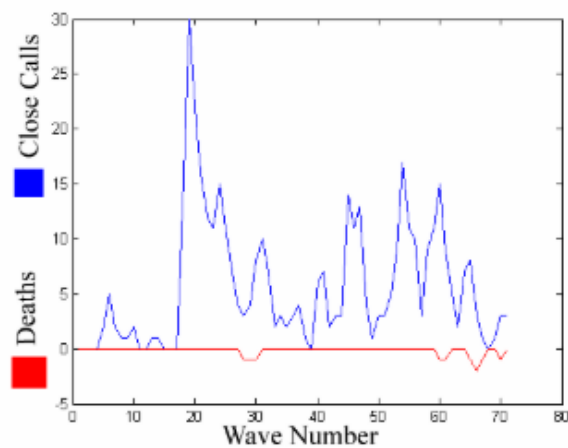
*Figure 2 amount close calls and deaths for player 1*

The second player got to 35 waves and most of the game had several close calls which means it was mostly challenging throughout the entire game.



*Figure 3 amount close calls and deaths for player 2*

The final player makes it to 72 waves and shows that the player was getting a little caught out with the increase of difficulty because it was constantly fluctuating around 8 close calls.



*Figure 4 amount close calls and deaths for player 3*

## 2. 6 Machine learning agents (ML-agents)

Reinforcement learning (RL) is the element of machine learning that examines training agents through a series of definitive solutions. For both RL researchers and game designers, the ML-agents plugin is beneficial as it provides a chance to work on a system where reinforcement learning progress can be evaluated on different Unity environments and then made understandable for deeper analysis and game developer communities.

The aim in this paper is to develop an ML-Agent that would be able to hide for an infinite period from a scripted AI and would be able to evade the standard AI's Chasing if detected. This project works by having the AI constantly moving around the environment and requires the ML-Agent to be in its field of view and then an identifiable route within that field of view. Each short amount of time the ML-agents sends raycasts to the scripted AI to find out where its view is directed. The AI refreshes its objective location to the agents position at the moment where the ML-agent is within the visible direction and the field of view. If the ML-agent is out of view, the agents last known location becomes the destination. **(Nurkey et al., 2019)**

Below there is displayed the training results of the "Hide" test. The most effective configuration was found after testing various configurations of hyperparameters values that occurred by training the ML-Agent.

The tests below show that the ML-Agents results were varying a lot and that it needed more stability because it was getting stuck a lot and confused.

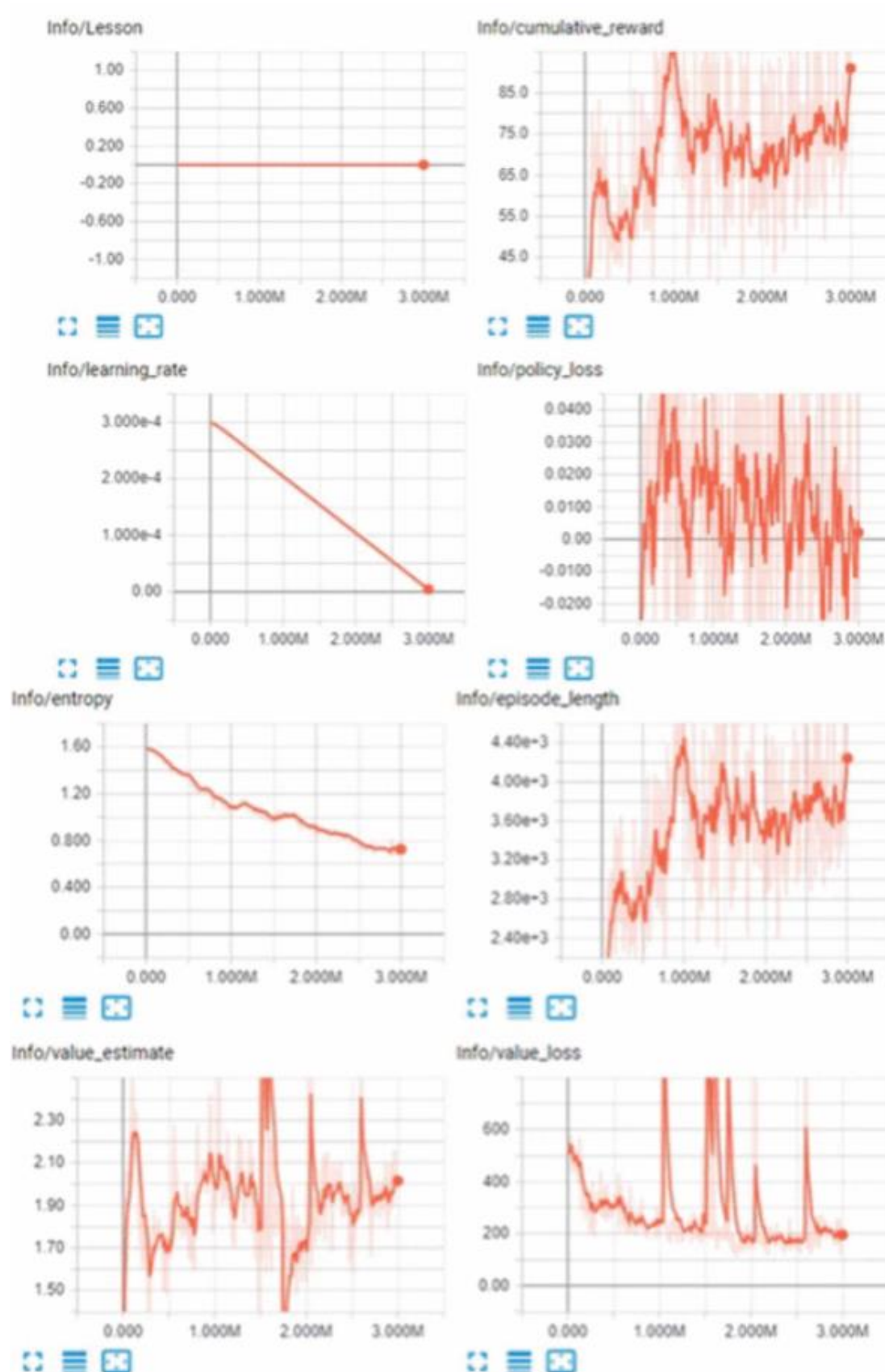


Figure 5 hide scenario agent training

Below show the updated “Hide” testing graphs. To get the ML-agent more stable the buffer size hyperparameter was increased, but there still was not a significant increase of smooth increment in cumulative reward. So, the agent needed to train until behavior enchantment occurred. With these changes the work showed improved training parameters which caused the agent’s behavior to show reduced uncertainty and less stuck events.

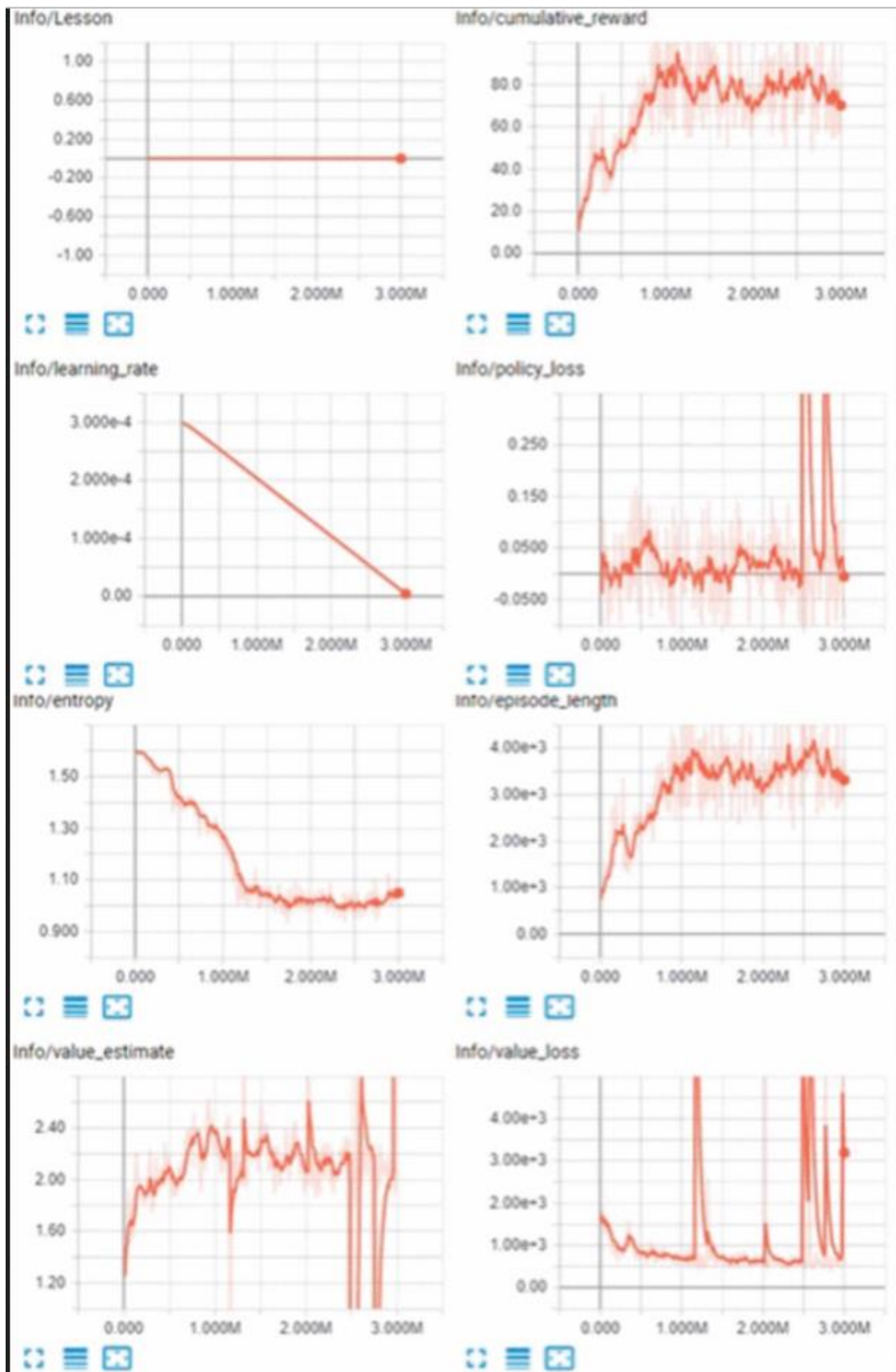


Figure 6 improvement of hide scenario agent training.

Below there is displayed the graphs for the escape scenario. Unlike the hide scenario the escape scenario worked splendidly as the accumulated reward for the chaser reached the value of 9.48 in the case of its highest speed out of 10 which is as perfect as it can achieve with the penalty feature that happens in the escape stage.

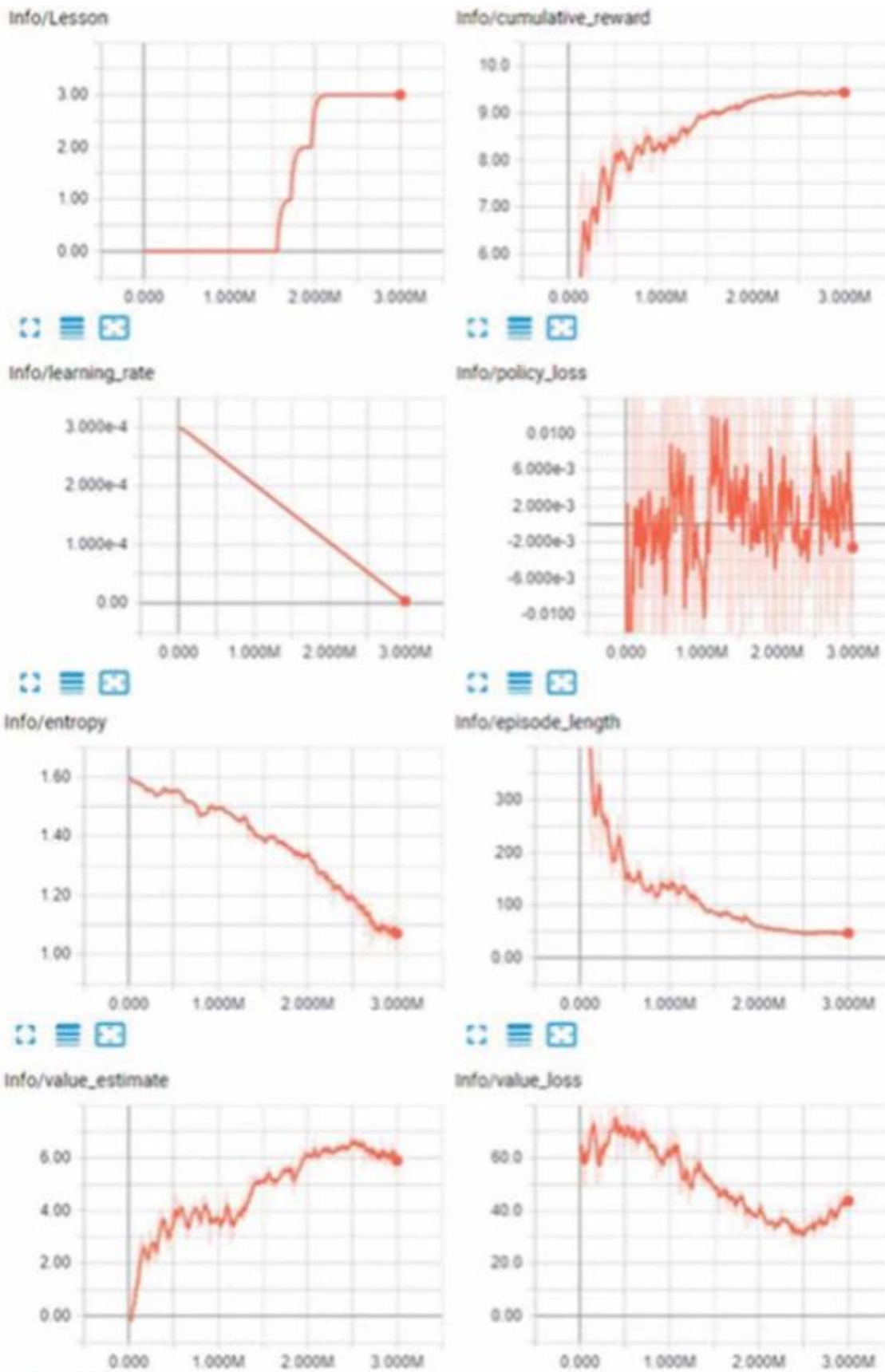


Figure 7 escape scenario agent training



This work demonstrates a single machine learning agent that learns both by reinforcement learning and by imitation learning. The reinforcing learning used in this work is the same as the previous paper by (Nurkey et al., 2019). The outcome of reinforcement learning method mostly depends on the hyperparameters settings, reward and penalty system, rules of the environment and training time.



