March, 2020

Improvement of Player Experience through Dynamic Difficulty Adjustment using Behavioral Patterns and Machine Learning

Supervisor: Koji Mikami

Tokyo University of Technology

Graduate School of Bionics, Computer and Media Science

Henry Daniel Fernández Balda

To the Japanese Ministry of Education, Culture, Sports, Science and Technology and the Japanese embassy in Venezuela, for giving me the opportunity be part of the MEXT scholarship program, which was the essential for the completion of my PhD.

To my professors, from those who taught me the basics of Computer Engineering during my undergraduate studies to those who tutored me and gave me the tools to overcome those challenges all these years, your guidance was fundamental to succeed.

To my family, friends and girlfriend, for supporting me, your thoughts and words gave me the necessary strength to keep going until the end.

Thank you all for your support, this thesis wouldn't have been possible without you.

Contents

List of Figures

Chapter 1

Introduction

When there is no proper balance between the level of difficulty of a game and its players, players might have a negative experience and quit playing as a consequence [1]. If challenges are too difficult to accomplish, players might get frustrated; if challenges are too easy, players might get bored [2, 3]. In any of these cases, the impact that a negative experience has in players, can negatively affect game creators (reputation, sales, etc).

According to the international standard on ergonomics of human-system interaction, ISO 9241-210 [4], user experience (UX) is defined as perceptions and responses that occur as a result of using a product, system or service. These responses involve emotions, preferences, perceptions, physical and psychological responses, etc, that are related to the use of that service or product.

User experience has been widely researched in different fields around the world, starting from an old research that was presented on the SIGCAPH Computers and the Physically Handicapped, a study that delved on how to improve the experience of users with disabilities when using a web page [5], they discussed about how those users can control and adapt elements from a website for better usability. Some other notable examples from the past, include the work of Juha Arrasvuori et al. [6], which explored how users interact with products and as a result, their study would help UX researchers to focus on the most pleasurable elements from the user experience. There was a different approach, which aimed to understand how to make an uncomfortable experience for users and how to avoid doing so [7].

This subject has not only being studied in the past but also nowadays, researchers put a lot of efforts on finding better ways to offer users a more comfortable experience. There is a publication from Paul Dourish [8] where he discusses about how technology and data management have impacted the concept and understanding of user experience and Human-Computer Interface (HCI). There is also research that focuses on the concerns cybersecurity and how this can impact the overall user experience when using technology and products made by big or small companies [9]; these researchers also advice people to be more informed in order to have a better end-user experience, caring about security, personal information and interaction is important. In addition, there is extensive research about how to effectively measure the user experience, due to the impact that this has on the success of companies that create products of services [10]. Evelyn Tio et al., performed a qualitative research about what really matters to the user, to help teams prioritize what is really important for their users.

Finally, for the matter of this research, there is the relationship between user experience and games. From some years ago, we have that user experience started to be a very important field of study in games, the work of Komulainen et al. [11], shows that the user experience is defined by four different components: cognition, motivation, emotion and attention, which served as the based of an empirical model designed by these researchers to measure UX in games; in addition, the work of Zhan Ye, showed how using genre theory (widely used in other types of media) to analyze the user experience in games [12].

From more recent research, there is the work of a group of researchers that focused on the user experience in serious games [13], that successfully designed and described how to evaluate the user experience in serious games, specifically, games for learning. Some other researchers focused on describing the user behavior in mobile gaming [14], for which they identified ten different solutions that explain the intention of users to download a mobile game. Lastly, in a combination of interaction, user experience and immersion, Don D.H. Shin, focused his research on augmented reality (AR) in videogames. Basically this researcher proposed a model to explain and predict the user experience of AR in games, as a result, the model successfully measured elements of engagement and immersion from users [15].

Besides user experience, other researchers have tried Dynamic Difficulty Adjustment to solve this issue of an between skills and challenges [16, 17, 18]. In addition, Procedural Content Generation has been combined with other techniques to help adapt the game automatically, depending on how players play at a determined time [19, 20, 21, 22, 23].

As a motivation for this research, the author wanted to contribute to the field and improve the game experience for players with different skills, the focus was to help create games that adapted depending on how players play in order to offer a more suitable level of difficulty, thus a better overall experience.

The main purpose was to create novel and effective ways to avoid the gap that occurs between players' skills and game difficulty. As a natural consequence of adapting the difficulty according to the player's skills, the experience as a whole would improve as well, making it more enjoyable and a better overall experience.

In this manuscript, the definition of *better experience* or *enjoyable experience* is derived from the concept of Flow, stated by the famous psychologist Mihaly Csíkszentmihályi in 1975 [24]. When a person is performing an activity, that person is said to be in a flow state or flow zone, when he or she reaches a mental state of full immersion, completely focused on the current task. In this research, the author tries to create an experience for players to get closer to the flow state, by encouraging them to avoid the zones outside the flow zone which are: boredom and frustration (also called anxiety).

Specific hypotheses for this research are stated as follows:

- Emotions can be predicted using behavioral patterns
- There is a relationship between how players feel while playing and the interaction they have with the controller
- Brain computer interfaces can contribute to the adaptation of games to players
- It is possible to create stages with a specific degree of difficulty
- Dungeon creation techniques can be effectively used to create automatically generated levels in other genres

Some of these assertions have been partly answered in previous research or there is evidence that show a tendency to demonstrated they are correct. The aim is to find clear answers and concrete solutions to each one of these hypotheses.

Starting from the previously mentioned hypotheses, the following goals have been set:

- Find novel and effective ways to improve the player experience through difficulty adjustment
- Prove how can emotions be predicted using behavioral patterns
- Propose a concrete method to connect behavioral patterns and player's emotions
- Evaluate the contribution of brain computer devices on game adaptation

- Find ways to create levels with a specific degree of difficulty
- Propose a concrete method to use dungeon creation techniques in other genres
- Demonstrate how effective can dungeon creation techniques be when creating levels for other genres
- Adapt difficulty of a game according to the player's skills

The content of this thesis is divided by type in two main parts: (A) Behavioral Patterns, which comprehends chapters: 3 and 4; (B) Dynamic Difficulty Adjustment, Procedural Content Generation Methods, which comprehends chapters: 5 and 6.

The overall approach for part (A) of the research, was to find a relationship between players' behavior and how that behavior is connected to the way they feel while playing. The main benefit from understanding emotions through physical behavior, would be to adapt the content of a game to encourage specific emotions of feelings and then be able to improve the experience as a whole. These two chapters of the book address the first two hypotheses.

Chapter 3 describes the experiment and how data was collected. A 2D shooting game was designed to make players interact with the button of a PS3 controller while shooting, pressure exerted on the button was recorded while playing and participants were required to answer a questionnaire about their experience while playing and the emotions they had towards the game in each session. The correlation between the answers given by players and the pressure exerted on the button was also calculated.

Using the results obtained from the experiment explained in **Chapter 3**, in order to find patterns between pressure sensitivity and the player behavior, machine learning methods such as Neural Networks and Support Vector Machines were tested. The explanation about how these methods were designed, the simulated conducted and best parameters are described in **Chapter 4**.

A natural proposal that arises after finding patterns between the player's behavior and his/her feelings is also discussed. A new method that uses the best parameters obtained from the tested machine learning methods was proposed. After detecting how players perceive the game in real time and how they feel towards it, one way to use these results to adapt the game and in consequence, change the experience for players was explained.

Chapter 5 consists of the explanation of results from combining DDA with attention levels obtained from a biosensor device that players used while performing experiments. 25 Players played a simple 2D platform game while wearing en EEG device that captured the levels of attention they triggered while playing, levels were then designed according to their performance and those attention levels obtained from the EEG. In addition, 29 players played the same game, without the EEG component, only performance was considered to design levels in real time. In this chapter, it was demonstrated the importance of including brain computer devices to experience adaptation.

Chapter 6 describes the results of applying Graph Grammars to 2D platform games. Graph Grammars is a procedural content generation method originally designed to create dungeons automatically, in this research, it's used for the creation of complex multi-path levels, adding variety and diversity to level design in traditional platform games. 16 players were gathered and played a simple 2D platform game (Super Mario Bros style), which generated levels in an automatic way with specific degrees of difficulty. Players then rated the perceived level of difficulty and was then compared to the one generated by the proposed method. This chapter addresses the last two hypotheses previously explained: level creation with a specific degree of difficulty and taking advantage of dungeon creation techniques in other genres.

The following section gives a brief introduction about the general literature review about the described topics, the last chapters of the manuscript discuss the results of the overall research and what would be the future steps to improve the current state of this study.

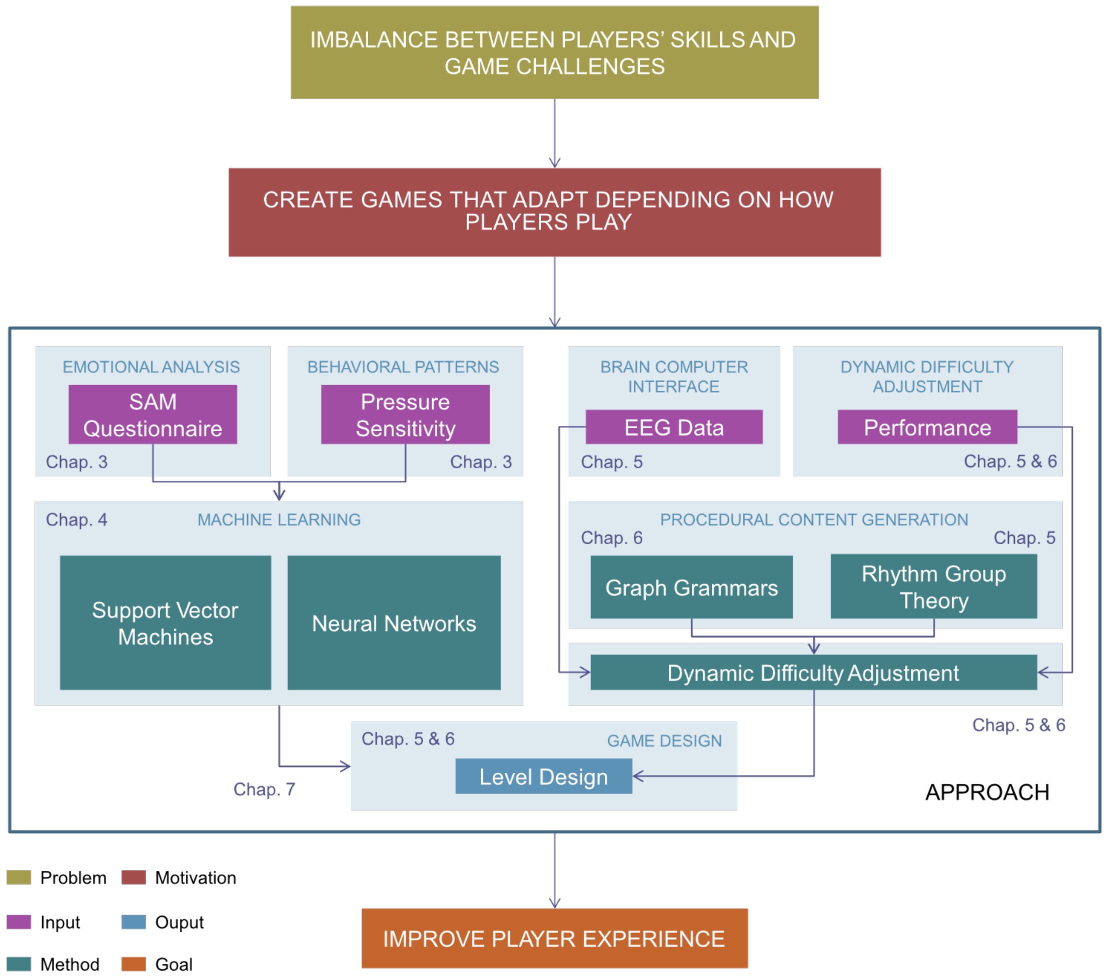


Figure 1.1: Research Map. Problem: The imbalance between players' skills and game challenges. Motivation: To design better methods to improve the player experience. Approach: All the methods tested in this research. Goal: Improve the player experience.

Chapter 2

Related Work

One of the reasons for imbalance between skills and challenge to exist is the lack of harmony in game design, which can be solved by changing the rules of the game and manually adapting its difficulty [1]. The problem with this solution is that not all players have the same skill set and it's very difficulty to match every kind of player's skills.

A common way to tackle this issue is to include Dynamic Difficulty Adjustment (DDA) techniques in games [16][17][18]. By doing this, the game adapts itself to the player, creating a suitable experience regardless the players' skills.

Dynamic Difficulty Adjustment improves the player experience in different ways [2, 3] and even in its most basic or elemental form, when done in the appropriate way, it could successfully adapt the game, making the levels of challenge more suitable for players.

This method is useful and can be implemented in games from different genres, from racing games [16], to platform games [19], from multiplayer digital games [17] to board games [25]; Dynamic Difficulty Adjustment is a method that can be adapted to each developer's or designer's necessities when using the proper approach.

In combination with Dynamic Difficulty Adjustment, techniques such as Procedural Content Generation (PCG) have been used in order to modify the players' experience in real time [19],[20],[21],[22], [23]. Being able to change the content of the game on the fly, gives developer a powerful tool to adapt the overall experience for players, when understanding how a player is performing or what kind of perception he or she has while playing a game, it is possible to adapt the content using algorithms.

There are active researchers such as Gillian Smith, Noor Shaker, Georgios Yannakakis, etc, who work towards personalizing the player experience automatically using PCG and Artificial Intelligence (AI) techniques.

The two following sections of this manuscript, describe the proposed approach when combining Dynamic Difficulty Adjustment and Procedural Content Generation methods with Electroencephalographic data and Graph Grammars, which is a PCG method originally designed for dungeon creation but was tested it in this research with 2D platform games.

Another perspective to help finding better ways to improve the player's experience, is to find patterns on the behavior of players and use them to change the game accordingly. This is the concept of behavioral patterns and has been used by several researchers in this topic [26], [27],[28].

Chapter 3

Behavioral Patterns: Data Collection

3.1 Introduction

This part of the research was conducted with the guidance of professor Koji Mikami (supervisor) and professor Kunio Kondo, who supported the author from the phase of planning and design to implementation and analysis.

In previous sections we have shown the results of different approaches that involve understanding the players' results and status when playing, analyzing it and changing the difficulty of games accordingly. For this part of the research, we decided to explore a different strategy towards finding solutions for our overall goal of improving the player's experience.

Understanding players' feelings, reactions against specific stimuli and how they behave when playing, would be useful when dynamically modifying and adapting a game in real time. This argument led us to delve in the behavioral patterns field.

A three general steps plan was designed to conduct this study: (1)data collection, (2)classification methods and (3)difficulty adaptation. Data collection was carried out using a 2D shooting game and questionnaires for feedback and perception evaluation. From the collected data, we designed machine learning methods: Neural Networks and Support Vector Machines, to find a relationship between the pressure exerted on the button of the controller and how players feel. Finally, using the best results from the classification methods, we propose a new approach that adapts the game to improve the experience for players.

We analyzed the relationship between the levels of pressure exerted on the button of the controller and the players' results, obtained from questionnaires about experience and perception.

In this part we analyzed the correlation between pressure sensitivity values and answers from a difficulty, fun, frustration and Self Assessment Manikin (SAM) questionnaires. We found trends and a close correlation between the parameters. Older players tended to press the button harder than young players (correlation or 0.44). Players with more experience tended to press the button softer than players with less experience (correlation of -0.72 for the amount of time playing and -0.64 for the experience playing videogames).

This section represents the first step of our plan, gives an introduction on the general approach for this new study and summarizes the findings of analyzing the data from a group of players.

3.2 Related Work

The idea of using behavioral patterns to detect the player's current status came from reviewing different authors in previous research related to this topic. Back in 2003, Jonathan Sykes and Simon Brown conducted experiments measuring the pressure used to press buttons on a PlayStation 2 controller while playing clone of *Space Invaders*, their results indicated that it is possible to determine the players level of arousal by using the pressure exerted on the gamepad [26]. In our research, we included some elements from this study, we decided to use a 2D shooter too (of a different kind) and the PlayStation 3 controller, which also includes pressure sensitive capability.

In a race game, there is a study about immersion and its relation with physical behaviors. Their results indicate that the pressure applied to the controller (not the button) was highest when people felt more aroused, more present and felt most dominant while playing the game [27]. Our results accord with the conclusion presented in this previous research about arousal, however, for dominance we obtained the opposite result, the reason for this might be related to the type of game that was used and the way of measuring the pressure on the controller.

Additionally, it was demonstrated that player behavior can be useful as a qualitative measure of player experience, using as a case of study fast-paced action games, their results corroborate the relation between levels of arousal and the pressure exerted on a keyboard [29].

A similar perspective to our general approach of detecting players' emotions through behavioral patterns has been revised by previous researchers in recent years, creating a recognition system using a Support Vector Machine to classify affective states from players using eye tracking and speech signal [28]. This method achieved results with high accuracy and proved to be helpful for being considered in game interface to enhance interaction.

We decided to use the Self-Assessment Manikin (SAM) for valence, arousal and dominance to study and compare emotions when playing the video game. This scale was successfully used for studying emotions in people while playing sounds in a portuguese context [30], in addition, it was used to predict playing time and playing preferences in games [31] and was also used by researchers to maintain engagement by adapting the difficulty [32] all with positive results.

The experiment we designed was based on a research that classified boredom, anxiety and engagement using Support Vector Machine to separate emotions in a Tetris game with different difficulty levels [32]. In our research, we used the same Threshold method approach that was successful in this previous work.

Efforts have been made to detect emotions such as boredom, frustration or state of flow. Particularly, researchers created a computational model to detect frustration in a game called Kane & Lynch 2, the method is capable of informing developers when a sequence of actions might be frustrating for players [33]; although it's a model designed for only one game, its approach could be useful for designing a more general method to be applied in different games.

In the same vein, biometric sensors and self-reports were used to classify negative and positive emotions among a group of programmers, results indicated an accuracy of 71.36% and progress in 67.70% [34]; in spite of the fact that participants are not players and the subject is not specifically related to games, their study involved flow theory, boredom, emotions and the same parameters we evaluate in this research.

Another attempt to improve the player's game experience was to create a method to predict difficulty, immersion and amusement, using a machine learning method to classify the data depending on physiological modality, behavioral modality and meta-information, researchers used peripheral physiological signals, facial recognition, game screen recording and in-game data to analyze the results [35]. As a result from this research, they showed that physiological signal is effective on the prediction of the player's game experience, which leads us to reassure our hypothesis about finding patterns between pressure sensitivity and player experience.

Finally, in a puzzle-based videogame, a new adaptive algorithm was created using flow theory and cognitive load theory to trigger engagement on players, the method was tested comparing it with traditional gameplay and choice-based gameplay, con-

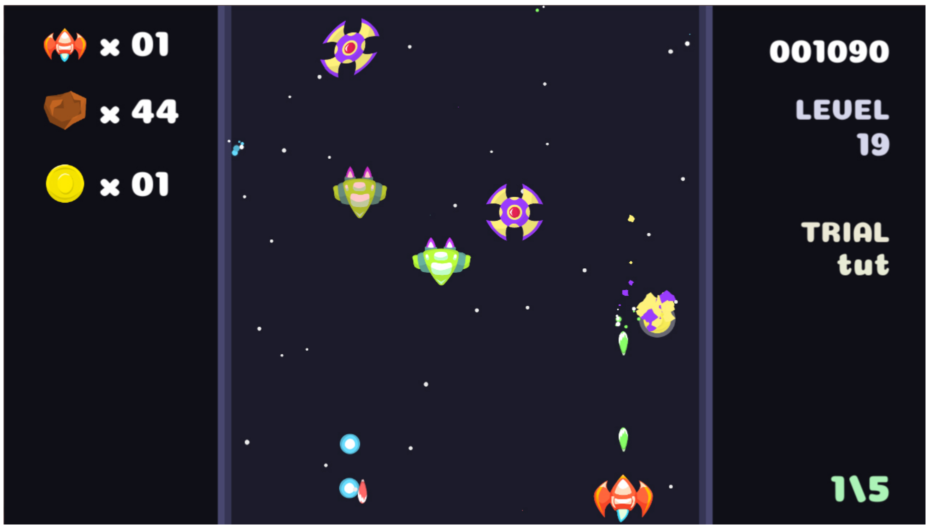


Figure 3.1: A screenshot of a gameplay scene. Players control the spaceship against enemies that appear from the upper part of the screen, the goal is to destroy them and avoid being destroyed.

cluding that those participants that used the adaptive method obtained better results than the other two methods.

In our research, we combine approach and methods that showed positive results in previous studies and consider that our contribution would help to clarify previous conclusions and end up creating a method that successfully classifies players' emotions. Analyzing the relationship between pressure and the parameters that we evaluated for this section adds up to the current status of the field, allowing future research to achieve better results.

3.3 Implementation

For data collection, we implemented a 2D space shooting game. The main goal with our experiments was to gather as much data as possible from the pressure exerted on the button of a controller and, in this type of games, the player has to press the button repeatedly to finish. In addition, shooters were used in previous research [26] and showed positive results. Choosing a similar genre also let us establish a comparison between our results and previous results in an easier way.

Players control a spaceship that can move throughout the whole game space without rotation, when the game starts, the player has three lives and each life includes three points of health. The game is divided in waves, each wave has a fixed amount of enemies that appear from the upper part of the screen, the player has to destroy all

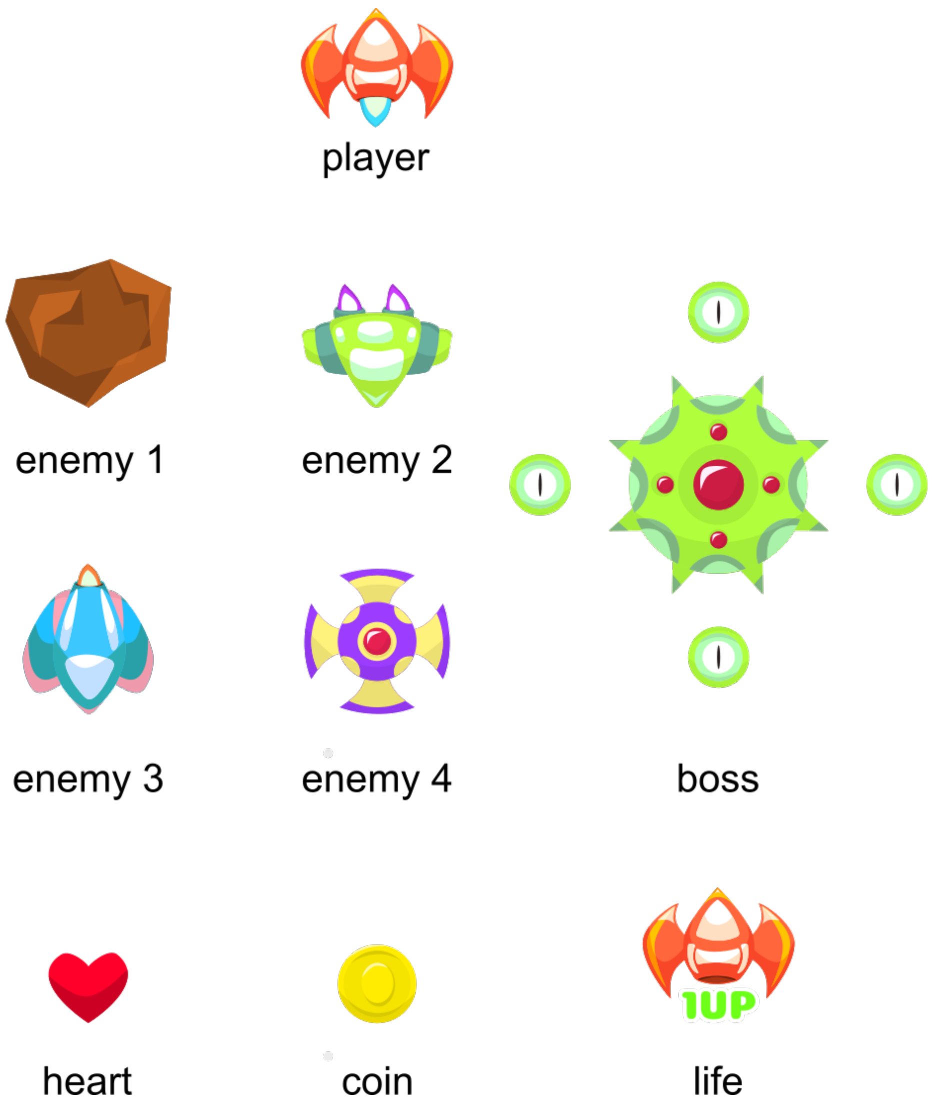


Figure 3.2: Game objects presented in the game.

the enemies that appear on the screen to clear a wave and continue to the next one, a sample of a gameplay screen can be seen in figure 3.1. Powerups were not included for simplicity, the player can shoot using one button and there is no limit in the amount of shots that can be shot. Enemies attack and move in different ways depending on their difficulty and there are three items that aid the player when playing: coins, lives and hearts (health). When players collect 10 coins, a life appears from the upper part of the screen. Coins and hearts are created randomly when destroying enemies. All the elements of the game can be seen in figure 3.2.

3.3.1 DUALSHOCK 3: PlayStation 3 Controller

We decided to use the DUALSHOCK 3 (PlayStation 3 controller) for our experiments due to its pressure-sensitive buttons capability. The controller's analog buttons enable developers to accurately capture the pressure when pressing buttons on the controller. Players used the left thumbstick to move the avatar on the screen and the X button to shoot.

3.3.2 Tools: ScpToolkit and Unity

The source code of a Windows driver designed for the DualShock 3 and 4 controllers was used to capture the pressure sensitive exerted on the buttons of the controller. The source of this tool is available under the GNU General Public license [36]. We wrote a layer of code that connected the libraries of this tool with Unity to develop the game and use the pressure sensitivity in real time.

3.3.3 Level Design

Levels were manually designed considering the following parameters: number, type and behavior of enemies for each wave. The difficulty of each level was designed in a way that the next wave's difficulty was always equal or higher than the previous wave. In order to calculate the difficulty for each wave, we used equation 3.1, which was proposed in a previous research [37].

$$d = \frac{n}{m}W_1 + \sum_{i=1}^{n} d_i W_2 \tag{3.1}$$

W1 was set to 0.9 and W2 to 0.1, in the same way that it was done in the previous work [37]. The number of enemies affects the overall difficulty of a the game more than the type of those enemies because they can cover more space on the screen and attacking at the same time creates a higher challenge for players.

Where:

- d: Difficulty of a wave
- n: Number of enemies in the wave
- m: Maximum number of enemies in that wave
- di: Difficulty of each enemy (see table 3.1). This was normalized for the calculation.

Table 3.1: Enemies features: Binary representation for the skills: 1=true,0=false

Skill	E1	E2	E3	E4
Bullets	0	1	2	4
Direction	0	1	1	2
Health	1-3	3	4	5
Speed	1	1	2	3
Total	2-4	6	9	14

- Wi: The influence of each part of the equation in the final decision

To calculate the difficulty for each enemy we used table 3.1. Enemies were designed to be different, add variety to the game and represent different difficulties.

3.3.4 Enemies Behavior

Although all waves were manually designed, enemies behave in a random way according to specific parameters. To increase difficulty and add variety to the game, we allow enemies to shoot and move in different ways, speed and number of bullets shot consecutively could be chosen from the design of the wave, enabling us to specifically decide how could they behave but when and how to move and shoot was randomly selected so players didn't notice that sometimes enemies were the same with different abilities.

3.4 Experiment

Using previous research as a model [32], we divided the experiment in two phases: threshold phase and Trials phase. Before starting the experiment, players were required to fill out a questionnaire about themselves and their experience playing games. Questions included: age, gender, playing frequency ,time playing videogames in a session, experience playing games, player's skills, experience playing 2D shooters and skills playing 2D shooters.

3.4.1 Threshold Phase

Since every player has different skills and experience, we needed to know what level of difficulty could they handle inside the game, in order to determine that, we used the threshold method. We started the experiment from wave 1 and every time the player cleared the current wave, they were taken to the next one which was more difficult than the previous one but not so difficult so it was impossible to clear.

Considering that this process could have taken a lot of time from the experiment, we included another condition during this phase, a challenge that represented a difference between players with more experience than others, if they overcame that challenge they would skip several waves to play levels that would be more suitable for their skills and if they didn't overcome the challenge, they would just be taken to the next wave. The challenge we include is the boss enemy (can be seen in figure 3.2 as *boss*).

The boss appeared during this phase every 12 waves, when the challenge started, the player's health was restored to 100% so we could evaluate if they were capable of beating the boss regardless their performance before reaching that wave. If the boss was defeated, players had their health restored to 100% again and then taken 6 waves ahead to keep playing; in case they lost one life (were defeated by the boss) then the boss disappeared and players had one life restored to 100% of health and continue playing on the next wave. This condition was included to speed up the experiment and to avoid boredom experienced during this phase to affect the next phase of the experiment. This character appeared only during this part.

The stopping condition for this phase was that the player lost all lives and died. Once that this happened, players took a rest for 1 min. and filled out a questionnaire about the the experience they had while playing. Items such as lives and hearts were removed from this stage to avoid extending the game indefinitely.

The maximum wave reached during this phase was used a pivot for determining the player's skills and suitable level of difficulty.

3.4.2 Trials Phase

During this part of the experiment, players had to complete 6 different trials of a specific number of waves each, two trials were created with medium difficulty (calculated during the threshold phase), two trials were created with easy difficulty (the medium wave - 16 waves) and two trials were created with hard difficulty (the medium wave + 16 waves). In the original experiment conducted in a previous research, the difference between levels was 8, not 16 but after doing preliminary experiments with players and receiving feedback on the difficulty being almost the same for all trials, we decided to increase and decrease the difficulty 8 more waves for the hard and easy trials respectively.

Each trial had 2 different stopping conditions: (1) players cleared 16 waves (which would be completing all the waves before reaching the next difficulty) or (2) 3 minutes passed while playing the game and they were still alive (in previous research each trial consisted of 5 min. but we decided to reduce it to keep the experiment shorter and avoid distractions/fatigue). In case players died, they were asked to continue playing (from the start of the trial) but the time condition was not restarted, it means that the maximum amount of time they would spend in each trial would be 3 min. The reason why a game over condition was included is that we wanted to give players all the features of the game because we didn't want them to get discouraged while playing. Trials were presented to the player randomly.

3.4.3 Experience Questionnaire

After completing each session of play, each trial, players were asked to fill out a questionnaire about their experience about the part of the game they played. The first group of questions were related to difficulty, fun, frustration and boredom, they were designed using a 5 points Likert Scale.

In addition, a Self-Assessment Manikin (SAM) test was conducted regarding the trial they played, players were asked to rate their emotions toward the play session using a valence-arousal-dominance space. A sample of the SAM test can be seen in figure 3.3.

The questionnaire for evaluating each session of play is shown as follows:

Questionnaire 1: Playing Experience

1. How would you rate the difficulty of the levels you just played?

2. How would you rate the level of fun you had playing these levels?

3. How would you rate the level of frustration you had playing these levels?

4. How would you rate the level of boredom you had playing these levels?

5. In the following categories, please choose the best approximation on how playing these levels made you feel. Use the numbers and pictures as a reference to decide. Figure 3.3 was displayed after this question and participants were able to choose between 1 - 9 for each category (Valence, Arousal, Dominance)

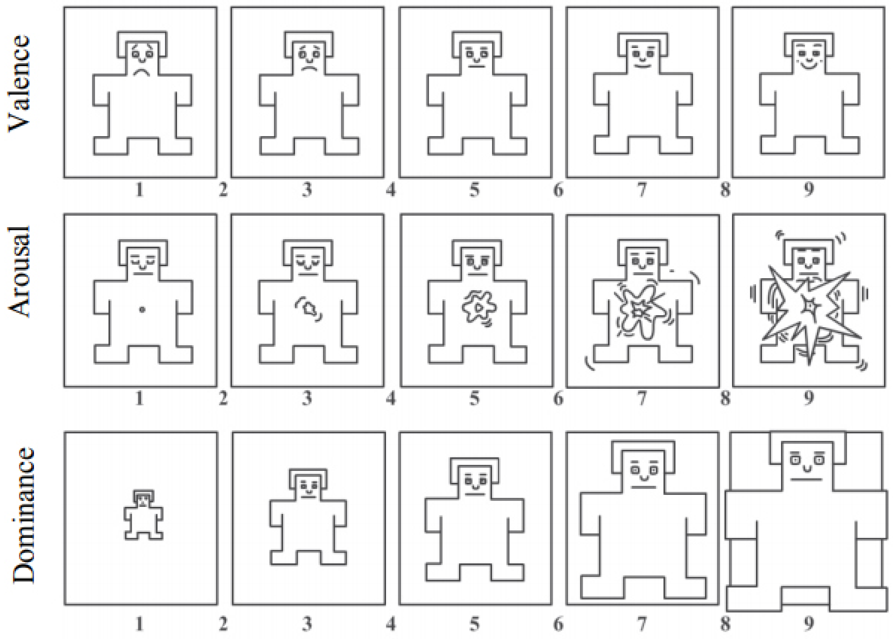


Figure 3.3: Self-Assessment Manikin Test: Players were asked to rate their feelings after playing each trial.

3.5 Results and Analysis

Collected data was gathered from 20 participants with different game skills and characteristics. 75% of the players were male participants; their age ranged between 12-44 years old and players with low, medium and high experience were tested.

3.5.1 Player's Profile and Experience

We calculated the correlation between the average pressure applied to the button of the controller per each participant and the answers obtained from the questionnaire before starting the experiment: age, gender, play frequency, play time, experience ans skills. Results for this correlation are shown in table 3.2.

The questionnaire for evaluating the player profile information is shown as follows:

Questionnaire 2: Player Profile

1. What is your age?

2. What is your gender?

3. How often do you play videogames?

Table 3.2: Player's Profile. Correlation between pressure and: (Q1)age, (Q2)gender,(Q3)playing frequency ,(Q4)time playing videogames in a session, (Q5)experience playing games(general), (Q6)player's skills(general), (Q7)experience playing 2D shooters, (Q8)skills playing 2D shooters

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8
0.44	0.01	-0.52	-0.72	-0.64	-0.31	-0.27	-0.30

4. How much time do you usually spend playing video games?

5. How much experience do you have playing games in general?

6. How would you rate your skills playing games in general?

7. How much experience do you have playing 2D shooting games?

8. How would you rate your skills playing 2D shooting games?

Pressure and age are correlated in 0.44, a weak positive correlation showing that the older players are, the harder they press the button of the controller; we consider that this result is connected to the loss of motor skills that people experience when ageing, people with low motor skills would try to compensate the lack of accuracy by pressing the button harder trying to overcome a challenge.

Gender has a a very low positive correlation of 0.01, it doesn't seem to be an important factor but in order to get better results on this parameter we would have to conduct the experiment with more participants, including more female participants.

For the rest of the parameters, all of them related to the experience of players, we have that playing frequency, playing time and amount of experience playing videogames have a moderate and strong correlation with pressure. Players with more experience are expected to being more used to handling a game controller, they very likely have faced more difficulty challenges than the ones designed for this experiment therefore, it's an expected result. The rest of the parameters, despite the fact that they represent a weak correlation, support the assertion about experience.

3.5.2 Player's Perception and Feelings

The correlation between average pressure for each player and the results obtained from the questionnaire of experience when playing the game can be seen in table 3.3. Some of the calculations were undefined due to the way players answered (all answers were the same) and these results are not considered for the analysis.

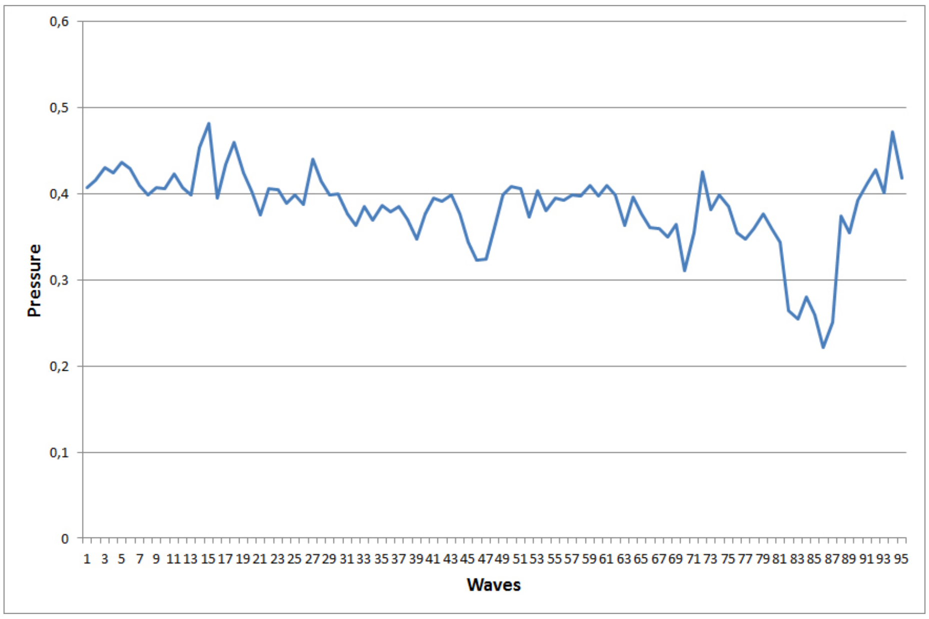


Figure 3.4: Average pressure results for all players per each wave

Difficulty shows a positive correlation for 73.33% of the participants, ranging from moderate to strong correlations depending on the player. We can clearly see a trend and we consider it's expected from players to press the button of a controller harder when a game is more difficult, the higher the challenge, the more focused they have to be and the faster should react. Two players showed a strong correlation (0.81, 0.70) and the rest of them between weak and moderate positive correlation. From players that showed negative results, only one has a strong correlation of -0.75.

We calculated the average pressure for every wave in the experiment, including the threshold phase and trials phase as well, the graphic is shown in figure 3.4. In previous research we have seen that the more difficult a challenge is, the higher the pressure players apply to the button of a gamepad. In our analysis, we also found out that the majority of players showed a positive correlation for these two parameters. Despite the fact that the overall graph shows a decreasing tendency, not all players were able to play all waves in the game, only high skilled players reached the last waves seen in the graph. This confirms that players with higher skills in fact exert less pressure on the controller and the more difficult the challenge, the higher the pressure (evaluating chunks of the curve, not as an overall).

From the results obtained for the question regarding fun, 76.92% of the participants (excluding the undefined results) showed a positive correlation, however, the majority is a moderate or weak correlation, there are only two players that had a very strong correlation (0.96, 0.75), on the other hand, for players that showed a negative

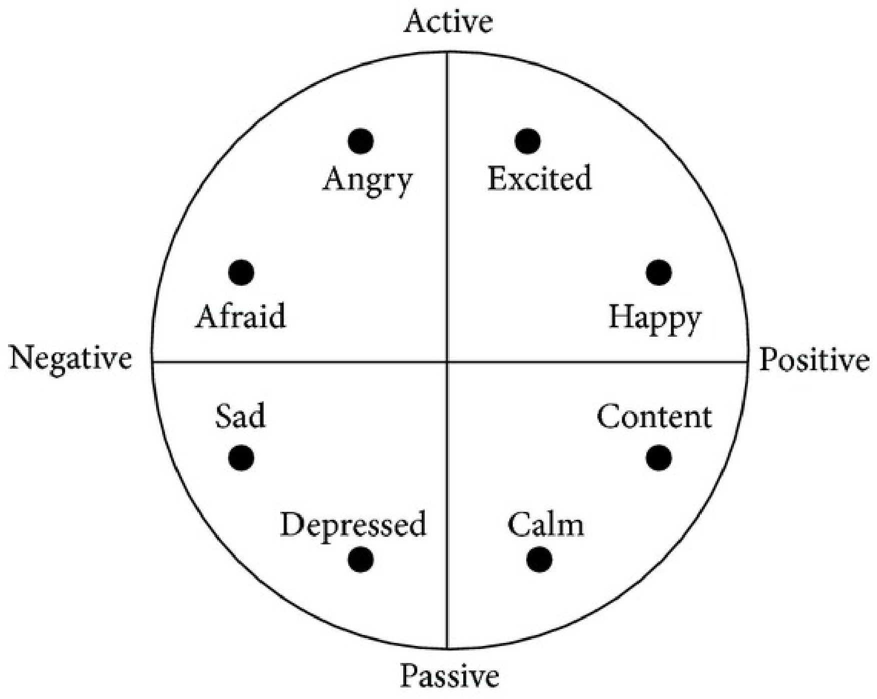


Figure 3.5: valence-arousal space representation. Emotions represented in a 2D space for classification.

correlation, all of them have a moderate strength. It's difficult to conclude something from these results, apparently more people press the button harder when they are feeling fun but for people whom experience the opposite, results are more consistent.

Frustration had 61.54% of positive correlation from all the participants that could be calculated, two of them with strong correlations, the rest were negative with weak correlation, except for one participant that showed -0.75. The majority of players pressed the button harder when they felt more frustrated, previous research showed that players increased the force on the controller when playing harder levels, when players play games with a degree of difficulty higher than their skills they tend to feel frustrated too. For the specific case of players 4 and 5, both showed negative correlations for difficulty and frustration, in these particular cases we consider that this might be related to how the player handles pressure in general.

For boredom, 53.85% of the players showed a negative correlation, ranging from moderate to strong. The softer they pressed the button, the higher the boring feeling was. Boredom manifests in lack of motivation, players were not motivated enough to press the button harder when they felt bored, this is an expected result from this particular case. For the players that showed the opposite results, we don't have clear

Table 3.3: Correlation between pressure pressure and: (Q1)difficulty, (Q2)fun,(Q3)frustration ,(Q4)boredom, (Q5)valence, (Q6)arousal, (Q7)dominance

Pla.	Q1	Q2	Q3	Q4	Q5	Q6	Q7
1	-0.21	–	0.79	0.36	-0.36	0.51	-0.91
2	0.20	0.37	0.08	-0.64	-0.54	0.07	-0.35
3	-0.11	-0.58	0.25	0.38	-0.43	-0.61	-0.24
4	-0.75	-0.49	-0.75	–	–	-0.15	–
5	-0.15	0.96	-0.17	-0.08	0.32	0.17	-0.45
6	0.15	0.25	-0.14	-0.72	0.25	-0.08	-0.54
7	0.70	–	0.15	–	–	0.46	-0.22
8	0.30	-0.54	–	0.51	-0.30	0.07	-0.83
9	0.61	0.33	–	-0.08	0.29	0.65	-0.49
10	0.81	0.48	0.81	-0.60	0.63	0.88	-0.42
11	0.22	0.14	-0.11	-0.20	0.19	0.16	0.59
12	0.33	0.29	0.03	0.06	0.40	-0.21	-0.41
13	0.35	0.44	-0.11	-0.09	0.03	0.52	0.30
14	0.46	0.75	0.46	0.05	0.87	0.71	0.14
15	0.01	0.13	0.41	0.19	-0.27	0.08	-0.09

conclusions but assume it's related to specific ways of reacting to boredom.

The analysis of valence, arousal and boredom, includes the arousal-valence space model shown in figure 3.5, in addition to evaluating the correlation of each parameter with pressure values, we displayed the results on this space in figure 3.6. We can observe in the graph that the majority of the results are located around on the positive quadrants, excited, happy, content and calm, however there are some results on the neutral state and few ones on the sad, afraid and depressed quadrant. In general, this is an expected result from this experiment, players were focused on trying to obtain good scores while playing, positive and negative emotions that translated into excitement or calmness, triggered by the difficulty of the levels.

Valence showed a result of positive correlation for 61.54% of the players, being almost all of them weak or moderate, except with one strong correlation (0.87). The happier players felt when playing, the harder the pressed the button, according to these results. Considering the values in table 3.4 showing the correlation between the parameters (not pressure), we can see that the highest correlation is fun/valence, the more fun players experienced, the happier they felt and harder they pressed the button on the controller. These are natural reactions to experiencing positive stimuli, projected physically on the behavior of these players. In previous research [27], researchers presented results for arousal but not valence, our results suggest that

Table 3.4: Enemies features: Binary representation for the skills: 1=true,0=false

	Dif.	Fun	Fru.	Bor.	Val.	Aro.	Dom.
Dif.	–	0.39	0.51	-0.12	0.31	0.34	-0.15
Fun	–	–	0.22	-0.37	0.65	0.33	-0.08
Fru.	–	–	–	0.11	0.07	0.52	-0.10
Bor.	–	–	–	–	-0.37	0.06	-0.07
Val.	–	–	–	–	–	0.34	0.26
Aro.	–	–	–	–	–	–	0.09

the correlation on valence is not clear enough for all players but it is closely related to the factor fun.

73.33% of our participants' results for arousal show a positive correlation with pressure, some of them moderate and one strong correlation. These results accord with previous research [26, 27, 29], arousal and pressure are directly proportional.

Finally, dominance has the the most solid results of the whole study, 78.57% of the participants evaluated show a negative correlation with pressure, the harder the press the button the less dominance they feel. We consider that this is related to how powerful players feel when playing, if they are stressed out by difficult challenges, they are not in control of the situation, feeling less dominant towards the game. In previous research [27], the opposite conclusion was presented, we believe it's likely related tot he fact that both experiments were different and the parameters we chose for measuring the pressure, we were focused on the button itself, previous research was focused on the controller as a whole.

3.5.3 Parameters Correlation

We calculated the correlation between all the parameters that we considered for this study, table 3.4 shows all the results. We found out that valence and fun are moderately correlated with a value of 0.65, that's why we mentioned the importance of these two factors in the previous section. The more fun players are having, the happier they feel. Frustration and arousal also have a moderate correlation of 0.52, which is an expected result. When players are frustrated, they tend to get angry or excited and it's exactly what we can see in figure 3.5 that shows the emotions triggered by each parameter. Finally, difficulty/fun and difficulty/frustration have a weak and moderate positive correlation of 0.39 and 0.51 respectively, players find challenges more fun than easier tasks and the more difficult a challenge is, the more frustration it can cause. This could be explained using the flow state, players' skills

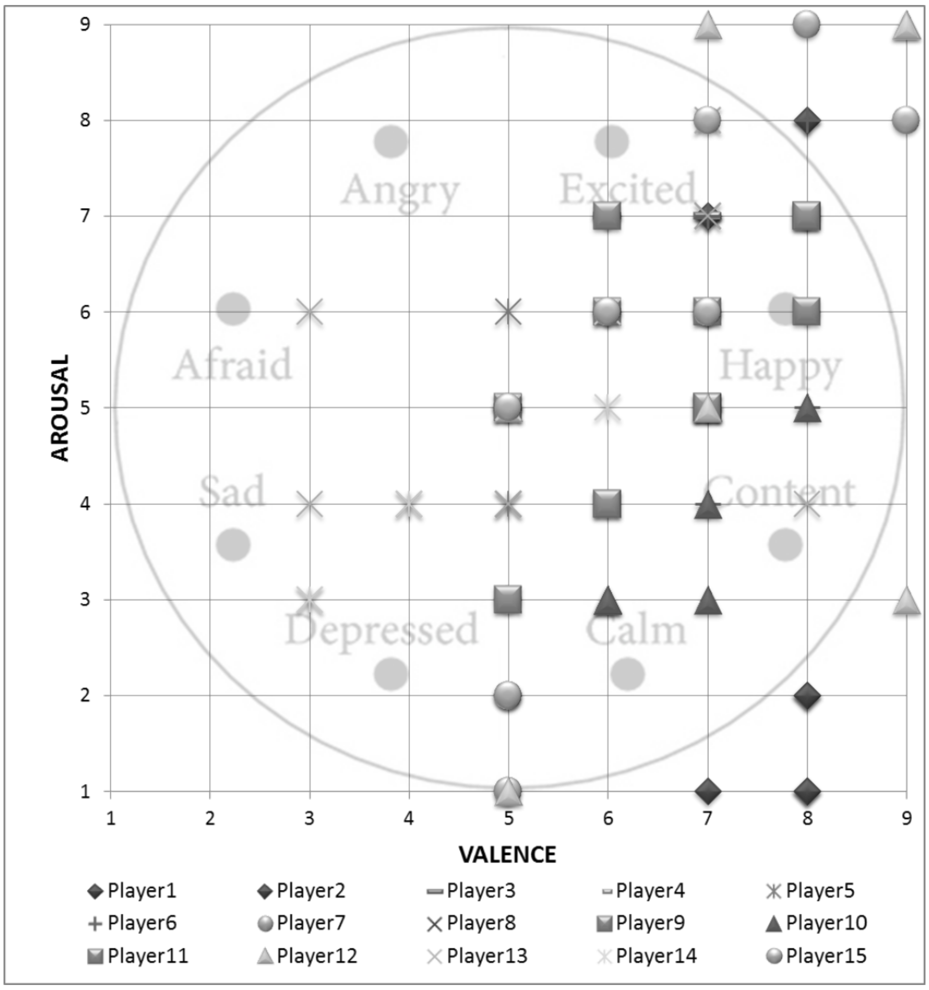


Figure 3.6: Results from the experiment displayed in a valence-arousal space to classify players' feelings.

and level of challenge they face, players tend to perceive little challenges over their skills more fun than challenges under their skills which ultimately trigger boredom.