*Figure 8 reinforcement learning cycle*

For imitation learning to happen it needs two agents, one which is the teacher and the other which is the student. The teacher can be a person, another neural network, or a deterministic algorithm. The most successful results are achieved when the teacher is an actual human. Imitation learning was implemented to train the ML-Agent in the same environment by having the teacher play around the environment and depending on the task difficulty a certain amount of time is needed for the student learn by watching the teacher's actions and try to imitate them. Comparing these two methods is a bit problematic seeing as they achieved various findings and behaviors. In addition, the intersection results might have only happened for early training

which might have been random and not sufficient. Because of recording real-time human behavior, that also takes up more human resources, the imitation learning method took more time to execute. For the most efficient outcomes the method of reinforcement learning is more beneficial. On the other hand, if the aim of the developer to create an AI which is closest to human actions, then imitation learning method would be the better choice. **(Alimanova et al., 2020)**

## **2.7 Reinforcement learning**

In this experiment a simple kart racing game is presented with the ML-Agents toolkit. For this racing game, the agents are given several laps to navigate through the course created. In this experiment the RL agent will be given a slight reward if they move in the right direction, they are given a significantly reward for going through a checkpoint and will receive a crucial punishment if the agent collides with a wall. After 45 minutes of training, most of the agents were doing quite well as they were getting close to the corner to reduce their lap times. On the other hand, some of the agents where still struggling a bit to adapt the track, but they still improved greatly from the beginning which meant that they just needed more time to properly adapt. **(Haring, 2020)**

The performance of AI is typically calculated by games and the aim is to create an opponent which plays the game better than humans. The focus of this paper is to create a suitable AI for players that aren't good at playing games. The AI player created learns by playing against a computer opponent instead of a human player. This study uses Reversi a one-on-one game as the case study used. The learning reinforcement problem with this aspect is getting the AI score close to the human players. For the system to learn at a fast pace they adopted the temporal difference (TD) method that uses state values. In this method an action is taken by

the AI player at a state  $S$  and it gets the reward  $r$  and the next state  $S$ . For this method the reward system works by giving the AI a reward when his score and opponents score are close and the closer the score gets the bigger the reward. (Yaguchi, lima, 2020)

### **3. Methodology**

#### **3.1 Methodology Introduction**

In this chapter the primary goal was to implement a ML-agent with dynamic difficulty adjustment that has the ability to navigate through the course and adjust speed according to the opponent's performance during the game. Reinforcement learning, a subtype of machine learning, was used to train the agent. With this type of learning, the AI was able to learn the course by navigating around it until it was determined that the agent's performance was good enough to be used in the game. In addition to implementing the ML agent with DDA, the other main objective was to create AIs for the Difficulty Level's that are easy and hard so that these systems can then be compared to each other.

#### **3. 2 Tools and Resources**

Tools and resources needed to build the prototype.

Unity 3D was used to create the environments, AI and to implement and train the ML-Agent.

The cars used for the player and AI were imported from the Unity assets store: Realistic Car Kit. The track texture was from Modular Track which was imported from the asset store.

The library ML-Agents Version 1.07 which is developed by Unity Technologies was used to create the agent.

From the Karting Microgame which is also developed by Unity technologies, scripts like the Arcade Kart and Kart Agent from that project were used in the prototype.

Python was used to access and launce the ml-agent training from the ml-agent library.

### **3.3 Implementation**

#### **3.3.1 Creation of Difficulty level System (Easy and Hard levels)**

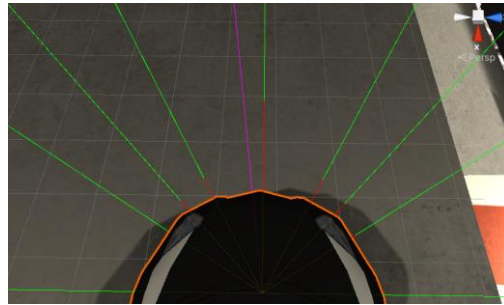
In the prototype the first action that needed to be taken was to create the easy and hard AI's for the difficulty level. For this to be achieved waypoints were put around the track to show the AI's racing line. After this an invisible cube was put in front of the car so it would be the eyes of the car and know where it is going. A waypoint tracker script was put on the AI car and was assigned to be cube. With this the car was able to keep with the racing/waypoint line that was created for it to follow even when it collides with something and gets through of the line with the help of the invisible cube in the front it is able to get back on track. The final step for the AI was to give it movement and additionally decide the amount of maximum speed, acceleration, and the deceleration when the AI turns at a 30f angle. The top speed and acceleration were calculated by having a couple a companions testing the game before it was ready to give to the participants to find the most appropriate speed for the easy and hard AI's. The same concept was used for both Difficulty levels AI, the difference was to give the Hard AI a more professional racing/waypoint line for it to follow and increase it speed to make the level more challenging.

#### **3.3.2 Player Car**

For the player car to move wheel colliders were placed at every wheel of the car which where than called in the Player Car script. The car has three main fields which are the motor force which determine the speed the car can generate, the break force which determine the amount of force when the car brakes and the max steer angle which determines the max angle that the car wheels can turn. From the way that the car is the user cannot just expect to press the brake and the car will immediately stop, they must calculate the speed of the car and be prepared to break before they arrive with the turn.

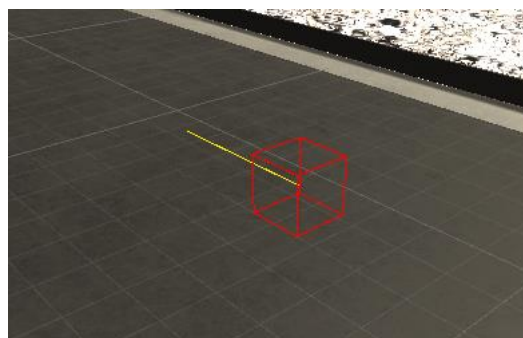
### 3.3.3 ML-Agent implementation

Initially sensors were needed before the ML-Agent could start its training. The sensors were game objects and put as children under one game object so they will always move as one. In total there are 9 game objects starting from -90 to 90 degrees. These sensors were needed so the agent could get the distance from the walls to the sensors.



*Figure 9 ML-agent with sensors*

In addition, checkpoints were created all over the map to indicate which in which direction the agent should be going. These checkpoints were cubes which only had box colliders assigned, that were wide as the track to make sure the agent will collide with them. The agent checkpoints were all put under a parent game object to group them and had a script assigned to it called “Debug checkpoint ray” which gives a yellow ray showing the direction forward to each collider of the checkpoints, so the agent knows where it needs to move to get to the next checkpoint.

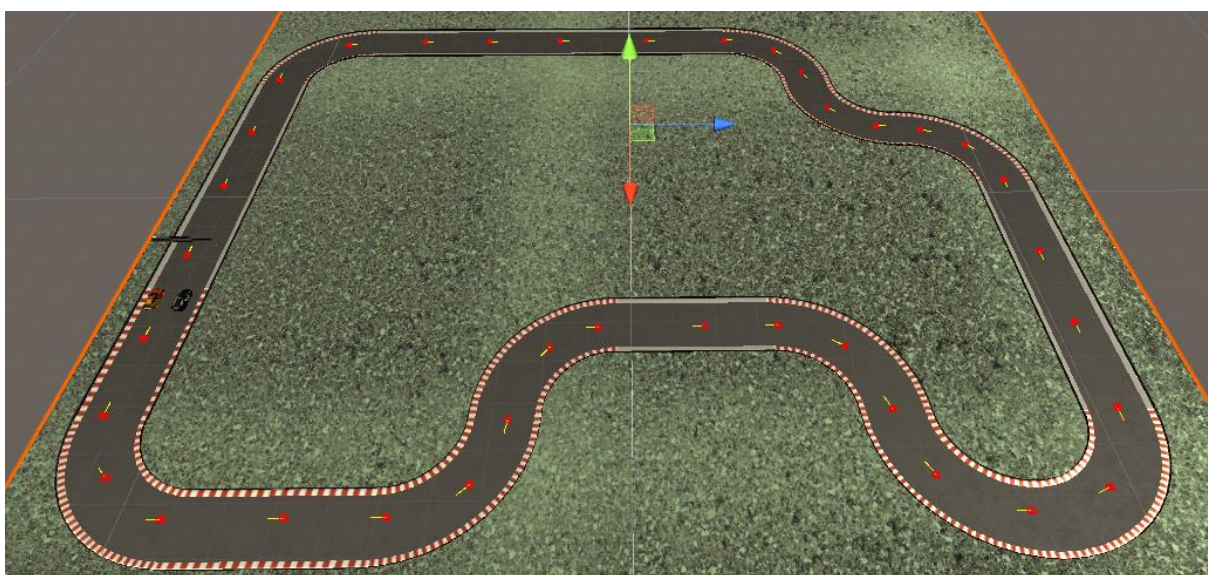


*Figure 10 checkpoint that aids the agent*

The next step was to give the AI agent the ability to move and to this a script was assigned to it which included the cars stats which were needed to make the car drive, these stats include acceleration, Top speed, Braking and turning input. These stats were then called into the “Move Vehicle” method to make the car move. The final step to make the agent able to move was to make wheel collider for each wheel of the car and assign the colliders to the script and in addition also assign the physical wheel of the car to the script.

### 3.3.4 Track

This was the track used for the AI to train on and then race on against the participants, which was created using parts of the modular track kit. This track format was used for all levels to make it fair when comparing lap times. This track has a lot of corners which are different from each other. The reasons for this are to see if the agent with the training given is able to go around every corner without any issues and since it did manage to go around the track the neural network which was trained for this agent and track will have a good chance at succeeding in simpler tracks or tracks with similar corners which most tracks have some of these types of turns.



*Figure 11 track used for racing and training.*

### 3.3.5 Training of Agent

For the AI agent to be able train three scripts are needed, these are the Decision requestor, Kart Agent and the behaviors parameters scripts.

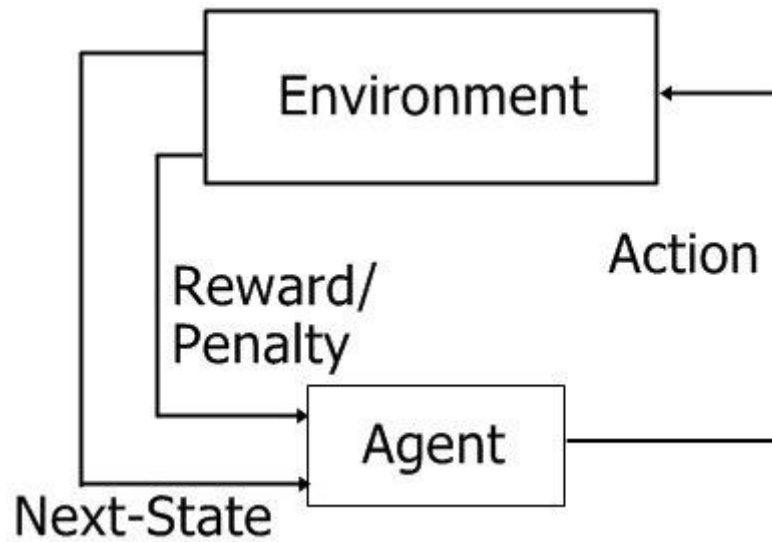
The decision requester was used to request a decision per step from the AI agent depending on the selected decision period number.

The behaviour parameter was used to let the agent interpret its actions to try and mimic human behaviour, which happens in the vector action consisting of the space type and branch sizes. The space type is set to discrete, to define the agent's actions as an array of integers, and the branches are representatives of our X and Y input axes.

The kart agent script was used to mainly observe the training environment and generate the inputs needed to move the car along with the arcade kart script. The observation parameters include the raycast distance, the sensors, and the mask. These work together to allow the AI agent to detect how close it is to the wall and detect when it crashes. These were altered by going into the sensors and then altering each hit threshold of the cars sensors to what felt appropriate so the raycasts would not be too close or too far from the walls. This script also contains the agent checkpoint colliders that are used to guide the agent while driving, the checkpoints are also used so that in any case that the agent crashes during training, it is reset to a random checkpoint and continues driving from that point. There is also the reward system in the kart agent script.

Before training can take place, it must be ensured that no neural network is assigned to the model in the behaviour parameter script and the mode in the kart agent script is set to training. A local virtual environment is then created using Python so that the Unity editor can interact with TensorFlow. The final step is to execute the command that allows the agent to start training, the agent will keep learning until it is stopped by me or exceeds its maximum steps.





*Figure 12 learning cycle for the agent*

Reinforcement learning was the training method used to train the agent, and it works by rewarding the agent depending on the actions the AI agent makes during training, this system shows if the AI does something as intended or not. For this study the chosen agent had a penalty of -1 each time it collided with the wall, it received a small reward of 0.03 each time it moved towards the checkpoint, a reward of 0.02 for increasing its speed so it could finish the lap faster and a reward of 1 for passing a checkpoint. In the event that the agent gets stuck in the wall after a crash, it will receive a penalty but will be reset to another checkpoint to continue the training.