3.5.4 Parameters Classification Proposal

As an overall goal for our research, we plan to use the pressure on the button of a gamepad to determine how players feel when playing a game and modify the game accordingly to improve the experience. After analyzing the obtained results from the first step of our approach, we consider that some of the parameters we evaluated could be classified according to the pressure levels. Having high negative correlation values on the player experience and pressure, we can estimate the overall experience that a player has by looking at patterns from the pressure on the controller; we can also assume that since dominance is inversely proportional to the pressure, players

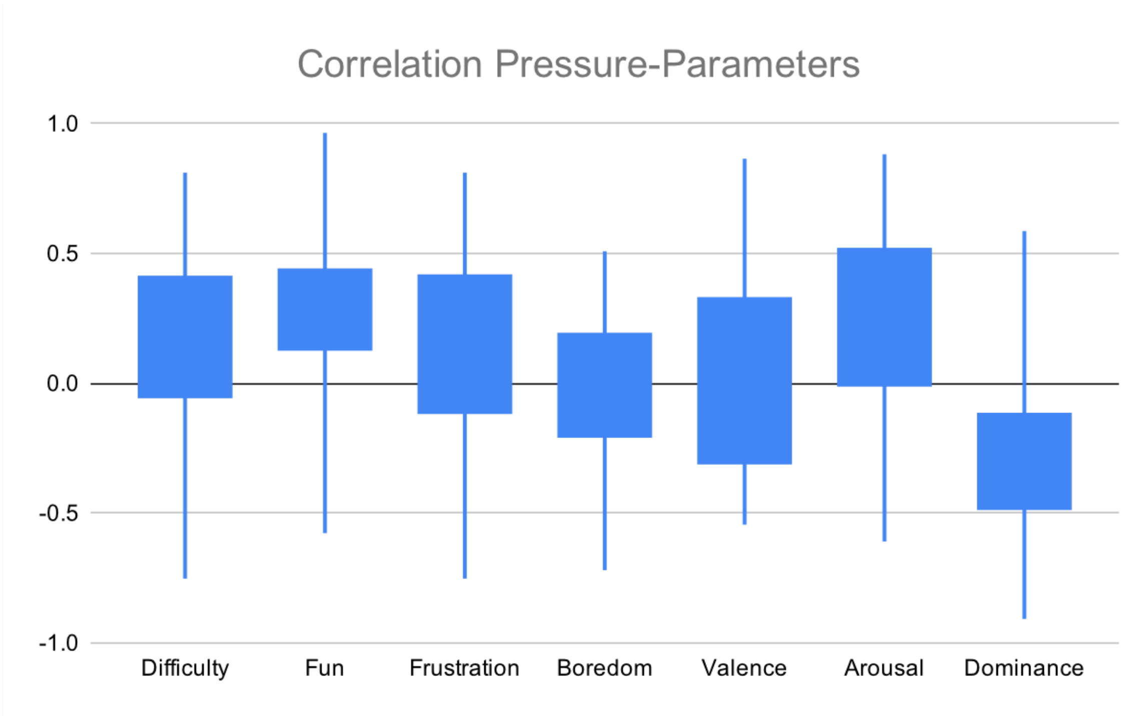


Figure 3.7: Correlation: Pressure-Parameters. Results for all experiments.

that show high pressure values during playtime, might not feel in power. The real importance of analyzing these results before continuing with the next step of the plan, is to focus on the parameters that best represent the players' emotions, by understanding these results, we will be able to make a more accurate method to classify these factors and offer an improved experience as a whole.

Table 3.5 shows the results for all the experiments conducted in this part of the research. In addition, figure 3.7 shows a box plot applied to these results.

3.6 Discussion

Although our approach with this research is the result of a combination of elements from previous studies, using pressure sensitivity exerted on a button's gamepad to recognize emotions felt by players, is a novel way to determine players' behavior. We consider that with this new idea, it will possible to dynamically change the content to make games that are more engaging and enjoyable for players.

According to the results obtained from our experiments, there is a significant correlation between pressure sensitivity and some of the parameters evaluated when conducting questionnaires to players. This means that there is a relationship between

Table 3.5: Correlation Pressure-Parameters

Player	Difficulty	Fun	Frustration	Boredom	Valence	Arousal	Dominance
1	-0.21	NAN	0.79	0.36	-0.36	0.51	-0.91
2	0.20	0.37	0.08	-0.64	-0.54	0.07	-0.35
3	-0.11	-0.58	0.25	0.38	-0.43	-0.61	-0.24
4	-0.75	-0.49	-0.75	NAN	NAN	-0.15	NAN
5	-0.15	0.96	-0.17	-0.08	0.32	0.17	-0.45
6	0.15	0.25	-0.14	-0.72	0.25	-0.08	-0.54
7	0.70	NAN	0.15	NAN	NAN	0.46	-0.22
8	0.30	-0.54	NAN	0.51	-0.30	0.07	-0.83
9	0.61	0.33	NAN	-0.08	0.29	0.65	-0.49
10	0.81	0.48	0.81	-0.60	0.63	0.88	-0.42
11	0.22	0.14	-0.11	-0.20	0.19	0.16	0.59
12	0.33	0.29	0.03	0.06	0.40	-0.21	-0.41
13	0.35	0.44	-0.11	-0.09	0.03	0.52	0.30
14	0.46	0.75	0.46	0.05	0.87	0.71	0.14
15	0.01	0.13	0.41	0.19	-0.27	0.08	-0.09

feelings and perception experienced by players and the way they physically behave when using the controller.

Indications of this relationship were also shown in previous research [26, 34]. Our approach differs from previous ones in the way that elements were combined for the experiments and their evaluation, in contrast with the related literature, we found out that some parameters such as dominance and fun, have a positive correlation.

Some of the parameters evaluated in this part of the research, were not evaluated in previous studies or were evaluated in unrelated works. We consider that a combination of different parameters can lead to more interested ways to predict the user's behavior in a better way.

In addition, the correlation obtained from analyzing the results of our experiments, show that there is a clear relationship between how players feel and the interaction that they have with the controllers, which was one the previously defined hypothesis. These results add up to the previous work [26], which also indicated that there was a relationship between these two elements.

This part of our research contributes to clarify a couple of objectives defined in the beginning of this document: (1) proving how can emotions be predicted using behavioral patterns; (2) proposing a concrete method to connect them. This is the first step to propose a concrete method to link behavioral patterns and player's emotions. Despite the fact that we only calculated the correlation between pressure sensitivity values and a group of pre-defined parameters. Having a way to predict these parameters using players' behavior would be big step towards finding a concrete method to adapt the player experience.

Among the things that can be done to improve this research, we consider that analyzing the data with statistic methods or including other parameters would contribute to obtain more significant results.

3.7 Chapter Conclusions

We proposed a general approach to improve the experience for players when playing video games through behavioral patterns, specifically pressure sensitivity. As a first step of this proposal, data from 20 participants with different skills and characteristics was obtained and analyzed using the correlation, self-reporting methods and previous research.

Results corroborated some of the hypothesis and results done by previous researchers. We showed that pressure and difficulty are directly proportional. Arousal and pressure also had a moderate to strong positive correlation. Players pressed the button of the controller harder when they face more difficult challenges and the happier they feel when playing, the harder they press the button.

In addition, one of the clearest results we obtained was re relationship between dominance and pressure. There was a negative correlation between these two parameters for 78.57% of the participants, demonstrating that the harder players press a button, the less dominant they feel. We consider that this is related to how powerful players feel when playing and it's triggered by stress felt due to complicated situations such as hard challenges.

Another consistent result with 76.92% of the participants showing a positive correlation was the relationship between fun and pressure, people tend to press the button harder when having more fun, players express the excitement through the controller.

Analyzing the results with the personal data obtained from players and experience reported by them, we observed that older players tend to press the button harder, which could be a consequence of deterioration of motor skills, leading to lack of accuracy and triggering physical actions to compensate.

Players with more experience tend to press the button softer, which is an expected results after analyzing it, players with more skills are more used to handling a controller, they have also faced more challenges than players with less skills, therefore, these players react calmed and can easily overcome challenges with less difficulty.

Finally, we consider that this study is not only important for our overall research approach because it shows us which parameters are more correlated to other and helped us make a clear proposal about the design of a better method to classify emotions for players, but also the conclusions of this research serve to corroborate previous results and to show new analysis that contrast with other results.

Chapter 4

Behavioral Patterns: Classification Methods

4.1 Introduction

As well as the previous chapter, this part of the research was conducted with the guidance of professor Koji Mikami (supervisor) and professor Kunio Kondo, who supported the author from the phase of planning and design to implementation and analysis.

This section constitutes the second step of our new approach: Classification Methods, which is part of a more general goal of improving the players' experience by analyzing their behavior through the pressure exerted on a gamepad's button.

We used the data collected from an experiment conducted with 20 different players and design classification models: Neural Networks and Support Vector Machines, to predict, in real time, how players perceive the game or how they feel.

Support vector machines and neural networks are widely used to classify data and make predictions in real time [38, 39, 40]. For our specific problem, which involve behavioral patterns related data, emotions and player's perceptions, some researchers working in the same feel have achieved good results using machine learning methods [41, 32, 28]. Due to the nature of our problem and the effectiveness of neural networks and support vector machines, we decided to used them for classification.

Players played 6 levels of a 2D space shooter and, after finishing each level, questionnaires to measure their perception towards the game were applied, among the measured parameters we have: difficulty, fun, frustration, boredom, valence, arousal and dominance. In addition, in-game data was collected for future analysis.

All the collected data was used with a set of different machine learning algorithms, three different types of neural networks and three support vector machines for each parameter measured in the questionnaires were designed, we tested several times to obtain the best possible results and input values for each method.

The following subsections describe how we designed the learning algorithms and results for the best method and parameters.

4.2 Related Work

There are several studies that involve the recognition of emotions and in-game player related parameters. Creating a recognition system using Support Vector Machines, researchers classified affective states from players and collecting speech signal data and an eye tracker [28]. Results showed high accuracy by applying these methods and they are being consider for improving the player interaction.

In order to predict valence, arousal and dominance, researchers focused on using random forests to prevent undesired emotions when playing video games [41]. As a result from this study, Frommel et al. proposed an architecture to react to player's emotions using procedural content generation and emotion detection.

Support vector machines were used to predict emotions and to keep players engaged while playing a game by adapting its difficulty automatically according to the predicted outputs [32]. Positive results with an accuracy of 53.33% were achieved by researchers in this study. We decided to design our experiments in a similar way to what Chanel et al. did, we adapted and changed their approach to include it in our proposal.

Using physiological signals, it was shown that game experience can be accurately predicted [35]. A machine learning method was implemented to classify difficulty, immersion and amusement from players while playing *FIFA 2016*. This study involved physiological signals, facial recognition, game screen recording and in-game data.

Using flow theory and cognitive load theory to trigger engagement on players, researchers designed a new adaptive method that showed successful results by comparing their proposal with traditional gameplay and choice-based gameplay [42, 43].

Considering that previous research has demonstrated that player's behavior is not only directly related with that players are experiencing but also can be classified and estimated, we decided to combine similar elements from the previous work and include it in our approach, using the features that lead to positive results. We attempt to measure and understand the player's emotions and game related parameters using machine learning methods.

Table 4.1: Number of samples per class.

Cla	Dif	Fun	Fru	Bor	Val	Aro	Dom
1	38092	49921	34067	29263	53513	65464	64987
0	85394	73565	89419	94223	69973	58022	58499

4.3 Machine Learning: Data Classification

For this part of our research the main goal was to analyze the collected data and search for patterns that could show a relationship between the specific behavior of pressing a button and how the player perceives game related parameters and emotions. To evaluate the data and find possible patterns, we used Neural Networks and Support Vector Machines and compare results of each method using accuracy calculated in the testing phase.

As a general approach for evaluating the results of each machine learning system, we took all the data gathered from our experiments, used that as input for different neural networks and support vector machines that were specifically designed for each parameter or emotion that was presented in the questionnaires to finally predict how the player was feeling or perceiving the game at that specific time. Figure 4.1 shows the general idea.

Different machine learning systems were created for each specific parameter evaluated in the questionnaire. Neural networks and support vector machines for difficulty, fun, frustration, boredom, valence, arousal and dominance were tested. In addition, depending on the parameter's output, a different system was created, the output for each question was converted to the space of 0s and 1s to achieve better results, more details about the structure and design decisions will be explained in the following sections.

4.3.1 Input Data

Collected data from all users (20 people) with a total number of samples of 123486 including all the player's results. We used the method Leave One Out (LOO), which is a special case of cross-validation where one participant's data is used to test and the rest of the data to train the model; this process is repeated until all participants' data is used [44].

The input size of our models was 5, we determined this value by testing with different input sizes and choosing the one with the best result.

Table 4.1 shows the details of each class for each parameter.

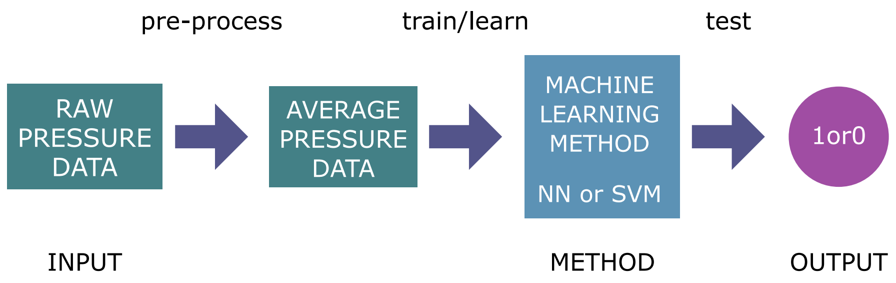


Figure 4.1: General Approach: Use input data in a machine learning method to predict parameters

4.3.2 Input Data Pre-Processing and Labeling

All data was separated and labeled depending on the answers given by players while doing the experiment. After completing one full session in the game, all the collected data for that specific session was classified using the given answers, as an example, if the player rates one session as 'very difficult', all the input will be labeled as 'very difficult' for that session.

Since the pace in one level can change from easier to harder while progressing, labeling all the input values in one specific category could lead to loss of granularity about what the player is really feeling in each part of the game, this is why we decided to pre-process the data and calculate the average of each input by the time the input was stored. Using the average data of each input value, allowed us to work with the tendency of that session instead of values in a specific time.

4.3.3 Output Data Codification

Questions for the first four parameters were designed using a Likert scale of 5 points; the second part of the questionnaire was designed using the SAM method which consisted of 9 different answers for each parameter.

In order to use the collected data for the mentioned machine learning methods, we grouped the output vector according to three different categories: low, medium and high. Parameters that were originally codified with 5 outputs were transformed to a binary output and parameters that were originally codified as 9 outputs, were transformed to a binary output as well. This means that we created one specific machine for each parameter and for each group of that parameter, being in total, 3 different machines per parameter and for each method respectively.

4.3.4 SciKit Learn

All the evaluation was conducted using a free software machine learning library called Scikit-learn. In order to choose the best parameters for each method, we conducted a series of preliminary simulations for each system and, in the end, we took the parameters with best results and use them to calculate what we present in this paper as our results.

Neural Network's Parameters:

- Rate: Constant
- Activation function: Relu
- Number of hidden nodes: 100
- Max. iterations: 1000
- Solver: lbfgs

Support Vector Machine's Parameters:

- Kernel: Radial Basis Function
- Degree: 3
- C-Parameter: 1
- Gamma: $\frac{1}{n}$ (being n the number of features)

4.4 Accuracy Calculation

We used the calculated accuracy to determine how well a system was able to predict the results for a specific parameter with a specific configuration using pressure sensitivity. High accuracy means better prediction when deciding what is the best classification for an input.

Accuracy for each model was calculated by obtaining the mean value of all accuracy results for each iteration of the LOO cross validation. Equation 4.1 shows the formula to calculate the overall accuracy for each model.

$$M(X,Y) = \frac{1}{n}\sum_{i=0}^{n-1} A(X_i, Y_i) \tag{4.1}$$

where:

- X represents a set with all the Leave One Out training data; Y is a set with all the expected outcomes for X.
- $A(X_i, Y_i)$ is the accuracy calculated for the specific set i. This function is defined in equation 4.2.

$$A(x, y) = \frac{1}{n} \sum_{i=0}^{n-1} f(x = y) \tag{4.2}$$

where:

- x represents the test data; y the expected outcome for x.
- $f(x = y)$ indicates whether x is equal to y or not.

4.4.1 Proficiency Levels of Players

Data from all types of players was collected during our experiments. We did not divide the data or the players' results by proficiency when designing and validating our models.

4.5 Results and Analysis

Collected data was gathered from 20 participants with different game skills and characteristics. 75% of the players were male participants; their age ranged between 12-44 years old and players with low, medium and high experience were tested.

4.5.1 Best Results

As a general result, support vector machines performed better than neural networks in all parameters except frustration, for which neural networks showed a better result. The average accuracy (all parameters) for neural networks was 68.55% and for support vector machines was 70.69%.

The best result was achieved with the boredom parameter, with 83.64% of accuracy, difficulty rated by the player, fun, frustration and dominance, were predicted with more than 70% of accuracy. All the average results can be seen in figure 4.2, showing all the evaluated parameters from the questionnaires in the horizontal axis, vertical axis shows the calculated accuracy.

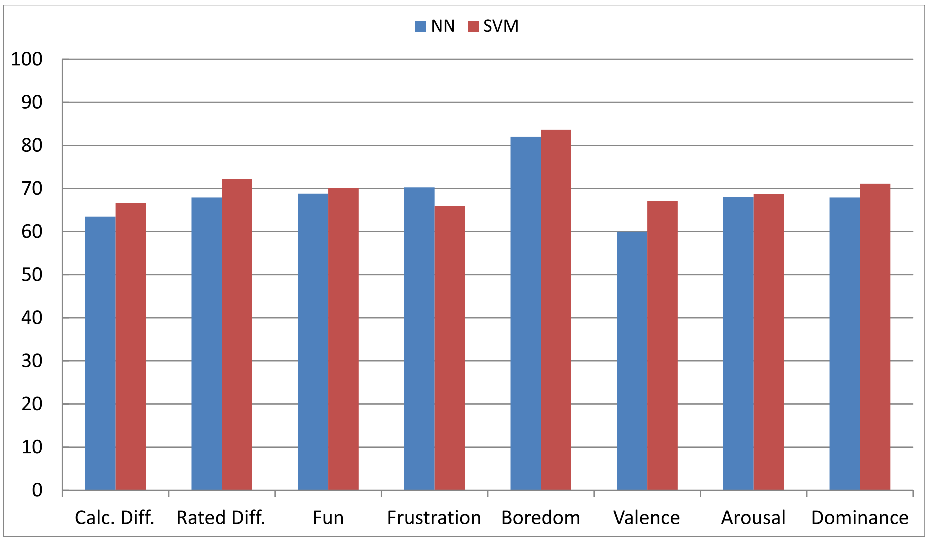


Figure 4.2: Average accuracy results for Neural Networks and Support Vector Machines

Figure 4.3 and 4.4 show the best achieved results with support vector machines and neural networks respectively. As well as figure 4.2, the horizontal axis shows all the evaluated parameters, obtained from data collected using the questionnaires about emotions and player's perception. Each parameter was tested with three different machines: detecting low, medium and high difficulty output configurations. Vertical axis shows the accuracy for each parameter and with each output configuration.

From these two graphs and the calculated average result of each output configuration, we have that recognition of the low codification was the highest (77.03% for SVM and 75.46% for NN), followed by the high codification (72.17% for SVM and 69.65% for NN) and in the last place we can see that medium showed the worst results (62.88% for SVM and 60.55% for NN).

Table 4.2 shows the results for all the experiments conducted in this part of the research, data was divided into Neural Network results and Support Vector Machine results and results about the accuracy for each machine and the Area Under the Curve (AUC) value. The same table includes all the parameters previously mentioned and a classification depending on the output code used for those experiments.

In addition, figure 4.5 shows a box plot applied to these results for Neural Networks; similarly, figure 4.6 shows a box plot with the results for Support Vector Machines.

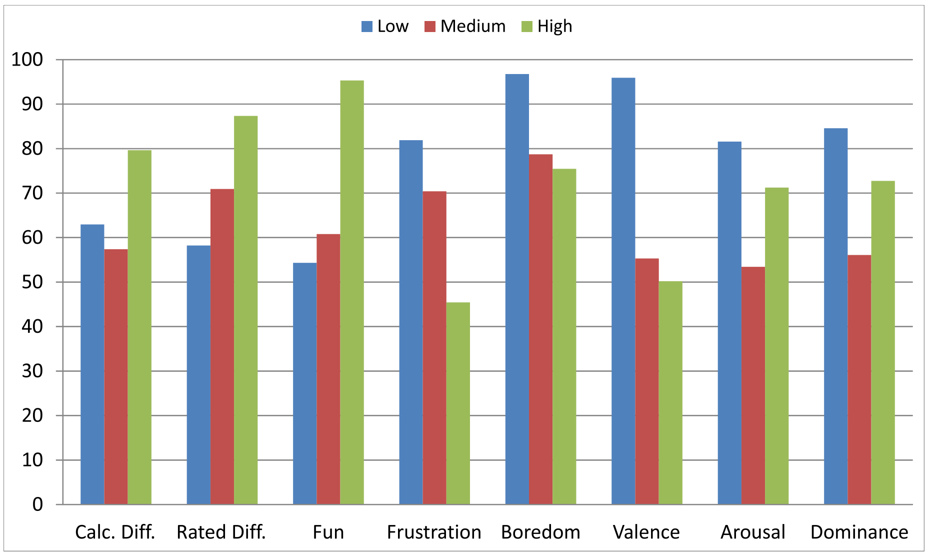


Figure 4.3: Accuracy results Using Support Vector Machines

4.5.2 Output Codification

The importance of understanding which output codification works best, gives us a hint about what would be better to use when designing a general method to predict these parameters' status. Considering that none of the evaluated methods is fully accurate in their predictions, it would be useful to understand what setup could contribute to achieve better results.

Another relevant conclusion from this specific part of the results analysis is that we could use different output configurations for different parameters, for instance, calculated difficulty and frustration work best with a high configuration, in contrast, the rest of the parameters would predict with higher accuracy using the low configuration.

4.5.3 How to Use these Parameters

Supported by the results presented in this paper, we can estimate that the best setup for predicting game experience related parameters and emotions would be using different neural networks as classification method and, depending on the parameter, use a specific output configuration. One possible approach would be to choose the highest accuracy network for each parameter and use it as is; a different approach could be to combine different neural networks to corroborate the results and use them. These are only our estimations, this needs to be tested with new experiments and new proposals.

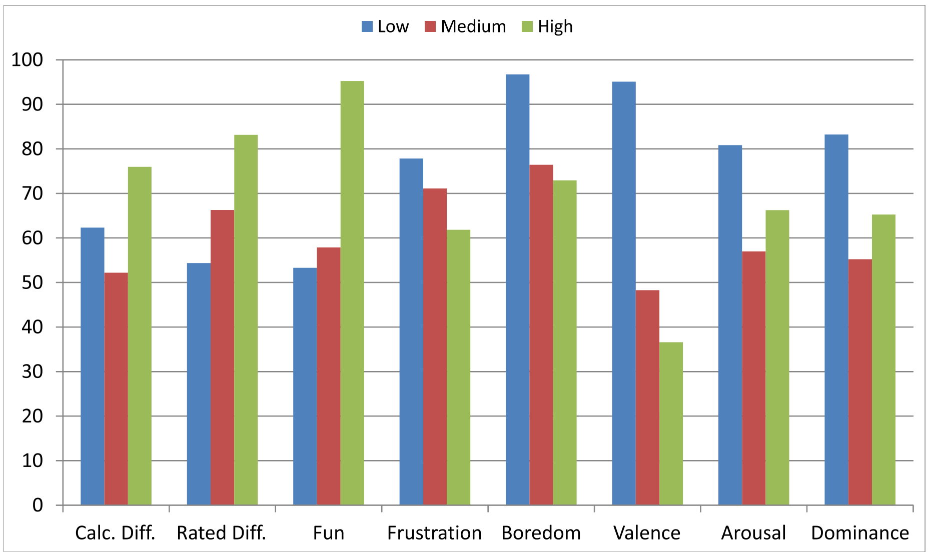


Figure 4.4: Accuracy results Using Neural Networks

4.6 Discussion

We tested two machine learning methods: Neural Networks and Support Vector Machines, to design a new model that could predict player's perception and emotions related parameters. In different studies from the previous literature, other researchers have tested the effect of Support Vector Machines and some of these parameters [32]. The novelty of our approach and the way we add up to this previous study, was to include parameters measured with SAM questionnaires and to test Neural Networks.

Our results show that Support Vector Machines (as in the previous work) classified he data with a higher accuracy. However, Neural Networks also performed well. In addition, the results obtained with our model were more accurate than the previous study, demonstrating that this approach could be a better solution for recognizing emotions/player's perception related parameters.

One of the Hypotheses established at the beginning of this document was that emotions can be predicted using behavioral patterns. We have demonstrated that it is possible to do it with an accuracy higher than 70.69% as an overall results for the analyzed parameters. This also corroborates that there is a relationship between players' perception of the game and the way they behave.

We propose the models presented in this research to classify emotions and player related parameters, using pressure sensitivity. Combining this method in order to decide how to change the content of a game according to how the player feels at a determined time, would help developers adapt games for a better experience.

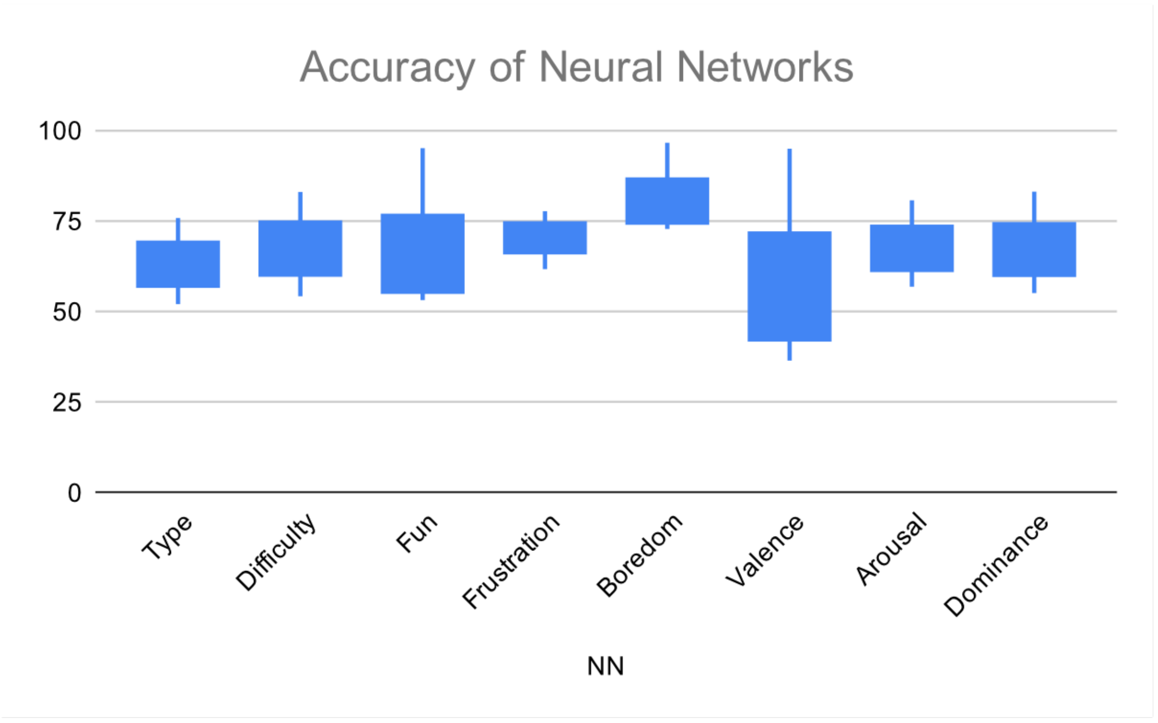


Figure 4.5: All Results for Neural Networks (All Codes)

Concretely, we believe that a combination of different Support Vector Machines and possibly a Neural Network, could be a support to decide when to make the game more difficult or easier, depending on how that player is feeling. This could also encourage players to rest when they probably feel to excited or they are not having fun with the game.

With this results and our proposal, we have shown a way to connect behavioral patterns and player's emotions, which is one of the objectives of this study.

For improving this research, we consider that would be assertive to conduct more experiments with different parameters for the machine learning methods we tested. In addition, it would be good to try other types of classification methods and compare with the results presented here.

4.7 Chapter Conclusions

Understanding how the players' perception of game related parameters and emotions is related to pressure sensitivity, would help developers to use that knowledge to create more appropriate experiences. We presented the results of the second step of that general approach: finding patterns between pressure and player-related parameters. Data from 20 participants was analyzed through machine learning methods.

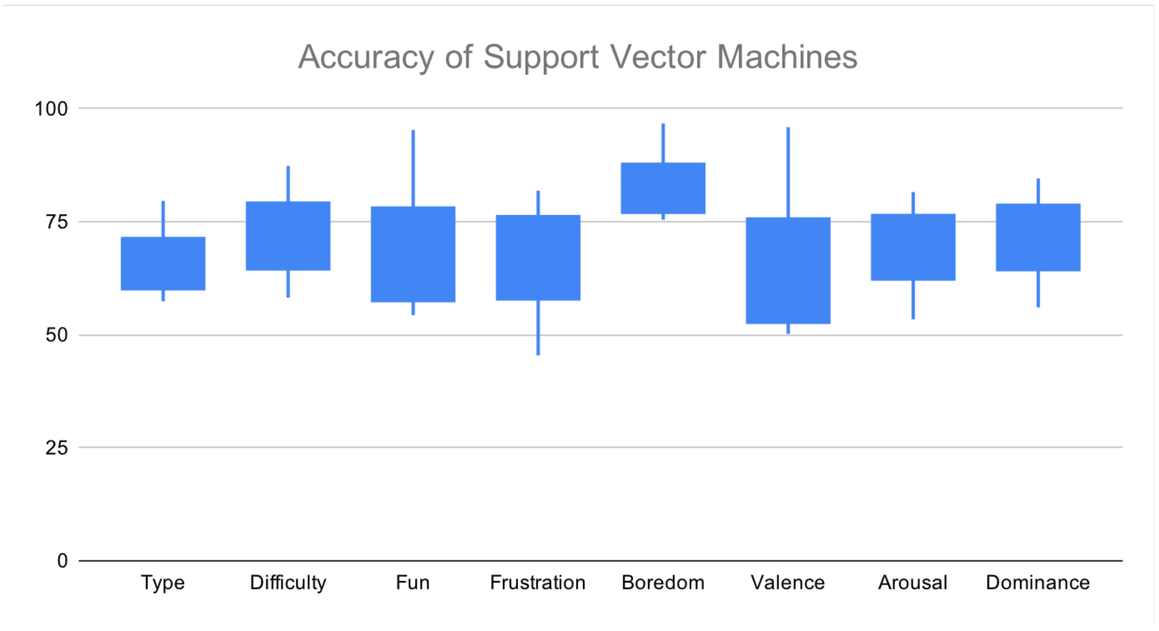


Figure 4.6: All Results for Support Vector Machines (All Codes)

Support vector machines demonstrated to be a better method to classify this type of data according to our results, neural networks average prediction accuracy was 68.55% and support vector machines showed 70.69%.

Despite the fact that the analysis for both research was different, our average best result using neural networks, are better than a previous research [32] which involved the use of support vector machines to classify boredom, engagement and fun, achieving an accuracy of 53.44%. Our results for those three parameters, were higher. It would be meaningful to analyze these three parameters using neural networks in the way that the previous research's study was conducted and this might contribute to a better understanding of this data.

Results show that boredom was the parameter with the highest accuracy (83.64%), which means that with that percentage of accuracy, it would be possible to determine when the player is bored using pressure sensitivity collected while playing. This result can be used by game developers to include more exciting challenges or content when determining that the player is feeling bored about what they are playing, this could also mean that the game should stop at this point to give the player some time to relax or recover the pace. In addition, frustration, fun, difficulty and arousal were predicted with more than 70% of accuracy.

Analyzing all the output configurations and their results, we can say that depending on each parameter, the results vary. Calculated difficulty and boredom showed

Table 4.2: All Results for Neural Networks and Support Vector Machines (All Codes)

Method	Fea	Typ	Dif	Fun	Fru	Bor	Val	Aro	Dom	Code
NN	Acc	75.95	83.14	95.23	61.84	72.93	36.59	66.24	65.27	HIG
	AUC	0.61	0.68	0.68	0.58	0.5	0.62	0.55	0.66	
SVM	Acc	79.64	87.35	95.31	45.45	75.47	50.19	71.23	72.73	
	AUC	0.5	0.65	0.67	0.5	0.5	0.56	0.5	0.64	
NN	Acc	62.32	54.35	53.29	77.83	96.73	95.08	80.84	83.23	LOW
	AUC	0.65	0.67	0.57	0.73	0.59	0.67	0.71	0.57	
SVM	Acc	62.96	58.22	54.34	81.89	96.75	95.93	81.57	84.58	
	AUC	0.6	0.5	0.5	0.7	0.5	0.5	0.65	0.5	
NN	Acc	52.19	66.28	57.89	71.13	76.43	48.26	56.99	55.23	MED
	AUC	0.62	0.61	0.65	0.64	0.6	0.63	0.68	0.65	
SVM	Acc	57.4	70.93	60.78	70.4	78.71	55.32	53.44	56.08	
	AUC	0.56	0.54	0.6	0.63	0.5	0.57	0.63	0.63	

their best results with a high output codification; the rest of the parameters showed their best results with the low configuration. It's easier to classify the extreme values of each parameters, which could be helpful for developers.

Chapter 5

Rhythm Group Theory and EEG

5.1 Introduction

Following the same approach of previous chapters, this part of the research was conducted with the guidance of professor Koji Mikami (supervisor) and professor Kunio Kondo, who supported the author from the phase of planning and design to implementation and analysis.

As mentioned in the previous section, Dynamic Difficulty Adjustment is one suitable solution for avoiding unbalanced difficulty when designed well and it has proved to be successful in the past [45]. However, the experience of players can be ruined by a poor implementation when they realize that difficulty is being adjusted deliberately [46]. In order to avoid this issue, we focused on finding a complementary component that could support traditional difficulty adjustment, adding variety and better results.

The level of challenge in video games is one of the most relevant aspects that affect the player experience, however it's not the only one [47]. Immersion plays a very important role determining how the player experience is shaped in general [48] and the levels of attention of players while playing a game influence how immersion fluctuates through time [49].

The word immersion in the video games field is used in a symbolic way for explaining the experience of feeling surrounded by a different reality as if players were submerged in water, it's a way to refer to the sensation of a deep feeling, having all our senses focused on a specific reality, the virtual reality [50].

Player experience is defined as the relationship between the player and the game, the influence that causes the game on the player while playing and the reactions triggered by that interaction [51]. The proper balance between frustration, challenge, and immersion, transforms into a good player experience [52].

Prompted by the relationship between immersion and player experience, we decided to include an Electroencephalography (EEG) component to measure attention values. This adds variety to the adjustment, making it less predictable for players.

For the automatic creation of levels we used Rhythm-Group theory [53], a successful Procedural Content Generation method to construct levels with a sense of rhythm for the player.

This section presents the results and analysis of the combination of these two components to adapt the difficulty for a 2D platformer and an attempt to improve the player's experience.

5.2 Related Work

For the particular case of this part of the research, one of the most relevant studies is the work of Martin Jennings-Teats et. al [19] which propose an approach similar to ours; using DDA and Procedural Content Generation, they created a personalized and structural experience for players. Our original contribution is the inclusion of the EEG component to the previous approach.

In recent years there have been several of studies related to BCI and games, mobile games [54] showing how useful these devices could be working together with mobile devices; PC-based FPS games [55] focused on the player experience and interaction; and even for conducting research related to how players learn to play [56], etc. These devices, specifically EEG-based devices, have also proved to be useful for making adaptive games [57], which is particularly our field of study.

EEG biosensors, are suitable to be used for cognitive games [58]; in multiplayer cooperative games [59], for evaluating how the cognitive activity changes while playing with teammates; also in serious games, in order to promote rehabilitation with patients that suffer from motor deficits due to a stroke, these researchers developed a new game that aims to help them with the process of rehabilitation [60].

Neurosky Mindwave Mobile device [61], which has shown positive results in reliability and commercial contexts, was used to capture the attention data from players [62]. In addition, a research particularly focused on video games has shown that the device accurately reads values of attention from players [63].

Previous researchers used a similar approach combining DDA and BCI to improve performance while doing a specific task [64]. Our research differs from theirs in different aspects: we use EEG and they use fNIRS; our field of test was games, theirs was general tasks' performance; results show that performance was improved

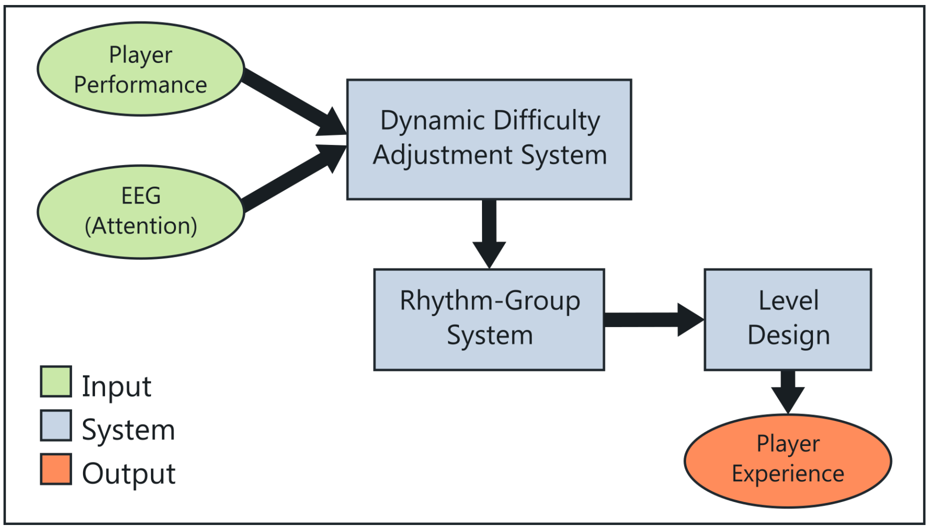


Figure 5.1: General design of our approach

by detecting boredom, in our case, we didn't only improve performance but also adapted the difficulty depending on the players' skills to achieve a more suitable experience.

Procedural Content Generation (PCG) methods are a good alternative to automatically modify the level design according to how players play in a game [65].

In previous research the effectiveness of PCG was demonstrated using games [66]. Georgios N. Yannakakis and Julian Togelius introduced a framework for procedural content generation applied with computational models of user experience, they created a method for developers to trigger specific experiences depending on the user decisions or status inside the game.

5.3 Method

Our method consists of the combination of three different elements: Dynamic Difficulty Adjustment, Rhythm-Group Theory and Brain Computer Interface. A general flow of our approach can be seen in figure 5.1.

5.3.1 Dynamic Difficulty Adjustment

Dynamic Difficulty Adjustment (DDA), is the process of changing game elements automatically in real-time, based on the player's performance, in order to adapt the game to each player and avoid frustration or boredom [67].

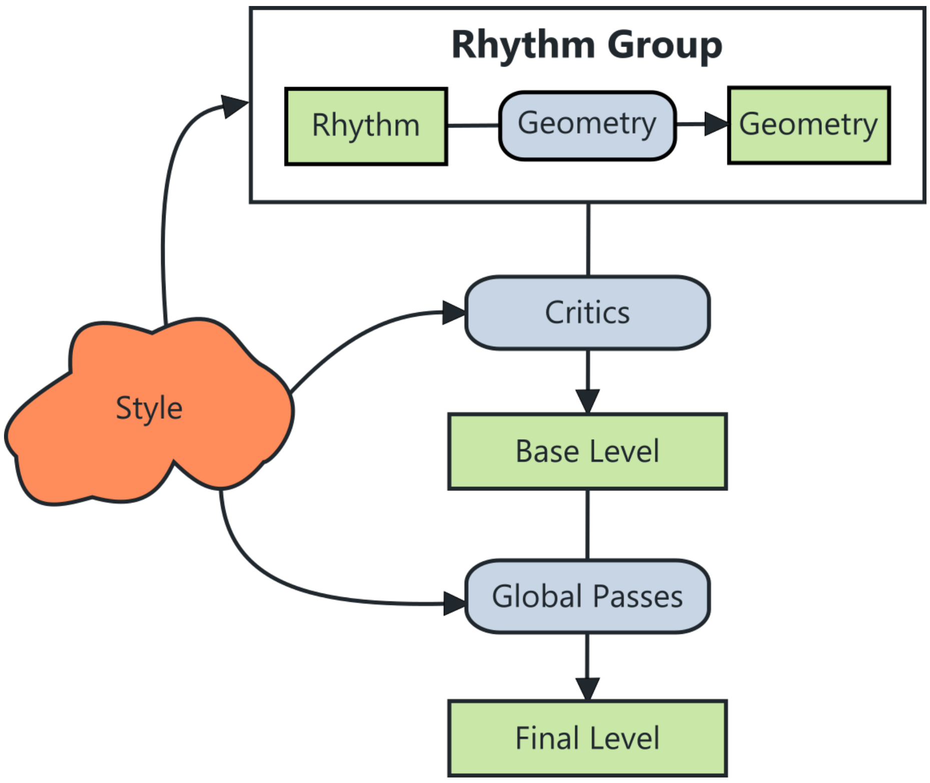


Figure 5.2: Level generation algorithm

We calculate the difficulty using the numbers of threats present in a level, specifically the number and width of gaps, number of enemies in each platform and the type of these enemies (beatable or unbeatable).

Difficulty is adapted according to performance and attention levels calculated by the EEG device while playing.

5.3.2 Rhythm-Group Theory

A Rhythm-based method for 2D platform games is a type of technique for automatic level creation in which rhythm is what the player feels with his hands while playing [53]. This method is a tool for the developer to create levels with a sense of rhythm, levels that are playable and have a natural feeling for players when it comes to level shape and experience. We are using only a part of the method created by Gillian Smith et al., here we describe the parts of the method used for our research, to read the details about the original method, see [53].

Figure 5.2 (taken from the original source) shows the overall method's flow: the creation of a rhythm using a Rhythm System; a set of physical constraints are applied to ensure that the level is well formed (it can be played and completed by the player);

the result is put in a Geometry System that translates all the elements of the rhythm into physical elements (platforms, enemies, etc); finally the level is created.

5.3.3 Brain Computer Interface

Brain Computer Interface or BCI [68] systems are based on obtaining electroencephalogram (EEG) data, extracting relevant information to translate it into commands to be read by software or particular applications [69].

For input extraction we are using *Neurosky Mindwave Mobile*, a bioensor that digitizes brain data into concise inputs for developers to be able to interact with it in real time, using a set of pre-designed algorithms API to monitor brain activity.

For this research we use the eSense value *Attention*, which is a value between 0 and 100 (being 0 the least focused and 100 the most focused) that describes the levels of attention of the user in real time.

5.4 Implementation

The game is a side-scrolling 2D platformer in which the player has to reach a goal placed on the right-most part of the level, very similar to *Super Mario Bros* [70]. The player is required to overcome simple challenges: gaps between platforms, beatable enemies and unbeatable enemies, both types of enemies static. In the beginning of each level players get two bars of health, once the player loses both, has start from the beginning of the level, in addition, the game time is shown on top-center of the screen.

5.4.1 EEG Data

According to the documentation [61], values from 80 to 100 are considered *elevated*; values from 60 to 80 are considered *slightly elevated*; values from 40 to 60 is *neutral*; values from 20 to 40 *reduced* and values from 1 to 20 *strongly lowered.*

By default, the device outputs data once a second, it means, we get as many values as the time the player plays a level. We calculate the average of all values obtained in a level, the calculation is shown in equation 5.1.

If a player is concentrated in a task, would perform better than if concentration levels were poor [71]. Considering this, when attention values are high, it means the player is concentrated in the game, it also means should perform better so we decided to add a higher level of challenge when the player is focused, in contrast, if the player is not focused, we reduce the level of difficulty.

$$A = \frac{1}{n}\sum_{i=0}^{n} a_i \tag{5.1}$$

Arithmetic Mean (A): a_i is the attention calculated by the device per second and n is the number of seconds that the player takes to finish a level. The interval time for this equation depends on how long each player takes to complete a level.

5.4.2 Player's Performance

To calculate the player's performance we considered two parameters: number of deaths or hits by an enemy and gameplay time (time from start to end). Low values for number of hits and play time result in a high performance, on the opposite case, high values for these parameters result in low performance.

$$P = \frac{1}{1+h+d}X_1 + \frac{g}{e}X_2 \tag{5.2}$$

Performance (P): d is the number of deaths; h is the number of times hit by an enemy; g is the gameplay time; e is the expected completion time and X_1, X_2 are weights that represent the influence of each term in the final calculation.

The term $\frac{1}{1+h+d}$ is calculated by cross-multiplication, the number of times hit or deaths is inversely proportional to how well players are playing, we sum 1 to avoid division by zero. The term $\frac{g}{e}$ is calculated by the same cross-multiplication principle, when players take more time to complete a level, performance decreases.

5.4.3 Dynamic Difficulty Adjustment

We used the attention value calculated in equation 5.1 and the player's performance value calculated in equation 5.2 and combined them in equation 5.3 to get a general a global value that involves both parameters. These two values can complement each other and affect the final calculation. Weight for this equation were both set to 0.5, same amount of influence for both parameters, attention and performance.

$$G = PW_1 + AW_2 \tag{5.3}$$

Combination of performance and attention (G): P is the performance calculated in equation 5.2; A is the attention average calculated in equation 5.1 and W_1, W_2 are weights that represent the influence of each term on the final calculation.

$$A_t = \frac{1}{n_t} \sum_{i=0}^{n_t} a_{ti} \tag{5.4}$$

Arithmetic Mean (A_t) for specific elements of type t: a_{ti} is the attention obtained from the device when the player interacts with elements of type t ; n_t is the number of times the player interacts with elements of type t and t = [low enemy, medium enemy, high enemy, gap]. The interval time for this equation depends on how many times the player interacts with elements of type t.

$$P_t = \frac{1}{1 + (h_t + d_t)} \tag{5.5}$$

Performance per type (P_t): h_t is the number of times the player has been hit by enemies of type t; d_t is the number of times a player has died due to enemies of type t; t = [low enemy, medium enemy, high enemy, gap].