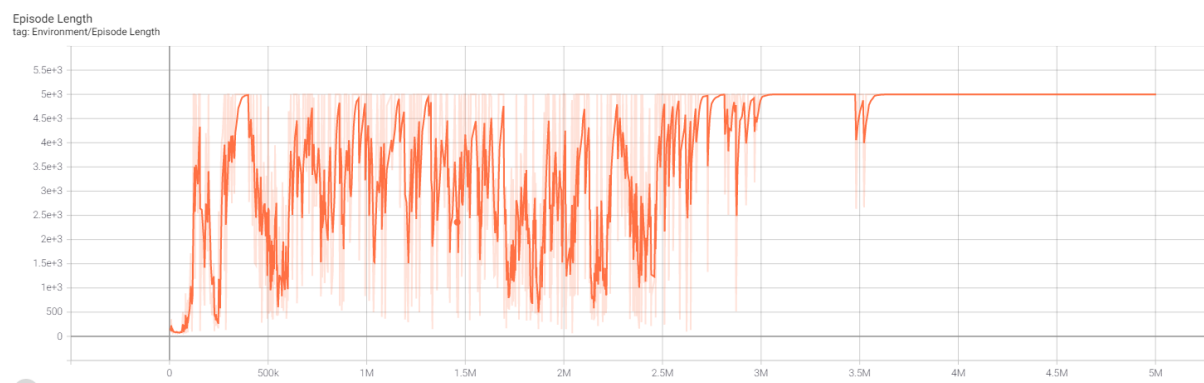
### 3. 4 Training of different agents

#### Agent 1



*Figure 13 cumulative reward for Agent 1*

As can be seen in the Figure 13, results varied greatly but around the 3M step mark, the AI agent lost a lot of performance to the extent where the agent was getting stuck most of the time and unable to complete a lap. For this reason, it was decided to let the AI agent train until the 5M step mark so that it could perhaps relearn what it had lost. From what was observed during the training, the AI agent had crashed and got stuck for a while and failed to recover from all the punishment it had received, so there was no hope of continuing the training any further.



*Figure 14 Episode length for Agent 1*

From what could be seen from the image above it seems that the agent did not manage to do much more training from the 3M mark onwards, so that might have been a main reason why it did not relearn what it had lost.

**These were the rewards for the actions of the AI agent:**

**Hit penalty -1**

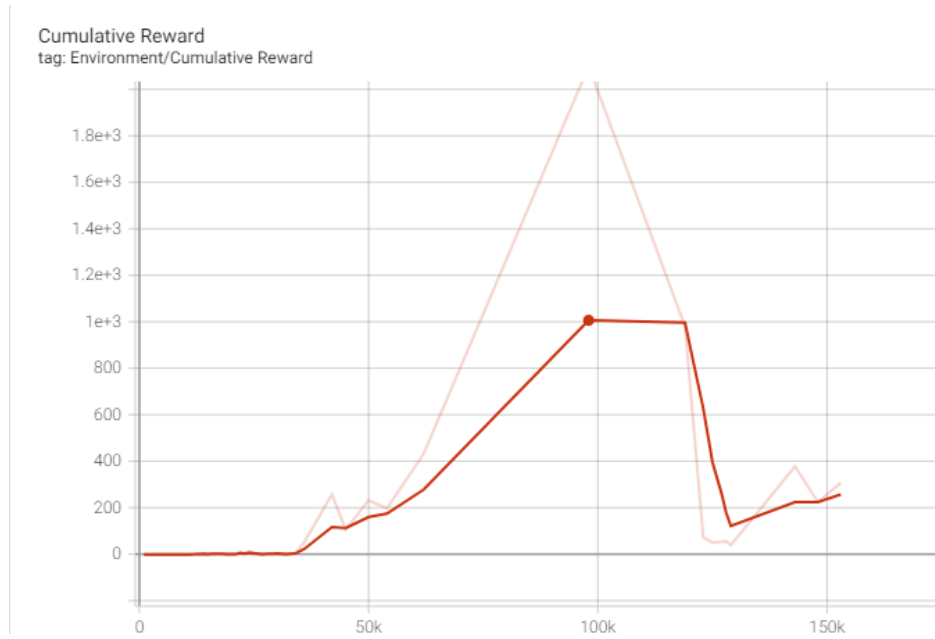
**Pass through checkpoint 1**

**Move towards checkpoint 0.02**

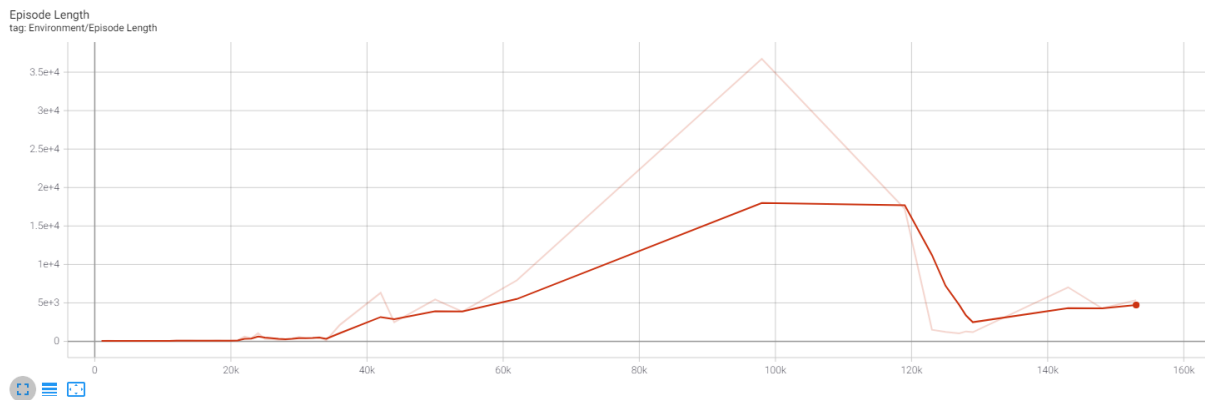
**Speed reward 0.01**

Overall, with this reward system setup the AI agent was improving with time and was doing a lot of good laps but since it had crashed and lost everything it learned it could not be used.

## Agent 2



*Figure 15 cumulative reward for Agent 2*



*Figure 16 episode length for Agent 2*

As can be seen in the above Figure, this AI agent performed very well from the beginning, but then suddenly dipped and stopped receiving rewards, even though it successfully completed

the track. The AI agent took more than 400k steps, but since it did not receive any rewards, they didn't register.

**These were the rewards for the actions of the AI agent:**

**Hit penalty -1**

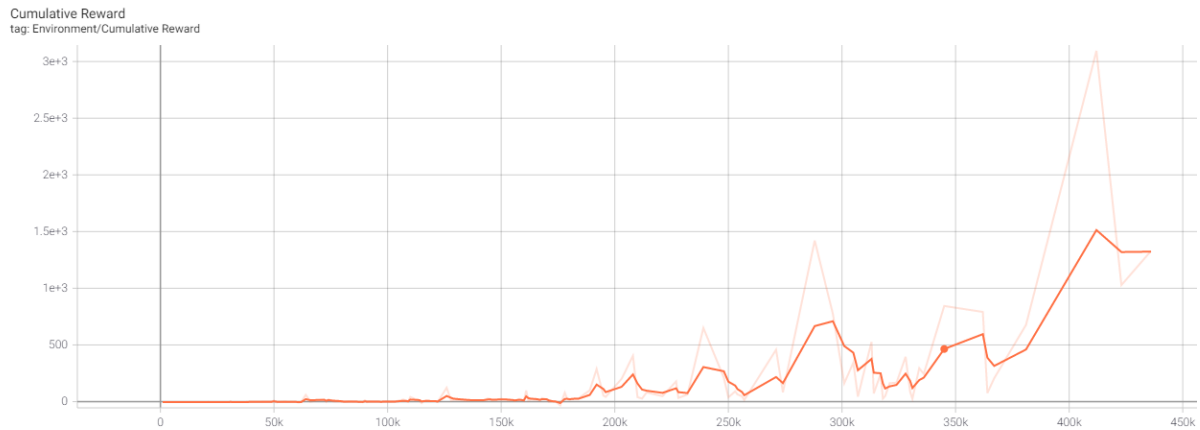
**Pass through checkpoint 1.2**

**Move towards checkpoint 0.04**

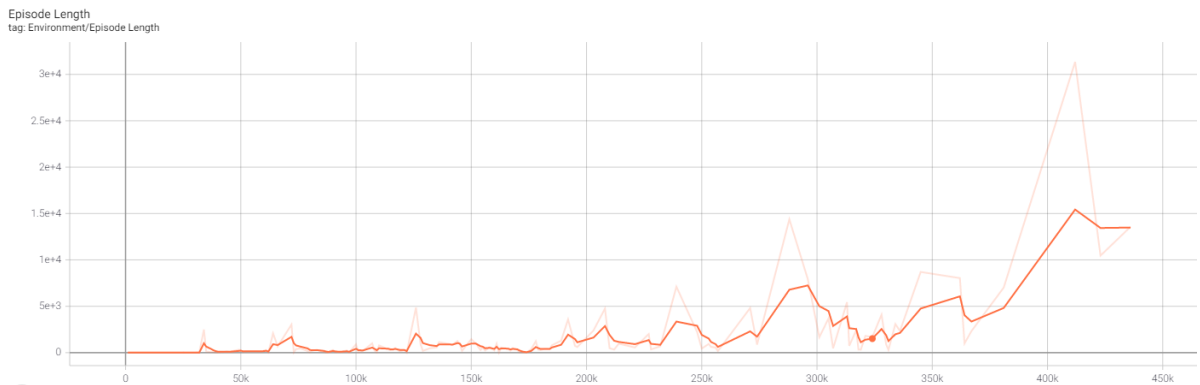
**Speed reward 0.03**

Overall, this agent when put into the track to see if it had worked was able to navigate around the track as intended but, this may have resulted in this since it was given higher rewards compared to punishment.

## Agent 3



*Figure 17 Cumulative reward for Agent 3*



*Figure 18 episode length for Agent 3*

As can be seen from the above Figure, the AI agent did not get much reward at the beginning of the training, but from the 200k step onwards it steadily went uphill, to the point where the AI agent training was paused to prevent the same problem from occurring as before, where the agent simply crashes and cannot recover. This was also done to see if the agent had trained enough and if it wasn't ready, training have resumed.

**These were the rewards for the actions of the AI agent:**

**Hit penalty -1**

**Pass through checkpoint 1**

**Move towards checkpoint 0.03**

**Speed reward 0.02**

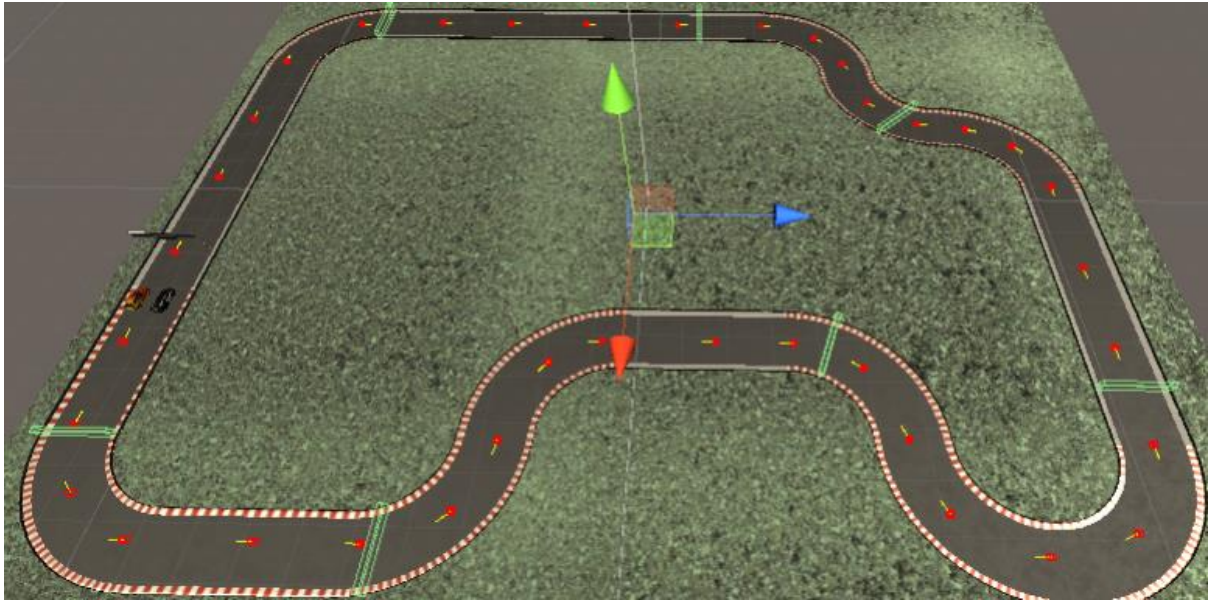
Overall, this agent performed the best from the other agents and worked as intended even though it had a lot less steps in comparison to the first AI agent and from what was planned for it.

### **3.5 Testing of agent**

After the AI agent was done with its training, the stored neural network was placed on the model of the AI's behavioural parameter scripts. In addition, the mode in the Kart agent's script was set to "Inferencing". The AI agent was then activated and allowed to drive around the track to see how it would behave independently and whether it would have difficulty with certain parts of the track. As it turned out, the AI agent had no trouble navigating the track, so it was time to introduce the player car to see if there would be any issues when the player competed against the agent. With the player versus the AI agent, the AI agent still worked successfully as intended, even when there was a collision between the two.

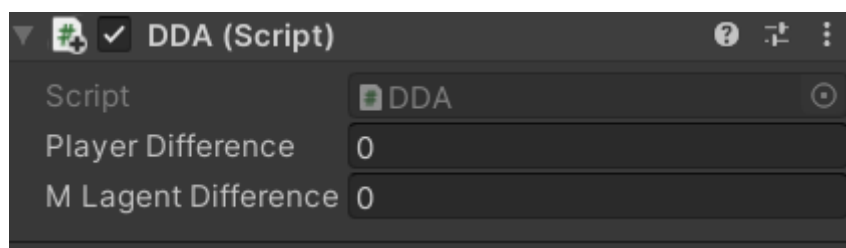
### **3.6 Dynamic Difficulty Adjustment (DDA) implementation into ml-agents**

Once it was certain that the agent was fully functional it was time to implement the dynamic difficulty adjustment (DDA) to the AI agent. The first thing that was created for the DDA were 7 empty colliders which were placed around the track were looked the most necessary to get the most out of the DDA.



*Figure 19 Track with the 7 colliders used for DDA.*

Subsequently, two scripts named "DDA" and "DDACounts" were created. The "DDACounts" script was assigned to each of the new colliders, while the "DDA" script was assigned to the AI agent. The "DDA" script was used to change the AI agent's top speed and acceleration based on who was ahead and by how much. To calculate the lead, the script "DDACounts" was used, which calculates who is ahead by seeing who goes through the colliders first, and calls the script "DDA" to show that one car is ahead of the other by giving that car +1, but as soon as the other car also goes through the collider, it reduces back to 0.