The term $\frac{1}{1+(h_t+d_t)}$ is calculated by cross-multiplication, the number of times hit or deaths is inversely proportional to how well players are playing, we sum 1 to avoid division by zero.

Equation 5.6 shows how to calculate the global value to decide how many elements of each kind are included in the level. Equation 5.5 shows the performance calculation for a particular type of element and the calculation for attention values of a particular type of element is the same as equation 5.1 but instead of using all values, we calculated how attention values behaved when the player interacted with elements of that kind.

For example, *low jump* is one of the element types; to decide how many elements (percentage of occurrence) we assigned to *low jump* actions, when the player dies due to an element of type *low jump*, we reduce the number of *low jump* type elements in the next level, it means, we are trying to add elements that increase the player's performance. In the case of attention, if the calculated attention was registered high for *low jump* type elements, we increase the number of this type of elements to increase get better results in the next level.

There is a compensation between both values, performance and attention, that work together to calculate the final percentage of occurrence of each element, it really adds variation to the gameplay and the experience in general.

$$G_t = P_t Z_1 + A_t Z_2 \tag{5.6}$$

Combination of performance and attention for elements of type t (G_t): P_t is the performance calculated in equation 5.5; A_t is the attention average calculated in equation 5.4 and Z_1, Z_2 are weights that represent the influence of each term on the final calculation.

5.4.4 Rhythm-Group System

A set of parameters are important to take into consideration to construct a rhythm, these are: rhythm type, rhythm density, action types, number of actions. For our research we used the following values:

- Action Type: we chose the simplest ones run and jump. For the action jump, there are three different types: low, medium and high
- Rhythm Type: we are using a regular type of rhythm which means that actions are evenly distributed in the rhythm, other types of actions are random or swing.
- Rhythm Density: number of actions in a rhythm, we choose this depending on how the player results are, the better the result, the higher the density. The minimum value is 5 actions and maximum value is 50 actions.
- Rhythm Length: this is how long the gameplay time should be according to the level length (horizontally), this is decided depending on the player results, the better the results, the longer the level. The minimum value is 5 seconds, the maximum value is 30 seconds.

Figure 5.3 shows an example of rhythms that our system can create.

To calculate the number of elements for each rhythm's value, we defined a difficulty function that is shown in equation 5.7.

$$D_1(p_0, p_1) = \begin{cases} D_0 + SG & \text{if } p_1 \geq p_0, \\ D_0 - SG & \text{if } p_1 < p_0 \end{cases} \tag{5.7}$$

Difficulty ($D_1(p_0, p_1)$): p_0 is the performance calculated in the previous level with equation 5.5; p_1 is the performance calculated in the current level with equation 5.5; D_0 is the difficulty of the previous level; S is a constant value to make the change between level and level smoother; G is calculated with equation 5.3.

The difficulty for the current level is calculated using the difficulty of the previous level plus a variation of the global value. This variation can take positive or negative

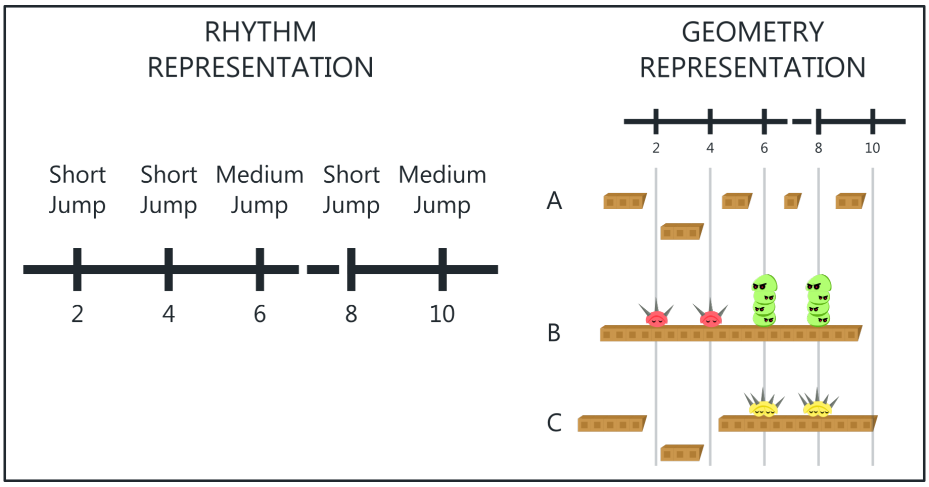


Figure 5.3: Rhythm Representation

values depending on whether we make the level more difficult or easier. The constant S represents a STEP value defined to make smooth changes of difficulty between levels so players do not feel an abrupt change.

For our implementation, the value S was set to 0.125 (calculated empirically after testing with other values). The value v is 1 if the difference between the performance for the current level and the previous level is positive and -1 if the difference is negative.

$$E_1 = E_0 + D_1 M \tag{5.8}$$

Rhythm density (E_1): E_0 is the rhythm density calculated in the previous level; D_1 is the difficulty calculated with equation 5.7; M is the maximum value for rhythm density.

$$L_1 = L_0 + D_1 N \tag{5.9}$$

Rhythm length (L_1): L_0 is the rhythm length calculated in the previous level; D_1 is the difficulty calculated with equation 5.7; N is the maximum value for rhythm length.

The rhythm density and rhythm length are calculated using equation 5.8 and 5.9 respectively. Density and Length are directly proportional to difficulty.

We simplified the elements that could be built by our system to the simplest elements in platformers, there are no special items or moving enemies, only the basic features to show the rhythm-group method working with the rest of our system.

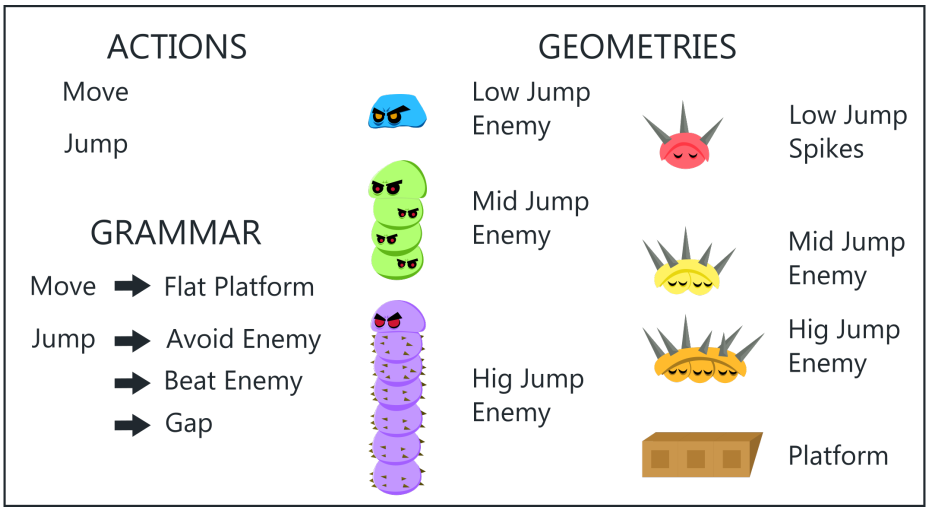


Figure 5.4: Rhythm System Elements

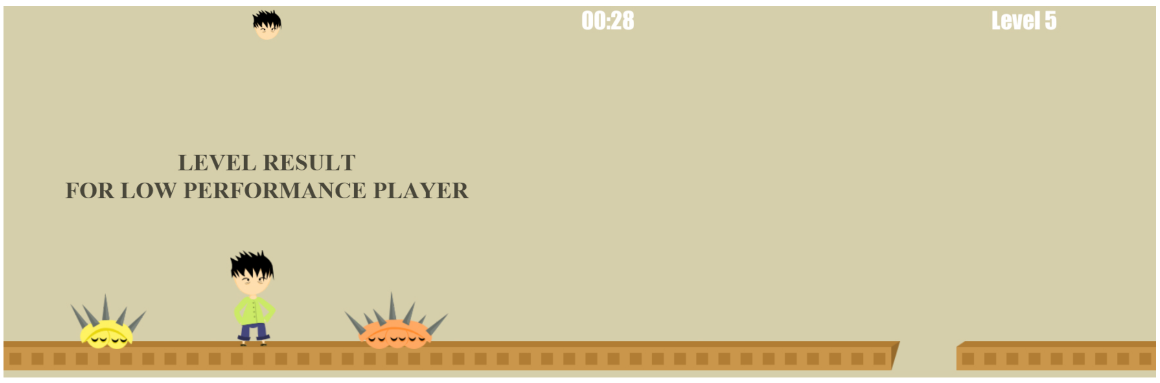


Figure 5.5: Low performance player: Few challenges, easier to complete.

We selected three types of challenges for each jump type, it means, we have three types for low jump, three for medium jump and three types for high jump, in total nine different elements. The first element is a gap, a separation between platforms, if the player falls through a gap, dies; the second element is a spike, the player dies when touching a spike; finally an enemy that can be beaten by the player jumping on its head. Figure 5.4 shows the geometry elements that the current geometry system can create.

Figure 5.5 shows a piece of the level result for a low performance player, it's an easy level with not so many enemies or gaps; on the other hand, we can see the result for a challenging level in figure 5.6 which is different from the low performance result, with more enemies, gaps and challenges.

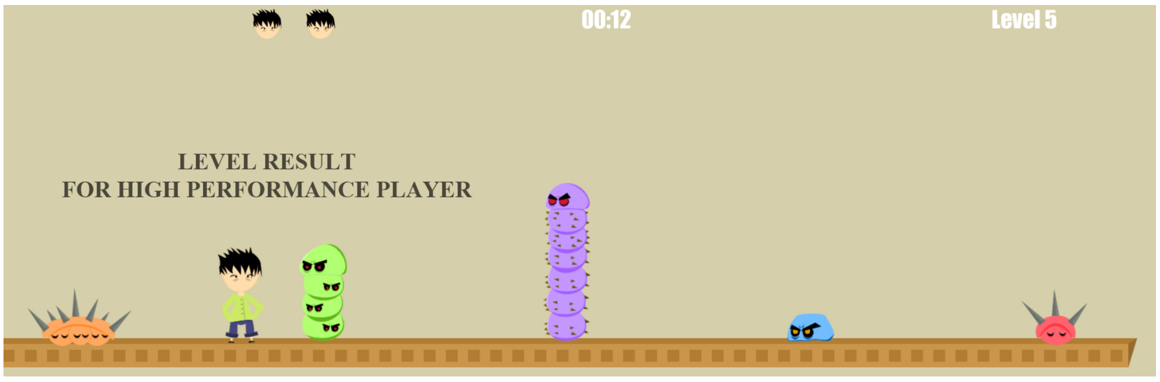


Figure 5.6: High performance player: More challenges, more difficult to complete.

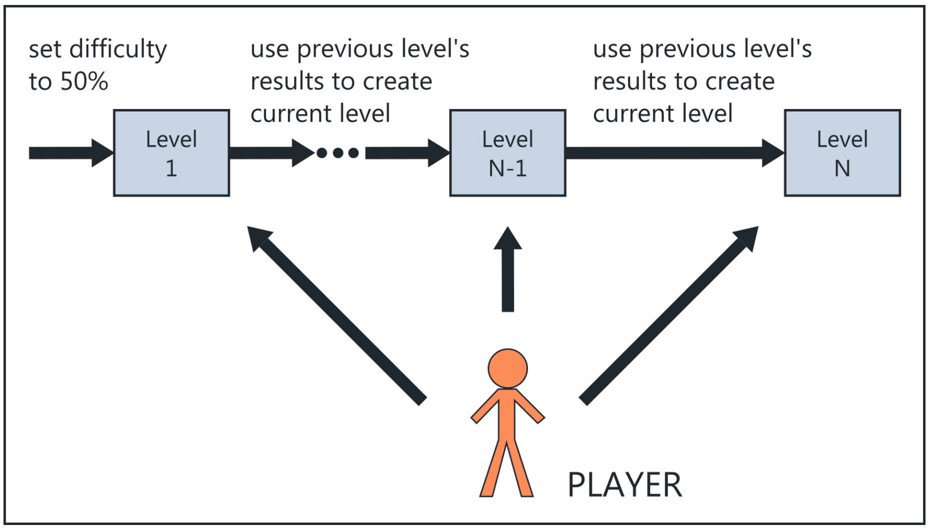

set difficulty to 50%
use previous level's results to create current level
use previous level's results to create current level
Level 1
Level N-1
Level N
PLAYER

Figure 5.7: Experiments Process

5.5 Experiments & Results

Players were asked to complete five levels created automatically by our method and they wore the biosensor while playing to record their brain activity and adapt the difficulty of each level. Figure 5.7 shows the process.

5.5.1 Players and Environment

25 people between 21 and 30 years old, 7 (28%) women and 18 (72%) men completed the experiment. Before starting, players were asked: "Have you played Super Mario Bros before?" (A) and "Have you completed Super Mario Bros?" (B).

For question **A**, only 3 people (12%) answered *no*, the rest (22 people, 88%) answered *yes*; for question **B**, 9 people (36%) said *yes*, the rest (16 people, 64%) answered *no*.

The game was played on a 15inch widescreen monitor of a Dell XPS LX502 laptop, at approximately 60cm from the player, using an *Xbox360 gamepad* , with the left thumbstick to move and A button to jump. Sound was played using the laptop's speakers at a volume of 25%.

5.5.2 General Features

For the first level that players played, we set the difficulty to 50%. Depending on the results of the first level the second level was created to adapt it to the player's results. Same process is repeated until players completed five levels.

5.5.3 Limitations

We faced some issues with the biosensor when performing the experiments. Sometimes the connection between the device and the computer (bluetooth) was not strong enough to establish a successful connection.

Besides connection issues, we found that after players wear the device for more than 10 min., they didn't feel comfortable, which lead us to decided to keep the experiments short.

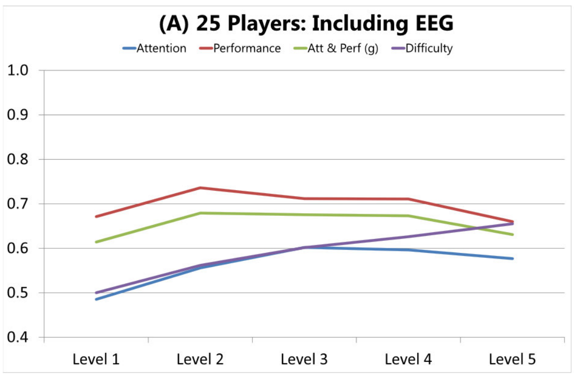
For some players the device just didn't fit well, regardless its adjustable functionality, sometimes is not easy to fit comfortably in all types of players.

The result of those players that experienced problems with the connection while playing, nuisance or any kind of bad experience that could affect the results were excluded from the experiment.

In addition to Neurosky related limitations, we consider that this method would be suitable for games that involve jumping as their main mechanic and games that involve some kind of rhythm in their core mechanics. Runners, 2D or 3D, sidescroller platformers and rhythm games are among the types of games we consider this would be a good method for.

5.5.4 Analysis

The overall results for all players across time are shown in figure 5.8 (A). The graph shows the average results for all 25 players, attention, performance, global value (attention & performance) and difficulty.



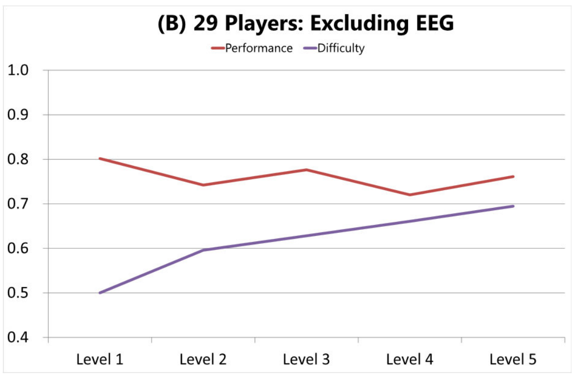


Figure 5.8: Experiments results: Including EEG (A), Excluding EEG (B)

5.5.4.1 General Results

Comparing the behavior of the global value (green curve) and the difficulty (purple curve) from level to level, we can see how in the end of the experiment both curves get close to each other, an increasing difficulty higher than the global value. This means that the method is matching the difficulty to the player's skills.

In addition there is a balance between attention (blue curve) and performance (red curve), the method combines both values and makes sure that both of them contribute with the final calculation. For example we can see the result from level 2 to level 3, performance decreases and the attention value increases, the result (green curve) is a stable value, which turns into an increase of difficulty, keeping the pace and attention for players.

Difficulty increases, comparing level 1 and level 5, showing that the level of challenge is changing, from 0.5 to 0.66, an overall increase of the difficulty until gets closer to the global value, we estimate that difficulty in upcoming levels would decrease a little and then raise along with the global value. We have to test with more levels to confirm this.

With the challenge increasing, we can also see how the global value changes, in the end of the experiment it ends up slightly higher than the beginning, meaning that players perform better with a higher difficulty.

If we performed experiments including more levels, we would expect that curves vary together, specially the global value and difficulty, the expectation is to keep increasing smoothly and both of them close to each other, demonstrating the adaptation process. We still have to make more experiments to confirm this.

All the results for the experiment conducted using EEG can be seen in table 5.1, these results include all players (25) and the data from level 1 to level 5. Figure 5.9

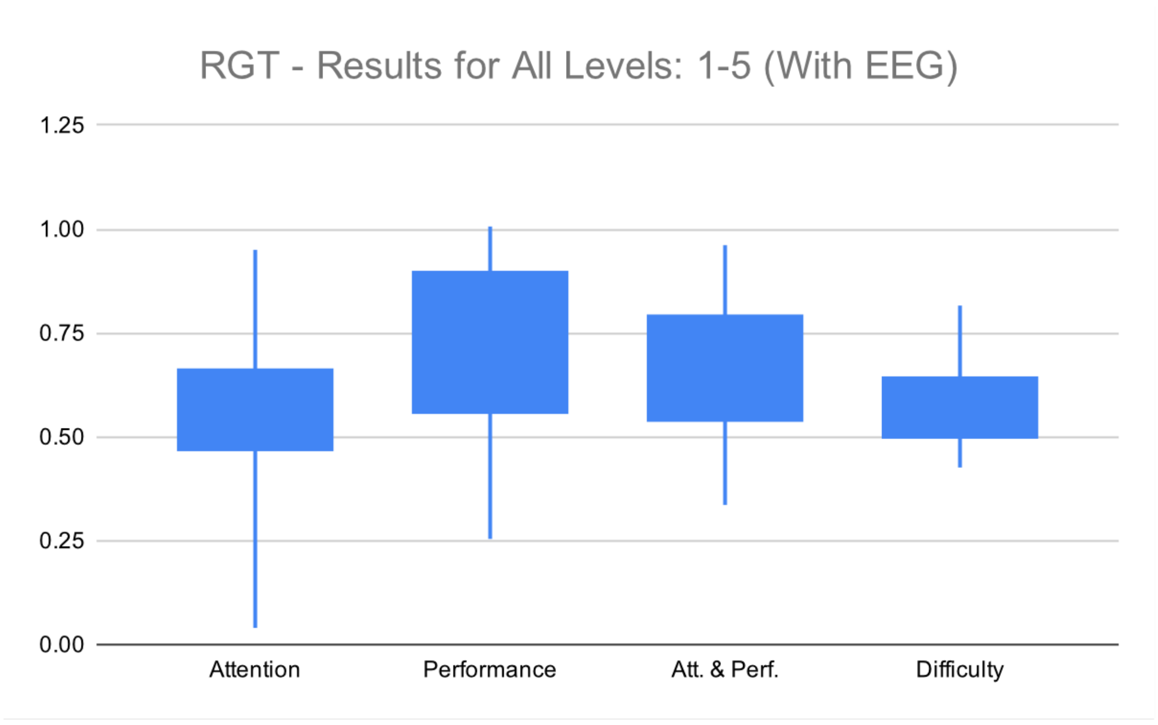


Figure 5.9: Rhythm Group Theory Results: All Results (With EEG)

shows a box plot applied to these results.

In addition, table 5.2 shows all the results for the experiment conducted without EEG, this includes data from 29 players and from levels 1 to 5. Figure 5.10 shows a box plot applied to these results.

Table 5.1: Rhythm Group Theory Results. A: Attention, P: Performance, C: Attention & Performace, D: Difficulty

Pla	Level 1				Level 2				Level 3				Level 4				Level 5			
	A.	P.	C.	D.	A.	P.	C.	D.	A.	P.	C.	D.	A.	P.	C.	D.	A.	P.	C.	D.
1	0.30	0.98	0.78	0.50	0.14	0.72	0.54	0.60	0.29	1.01	0.79	0.54	0.81	0.59	0.65	0.64	0.40	0.72	0.62	0.59
2	0.43	0.57	0.53	0.50	0.61	0.58	0.58	0.57	0.60	0.31	0.39	0.64	0.58	0.63	0.61	0.56	0.46	0.50	0.48	0.64
3	0.80	0.64	0.69	0.50	0.54	0.90	0.79	0.59	0.73	0.59	0.63	0.68	0.65	0.94	0.86	0.64	0.61	0.52	0.54	0.75
4	0.56	0.51	0.52	0.50	0.77	0.81	0.81	0.57	0.66	0.89	0.82	0.67	0.61	0.39	0.45	0.77	0.64	0.85	0.79	0.70
5	0.74	0.43	0.52	0.50	0.63	0.60	0.61	0.56	0.70	0.87	0.82	0.64	0.71	0.59	0.62	0.74	0.69	0.64	0.65	0.70
6	0.47	0.47	0.46	0.50	0.58	0.92	0.82	0.43	0.65	0.42	0.48	0.54	0.51	0.65	0.60	0.47	0.48	0.71	0.64	0.55
7	0.66	0.46	0.52	0.50	0.43	0.39	0.39	0.57	0.61	0.88	0.80	0.49	0.57	0.47	0.49	0.59	0.58	0.87	0.79	0.53
8	0.90	0.57	0.66	0.50	0.82	0.59	0.65	0.58	0.89	0.99	0.96	0.54	0.92	0.73	0.78	0.66	0.95	0.70	0.77	0.63
9	0.49	0.99	0.84	0.50	0.73	0.99	0.91	0.60	0.63	0.98	0.87	0.72	0.67	1.01	0.91	0.70	0.81	0.58	0.64	0.82
10	0.38	0.85	0.71	0.50	0.61	0.66	0.64	0.59	0.58	0.68	0.65	0.54	0.58	0.96	0.85	0.62	0.56	0.61	0.59	0.73
11	0.08	0.88	0.64	0.50	0.61	0.99	0.88	0.58	0.53	0.60	0.58	0.69	0.45	0.66	0.59	0.64	0.55	0.61	0.59	0.71
12	0.48	0.98	0.83	0.50	0.59	0.67	0.64	0.60	0.71	0.97	0.89	0.56	0.56	0.59	0.57	0.67	0.67	0.94	0.86	0.62
13	0.47	0.91	0.78	0.50	0.37	0.67	0.58	0.60	0.80	0.65	0.69	0.54	0.65	0.96	0.87	0.63	0.37	0.62	0.54	0.74
14	0.04	0.66	0.47	0.50	0.23	0.96	0.74	0.43	0.31	0.99	0.78	0.53	0.35	0.66	0.56	0.62	0.43	0.99	0.82	0.57
15	0.44	0.41	0.41	0.50	0.47	0.96	0.81	0.43	0.31	0.63	0.53	0.53	0.40	0.65	0.57	0.47	0.36	0.68	0.58	0.54
16	0.50	0.58	0.55	0.50	0.79	0.49	0.57	0.57	0.67	0.56	0.58	0.64	0.73	0.45	0.52	0.71	0.90	0.56	0.65	0.65
17	0.68	0.57	0.60	0.50	0.56	0.57	0.56	0.58	0.54	0.52	0.52	0.52	0.54	0.60	0.58	0.46	0.47	0.49	0.48	0.53
18	0.65	0.77	0.74	0.50	0.49	0.84	0.74	0.59	0.76	0.98	0.92	0.56	0.61	0.95	0.85	0.67	0.55	0.96	0.84	0.66
19	0.72	0.54	0.59	0.50	0.57	0.94	0.83	0.57	0.57	0.54	0.54	0.68	0.54	0.93	0.82	0.62	0.68	0.64	0.65	0.72
20	0.43	0.57	0.53	0.50	0.70	0.54	0.58	0.57	0.87	0.69	0.74	0.64	0.87	0.57	0.66	0.73	0.85	0.60	0.67	0.69
21	0.41	0.56	0.51	0.50	0.64	0.66	0.65	0.56	0.48	0.46	0.46	0.65	0.55	0.99	0.86	0.58	0.55	0.25	0.34	0.68
22	0.33	0.91	0.74	0.50	0.33	0.53	0.46	0.59	0.52	0.97	0.83	0.52	0.47	0.50	0.48	0.63	0.54	0.66	0.62	0.56
23	0.41	0.57	0.51	0.50	0.43	0.93	0.78	0.56	0.55	0.66	0.62	0.66	0.47	0.92	0.79	0.61	0.41	0.48	0.45	0.71
24	0.39	0.56	0.50	0.50	0.64	0.60	0.60	0.56	0.60	0.51	0.53	0.64	0.58	0.96	0.85	0.58	0.54	0.86	0.77	0.69
25	0.37	0.85	0.71	0.50	0.62	0.88	0.80	0.59	0.49	0.44	0.45	0.69	0.53	0.43	0.45	0.62	0.37	0.44	0.41	0.68