*Figure 20 Script used to show which car is ahead.*

As can be seen in the Figure below, in this case the AI agent's speed was set to 14f and acceleration to 2.5f when both vehicles are on the same collider (close to each other). On the



other hand, if the AI agent is ahead by 2 colliders or more, its top speed is reduced to 10f and its acceleration is reduced to 2f. Finally, in the case where the player is ahead by one checkpoint or more, the AI agent will receive an increase in top speed of 25f and acceleration of 4f. The top speed and acceleration were calculated by having a few companions test the level to see what speed is most appropriate for the AI agent used in the DDA system. The speed of the AI agent will always depend on the player's performance because if both cars are always with each other than the AI agent will stay at a constant speed, but if the player speeds up or slows down the AI agent will react depending on the scenario.

```
if (MlagentCar)
{
    if (MlagentDifference >= 2)
    {
        baseStats.TopSpeed = 10f;
        baseStats.Acceleration = 2f;
    }
    if (MlagentDifference <= -1)
    {
        baseStats.TopSpeed = 25;
        baseStats.Acceleration = 4;
    }
    if (MlagentDifference == 0)
    {
        baseStats.TopSpeed = 14f;
        baseStats.Acceleration = 2.5f;
    }
}
```

*Figure 21 Different speeds for the AI agent*

### **3.7 Methodology summary**

In summary, this chapter mentioned what tools were needed and used to create the machine learning agent with dynamic difficulty adaptation and how this process took place. There is also an explanation of the scripted AI with the easy and hard difficulty settings and the player car. In addition, there is a summary of the track used and the checkpoints that help the AI agent track his progress as he goes. There is an explanation of how the AI agent is trained and how the reward system for the AI agent works. Finally, the description of how the dynamic difficulty adjustment was created and implemented with the agent.

## **4 Findings**

### **4.1 Overview**

The main purpose of this study was to find out whether dynamic difficulty adjustment implemented in an artificial intelligence with machine learning provides a better gaming challenge and player experience than the difficulty level (Easy and Hard). The discussion section will evaluate this point in detail and see the result depending on the results and findings obtained from the data collected from the application and from the survey.

### **4.2 Analysis of implementation and Limitations**

During the creation of the prototype, a number of limitations were identified. Most of these limitations were related to the training of the AI agent. One main limitation was that the AI agent took a very long time to make visible progress during training. This is caused by reinforcement learning, as this type of learning usually takes a long time to train the AI agent properly. The most time-consuming limitation was that the AI agent crashed and lost all the rewards it had received during training, after making great progress and having already taken many steps. For some reason, while it was moving smoothly and constantly around the lap, it suddenly crashed and did not manage to recover from its mistake, even when it was teleported to another checkpoint to continue its training, it did not seem to respond. Unfortunately, this happened quite often when the AI agent was left to train for an extended period of time, and every time it was tested it would either stop where it started or simply crash into the wall. To fix this limitation, several adjustments had to be made to the reward system, but this caused the AI agent to start over each time, which took even more time to try out each adjustment. Once the best reward system was found, the AI agent's training was stopped when it looked like it could easily go around the track without any issues, and the AI agent was tested to see if it had enough training or if it should resume its training for more steps.

Since when tested the AI agent managed to go round the track as intended, the AI agent training was cut short even though it is recommended to allow the agent to train for a longer period of time. Also, since the AI agent did not have any issues driving on the track nor racing against the participants it did not require any additional training. This is due to the ray cast sensors that it has on the front of the car which allows it to see if it is close to crashing and will try to avoid it, this also worked when the participants were ahead of the AI agent and instead of trying to go through the participants it slowed down so it would not crash and tries to overtake them by going beside them.

### **4.3 Research method for data collected**

For this study, a mixed research approach was used, where the quantitative data was the data collected from the prototype, i.e., the lap times of all participants and also the lap times of the AIs from the easy, hard and dynamic difficulty adjustment levels. The qualitative data was the data collected from the survey where participants had to answer a series of questions after participating in the prototype. The quantitative data was collected to calculate the gaming challenge of each level by comparing player and AI lap times, in addition to asking a few questions about what participants found most challenging. The qualitative data was collected to see which system gave participants the best player experience of the two.

#### **4.3.1 Data collected from participants during the prototype testing phase**

Below shows the data collected from the participants during the game (quantitative data). It shows the lap times of the two laps that the AI and player completed for all three levels. It also includes the winner of each level between the participants and the AI's and the difference of by how many seconds the winner won by of the two laps combined.

Table 1 Shows lap times/ difference in time/ Winner

	Easy lap times	Difference in time	W/L	Hard lap times	Differe nce in time	W/L	DDA lap times	Differe nce in time	W/L
<b>Participant</b>	0.43.7	-10.1	Win	0.43.4		Lose	0.49.1	-1.4	Win
<b>1</b>	0.39.0			0.37.1			0.36.1		
<b>AI</b>	0.48.9		Lose	0.41.3	+0.4	Win	0.45.4		Lose
	0.43.9			0.38.8			0.41.2		
<b>Participant</b>	0.53.0		Lose	0.49.2		Lose	0.54.0	+0.8	Win
<b>2</b>	0.46.4			0.42.1			0.44.3		
<b>AI</b>	0.48.7	+6.7	Win	0.42.7	+10.9	Win	0.49.8		Lose
	0.43.7			0.38.7			0.49.3		
<b>Participant</b>	0.42.8	-10.8	Win	0.42.4		Win	0.42.9	-2.8	Win
<b>3</b>	0.38.8			0.36.7	-1.0		0.34.3		
<b>AI</b>	0.49.0		Lose	0.41.3		Lose	0.45.2		Lose
	0.43.4			0.38.8			0.34.7		
<b>Participant</b>	0.52.5		Lose	0.52.0		Lose	0.48.0		Lose
<b>4</b>	0.43.7			0.49.9			0.53.0		
<b>AI</b>	0.48.7	+3.7	Win	0.42.7	+20.5	Win	0.47.3		Win
	0.43.8			0.38.7			0.50.8	+2.9	
<b>Participant</b>	0.46.7		Lose	0.51.3		Lose	0.50.7	-3.6	Win
<b>5</b>	0.47.5			0.52.0			0.48.1		
<b>AI</b>	0.49.2	+1.0	Win	0.42.7		Win	0.49.9		Lose
	0.44.0			0.38.7			0.52.5		

<b>Participant</b> <b>6</b>	0.47.4 0.41.4	-0.1	Win	0.44.5 0.43.6		Lose	0.51.9 0.42.0	-3.5	Win
<b>AI</b>	0.47.7 0.41.2		Lose	0.42.4 0.38.8	+6.9	Win	0.51.6 0.43.8		Lose
<b>Participant</b> <b>7</b>	1.01.7 0.58.1		Lose	1.03.4 0.58.9		Lose	1.00.8 1.00.3		Lose
<b>AI</b>	0.48.9 0.43.3	+27.6	Win	0.42.5 0.38.6	+41.2	Win	0.56.9 0.55.4	+8.8	Win
<b>Participant</b> <b>8</b>	0.44.6 0.56.1		Lose	0.47.7 0.43.6		Lose	0.50.8 0.40.5	-1.5	Win
<b>AI</b>	0.48.9 0.43.5	+8.3	Win	0.42.6 0.38.5	+10.2	Win	0.50.3 0.42.5		Lose
<b>Participant</b> <b>9</b>	0.54.1 0.41.5		Lose	0.48.3 0.44.9		Lose	0.46.1 0.41.2	-0.4	Win
<b>AI</b>	0.49.3 0.43.6	+3.7	Win	0.43.0 0.38.9	+11.9	Win	0.47.7 0.40.0		Lose
<b>Participant</b> <b>10</b>	1.03.2 0.57.0		Lose	0.48.7 1.01.7		Lose	0.46.0 0.51.7		Lose
<b>AI</b>	0.48.6 0.43.1	+28.5	Win	0.42.2 0.38.4	+29.8	Win	0.46.3 0.50.2	+1.2	Win
<b>Participant</b> <b>11</b>	0.52.1 0.55.7		Lose	0.55.0 1.02.8		Lose	0.53.8 0.44.3	-0.9	Win
<b>AI</b>	0.48.8 0.43.5	+15.5	Win	0.42.6 0.38.5	+36.7	Win	0.52.2 0.46.8		Lose

<b>Participant</b>	1.03.9		Lose	1.00.4		Lose	0.57.7		Lose
<b>12</b>	1.06.9			1.07.3			0.55.7		
<b>AI</b>	0.48.8		Win	0.42.6		Win	0.53.4	+3.7	Win
	0.43.4	+38.6		0.38.5	+46.6		0.56.3		
<b>Participant</b>	0.45.1	-7.8	Win	0.42.9		Lose	0.46.0	-1.3	Win
<b>13</b>	0.39.1			0.40.1			0.41.8		
<b>AI</b>	0.48.8		Lose	0.41.5		Win	0.47.7		Lose
	0.43.2			0.38.4	+3.1		0.41.4		
<b>Participant</b>	0.48.1		Lose	0.55.1		Lose	0.51.5	-0.5	Win
<b>14</b>	0.47.8			0.49.1			0.44.8		
<b>AI</b>	0.47.5		Win	0.42.4		Win	0.51.4		Lose
	0.43.4	+4.0		0.38.4	+23.4		0.45.4		
<b>Participant</b>	0.41.2	-9.1	Win	0.42.5	-2.8	Win	0.43.6	-2.3	Win
<b>15</b>	0.40.9			0.35.9			0.37.2		
<b>AI</b>	0.48.9		Lose	0.42.4		Lose	0.44.9		Lose
	0.43.3			0.38.8			0.38.2		

The table below shows how many times the participants and AI's had won these levels.

*Table 2 Shows winner of every level*

Total wins and losses	Easy	Hard	DDA
Player	5	2	11
AI	10	13	4

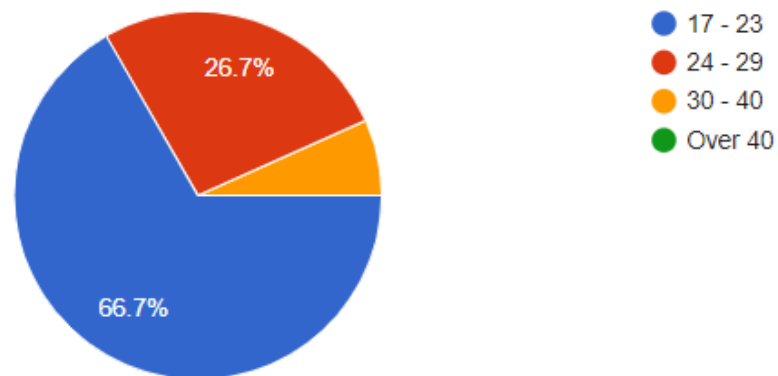
### 4.3.2 Survey

#### Question 1

Participants were asked to choose which age group they belong to in this question. 66.7% (10) were between the ages of 17-23 years old, 26.7% (4) were between the ages of 24-29 years old and only 6.7% (1) participant was between the ages of 30-40 years old.

#### 1. What is your age Group?

15 responses



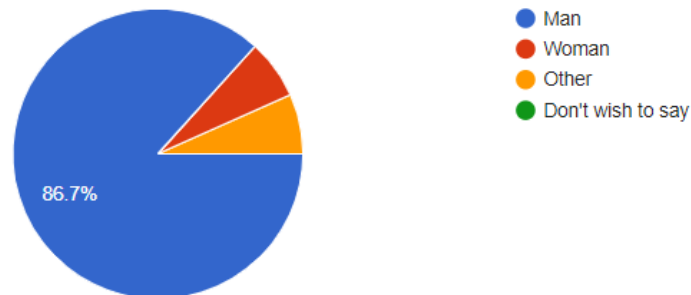
*Figure 22 Results of question 1*

#### Question 2

In this question participants were asked to specify their gender if they wanted to. 86.7% (13) participants identified as a man. 6.7% (1) one participant identified as a woman and 6.7% (1) another one identified as other.

## 2. What is your gender?

15 responses



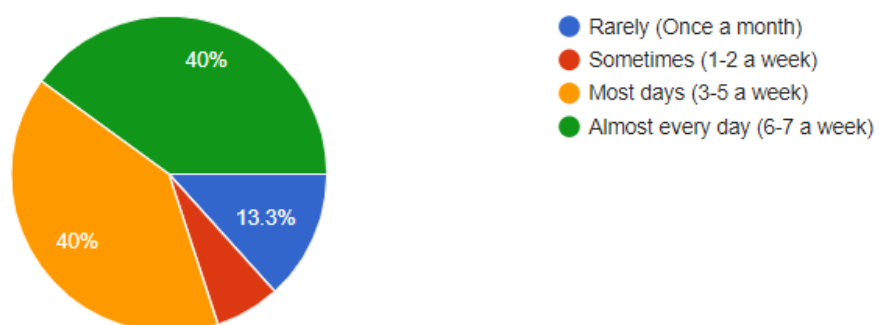
*Figure 23 results of question 2*

## Question 3

In this question the participants were asked how often they play games during the week, this is mostly asked to see if it will affect their skill level when it comes to the game. 40% (6) participants play between 3 to 5 times a week. Another 40% of the participants play almost every day between 6 to 7 days a week. 6.7% (1) of the participants play sometimes about 1 to 2 a week and 13.3% (2) play rarely about once a month.