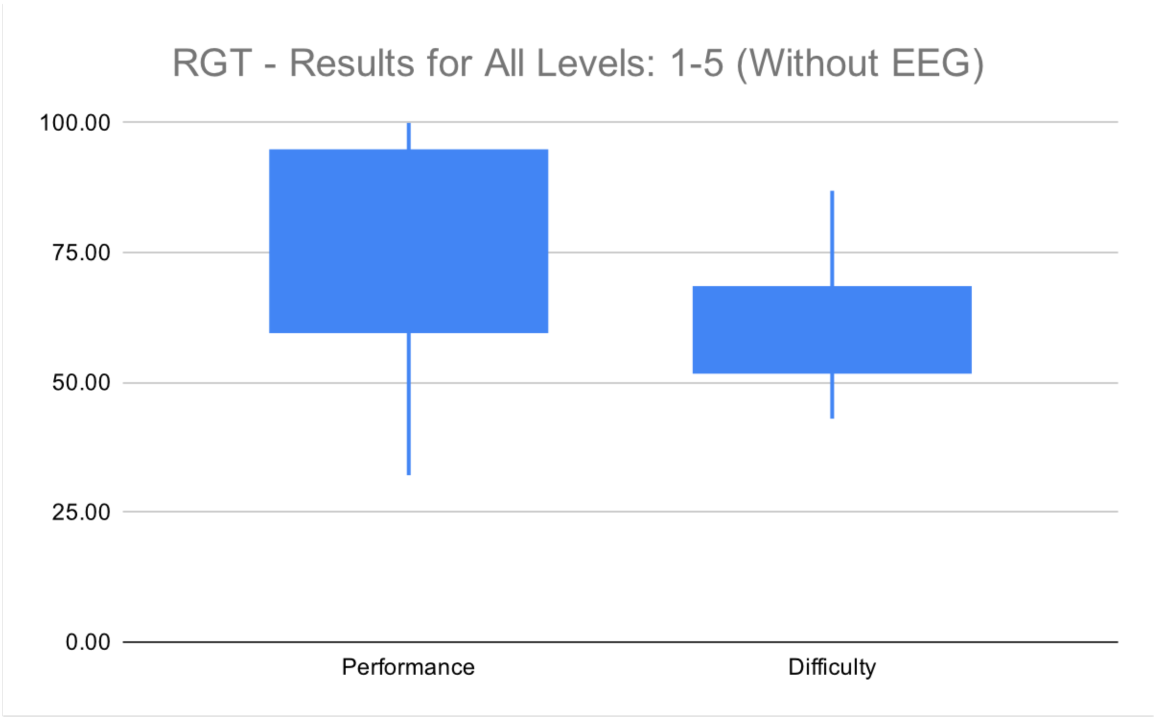


Figure 5.10: Rhythm Group Theory Results: All Results (Without EEG)

5.5.4.2 Player Groups

In addition to the general results of our experiments, we separated and classified players' results by experience. The main reason to do this is that these players have features in common and it makes it easier and more meaningful when analyzing the results.

Figure 5.11 shows the results for all grouped features. We know that the number of players from graph B is low however, we can see how the behavior for both types of groups (A and B) in the end is similar, curves get closer in the end, which means that for both types of players the algorithm is adapting the difficulty.

One of the interesting things about this particular group is we can see how the difficulty becomes lower to match players' results and in the end, curves get closer, reducing the gap while playing.

Comparing graphs C and D we can deduce that players from group C are more experienced than players from group D. In fact, by the overall performance of each group (73% for group C and 66% for group D), we can assume this. We can see that for players with different experience, the algorithm is, step by step, adapting to change and offer a suitable experience. Curves for both groups end up with a similar shape.

Table 5.2: Rhythm Group Theory Results (Without EEG)

Pla	Level 1		Level 2		Level 3		Level 4		Level 5	
	Perf.	Diff.	Perf.	Diff.	Perf.	Diff.	Perf.	Diff.	Perf.	Diff.
1	70.00	50.00	57.42	58.75	98.92	53.42	38.50	65.79	71.29	58.10
2	88.29	50.00	52.12	61.04	97.85	55.05	56.89	67.28	58.78	61.89
3	81.29	50.00	33.66	60.16	61.10	51.86	65.90	59.50	82.72	67.77
4	56.70	50.00	32.13	57.09	57.10	48.60	53.25	55.74	55.49	49.90
5	85.35	50.00	95.64	60.67	59.86	72.62	94.49	67.60	47.86	79.41
6	92.92	50.00	53.42	61.62	67.81	55.79	53.56	64.27	94.52	58.46
7	64.95	50.00	51.49	58.12	70.57	52.05	71.73	60.88	71.65	69.84
8	51.59	50.00	93.83	56.45	93.75	68.18	93.82	67.40	62.77	79.12
9	53.65	50.00	71.13	56.71	68.23	65.60	99.37	61.63	69.76	74.05
10	94.61	50.00	72.12	61.83	100.00	58.35	63.73	70.95	72.42	66.41
11	97.78	50.00	99.68	62.22	98.72	74.68	99.12	74.52	94.74	86.91
12	95.38	50.00	57.17	61.92	71.63	56.57	98.24	65.52	65.56	77.80
13	77.75	50.00	92.75	59.72	49.01	71.31	51.59	64.93	52.22	71.39
14	93.23	50.00	69.64	61.65	60.97	57.85	71.44	52.98	70.85	61.91
15	86.02	50.00	97.67	60.75	91.22	72.96	97.87	71.86	50.25	84.10
16	80.23	50.00	49.57	60.02	93.02	53.72	62.37	65.35	69.84	60.64
17	89.87	50.00	87.11	61.23	97.24	59.62	92.98	71.77	65.32	70.90
18	97.46	50.00	97.94	62.18	96.49	74.42	97.32	73.98	90.07	86.15
19	65.38	50.00	86.23	58.17	57.01	68.93	70.12	63.55	96.05	72.30
20	95.09	50.00	98.25	61.89	60.20	74.16	99.23	69.19	98.51	81.60
21	55.57	50.00	68.69	56.94	97.93	65.53	36.21	77.77	68.15	69.80
22	97.78	50.00	96.23	62.22	97.95	61.75	46.35	73.99	99.20	67.29
23	44.23	50.00	54.51	43.02	50.68	49.84	57.05	43.68	63.05	50.81
24	97.78	50.00	63.77	62.22	61.71	57.69	71.69	52.91	98.42	61.86
25	86.34	50.00	55.76	60.79	96.37	55.26	93.33	67.30	66.77	66.48
26	62.73	50	92.5	57.84	62.17	69.4	56.65	64.67	99.59	59.25
27	91.26	50	95.23	61.41	91.66	73.31	55.03	72.26	94.58	66.64
28	84.82	50	85.77	60.6	48.51	71.32	92.47	64.89	89.29	76.45
29	86.87	50	90.89	60.85	93.99	72.22	48.03	83.97	87.67	77.47

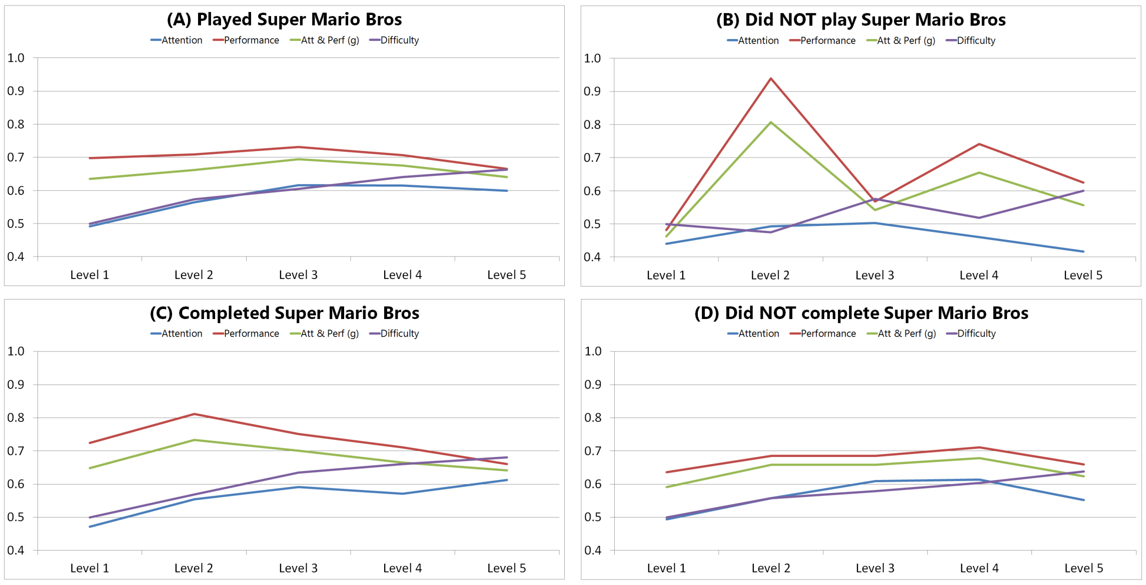


Figure 5.11: Results for Groups of Players

The difficulty for almost all groups adapts at a constant pace. Values of attention, difficulty and performance start at a particular point for each graph and in the end of the experiment they increase, it means players are getting better at the game and also challenge is increasing accordingly.

In general, for different types of players, with different characteristics and different experience we can see how the method is adapting, performance and attention are adjusting values of difficulty, the green and purple curve end close to each other.

We expect that if we perform experiments with more players and levels, the difficulty curve and globals (att. perf.) curve will keep moving at the same pace, ideally increasing with the player's abilities.

5.5.4.3 Players' Feedback

After playing each level of the game, we interviewed few players and collected feedback of their experience while playing. This section shows a short sample of those interviews.

All players said after that the game was fun to play however, they also mentioned that some levels (the easiest ones) are too simple and they would enjoy levels with more elements and challenges.

As a recurring comment from high skilled players, they all agree that difficult levels are more interesting than easy levels, in contrast, low skilled players felt better with easier levels but they also said that a higher level of challenge would be interesting.

When we designed experiments for this research, we implemented a game with the most basic elements from a platformer, to adjust the game and make it fit for the Rhythm-group theory parameters, in addition, graphics and in-game feedback are also very basic. Based on the feedback from players and the results of the experiment we should increase the level of challenge of the game, adding variety and improving the implementation of it.

5.5.4.4 EEG Influence

In order to validate the results obtained from our approach of combining player's performance and EEG to adapt the difficulty and to be able to prove the value and novelty of including this new component, we conducted new experiments without the EEG data.

Using the same experiment layout shown in figure 5.7 and with the same approach described in figure 5.1, we removed the biosensor data from the calculation and gathered results from 29 players. Results from this experiment can be seen in figure 5.8 (B).

Comparing the results for the approach with and without the EEG component, we can observe significant differences.

As we explained before, the adaptation appears for the result with EEG at the end of the experiment, where difficulty and performance/attention values are getting close to each other, in contrast, we can see that for the experiment without EEG, these values are more separated, this demonstrates that the adaptation with EEG is better, in less time, was able to adjust the difficulty accordingly.

Results for the experiment without EEG show how the adaptation occurs abruptly, between levels we can see that changes are not smooth. One could argue that increasing or decreasing a constant in the calculation to soften these changes, the abrupt change problem would be solved, however, this could make the game less interesting by lacking variety (levels would be too similar for too long).

We can see a behavior for the experiments without EEG, show a pattern where performances decreases and increases in each level, this is due to the difficulty being too high or too low in the calculation; on the other hand we can see the results for the approach with EEG, changes occur smoothly and slower than without EEG.

5.6 Discussion

The novelty of our method lies in the introduction of Brain Computer Interface elements for dynamically adapting a game according to the player. We demonstrated that using attention values to determine whether to increase or decrease the difficulty of a game, contributes to better calculate the difficulty of a game and to offer an adaptive experience.

In addition, the combination of the attention values with rhythm-group theory, is a new way to effectively create levels that adapt to the player in 2D platformers.

The hypothesis presented at the beginning of this manuscript: Brain computer interfaces (BCI) can contribute to the adaptation of games, was demonstrated to be true, by conducting experiments with and without the BCI component and obtaining better results with the inclusion of the brain related device.

We can see the influence of including the EEG component by looking at the results in figure 5.8. The adaptation occurs faster for the experiments that included the EEG component than those that didn't include it. This proves that brain computer devices can contribute positively to adapt the difficulty of a game and the overall player experience.

5.7 Chapter Conclusions

We created a new method that is capable of adapting the level of challenge for players depending on their performance and degree of attention (using EEG data).

The adaptation matches different types of player's skills and status, not only experienced players but also inexperienced players, people with different characteristics. We also validated our results by comparing the results with an approach that didn't include the EEG component, demonstrating that the inclusion of brain waves related data can lead to better results.

Due to the biosensor issues and limitations, also the necessity of having the device for the process, we do not envision this method for commercial use yet, however, with a better performance and more comfortable device, this method could be implemented in other genres that involve jumping as a core mechanic or elements in which the rhythm group theory can help to add a sense of harmony to the game. Among the types of games we consider this method could be suitable for are: endless runners, 2D or 3D side-scrolling platformer or rhythm games

For game developers, this would be a good playtesting tool, this method would enable them to gather helpful information to create better levels. For instance, when designing a platformer, developers could create a base design and iterate on it dynamically using our method, evaluate and improve the design.

The method presented in this section could be improved, testing with different EEG biosensors, planning and performing more experiments, testing with more players and modifying the initial values to compare which values adapt the best to this approach. In addition, it would be good to include other factors on the difficulty calculation, so far we have only tested with number of elements in the level, we should also consider position and relationship between elements, which describes another level of difficulty.

We also consider that it would be meaningful to include new elements, other than difficulty to evaluate the player's experience. The harmony that gameplay, graphics and sounds constitute all together to shape experiences for players influence how these players perceive the game in general.

Chapter 6

Graph Grammars: Difficulty Analysis

6.1 Introduction

As explained in previous chapters, this part of the research was conducted with the guidance of professor Koji Mikami (supervisor) and professor Kunio Kondo, who supported the author from the phase of planning and design to implementation and analysis.

Following our motivation of finding better ways to improve the overall experience for players by adapting the degree of difficulty of games they play, we want to design a new method that automatically creates levels with a specific level of difficulty. Using this design, we would be able to adapt the difficulty accurately for a better overall experience.

This section consists of a preliminary exploration conducted to test the base calculations of our proposal. This new method combines Dynamic Difficulty Adjustment (DDA) and Graph Grammars, which is a Procedural Content Generation (PCG) method that enables the creation of levels with high variation and expressiveness.

We introduce our vision on how to apply Graph Grammars to create multi-path levels in 2D platformers with specific degree of difficulty and show the results of experiments designed to test the player's perception of difficulty, in addition, we explain the relationship between the effectiveness of our method and the calculation of difficulty in general.

Results show that there is a strong linear relationship between the difficulty perceived by participants of our experiments and the difficulty calculated by the algorithm we created, with a correlation coefficient of 0.75. In addition, the correlation coefficient of difficulty and player's performance was -0.69, a moderate correlation that shows both variables are inversely proportional. These results indicate that the calculations are heading to the right direction.

6.2 Related Work

The new method we decided to use, Graph Grammars, was originally designed for automatic dungeon creation by David Adams [72] for his Bachelor's thesis research. We found in previous research that this technique has been tested and compared against other methods for automatic content creation [73], having interesting results creating dungeon maps driven by gameplay. In addition to this, the Graph Grammars method was used in a research about a learning game to teach parallel programming [74], which was partially successful showing positive results in automatic puzzle creation.

Some researchers have explored the idea of applying the same method to 2D platformers: proposing a graph-based representation of Super Mario Bros levels using graph grammars [20],[21],[22]. Joris Dormans and Sander Bakkes have focused on using generative graph grammars to procedurally generate missions for action-adventure games, which due to their nature of nonlinear structures creation make them a suitable solution for games that involve exploration [75],[76]; in this research they demonstrated that graph grammars could be a powerful tool to use the advantages of procedural generation and human design in the creation of mission and level design.

Using a dungeon crawler game case of study, researchers concluded that graph grammars can be expressive enough to construct design levels in this type of games, leading to the creation of a variety of possibilities with rich and interesting results for players to explore and enjoy [77].

Besides Graph Grammars, we focused this particular study on perceived difficulty, defined as relative difficulty minus the player's experience at meeting specific challenges [78].

A previous study that involves Bayesian optimization shows that there is a significant relevance between perceived difficulty and engagement [79]. Researchers found that players attributed changes in their performance to their own capacity, which was in reality affected by the covert manipulation of difficulty done by researchers.

We consider that our contribution adds up to previous research on the importance of examining difficulty from the player's point of view when analyzing games [80] and the evaluation of different factors (including difficulty) on the player's performance [81].

6.3 Graph Grammars

The concept of graph grammars involves the idea of having a grammar that handles graphs instead of strings. One of the reasons for replacing strings by graphs when automatically creating levels is that due to their nature, graphs are a good fit to partition and represent the 2D space. Among the advantages of using graph grammars to build levels instead of other kinds of grammars is that they provide much more flexibility when creating variation and randomness. In addition, they allow creating complex levels in a more natural way [72].

The following list shows a formal definition of a directed graph: G := (V,E,lV, lE, s, t), where:

- V(G) := V is a finite set of vertices
- E(G) := E is a finite set of edges
- lV(G) : V $\rightarrow$ LV is a labeling function for vertices
- lE(G) : V $\rightarrow$ LE is a labeling function for edges
- s(G) : E $\rightarrow$ V assigns each edge to its source
- t(G) : E $\rightarrow$ V assigns each edge to its target

A graph grammar is formally defined as a tuple (A,P) where A is a non-empty initial graph and P a set of graph grammar productions. Most approaches of graph production definition concur that each production consists of two parts: left-hand side and right-hand side. Being the left side of the production the one that defines how to replace and transform graphs in the grammar [72]. The set of productions defined in the method we propose are shown in section 6.3.1.

The method we designed to implement Graph Grammars in 2D platformers consists of three different systems that we called: topological map system; area designation system and a graphical map system. In a very brief way, the topological map defines how many nodes and how are the nodes connected in the graph; then, the area designation system assigns a type to each node in the graph; finally the graphical map system sets a representative game object for each node in the graph. Figure 6.1 shows the general approach of our method.

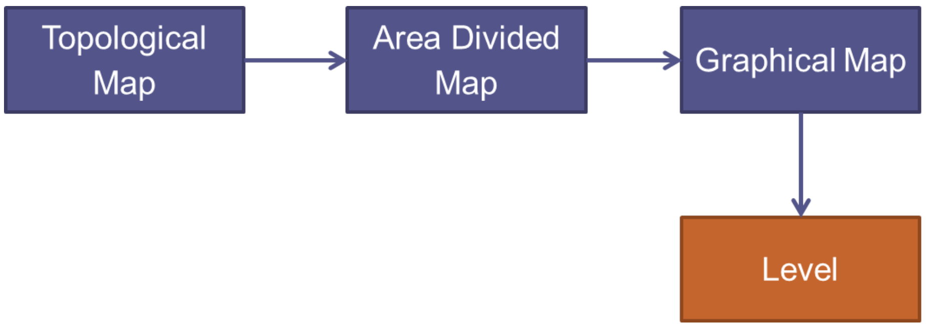


Figure 6.1: **Graph Grammars Creation Approach:**. The topological map creates a graph. The area divided map system assigns a type to each node of the graph. Finally the output area graph is used by the graphical map system to create a level.

6.3.1 Topological Map

This map is the logical representation of the level. The following rules are the set of productions designed to define the grammar used to create the topological map.

1. Starting Rule: This production ensures that there is always a start and there is always an end. It includes a start node S connected directly to the ending node E.
2. Adding Rule: The adding rule enables the algorithm to include more nodes between nodes. We have two different types of adding rules:
   (a) Linear adding rule: the node is created in one of the extremes of the path, increasing the length of this path.
   (b) Non-linear adding rule: The node is created in a different path to the one that is being added. This usually introduces a new level in the system.
3. Linking Rule: The linking rule takes a couple of nodes and links them. This link could be directed or not directed. Two main cases can be considered when linking nodes:
   (a) Case A: Two nodes of the same path are connected
   (b) Case B: Two nodes of different paths are connected
4. Changing Rule: The changing rule enables the algorithm to decide a direction change after connecting all nodes.
5. Deleting Rule: This production says that we can eliminate nodes from a graph when needed.

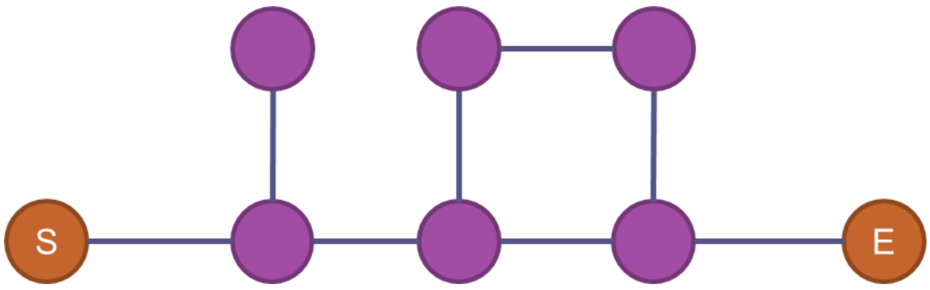


Figure 6.2: **Example of a topological map**. As we can see, there is a main path from S (start) to E (end) and, in addition, a second level (secondary path) with interconnected nodes.

6. Ending Rule: This production ensures that the result level can be cleared. In addition, it will ensure that there is always a complete path from the start of the level to the goal.

In this part of the process there are three main steps to ensure that the map is well created and will output an expected result, these are: main path, secondary path and arcs direction.

1. Main Path: This is the path that connects the start of the level to the end. This ensures that the level can always be finished by the player. This path is mandatory in every level.
2. Secondary Path: All paths that are connected to the main path and are not the main path, are called secondary paths.
3. Direction: The last segment of map generation enables the algorithm to change the direction of each arc to build a unique level. This adds variety and diversity and reduces the probabilities of having two similar levels.

All the previously explained steps guarantee the construction of a final map and sets the basis to create an output that can be used as a playable level in the end of the process. An example of a topological map can be seen in figure 6.2.

6.3.2 Area Designation Map

The result graph created by the topological system passes through a system that takes every node in the graph and assigns an area type, areas can be ”safe areas” or ”dangerous areas”.

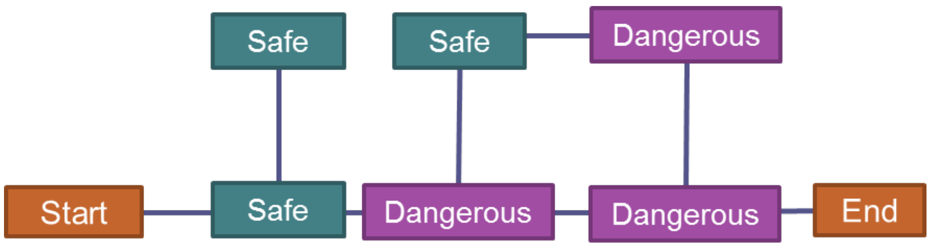


Figure 6.3: **Example of an area divided map**. Using the topological map from 6.2, this is an example of a possible area divided map, after assigning safe and dangerous areas.

A safe area is defined as a platform (of any length) where players can stand safely, it means, not harming game objects will be found in this type of area. Examples of this type of areas are: empty platforms (no enemies) and platforms that contain elements such as obstacles to stand on them, etc.

A dangerous area is defined as a platform (of any length) where players can die, usually represented by areas that contain enemies that can harm the player in any way, examples of this type of areas are: movable platforms, platforms with enemies, platforms that fall, etc.

The starting and ending node are both set to safe areas, however it is possible to modify these rules and adapt them depending on the designer's needs. An example of an area divided map that matches the topological map in figure 6.2 can be seen in figure 6.3.

The number of dangerous areas is decided using the expected difficulty for the level and multiply that by the total number of areas in the level. The calculation can be seen in equation 6.1.

$$a = l_d n \tag{6.1}$$

Where:

- a: Number of dangerous areas
- ld: Expected difficulty for the level
- n: Number of nodes in the graph

6.3.3 Graphical Map

The graphical map is the physical creation of the elements in the level. The main function of this system is to decide where to put each game object on the screen and

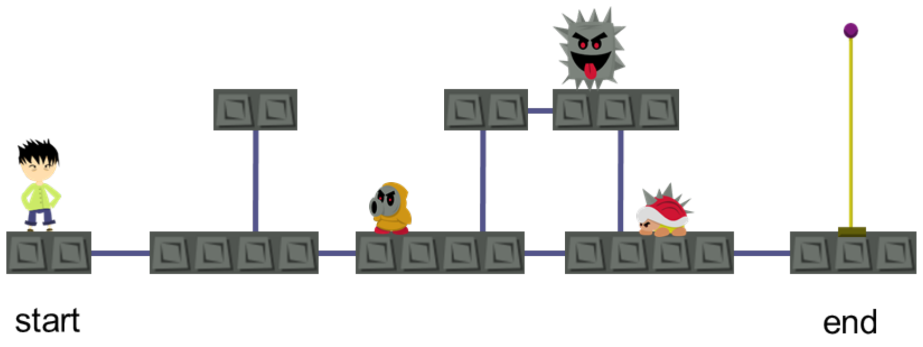


Figure 6.4: **Example of graphical map**. Using the area divided map of 6.3, this is an example of a possible graphical map.

how. This process translates a topological map to a graphical positions on the screen. Designers can decide and adapt this system according to the type of game elements that exist in their games. An example of a graphical map can be seen in figure 6.4.

6.4 Difficulty

The approach we took to calculate difficulty in a level is: assign difficulty values to each area in the graph and use those difficulties to calculate a general difficulty for the level. The general calculation of difficulty for a level can be seen in equation 6.2.

$$l_d = \frac{n}{m}V_1 + \sum_{i=1}^{n} d_i V_2 \tag{6.2}$$

Where:

- ld: Difficulty calculated for one level
- n: Number of dangerous areas in the level
- m: Number of total areas in the level
- di: Difficulty of each area (see equation 6.3)
- Vi: Weights that represent how much influence the component has

As preliminary parameters for this research we decided to start with V1 to 0.5 and V2 to 0.5 as well, due to the influence that both components have in the overall difficulty.

The calculation for each area is carried out using the game objects that are on it, difficulty values are assigned (by the designer, depending on the game) to each game object, for calculation difficulty only dangerous objects count.The difficulty calculation for each area is shown in equation 6.3.

$$a_d = \frac{n}{m} W_1 + \sum_{i=1}^{n} d_i W_2 \tag{6.3}$$

Where:

- ad: Difficulty calculated for one area
- n: Number of enemies in the area
- m: Maximum number of enemies in that area
- di: Difficulty of each enemy (see table 6.1)
- Wi: Weights that represent how much influence the component has

For this research, we set W1 to 0.9 and W2 to 0.1. As preliminary values we chose these two because the number of elements in a platform affects directly the performance, the more enemies, the higher the chance to get hit by them which means the performance would be lower.

To calculate the difficulty for each enemy, we interpret each skill shown in table 6.1 as one point of difficulty. Calculated difficulties are explained as follows:

1. Moves X Axis: Enemies that can walk or run. Moving makes an enemy more difficult than not moving.
2. Moves Y Axis: Enemies that can fall, jump, etc. Moving makes an enemy more difficult than not moving.
3. Shoots: An enemy that can shoot is more difficult than one that cannot shoot.

Table 6.1: Enemies features: Binary representation for the skills: 1=true,0=false

Enemies Difficulties					
	Enemy 1	Enemy 2	Enemy 3	Enemy 4	Enemy 5
Moves X Axis	0	0	0	1	1
Moves Y Axis	0	1	1	0	0
Shoots	1	0	0	0	0
Beatable	1	0	0	0	1
Visible	1	1	0	1	1
Timing Attack	1	1	1	0	0
Aim	1	0	0	0	0
Player Can Stand	0	0	1	0	0
Total	4	4	4	3	2

4. Beatable: Enemies that can be beaten by the player (jumping on its head), an enemy that cannot be beaten is more difficult than an enemy that can.
5. Visible: Enemies can hide, this makes it more difficult than an enemy that cannot hide.
6. Timing attack: The player has to be on alert when the enemy attacks, this adds a point of difficulty.
7. Aim: Aims at the player before attacking, this makes the enemy smarter, one more point of difficulty.
8. Player can stand: If the player can stand on this enemy sometimes and not be harmed by it, the enemy becomes easier. We add 1 point to all those enemies that the player cannot stand on them.

Depending on the kind of game that is being designed by the developers, the values in the difficulties table could vary to adapt to a more accurate result. For simplicity we decided to give one point of difficulty to each skill.

6.5 Player Performance

The player's performance calculation involves how many times is the player hit by an enemy, level clear time and number of collected coins. Equation 6.4 shows the calculation of the performance for this method.

$$p = \frac{1}{(d+1)}X_1 + \frac{eT}{gT}X_2 + \frac{c}{m}X_3 \tag{6.4}$$

Where:

- p: Calculated performance
- d: Number of deaths
- eT: Level estimated time (15 seconds)
- gT: Level cleared time (cannot be less than eT)
- c: Number of collected coins
- m: Maximum number of coins per level (3 coins)

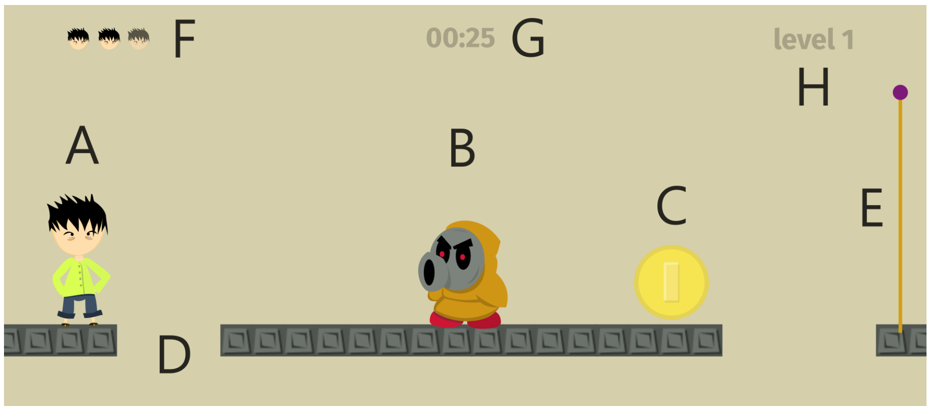


Figure 6.5: **Game Elements**. A: Player; B: Enemies, beatable and unbeatable; C: Coins, motivation to explore; D: Gap; E: Goal; F: Player's health and coins (3 per level); G: Game Time; H: Current Level.

- Xi: Weights that represent how much influence the component has

For this study we started testing with a set of parameters that we consider is the most suitable combination of values: X1: 0.5, X2: 0.15 and X3: 0.35, decided by the importance of each parameter in the equation.

6.6 Implementation and Experiments

The game is a side-scrolling 2D platformer in which the player has to reach a goal placed on the right-most part of the level. The player is also required to overcome very simple challenges: gaps between platforms, beatable enemies and unbeatable enemies, both types of enemies static. We also included 3 collectable coins to increase motivation for the player to explore different paths. Figure 6.5 shows the elements of the implemented game.

Players were required to play and clear 12 different levels generated automatically by our system. The first three levels were part of a tutorial to give players a sense of difficulty and rating. After finishing one level, players were taken to a results screen that showed a set of 10 different options to rate the level: [1-10]; being 1 the easiest and 10 the hardest values. Numbers were presented as integers to the player but they were normalized (0,1) for our internal calculations. The whole flow of the experiment can be seen in Figure 6.6.

We collected data from 16 different players. All participants were between 21 – 40 years old; almost all of them with previous experience playing *Super Mario Bros* (which we asked to have background information about their skills), one of the players

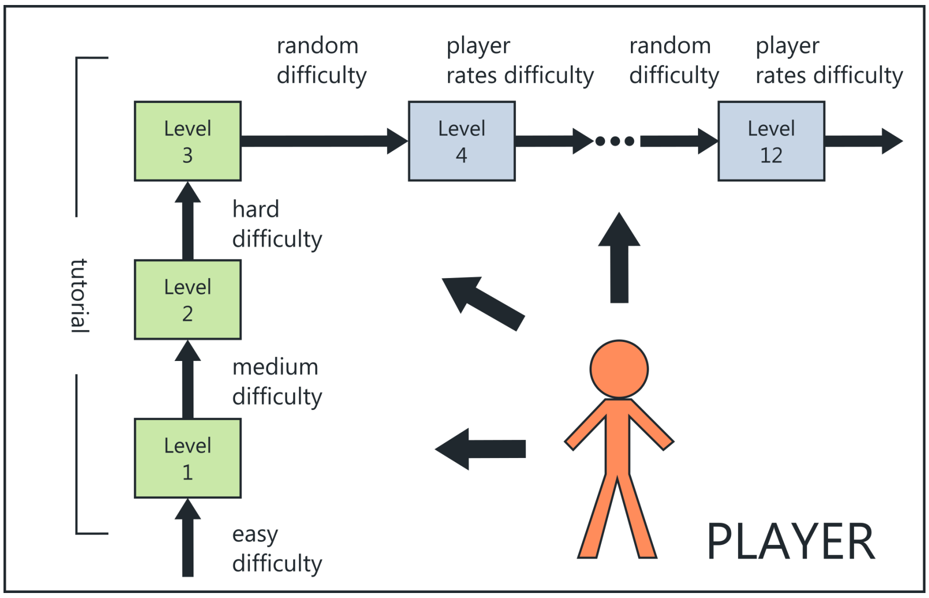


Figure 6.6: **Experiments**. Players played three tutorial levels and nine levels for which difficulty was randomly calculated. After playing a level players rated its difficulty.

(6.25%) has never played Super Mario Bros before, only the 7 players (43.75%) have cleared the game at least once.

For our implementation, we consider the following parameters: minimum number of nodes in the main path, maximum number of nodes, maximum number of secondary paths, minimum number of branch nodes and maximum number of branch nodes.

1. Minimum number of nodes in the main path
2. Maximum number of nodes in the graph
3. Minimum number of secondary paths
4. Maximum number of secondary paths
5. Minimum number of branch nodes
6. Maximum number of branch nodes

In our experiments, the parameters of this system we set to the minimum possible values for simplicity. The whole creation of a topological graph is done randomly and automatically. Since this step of the process does not involve any graphical elements, we do not consider difficulty when creating this first part of the graph.

Graph Grammars System

In our implementation, the Graph Grammars method involves three different subsystems: topological map system, area divided map system and graphical map system, each one with a clearly different purpose.

Topological Map: Graph Creation

A set of parameters are important to take into consideration to construct a graph, these are: minimum number of nodes in the main path, maximum number of nodes, maximum number of secondary paths, minimum number of branch nodes and maximum number of branch nodes. For our research we used the following values:

1. Minimum number of nodes in the main path: number of starting nodes between the node S (start node) and node E (end node). We set this value to 3.

2. Maximum number of nodes: Including all paths, main and secondary, the maximum number of nodes was set to 15.

3. Minimum number of secondary paths: number of paths (besides the main path) to start. The algorithm can create graphs with no secondary paths.

4. Maximum number of secondary paths: For simplicity, this parameter was set to 1, it means, in total, a level created by this algorithm could have 2 paths.

5. Minimum number of branch nodes: it's related to the minimum number of nodes that can be created as a branch of an existing node. We set this parameter to 1.

6. Maximum number of branch nodes: it's related to the maximum number of nodes that can be created as a branch of an existing node. We set this parameter to 3.

We set these parameters to the minimum possible values for simplicity and to demonstrate the capabilities of our method.

Figure 6.7 and 6.8 show are examples of levels created for a low performance player and a high performance player respectively. Both results were generated using the Graph Grammars implementation explained in this section.

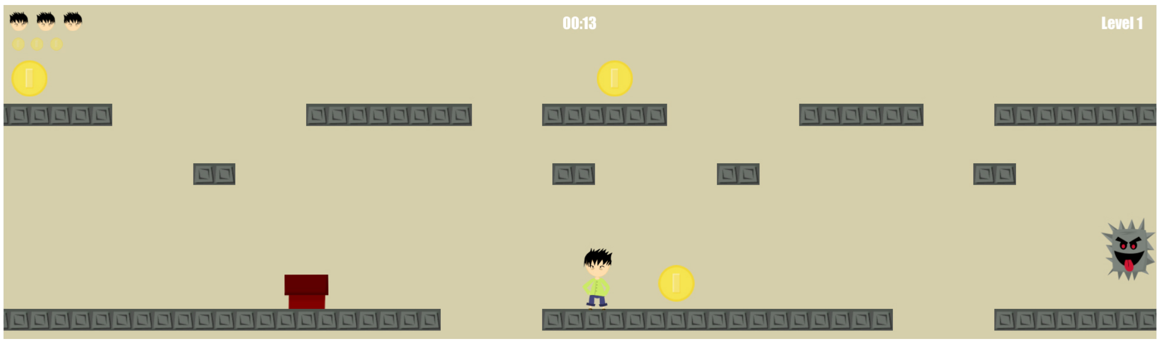


Figure 6.7: **Low performance player**. A level created for a low performance player, few challenges, easier to clear

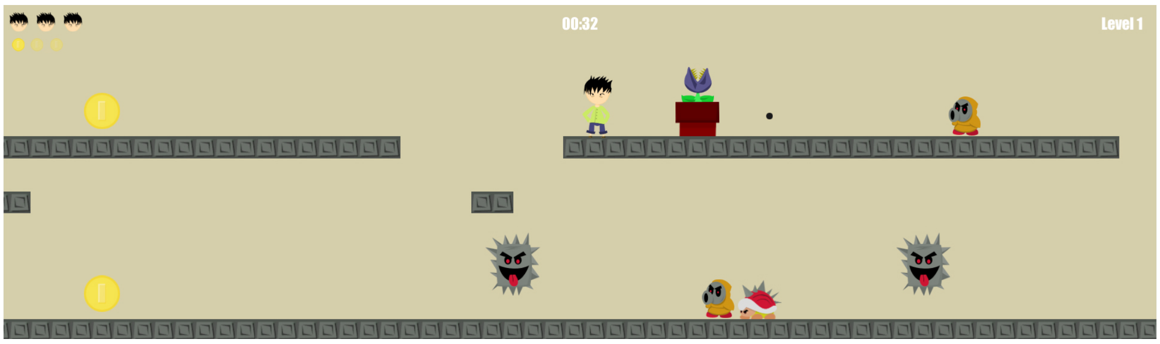


Figure 6.8: **High performance player**. A level created for a high performance player, more challenges, more difficult to clear.

Area Divided Map

In addition to assigning a type to each node (area) of the graph, this system calculates the difficulty of each dangerous are depending on the expected difficulty of the whole level. The difficulty calculation for each area is shown in equation 6.3.

Where:

- ad: Difficulty calculated for one area
- n: Number of enemies in the area
- m: Maximum number of enemies in that area
- di: Difficulty of each enemy (see table 6.1)

For this research, we set W1 to 0.9 and W2 to 0.1. As preliminary values we chose these two because the number of elements in a platform affects directly the performance, the more enemies, the higher the chance to get hit by them which means the performance would be lower.

To calculate the difficulty for each enemy, we interpret each skill shown in table 6.1 as one point of difficulty. Calculated difficulties are explained as follows:

1. Moves X Axis: It means the enemy can move in the X axis, enemies that can walk or run. Moving makes an enemy more difficult than not moving, if the enemy moves it gets one point o difficulty.
2. Moves Y Axis: It means the enemy can move in the Y axis, enemies that can fall, jump, etc. Moving makes an enemy more difficult than not moving, if the enemy moves it gets one point o difficulty.
3. Shoots: An enemy that can shoot is more difficult than one that cannot shoot, shooting gives one point of difficulty to an enemy.
4. Beatable: The enemy can be beaten by the player (jumping on its head), an enemy that cannot be beaten is more difficult than an enemy that can, an unbeatable enemy gets one point of difficulty.
5. Visible: Enemies can hide (like enemy 3), this makes it more difficult than an enemy that cannot hide. Hiding enemies get one point of difficulty.
6. Timing attack: since the player has to be on alert when the enemy attacks, this adds a point of difficulty to this kind of enemy.
7. Aim: Aims at the player before attacking, this makes the enemy smarter, one more point of difficulty.
8. Player can stand: If the player can stand on this enemy sometimes and not be harmed by it, the enemy becomes easier. We add 1 point to all those enemies that the player cannot stand on them.

Depending on the kind of game that is being designed by the developers, the values in the difficulties table could vary to adapt to a more accurate result. For simplicity we decided to give one point of difficulty to each skill.

Number of dangerous areas

To decide the number of dangerous areas, we use the expected difficulty for the level and multiply that by the total number of areas in the level. The calculation can be seen in equation 6.1.

Graphical Map: Level Creation

An explanation on how to create game objects for each area is as follows.

1. Coins: Coins can be placed in any type of area, safe or dangerous and there is a fixed amount of coins per level (3 coins) so we randomly choose 3 areas (could be the same area) and decide where the coins will be.

2. Safe Areas: When geometrically construction safe areas, we can decide to add a coin, a harmless obstacle or nothing. Coins have priority because it is mandatory that each area with a coin shows that coin somewhere. After deciding where the coin should be placed, the rest of the available spaces in the are set to having an obstacle or being empty. We do this randomly.

3. Dangerous Areas: To decide what enemies will be placed in a dangerous area, we use the same approach that we used to calculate how difficult an area should be but instead of using the whole level's difficulty, we use the area's difficulty and distribute the enemies on the area in a random way.

6.7 Results and Analysis

We calculated the linear regression of the computed difficulty, perceived difficulty and performance in two different plots. Figure 6.9 shows the results of the calculated difficulty and perceived difficulty, figure 6.10 shows the results of the calculated difficulty and performance calculated by the algorithm.

All the results for this experiment can be seen in table 6.2, these include all players (11) and the data from level 1 to level 5. In addition, figure 6.11 shows a box plot applied to these results.

6.7.1 Difficulty Calculation

The correlation coefficient for calculated vs perceived difficulty variables was 0.75, which is considered a strong correlation (more than 0.7) using statistical basics. This means that both results are strongly related, which demonstrates that the approach of the calculations of difficulty in our algorithm are heading in the right direction.

One of the reasons that these results might have been affected is the way the experiment was designed, each level's difficulty was decided randomly during the experiment, with no particular order in way the difficulty was presented to players. We

Table 6.2: Graph Grammars Results

Player	Level	Difficulty	Performance	Player	Level	Difficulty	Performance
	1	0.1	0.937004		1	0.1	0.936488
	2	0.217126	0.955643		2	0.217061	0.433379
1	3	0.336581	0.927989	7	3	0.146233	0.923474
	4	0.327579	0.423869		4	0.261668	0.936667
	5	0.255563	0.91838		5	0.378751	0.945404
	1	0.1	0.888843		1	0.1	0.938433
	2	0.211105	0.930866		2	0.217304	0.942267
2	3	0.327464	0.0842169	8	3	0.335088	0.941259
	4	0.212991	0.0874164		4	0.327745	0.4258
	5	0.223918	0.60762		5	0.25597	0.944469
	1	0.1	0.943615		1	0.1	0.931607
	2	0.217952	0.443859		2	0.216451	0.953831
3	3	0.148434	0.943801	9	3	0.33568	0.91068
	4	0.266409	0.942995		4	0.324515	0.931903
	5	0.259284	0.438822		5	0.441003	0.429805
	1	0.1	0.941581		1	0.1	0.874136
	2	0.217698	0.469794		2	0.209267	0.385828
4	3	0.151422	0.934526	10	3	0.132495	0.929827
	4	0.268238	0.414802		4	0.248724	0.92858
	5	0.195088	0.937996		5	0.239796	0.401388
	1	0.1	0.920629		1	0.1	0.964199
	2	0.0900787	0.897157		2	0.220525	0.409045
5	3	0.0772233	0.919488	11	3	0.146655	0.937766
	4	0.192159	0.919229		4	0.263876	0.90859
	5	0.182063	0.436498		5	0.25245	0.426269
	1	0.1	0.410059				
	2	0.0262574	0.410704				
6	3	0.0775954	0.536723				
	4	0.0196858	0.40721				
	5	-0.054413	0.543851				

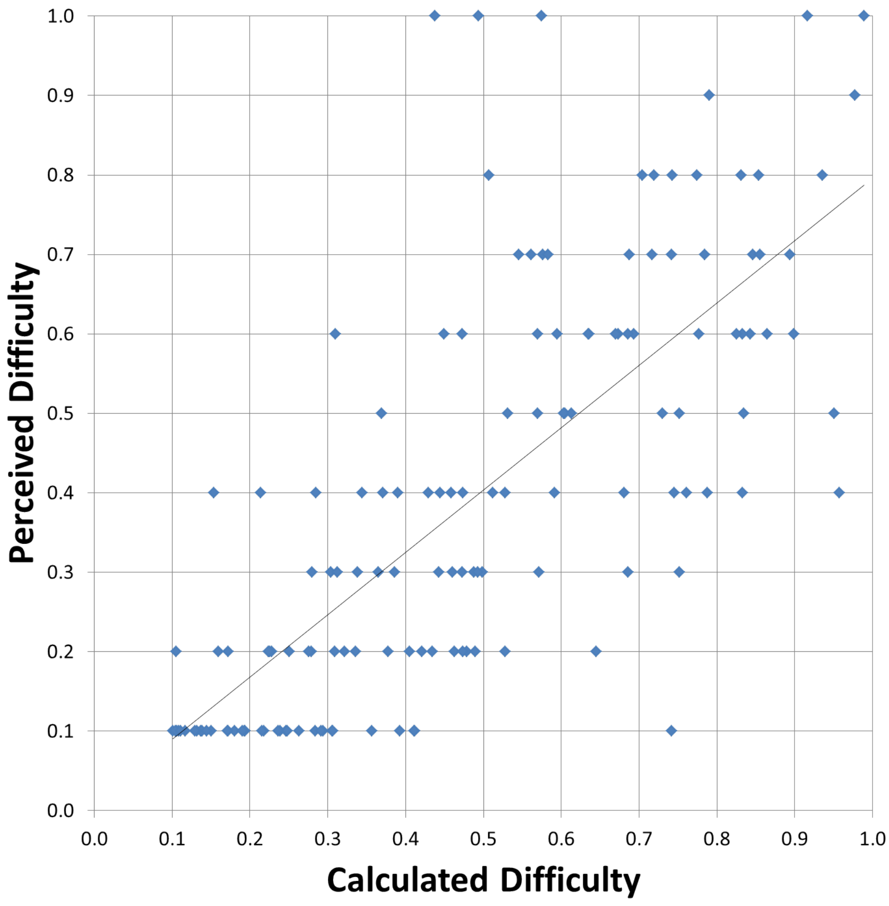


Figure 6.9: **Perceived Difficulty vs Performance.** The horizontal axis shows the calculated difficulty, vertical axis the perceived difficulty. We can clearly see a strong linear relationship between the variables.

consider that a way to improve this experiment is to control exactly which difficulty and when to present it to players.