## 3. How often do you play games

15 responses



*Figure 24 results of question 3*

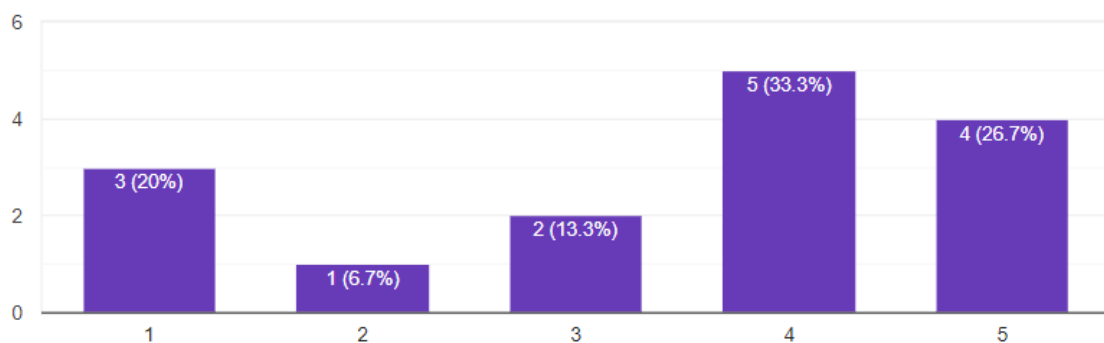


#### Question 4

This question gets a little inside to whether the participants have had any experience with playing car racing games. 20% (3) of the participants have stated they are of an amateur level when it comes to car racing games. 6.7% (1) have stated that they slightly more experience putting them at beginner level. 13.3% (2) of the participants have a decent amount of experience in car racing games. 33.3% (5) have a good amount of experience when it comes to racing games. Finally, 26.7% (4) participants have said that they are rally experienced when it comes the car racing games.

4. How much experience do you have with car racing games?

15 responses



*Figure 25 results of question 4*

1 = Amateur      2 = beginner      3 = decent      4 = good      5= Experienced

#### Question 5

For this question, the participants were asked whether they preferred to play games in competitive way which means mostly to win or in a casual way which is just to have fun playing. 53.3% (8) participants prefer to play games in a casual way, on the other hand 46.7% (7) participants like to play games in a competitive way.

### 5. Do you prefer Casual or competitive play?

15 responses

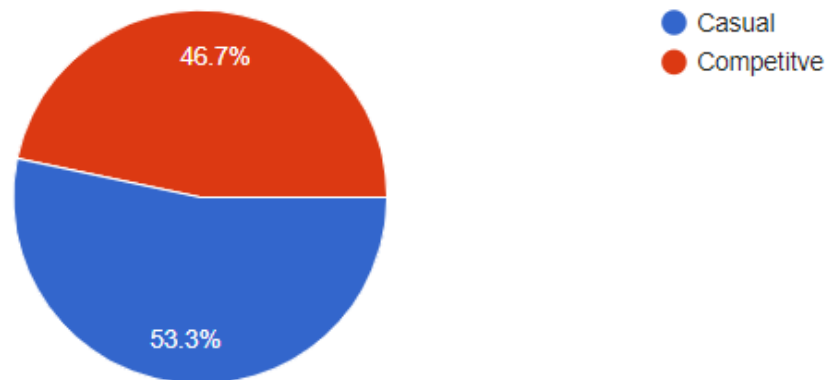


Figure 26 results of question 5

### Question 6

In this question the participants selected how much they felt that the easy level AI opponent was challenging to race against. 13.3 (2) participants did not feel that the easy AI was that difficult, 46.7% (7) participants found the difficulty of the easy AI at a normal difficulty. Finally, 40% (6) found the difficulty of the opponent to be fairly competitive.

### 6. Rate how much you felt the opponent challenge in the Easy level

15 responses

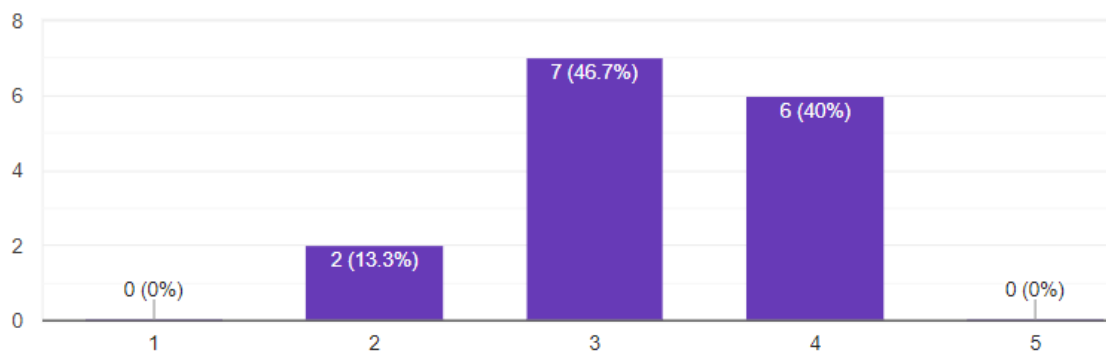


Figure 27 results of question 6

1 = Not competitive

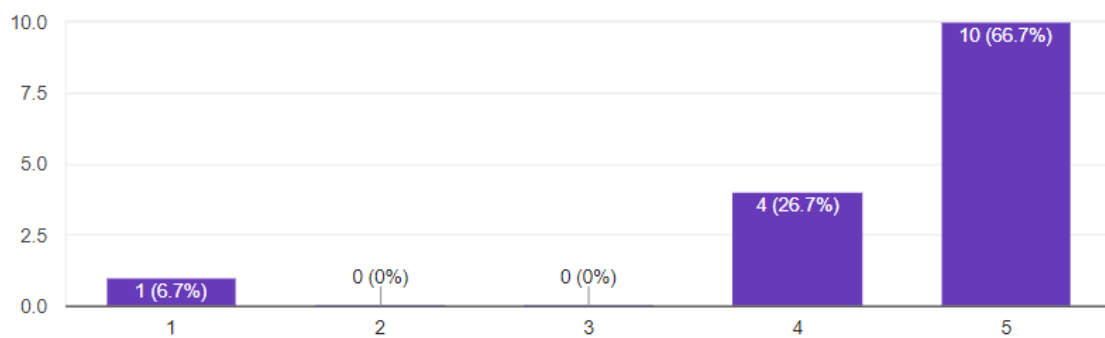
5 = Very competitive

### Question 7

In this question the participants selected how much they felt that the hard level AI opponent was challenging to race against. 6.7% (1) of the participants found that the hard level opponent was not competitive. 26.7% (4) of the participants found that hard opponent was pretty competitive. 66.7% (10) of the participants found that the hard opponent was very competitive.

7. Rate how much you felt the opponent challenge in the Hard level

15 responses



*Figure 28 results of question 7*

1 = Not competitive

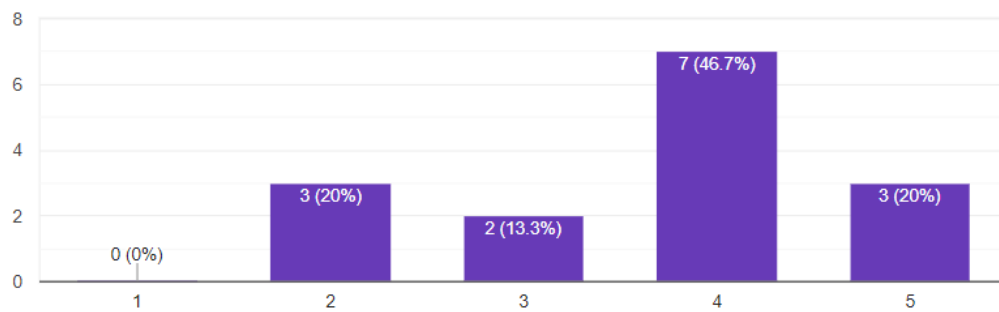
5 = Very competitive

### Question 8

For this question, the participants had to choose how much difficult they felt the DDA ML agent was as an opponent. 20% (3) of the participants did not feel that the DDA AI opponent was that competitive. 13.3% (2) participants felt that the DDA opponent was at a normal difficulty. 46.7% (7) felt that DDA AI opponent was family competitive. 20% (3) found that the DDA AI opponent was very competitive.

8. Rate how much you felt the opponent challenge in the DDA integrated with MLagents level

15 responses



*Figure 29 results of question 8*

1 = Not competitive

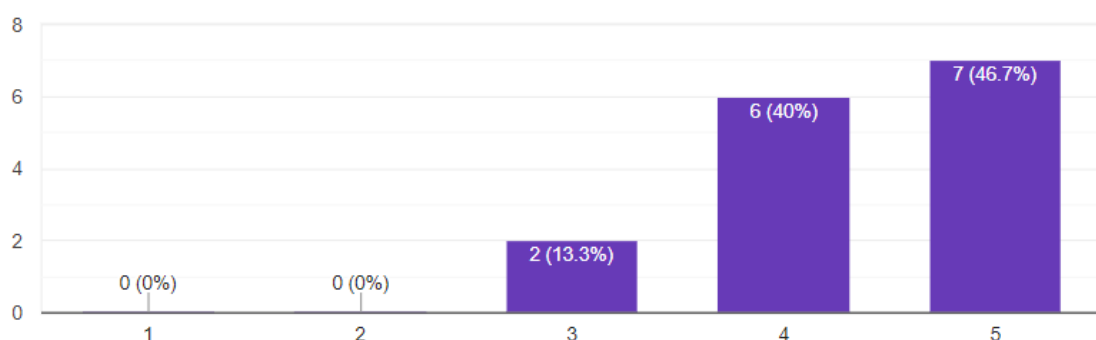
5 = Very competitive

### Question 9

For this question participants had to choose how much they enjoyed the player experience of the DDA level which is integrated with ML-agents. 13.3% (2) of the participants felt that the DDA level was in the middle in terms of player experience enjoyment. 40% (6) of the participants enjoyed the player experience of the DDA level. 46.7% (7) of the participants have enjoyed the player experience of the DDA level the most.

9. Rate how much you enjoyed the player experience (Enjoyment) of the DDA integrated with MLagents

15 responses



*Figure 30 results of question 9*

1 = unenjoyable

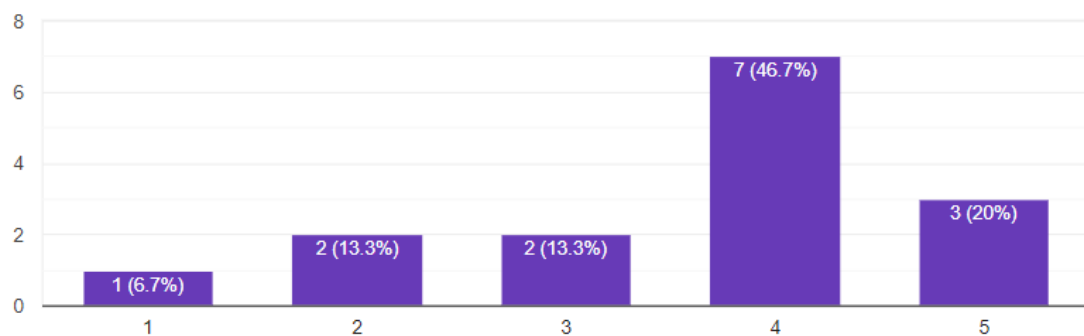
5 = Enjoyed a lot

## Question 10

For this question participants had to choose how much they enjoyed the player experience of the Difficulty level which are the easy and hard levels. 6.7% (1) participants felt that in terms of player experience was unenjoyable. 13.3% (2) of the participants felt that the player experience of the difficulty level was not that enjoyable. 13.3% (2) felt that difficulty level was in the middle in terms of player experience. 46.7% (7) of the participants quite enjoyed the player experience of the difficulty level. Finally, 20% (3) of the participants enjoyed the difficulty level player experience a lot.

10. Rate how much you enjoyed the player experience (Enjoyment) of the Difficulty Level (Easy and Hard)

15 responses



*Figure 31 results of question 10*

1 = unenjoyable

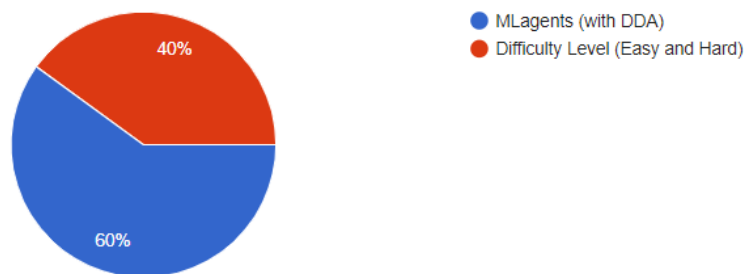
5 = Enjoyed a lot

## Question 11

In this question the participants were asked which was the preferred system between the difficulty level and dynamic difficulty adjustment integrated with ML-agents. 40% (6) participants felt that the Difficulty level (easy and hard) was the better system between the two, while on the other hand 60% (9) of the participants prefer the DDA with ML-agents.

11. Which do you personally feel was the best System overall considering the games challenge and player experience?

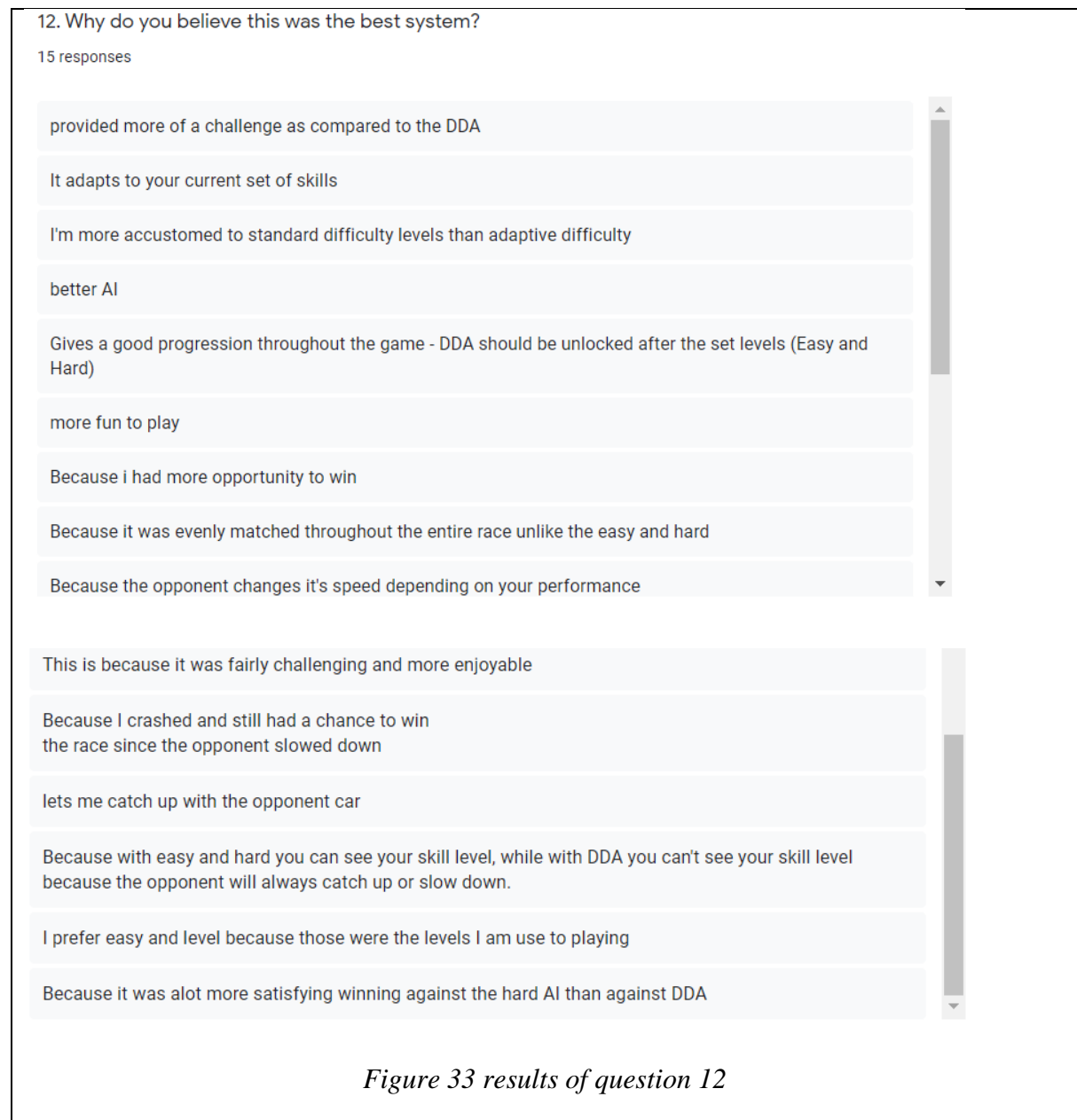
15 responses



*Figure 32 results of question 11*

## Question 12

In this question the participants were asked to give a reason to why they believe the system which they choose is the best system, which will give an insight to their perspective to why one is better than the other.

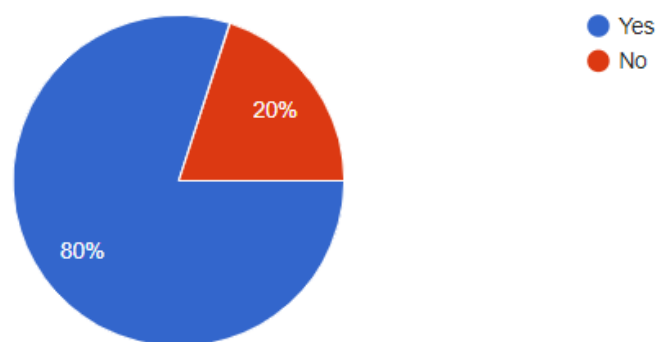


### Question 13

For the final question, the participants were asked whether they would consider playing a game with DDA integrated with machine learning. 20% (3) participants chose that they wouldn't consider playing a game with DDA integrated with Machine learning, while 80% (12) have chosen that they would consider playing a game with DDA integrated with Machine learning.

13. Would you consider playing games with DDA integrated with Machine Learning ?

15 responses



*Figure 34 results of question 13*



## 5 Discussion of Results

### 5.1 Discussion concerning game challenge.

To further evaluate the gaming challenge of the three levels, the results collected from the participants shows information like how many times the participants and AI's had won these levels against each other and there lap time difference. The average of the time difference was calculated to find out the level of competitiveness in each level.

This was calculated for each level by finding the number of wins for both participants and AI in each level, which are shown in Table 3. Using the time difference taken from Table 5, the player average was calculated by adding all the time difference values and then dividing by the number of total wins in that level. This calculation was used to find the average difference in winning times for both the AI and the participants in each level, and the results are shown in Table 4.

In the easy level, 5 out of 15 participants were able to win and have an average lead of 0.07.58s over the AI. In the hard level, only 2 out of 15 participants could win and have an average gap time of 0.02.40s. Finally, in the DDA with ML -agents, 11 out of 15 participants were able to win and have an average gap time of 0.1.73s

Considering this information, it is clear that the participants in the DDA with integrated ml-agent level had the lowest average gap time when they won the race, indicating that the race was very tight compared to the difficulty levels (easy and hard) until both cars crossed the finish line. Also, the DDA level with 11 participants still had a better average gap time than the other levels, which in theory, since it had more participants, could have easily had a higher average, but this shows how close its race was when the participants won.

The AI have won 10 out of 15 in the easy level and it had an average gap time of 0.13.76s over the participants when it won. In the hard level the AI managed to win 13 out of 15 against the

participants and had an average gap time of 19.75s over the participants. Finally, in the DDA with ML-agents the AI agent managed to win 4 out of 15 and had an average gap time of 0.4.32s over the participants.

With this data it shows that the DDA system is a better competitive system since in both cases when the participants or the AI had won the race it gap time was significantly closer than the easy and hard levels and no matter who won between the two the gap time was almost always very close. Even though the DDA was more competitive the hard level was more difficult to win against.

*Table 3 Shows winner of every level*

Total wins and losses	Easy	Hard	DDA (ML-agent)
Player	5	2	11
AI	10	13	4

*Table 4 average of time difference on winning times*

Average gap time when winning	Easy	Hard	DDA (ML-agent)
Player	0.07.58s	0.02.40s	0.01.73s
AI	0.13.76s	0.19.75s	0.04.32s

In Table 5, the difference in time column the (+) indicates that the AI managed to finish the race before the player and the value after the (+) indicates the number of seconds the player took to cross the finish line. While the (-) indicates that the player managed to cross the finish line before the AI and the value after the (-) indicates the number of seconds the AI took to cross the finish line.

Table 5 Shows lap times/ difference in time/ Winner

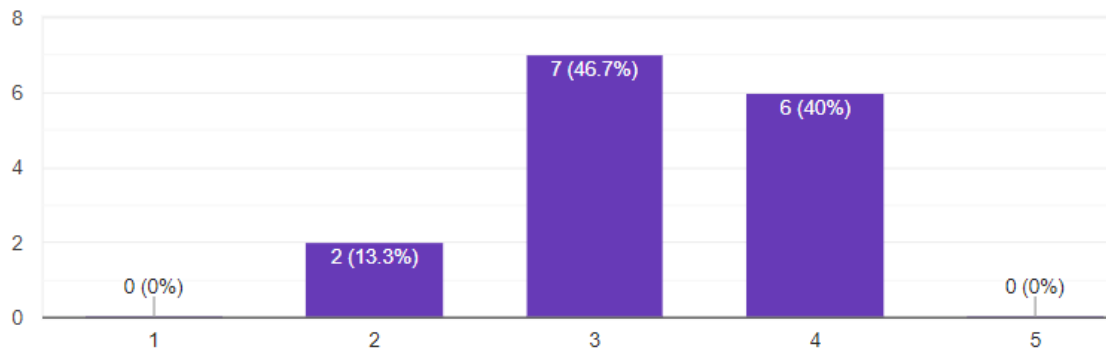
	Easy lap times	Difference in time	W/L	Hard lap times	Differe nce in time	W/L	DDA lap times	Differe nce in time	W/L
<b>Participant</b>	0.43.7	-10.1	Win	0.43.4		Lose	0.49.1	-1.4	Win
<b>1</b>	0.39.0			0.37.1			0.36.1		
<b>AI</b>	0.48.9		Lose	0.41.3	+0.4	Win	0.45.4		Lose
	0.43.9			0.38.8			0.41.2		
<b>Participant</b>	0.53.0		Lose	0.49.2		Lose	0.54.0	-0.8	Win
<b>2</b>	0.46.4			0.42.1			0.44.3		
<b>AI</b>	0.48.7	+6.7	Win	0.42.7	+10.9	Win	0.49.8		Lose
	0.43.7			0.38.7			0.49.3		
<b>Participant</b>	0.42.8	-10.8	Win	0.42.4		Win	0.42.9	-2.8	Win
<b>3</b>	0.38.8			0.36.7	-1.0		0.34.3		
<b>AI</b>	0.49.0		Lose	0.41.3		Lose	0.45.2		Lose
	0.43.4			0.38.8			0.34.7		
<b>Participant</b>	0.52.5		Lose	0.52.0		Lose	0.48.0		Lose
<b>4</b>	0.43.7			0.49.9			0.53.0		
<b>AI</b>	0.48.7	+3.7	Win	0.42.7	+20.5	Win	0.47.3		Win
	0.43.8			0.38.7			0.50.8	+2.9	
<b>Participant</b>	0.46.7		Lose	0.51.3		Lose	0.50.7	-3.6	Win
<b>5</b>	0.47.5			0.52.0			0.48.1		
<b>AI</b>	0.49.2	+1.0	Win	0.42.7	+15.2	Win	0.49.9		Lose
	0.44.0			0.38.7			0.52.5		

<b>Participant</b>	0.47.4	-0.1	Win	0.44.5		Lose	0.51.9		Win
<b>6</b>	0.41.4			0.43.6			0.42.0	-3.5	
<b>AI</b>	0.47.7		Lose	0.42.4	+6.9	Win	0.51.6		Lose
	0.41.2			0.38.8			0.43.8		
<b>Participant</b>	1.01.7		Lose	1.03.4		Lose	1.00.8		Lose
<b>7</b>	0.58.1			0.58.9			1.00.3		
<b>AI</b>	0.48.9	+27.6	Win	0.42.5	+41.2	Win	0.56.9	+8.8	Win
	0.43.3			0.38.6			0.55.4		
<b>Participant</b>	0.44.6		Lose	0.47.7		Lose	0.50.8	-1.5	Win
<b>8</b>	0.56.1			0.43.6			0.40.5		
<b>AI</b>	0.48.9	+8.3	Win	0.42.6	+10.2	Win	0.50.3		Lose
	0.43.5			0.38.5			0.42.5		
<b>Participant</b>	0.54.1		Lose	0.48.3		Lose	0.46.1	-0.4	Win
<b>9</b>	0.41.5			0.44.9			0.41.2		
<b>AI</b>	0.49.3	+3.7	Win	0.43.0	+11.9	Win	0.47.7		Lose
	0.43.6			0.38.9			0.40.0		
<b>Participant</b>	1.03.2		Lose	0.48.7		Lose	0.46.0		Lose
<b>10</b>	0.57.0			1.01.7			0.51.7		
<b>AI</b>	0.48.6	+28.5	Win	0.42.2		Win	0.46.3	+1.2	Win
	0.43.1			0.38.4	+29.8		0.50.2		
<b>Participant</b>	0.52.1		Lose	0.55.0		Lose	0.53.8	-0.9	Win
<b>11</b>	0.55.7			1.02.8			0.44.3		
<b>AI</b>	0.48.8	+15.5	Win	0.42.6	+36.7	Win	0.52.2		Lose
	0.43.5			0.38.5			0.46.8		

<b>Participant</b>	1.03.9		Lose	1.00.4		Lose	0.57.7		Lose
<b>12</b>	1.06.9			1.07.3			0.55.7		
<b>AI</b>	0.48.8	+38.6	Win	0.42.6	+46.6	Win	0.53.4	+3.7	Win
	0.43.4			0.38.5			0.56.3		
<b>Participant</b>	0.45.1	-7.8	Win	0.42.9		Lose	0.46.0	-1.3	Win
<b>13</b>	0.39.1			0.40.1			0.41.8		
<b>AI</b>	0.48.8		Lose	0.41.5	+3.1	Win	0.47.7		Lose
	0.43.2			0.38.4			0.41.4		
<b>Participant</b>	0.48.1		Lose	0.55.1		Lose	0.51.5	-0.5	Win
<b>14</b>	0.47.8			0.49.1			0.44.8		
<b>AI</b>	0.47.5	+4.0	Win	0.42.4	+23.4	Win	0.51.4		Lose
	0.43.4			0.38.4			0.45.4		
<b>Participant</b>	0.41.2	-9.1	Win	0.42.5	-2.8	Win	0.43.6	-2.3	Win
<b>15</b>	0.40.9			0.35.9			0.37.2		
<b>AI</b>	0.48.9		Lose	0.42.4		Lose	0.44.9		Lose
	0.43.3			0.38.8			0.38.2		

6. Rate how much you felt the opponent challenge in the Easy level

15 responses



*Figure 35 results of question 6*

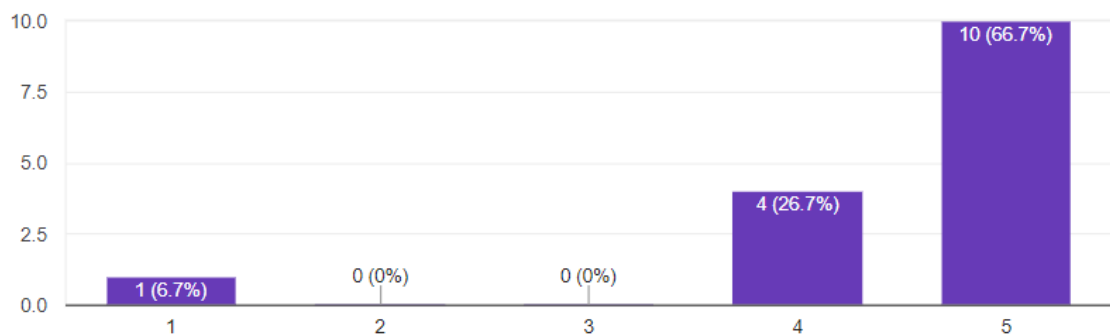
1 = Not competitive

5 = Very competitive

From Figure 35, it can be seen that most participants chose the easy level as their normal difficulty. Although only 5 participants won this level, you can see from their lap times that the level is winnable with a little more practice. Also, another reason is even though most participants lost by quite a margin the difficulty of the hard level made them decide to not select the easy level as a very competitive level because in comparison to the hard it is not.

7. Rate how much you felt the opponent challenge in the Hard level

15 responses



*Figure 36 results of question 7*

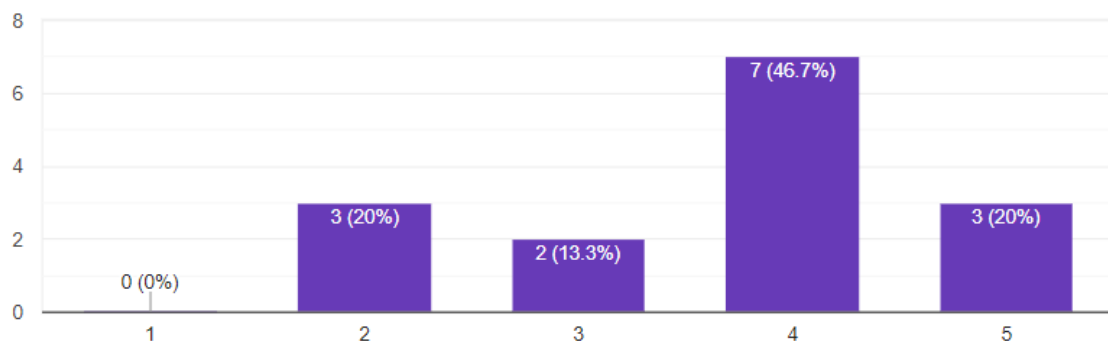
1 = Not competitive

5 = Very competitive

Most participants feel that the hard AI was very challenging, and this selection of participants is most likely chosen because only 2 participants managed to win and won by a really small margin, while the majority of participants lost by a large margin, which makes them feel that the AI was very difficult for them and that they had no chance against it. Even though it looks very difficult it is still feasible to win with more experience in this gaming genre as can be seen from the 2 experienced participants that won the level.

#### 8. Rate how much you felt the opponent challenge in the DDA integrated with MLagents level

15 responses



*Figure 37 results of question 8*

1 = Not competitive

5 = Very competitive

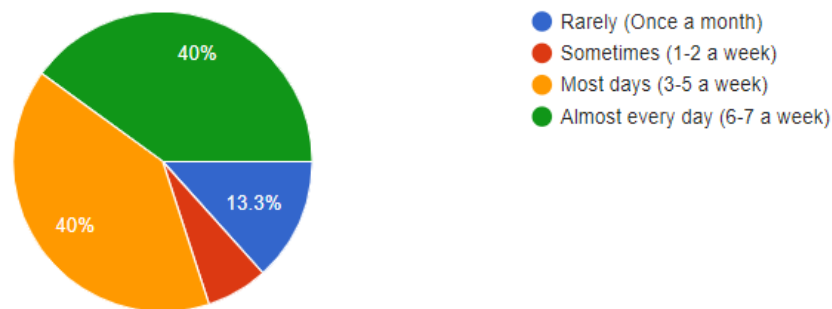
In the DDA integrated with ML-agents level the participants had a lot of mixed selections on to how challenging this level was. Seeing that 11 out of 15 participants had won the race it is interesting to see that most participants selected that the DDA level was on the challenging side, this is probably since most participants only lost or won by a few seconds which indicated that the race was consistently challenging for them.

For the participants that have voted to the middle or to the not so competitive side this is most likely caused to because once the participants overtook the AI agent it was pretty simple to keep it behind them unless they made a mistake that allowed the AI agent to overtake them back, even though the AI agent was always close behind. This happened because of the agents raycast sensors since the AI agent detects the participants car it opted to slow down and try to pass them from a different side so it

would not collide with them, and once the participants were seeing the agent catching up they were blocking it so they could keep their lead which wasn't giving the agent the opportunity to pass back.

### 3. How often do you play games

15 responses

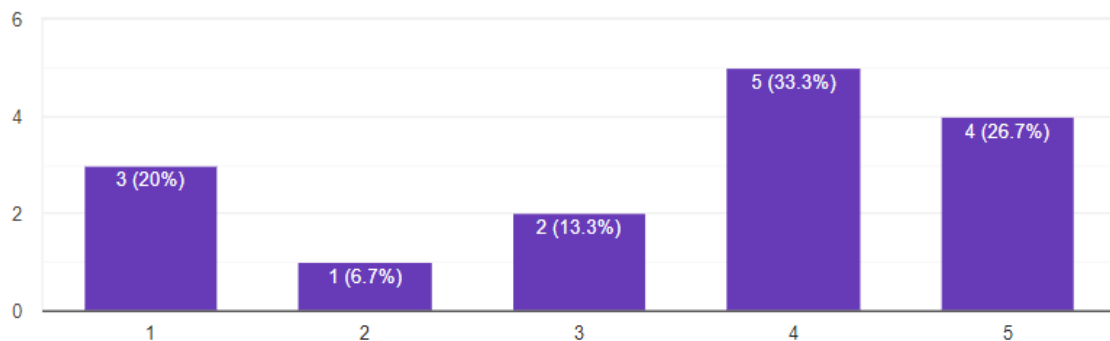


*Figure 38 results of question 3*

## 5.2 Discussion concerting player experience

### 4. How much experience do you have with car racing games?

15 responses



Amateur = 1

Experienced = 5

*Figure 39 results of question 4*

In Figure 38, the data show that 80% of participants play games most days of the week. In Figure 39, experience with car racing games is mixed among participants, but most participants still indicate that they have a decent amount of experience with car racing games.



Using Figure 38 and Figure 39, two main factors are considered, the amount of time spent playing games and the other is the total amount of experience participants have gained from playing car racing games. Therefore, these two factors form the lap times in Table 5, which contains the data of each participant.

In order to evaluate which system provided the best player experience in terms of satisfaction, enjoyment, and immersion, a few questions were conducted in the survey to ask participants which system they thought was the best after playing the prototype.

From the data shown in Figure 40 and Figure 41, the DDA system with ML- agents provided a better player experience than the difficulty level system (easy and hard). Although most participants indicated that the difficulty level was quite enjoyable, according to their selection, it was not as enjoyable compared to the DDA. This selection is also influenced by the results that the participants achieved in the different levels. Since 11 participants won in the DDA level and had a competitive race against the AI, they indicate that they had a good experience with that level. On the other hand, 5 participants won the easy level and only 2 won the hard level. In addition, most participants lost by some margin, which probably caused them to be frustrated in this level and did not have as good of an experience as in the DDA level.

9. Rate how much you enjoyed the player experience (Enjoyment) of the DDA integrated with MLagents

15 responses

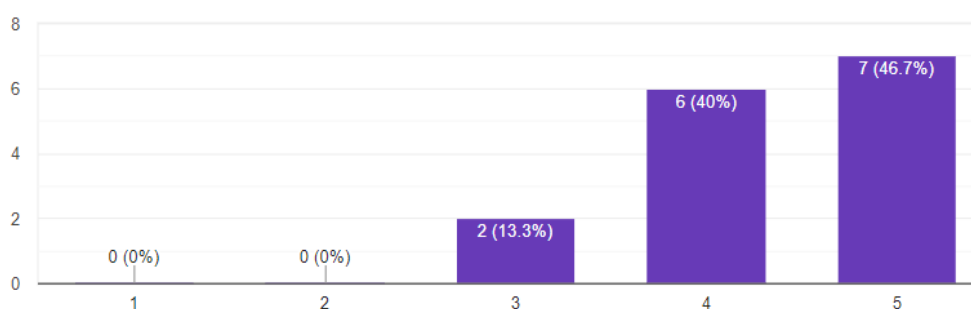


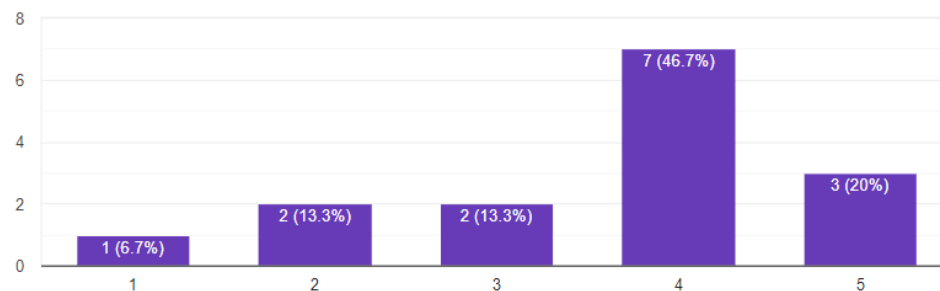
Figure 40 results of question 9

1 = unenjoyable

5 = Enjoyed a lot

10. Rate how much you enjoyed the player experience (Enjoyment) of the Difficulty Level (Easy and Hard)

15 responses



*Figure 41 results of question 10*

1 = unenjoyable

5 = Enjoyed a lot

From the data collected from Figure 42, Figure 43, and Figure 44 it shows which was the best overall system according to the participants and their reason for choosing that system.

The main reasons that majority of participants gave for voting in favour of the DDA integrated with ml agents were mostly because they felt that DDA gave them a better opportunity to catch up and attempt to win even if they crashed, kept the race competitive no matter the level of skill of the participant and because of this gave them a better opportunity to win.

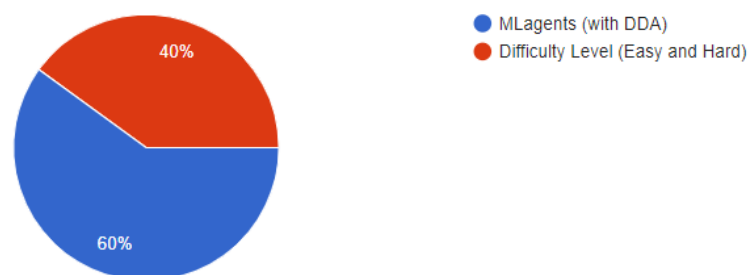
On the other hand, the main reasons the participants preferred the Difficulty level (easy and hard levels) was because they felt it allows them to know where they lie in terms of skill with comparison to the AI since the DDA will always catch up or slow down, some felt that it was more satisfying that they beat the hard level because of how challenging they found it to be, one participant believed that the DDA should be unlocked after a player beats both the easy and hard since it gives a good progression throughout the game and others preferred the difficulty level because that is what they are used to play and did not like the change too much.

Seeing the reasons why the participants preferred one system over the other could also be because it would depend on what they prefer between casual and competitive gaming. Because

a competitive gamer would like to know exactly the level of skill they are on and always like to set a target so they could improve on so a level like the Hard level could be a target for competitive players, while causal players would prefer to just enjoy the game and have a good experience playing it. To summarise on this alongside the results obtained concerning the DDA it can also cater for players that prefer a competitive experience.

11. Which do you personally feel was the best System overall considering the games challenge and player experience?

15 responses



*Figure 42 results of question 11*

12. Why do you believe this was the best system?

15 responses

This is because it was fairly challenging and more enjoyable
Because I crashed and still had a chance to win the race since the opponent slowed down
I prefer easy and level because those were the levels I am use to playing
Because it was evenly matched throughout the entire race unlike the easy and hard
Because i had more opportunity to win
I'm more accustomed to standard difficulty levels than adaptive difficulty
lets me catch up with the opponent car
Because with easy and hard you can see your skill level, while with DDA you can't see your skill level because the opponent will always catch up or slow down.

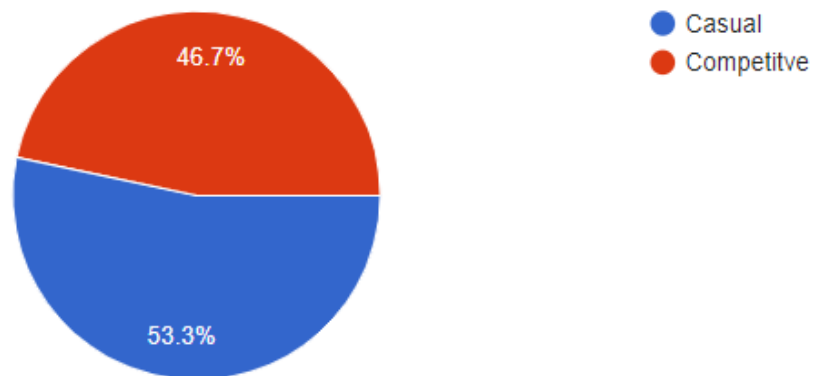
*Figure 43 answers of question 12 part 1*

more fun to play
Gives a good progression throughout the game - DDA should be unlocked after the set levels (Easy and Hard)
better AI
Because it was alot more satisfying winning against the hard AI than against DDA
provided more of a challenge as compared to the DDA
It adapts to your current set of skills
Because the opponent changes it's speed depending on your performance

*Figure 44 answers of question 12 part 2*

5. Do you prefer Casual or competitive play?

15 responses

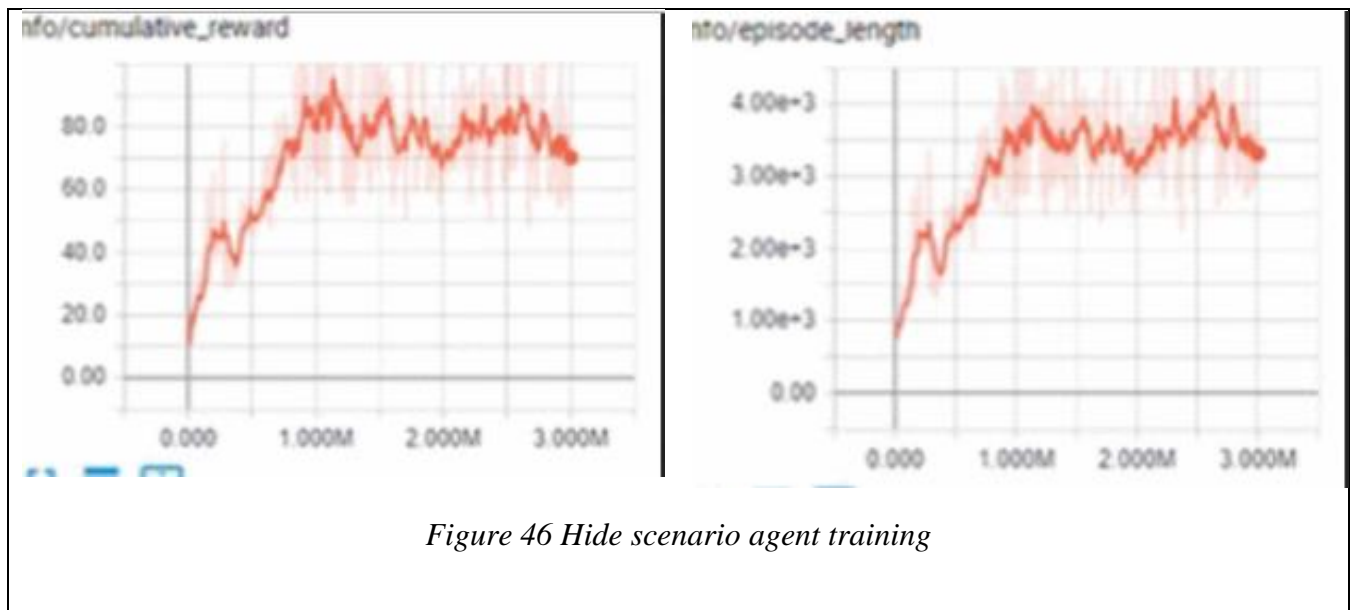


*Figure 45 results of question 5*

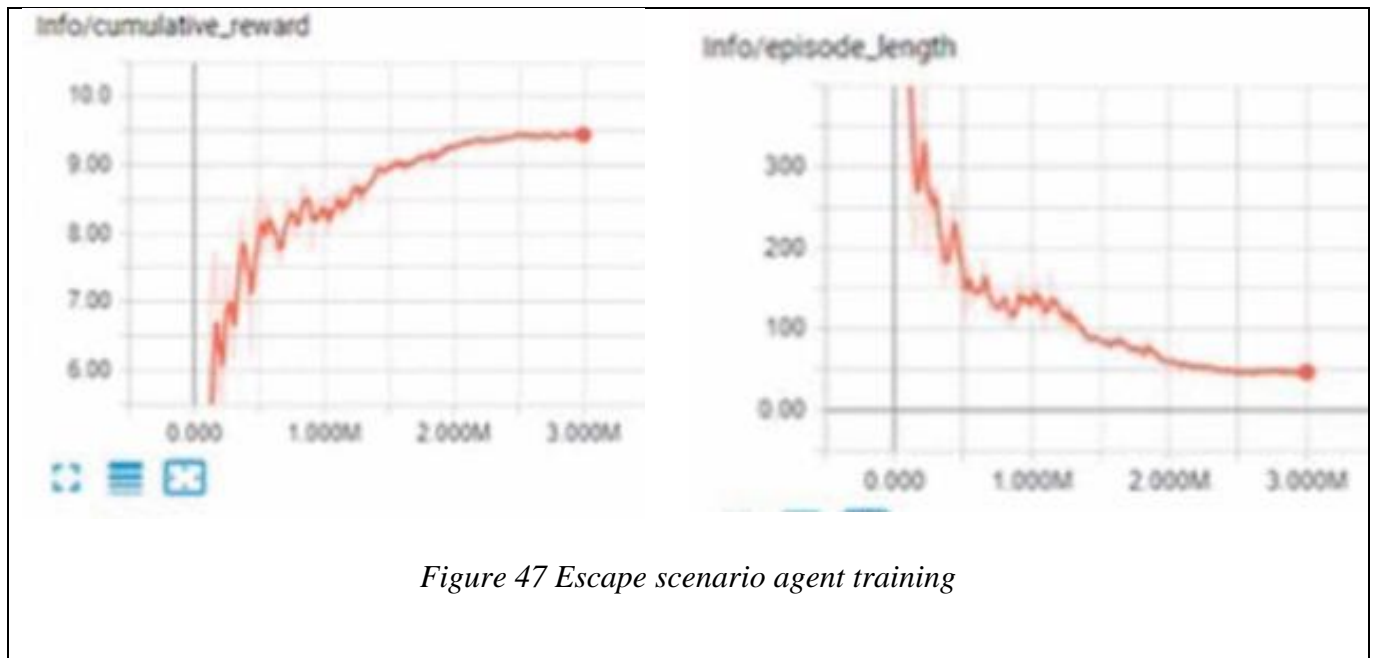
### 5.3 Observations

Here is the agent from (Nurkey, 2019) where the author uses machine learning agents with reinforcement learning to have an agent that hides infinitely and, when found by the scripted AI, tries to escape it. This is compared to the created machine learning agent that also uses reinforcement learning to drive on the track. In the study by (Nurkey, 2019), the hiding training did not result in a consistent slope, so the researcher left the agent to train until there was an improvement in behaviour. The escape scenario, on the other hand, worked very well from the beginning. Although the agent created for this study had considerably fewer steps than Hide and Escape, it is similar to Hide in some respects, as both varied a lot during the training.

Ultimately, the AI agents in both this study and the study by (Nurkey, 2019) were successful with their training, and both had functioned as intended in the project during testing.



*Figure 46 Hide scenario agent training*



## **5.4 Conclusion of discussion**

From the results collected it is shown that the DDA integrated with ML-agents level had a closer average time when the AI or player was victorious in comparison to the other levels. With the lap times data collected it also shows that the DDA worked as intended since it was always performing on the same level of the participants. An important observation that was made to conclude why so many participants managed to beat once they managed to take the lead from the AI agent is probably because when the agent got too close to the player it decided to slow down and try to pass from beside him.

From the survey it was seen that the participants believed that the DDA system was better for player experience, but they felt that the hard level of the difficulty system is more challenging which is probably due to the fact that most participants lost that race and by a huge margin.

## 6. Conclusion

With the data collected from the prototype and the data gathered from the survey it can be found which system was better for gaming challenge and player experience between Dynamic difficulty adjustment implemented with ML-agents and difficulty level with a non-adaptive AI with easy and hard levels. To evaluate the gaming challenge the lap times were taken and then depending on the winner the average gap time was taken to see how close the race was. In addition, the participants had a few question in the survey about the challenge of the game were in terms of difficulty most people chose the Hard level as most difficult but when seeing the average winning time for the AI in the hard level it was at 0.19.75s which means that the participants selected the hard level because it was the level that they felt did not manage to challenge the AI. When seeing how the DDA AI agent gap time was when beating the participants, it only had a gap time of 0.04.32s which when compared to the hard AI is significantly less and although it was significantly less most people voted that the DDA AI agent was really competitive. In conclusion by these results show that the DDA implemented with ML-agent was the most competitive in terms of having the opponent constantly challenging till the end but, in the case of opponent difficulty it was not as challenging as the hard AI to win against.

In terms of player experience the results from the survey were taken into account, and shows that people preferred the DDA with ML-agents because it was more enjoyable, quite competitive, more immersive since the participants had to keep focused till the end to make sure they will not lose the race or maybe still have the chance to win it, less frustrating because it allowed the users to have a chance to recover from a mistake and in terms of AI movement the AI agent was a lot more variant than the traditional AI's because the AI agent learned the track by using reinforcement learning which allowed to try different manoeuvre when the player was in its ways.



For future improvements multiple adjustments seemed necessary especially when the participants were playing the prototype and could be clearly seen where they were struggling. The first improvement needed was the players car acceleration because whenever they slowed down, they felt that the car was taking too long to get back up to speed. Another improvement needed in the players car was the turning power which they found to be very difficult to get use to and did not feel that the car was turning enough for tight turns which ended up causing most participants to crash repeatedly.

In the difficulty level some future improvements are needed in the easy and hard opponents. The first future improvement is that once the participant is ahead of the AI, and the AI try to overtake back, if the participant is on the AI's racing line it will go right through the participants car instead of trying to overtake the participant by going beside them. To conclude the last improvement needed for the difficulty level is that once the player collides with the AI from behind it ends up giving a huge boost to the AI making it even more difficult to pass.

In the DDA integrated with ML-agents level improvements that could be done after watching the participants race against it are firstly to try and make the agent a little more aggressive when overtaking because currently the AI agent is a little too cautious when trying to overtake because when it gets close to the participant car it decides to slightly brake so it would not collide and due to this the participants normally held the lead once managed to overtake the AI agent.

## 7. References

- Ang, D. (2017) ‘Difficulty in video games: Understanding the effects of dynamic difficulty adjustment in video games on player experience’, in *C and C 2017 - Proceedings of the 2017 ACM SIGCHI Conference on Creativity and Cognition*. Association for Computing Machinery, Inc, pp. 544–550. Doi: 10.1145/3059454.3078706. (Accessed: 8 December 2020).
- Bakkes, S. C. J., Spronck, P. H. M. And van Lankveld, G. (2012) ‘Player behavioural modelling for video games’, *Entertainment Computing*, 3(3), pp. 71–79. Doi: 10.1016/j.entcom.2011.12.001. (Accessed: 15 December 2020).
- Chu, K., Wong, C. Y. And Khong, C. W. (2011) ‘Methodologies for evaluating player experience in game play’, in *Communications in Computer and Information Science*. Springer, Berlin, Heidelberg, pp. 118–122. Doi: 10.1007/978-3-642-22098-2\_24. (Accessed: 3 February 2021).
- Cooper, N. (2005) *Machine Learning for Auto-Dynamic Difficulty in a 2-D Space Shooter*. Available at: <http://www.stanford.edu/~nacooper> (Accessed: 9 December 2020).
- Haring, M. (2020) ‘Reinforcement Learning – Introducing the Unity ML-Agents Toolkit’. Available at: <http://desres18.netornot.at/id/reinforcement-learning-unity-ml-agents-toolkit/>. (Accessed: 20 January 2021).
- Hunicke, R. (2005) ‘The case for dynamic difficulty adjustment in games’, in *ACM International Conference Proceeding Series*. New York, New York, USA: ACM Press, pp. 429–433. Doi: 10.1145/1178477.1178573. (Accessed: 28 December 2020).
- Hunicke, R. And Chapman, V. (2004) ‘AI for dynamic difficulty adjustment in games’, in *AAAI Workshop - Technical Report*, pp. 91–96. Available at: [https://www.researchgate.net/publication/228889029\\_AI\\_for\\_dynamic\\_difficulty\\_adjustment\\_in\\_games](https://www.researchgate.net/publication/228889029_AI_for_dynamic_difficulty_adjustment_in_games). (Accessed: 11 January 2011).
- Jennings-Teats, M., Smith, G. And Wardrip-Fruin, N. (2010) ‘Polymorph: Dynamic Difficulty Adjustment through level generation’, in *Workshop on Procedural Content Generation in Games, PC Games 2010, Co-located with the 5th International Conference on the Foundations of Digital Games*. Doi: 10.1145/1814256.1814267. (Accessed: 11 January 2021).

M. Urmanov, alimanova, m. And , suleyman demirel university, kaskelen, k. (2020) ‘training a single machine learning agent using reinforcement learning and imitation learning methods in unity environment’. Available at: <https://journals.sdu.edu.kz/index.php/nts/article/view/47>. (Accessed: 20 February 2021).

Pedersen, C., Togelius, J. And Yannakakis, G. N. (2009) *Modeling Player Experience in Super Mario Bros*. Available at: <http://www.mojang.com/notch/mario/> (Accessed: 9 December 2020).

Samuel, A. L. (2000) ‘Some studies in machine learning using the game of checkers’, *IBM Journal of Research and Development*, 44(1–2), pp. 207–219. Doi: 10.1147/rd.441.0206.(Accessed: 24 January 2021)

Spronck, P., Sprinkhuizen-Kuyper, I. And Postma, E. (2002) ‘Improving Opponent Intelligence through Machine Learning’, *Game-on*, pp. 94–98. Available at: [https://www.researchgate.net/publication/250423947\\_Improving\\_Opponent\\_Intelligence\\_by\\_Machine\\_Learning](https://www.researchgate.net/publication/250423947_Improving_Opponent_Intelligence_by_Machine_Learning) (Accessed: 8 December 2020).

Urmanov, M., Alimanova, M. And Nurkey, A. (2019) ‘Training unity machine learning agents using reinforcement learning method’, *2019 15th International Conference on Electronics, Computer and Computation, ICECCO 2019*. Doi: 10.1109/ICECCO48375.2019.9043194. (Accessed: 20 March 2021).

Yaguchi, T. And Iima, H. (2020) ‘Design of an Artificial Game Entertainer by Reinforcement Learning’, *IEEE Conference on Computational Intelligence and Games, CIG*, 2020-Augus, pp. 588–591. Doi: 10.1109/cog47356.2020.9231551. (Accessed: 10 February 2021).

Yannakakis, G. N. And Hallam, J. (2006) ‘Towards capturing and enhancing entertainment in computer games’, in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer Verlag, pp. 432–442. Doi: 10.1007/11752912\_43. (Accessed: 6 March 2021).

## 8. Appendix

This section contains the template used for the survey conducted.

1. What is your age Group? \*

- ☐ 17 - 23
- ☐ 24 - 29
- ☐ 30 - 40
- ☐ Over 40

2. What is your gender? \*

- ☐ Man
- ☐ Woman
- ☐ Other
- ☐ Don't wish to say

3. How often do you play games? \*

- ☐ Rarely (Once a month)
- ☐ Sometimes (1-2 a week)
- ☐ Most days (3-5 a week)
- ☐ Almost every day (6-7 a week)

4. How much experience do you have with car racing games? \*

- |         |                       |                       |                       |                       |                       |             |
|---------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-------------|
|         | 1                     | 2                     | 3                     | 4                     | 5                     |             |
| Amateur | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | Experienced |

5. Do you prefer Casual or competitive play? \*

- ☐ Casual
- ☐ Competitive

6. Rate how much you felt the opponent challenge in the Easy level \*

	1	2	3	4	5	
Not Competitive	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very Competitive

7. Rate how much you felt the opponent challenge in the Hard level \*

	1	2	3	4	5	
Not Competitive	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very Competitive

8. Rate how much you felt the opponent challenge in the DDA integrated with MLagents level \*

	1	2	3	4	5	
Not Competitive	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very Competitive

9. Rate how much you enjoyed the player experience (Enjoyment) of the DDA integrated with MLagents \*

	1	2	3	4	5	
unenjoyable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Enjoyed a lot

10. Rate how much you enjoyed the player experience (Enjoyment) of the Difficulty Level (Easy and Hard) \*

	1	2	3	4	5	
unenjoyable	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Enjoyed a lot

11. Which do you personally feel was the best System overall considering the games challenge and player experience? \*

- ☐ MLagents (with DDA)
- ☐ Difficulty Level (Easy and Hard)

---

12. Why do you believe this was the best system? \*

Long answer text

---

13. Would you consider playing games with DDA integrated with Machine Learning ? \*

- ☐ Yes
- ☐ No