Despite the fact that the calculations and the experiment should be modified and improved to avoid abrupt changes of difficulty while playing, having a 0.75 of correlation between the perceived difficulty and the calculated one is a positive result. This means we can keep enhancing this method to achieve better results.

6.7.2 Performance and Difficulty

In figure 6.10 we can see how the data has a strong linear inverse tendency, with a correlation coefficient of -0.69, despite the fact that is not as high as 0.7 to qualify as strong correlation, it's a high value that represents a close relationship between the difficulty and performance calculated by the algorithm. This result shows that

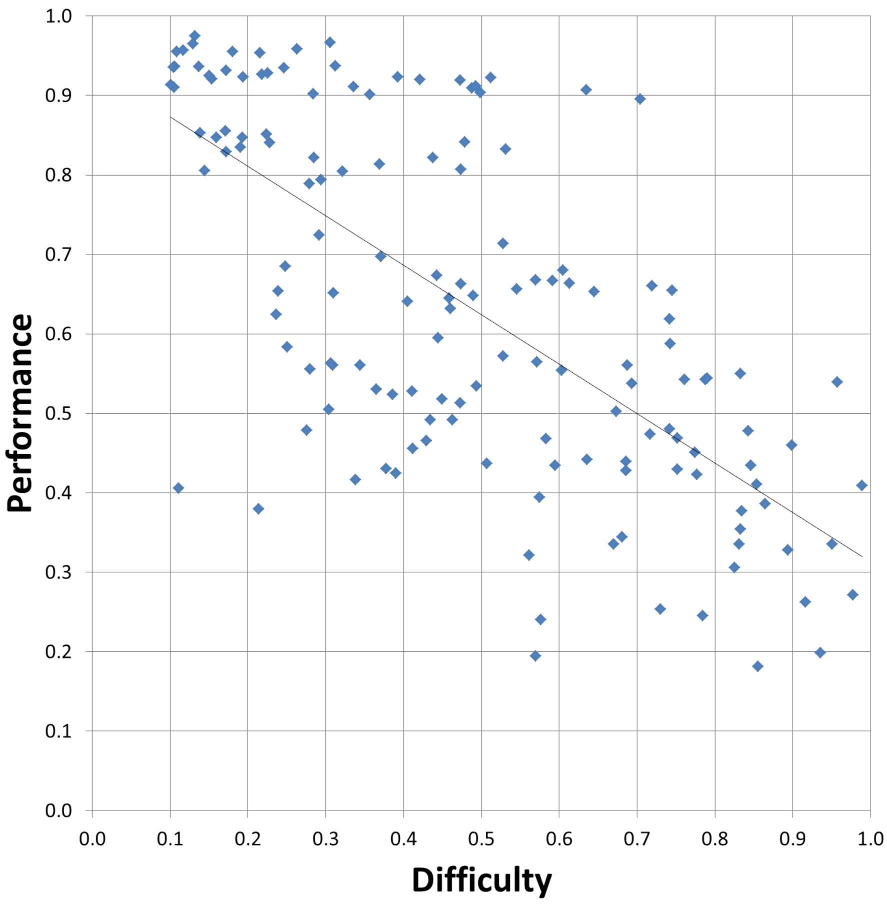


Figure 6.10: **Performance vs Calculated Difficulty.** The horizontal axis shows the calculated difficulty, vertical axis the performance. We can see a linear inversely proportional relationship between the variables.

difficulty and performance are inversely proportional in this case, the higher the difficulty, the lower the performance, which in a way would be an expected result, however it depends on the kind of player that is playing.

Since the conducted experiments did not involve any adaptation functionality, we consider this a positive result, nonetheless for the ultimate goal of our research, adapting the player experience according to the player's skills, this correlation should be the opposite.

6.8 Discussion

Graph grammars was a method originally designed to automatically create dungeons in dungeon crawlers, RPG games, etc. Due to the expressiveness and variety of the results that can be obtained with graph grammars, we decided to use them to

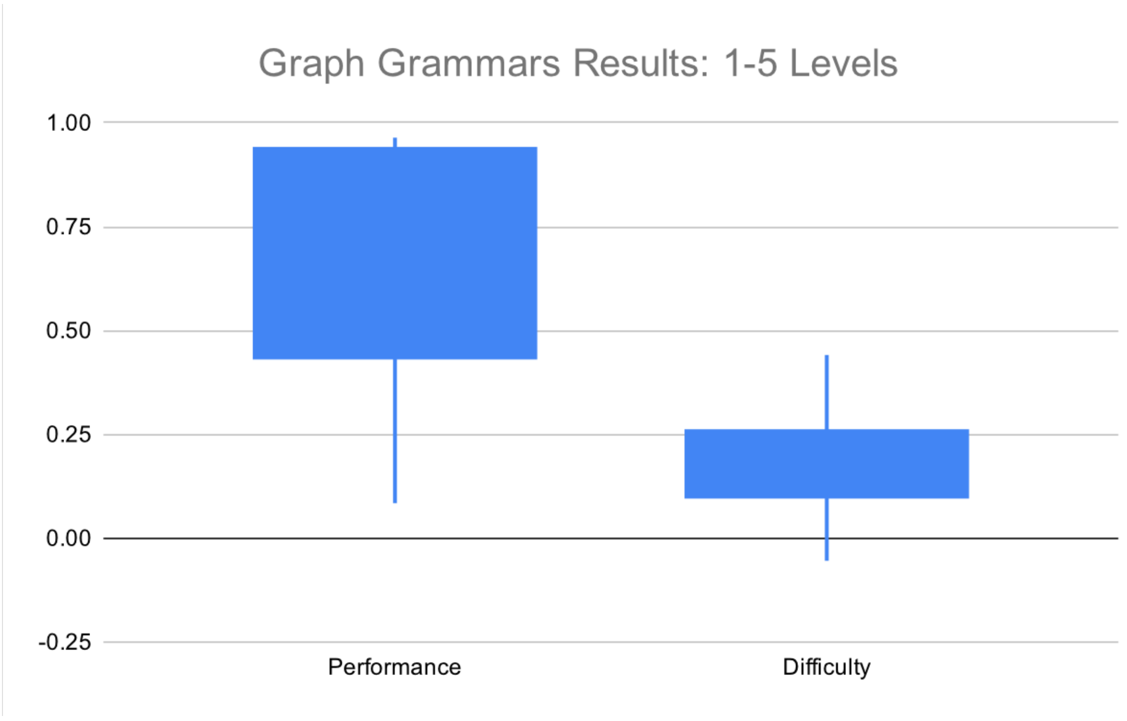


Figure 6.11: Graph Grammars. Results for all experiments.

create levels in 2D platformers, to evaluate their effectiveness and also to determine if they could be used to create levels with a specific difficulty.

The use of graph grammars in 2D platfomers is a new way to procedurally generated levels with a high level of variety. In addition, the nature of graph grammars, enabled us to come up with a novel way to calculate difficulty for each level that is created automatically.

Additionally, the levels created by our algorithm consists of multiple paths. Which encourage players to to explore and to enjoy a variety of possible maps using the same core concept.

The results of our experiments demonstrate that it is possible to create levels with a specific degree of difficulty, which was one of the hypotheses established in the introduction.

Being able to create stages with a specific degree of difficulty, helps developers to achieve a higher level of granularity and with this, it is possible to adapt the player experience in a more specific and detailed way.

Our method effectively creates levels with a specific level of difficulty. We believe that it is possible to improve the proposed model and also achieve better results by testing with other parameters, however, the presented method is one way to do it and we can corroborate that one of the objectives of our research was completed.

These results also prove our final hypothesis: dungeon creation techniques can be effectively used to create automatically generated levels in other genres. Of course we would need to test with genres besides 2D platformers, but we have at least one proof that this can be done.

In addition, we consider that Graph Grammars can be used to create 3D platformers as well. Although we haven't tested this approach for this research, using the same graphs generated by the current proposed method and mapping that graph to geometries in 3D, such as level areas and platformers located in the Z axis, it would be possible to automatically create levels in 3D games, particularly for platformers.

As for the final objective of our research in this part, we know that graph grammars can effectively help create levels automatically in platformrers, however, depending on the type of game, the algorithm needs to be adapted for achieving good results. In addition, we consider that for platform related games this would be a good method but perhaps not so effective in other genres, further experiments need to be conducted to prove it.

6.9 Chapter Conclusions

We successfully implemented a method that involves Graph Grammars to create multipath levels in 2D platform games, which increases expressiveness and variety of automatically created levels in procedural construction of levels.

This is only a first approach of a more general and bigger purpose method to adapt the player experience through difficulty balance and performance.

Our experiment showed a 0.75 correlation coefficient between the difficulty calculated by our algorithm and the difficulty perceived by players. This strong correlation demonstrates that the approach is heading to the right direction, however it's necessary to keep improving the current results to design a more robust method that accurately defines difficulty for players.

Results also showed that performance and difficulty have a correlation of -0.69, which is near to be considered a strong correlation and reinforces an expected outcome: the level's difficulty and player performance are inversely proportional. This result in the general approach of experience adaptation should be the opposite at some point where the performance of players increases along with the difficulty of the levels they play.

This method can be improved by changing the parameters of each formula and test accordingly with players to find the best values and get positive results. In addition, new experiments should be designed and conducted to evaluate the accuracy of this method. Finally we will include these calculations in a more general method designed to adapt player experience and compare results.

Chapter 7

General Discussion

In previous sections, it was discussed how results are connected with the hypotheses proposed at the beginning of this document, the novelty of each method or approach we designed and what could be improved from each method.

This research has as main motivation, to find new ways to improve the experience for players. Specifically, to contribute to reduce the gap that exists between levels of challenges and players' skills.

Although three different types of approaches that seem to be unrelated were tested, the author considers that there are important elements that connect them in a meaningful way. Behavioral patterns, Brain Computer Interfaces and Difficulty Perception are directly related to the player's status. The way that the player physically reacts can is part of the player behavior and contributes to determine how they feel at a determined time. The brain activity can accurately show, for example, how focused is a player in real time, which is part of the player behavior too. Finally, the perception of difficulty indirectly shows the player perception towards the game, indicating how is the current behavior and how possibly could be used to improve the overall experience.

These three ways of tackling the same problem have shown positive results despite the differences in their nature. Separately, it has been possible to adapt the difficulty or propose a way to adapt the difficulty of a game according to the player's skills. This was one of the objectives that were defined at the beginning of this research and the three of them have demonstrated positive results towards finding novel solutions to the main problem.

The author also considers that it would be interesting to combine some of these approaches to propose new ways to automatically create content for the end user and to evaluate their effectiveness for developers to create better games.

The combination between EEG and rhythm-group theory was the most innovative way to propose a solution for the problem. This method was not necessarily the best but it was a new perspective on a problem that has been tackled by previous researchers with similar tools.

Using Graph Grammars in a different genre also demonstrated to be effective and with better parameters, could be a good tool to create endless content in other genres. Perhaps this could be combined with the EEG component in order to make more interesting experience.

Finally, behavioral patterns are probably one of the most promising proposals. The author considers that combining different methods of classification in order to understand the player's emotions and perception, could lead to designed a tailored experience for players. This method can also be used commercially by developers, instead of measuring the pressure sensitivity, it is possible to measure the time between pressing an releasing a button in order to get an accurate approximation of the force exerted on the button of the controller.

Chapter 8

Conclusions

This research was started to find new approaches and solutions to the problem of the difficulty imbalance that exists in videogames in order to avoid a negative experience for players and with this, provide alternatives for developers to offer a adequate challenges in the games they create.

Different methods were designed, applied and tested in genres such as 2D platformers and 2D shooting games. From chapter 6 with the creation of multipath levels in 2D platform games; chapter 5 by using EEG and performance in 2D platformers; to chapters 3 and 4, that explore a three step general approach to use behavioral patterns to recognize how players experience 2D shooting games in real time and modify the game in consequence.

Graph Grammars proved to be a suitable solution to automatically create levels with multiple paths from start to end in 2D platformers. After conducting experiments and analyzing the results, the analysis shows that there is a 0.75 correlation coefficient between the perceived difficulty by players and the difficulty the designed algorithm calculated. With this not only it was able to propose the new model to design levels in real time but also a model with a specific difficulty that can be helpful when using procedural content generation methods.

In addition, results from these experiments also corroborate that players' performance and the difficulty they face is inversely proportional with a correlation coefficient of -0.69. This is actually an expected results but in this research it was possible to clearly show it after testing.

Rhythm-Group Theory, which has been successfully used in previous research to create levels automatically in 2D platformers, in combination with attention levels obtained in real time from a biosensor (Neurosky) by using EEG, demonstrated to be a good approach to adapt the levels of challenges for players. Including the new EEG component to adaptation approach worked to compensate the results obtained from calculated performance while playing and with this, add more variety to the levels designed by the proposed algorithm.

The skills of players that participated in these experiments vary from people that play games everyday to people that rarely play videogames, which shows that this type of approach could successfully adapt content for experienced and inexperienced players.

As a validation for the overall method and results, experiments without the EEG component were conducted to compare the effectiveness of this new model, results show that having the EEG component can improve the overall calculations.

For the second part of the research, the autho decided to delve in the field of behavioral patterns and try to find ways to predict players' emotions and perception towards games in real time. With the goal of improving the player experience by using behavioral patterns, this approach was divided in three steps that complemented each other: (1) data collection, (2) classification methods and (3)difficulty adaptation.

The focus was to analyze the pressure exerted on a gamepad's button, expecting that depending on how players felt while playing, would somehow be reflected by the way the press the button. For instance, when players are more excited while playing, press the button harder, etc.

Experiments were designed to trigger the action of pressing a button as many times as possible by playing a simple 2D shooting game. Players were required to fill in a questionnaire about their experience after playing each levels.

From this data, it was shown that pressure sensitivity (exerted on the PS3 controller) and difficulty were directly proportional, corroborating what was found in previous research. In addition, results showed that players tended to press the button harder when they played more difficult levels.

One of the parameters measured in this experiment was dominance and it showed a strong negative correlation with pressure. One can deduce that players pressed the button harder when they felt less dominant, related as well with the difficulty they were facing at that time.

Another clear result from this part of the data analysis was that fun and pressure are directly correlated, the more fun players had, the harder they tended to press the button.

Taking a look at the players' background and their experience while playing, it was found that players with more experience press the button softer than players with less experience. Considering that more experienced players are used to a gamepad and have played 2D shooting games in the past, it's a task that would be easier for them than for players without this experience.

For the second part of the behavioral patterns approach, several Neural Networks and Support Vector Machines were tested to find the best model that could predict emotions from players by looking at the pressure they exerted on a button.

Results showed that Support Vector Machines can predict this type of data better than Neural Networks. Support Vector Machines showed an average accuracy of with 70.69% and Neural Networks an average accuracy of 68.55%.

It was possible to improve previous research's [32] results with the proposed models, for which they achieved 53.44% of accuracy to classify boredom, engagement and fun. In this research's case, these three parameters had a better accuracy, including boredom with 83.64%.

In addition, the parameter with the best results was boredom, followed by frustration, fun, difficulty and arousal which were predicted with more than 70% of accuracy.

After completing the second step of the proposed new approach, the plan is to use these results to propose a final method that can adapt the game depending on the emotions and perceptions that players are having while playing.

The general idea is to use the models created in step 2, specifically Support Vector Machines with the best parameters: boredom, frustration, fun and difficulty and use them to modify the game in real time and offer a more adequate experience for players.

In this research,the author successfully designed and tested different approaches and methods to find new solutions about the problem that exists with difficulty imbalance in videogames. Combining new components (EEG), testing methods that were successful in other genres (Graph Grammars), focusing on finding patterns to detect how players behave or react towards the game (Pressure Sensitivity), etc, the author proposed different ideas that can be used by game developers in the industry or the academia to improve the overall experience they offer to players or simply to act in consequence for specific purpose.

There are still ways to improve the methods presented in this thesis, there are parameters to change or test, however, this first step towards making games more enjoyable for a wider range of people can work as a cornerstone to design better games for players.

Publications

Journals

[1] Henry Fernández, Koji Mikami, Kunio Kondo, Perception of Difficulty in 2D Platformers Using Graph Grammars.*International Journal of Asia Digital Art and Design Association*, 2018, Volume 22, Issue 2, Pages 38-46, Released October 12, 2018, Online ISSN 1738-8074.

[2] Henry Fernández, Koji Mikami, Kunio Kondo, Difficulty Adjustment Using Player's Performance and Electroencephalographic Data. *The Journal of the Society for Art and Science*, 2019, Volume 18, Issue 5, Pages 143-155, Released December 25, 2019, Online ISSN 1347-2267.

[3] Henry Fernández, Koji Mikami, Kunio Kondo, Pressure Sensitivity Pattern Analysis Using Machine Learning Methods. *The Journal of the Institute of Image Electronics Engineers of Japan.* Accepted but not published.

International Conferences

[4] Henry Fernández, Koji Mikami, Kunio Kondo, Pressure Sensitivity Pattern Analysis Using Machine Learning Methods.*IEVC 2019*, 2019, Bali. Double blind review.

[5] Henry Fernández, Koji Mikami, Kunio Kondo, Analyzing the Relationship Between Pressure Sensitivity and Player Experience. *VRCAI 2018*, 2018, Tokyo. Double blind review.

[6] Henry Fernández, Koji Mikami, Kunio Kondo, Perception of Difficulty in 2D Platformers Using Graph Grammars. *Adada International 2017*, 2017, Gwangju . Single blind per review.

[7] Henry Fernández, Koji Mikami, Kunio Kondo, Adaptable Game Experience Based on Player's Performance and EEG. *Nicograph International*, 2017, Kyoto. Pages 1-8. Reviewed by 3 people.

[8] Henry Fernández, Koji Mikami, Kunio Kondo, Adaptable game experience through procedural content generation and brain computer interface. *SIGGRAPH 2016*, 2016, California. Poster Presentation.

References

[1] Eva Kraaijenbrink, Frank Gils, Quan Cheng, Robert Herk, and Elise Hoven. Balancing skills to optimize fun in interactive board games. In *Proceedings of INTERACT '09*, pages 301–313. Springer-Verlag Berlin, Heidelberg ©2009, August 2009.

[2] Denisova Alena and Cairns Paul. Adaptation in digital games: The effect of challenge adjustment on player performance and experience. In *Proceedings of CHI PLAY '15*, pages 97–101. ACM New York, NY, USA ©2015, October 2015.

[3] Yun Chang, Trevino Philip, Holtkamp William, and Deng Zhigang. Pads: enhancing gaming experience using profile-based adaptive difficulty system. In *Proceedings of Sandbox '10*, pages 31–36. ACM New York, NY, USA ©2010, July 2010.

[4] Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems. Standard, Ergonomics of human-system interaction, July 2019.

[5] Vicki L. Hanson. The user experience: Designs and adaptations. *SIGCAPH Comput. Phys. Handicap.*, (76):4–5, June 2003.

[6] Juha Arrasvuori, Hannu Korhonen, and Kaisa Väänänen-Vainio-Mattila. Exploring playfulness in user experience of personal mobile products. In *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction*, OZCHI '10, page 88–95. Association for Computing Machinery, 2010.

[7] Steve Benford, Chris Greenhalgh, Gabriella Giannachi, Brendan Walker, Joe Marshall, and Tom Rodden. Uncomfortable user experience. *Commun. ACM*, 56(9):66–73, September 2013.

[8] Paul Dourish. User experience as legitimacy trap. *Interactions*, 26(6):46–49, October 2019.

[9] Richard F. Forno. Risk awareness and the user experience. In *Proceedings of the 37th ACM International Conference on the Design of Communication*, SIGDOC '19, New York, NY, USA, 2019. Association for Computing Machinery.

[10] Evelyn Tio, Megan Torkildson, Denise Su, Heidi Toussaint, Aditi Bhargava, and Dawn Shaikh. Measuring holistic user experience: Keeping an eye on what matters most to users. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '19. Association for Computing Machinery, 2019.

[11] Jeppe Komulainen, Jari Takatalo, Miikka Lehtonen, and Göte Nyman. Psychologically structured approach to user experience in games. In *Proceedings of the 5th Nordic Conference on Human-Computer Interaction: Building Bridges*, NordiCHI '08, page 487–490, New York, NY, USA, 2008. Association for Computing Machinery.

[12] Zhan Ye. Genres as a tool for understanding and analyzing user experience in games. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '04, page 773–774, New York, NY, USA, 2004. Association for Computing Machinery.

[13] Jonathan Moizer, Jonathan Lean, Elena Dell'Aquila, Paul Walsh, Alphonsus (Alfie) Keary, Deirdre O'Byrne, Andrea Di Ferdinando, Orazio Miglino, Ralf Friedrich, Roberta Asperges, and Luigia Simona Sica. An approach to evaluating the user experience of serious games. *Computers & Education*, 136:141 – 151, 2019.

[14] Ilias O. Pappas, Patrick Mikalef, Michail N. Giannakos, and Panos E. Kourouthanassis. Explaining user experience in mobile gaming applications: an fsqca approach. *Internet Res.*, 29:293–314, 2019.

[15] Don Shin. How does immersion work in augmented reality games? a user-centric view of immersion and engagement. *Information, Communication and Society*, 22:1–18, 12 2017.

[16] Jared E. Cechanowicz, Carl Gutwin, and Scott Bateman. Improving player balancing in racing games. In *Proceedings of CHI PLAY '14*, pages 47–56. ACM New York, NY, USA ©2014, October 2014.

[17] Alexander Baldwin, Daniel Johnson, and Peta A. Wyeth. The effect of multi-player dynamic difficulty adjustment on the player experience of video games. In *Proceedings of CHI EA '14*, pages 1489–1494. ACM New York, NY, USA ©2014, April 2014.

[18] Robin Hunicke. The case for dynamic difficulty adjustment in games. In *Proceedings of ACE '05*, pages 429–433. ACM New York, NY, USA ©2005, June 2005.

[19] Martin Jennings-Teats, Gillian Smith, and Noah Wardrip-Fruin. Polymorph: Dynamic difficulty adjustment through level generation. In *Proceedings of PCGames '10*. ACM New York, NY, USA ©2010, June 2010.

[20] Kate Compton and Mateas Michael. Procedural level design for platform games. In *AIIDE 2006*, pages 109–111. American Association for Artificial Intelligence 2006, June 2006.

[21] Wang Hanqing, Koji Mikami, and Kunio Kondo. *A Research on the method of Automatic Map Generation of Platform Game.* Tokyo University of Technology, Japan, 2010.

[22] Santiago Londono and Olana Missura. Graph grammars for super mario bros * levels. In *Sixth FDG Workshop on Procedural Content Generation, At Asilomar, Pacific Grove, CA, USA*, June 2015.

[23] Noor Shaker, Georgios Yannakakis, and Julian Togelius. Towards automatic personalized content generation for platform games. In *Proceedings of AIIDE '10*, 2010.

[24] Mihaly Csikszentmihalyi. *Flow: The Psychology of Optimal Experience.* Harper Perennial, March 1991.

[25] Gomez-Hicks Guillermo and Kauchak David. Dynamic game difficulty balancing for backgammon. In *Proceedings of ACM-SE '11*, pages 295–299. ACM New York, NY, USA ©2011, March 2011.

[26] Jonathan Sykes and Simon Brown. Affective gaming: Measuring emotion through the gamepad. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, pages 732–733, New York, NY, USA, 2003. ACM.

[27] Wouter M Van Den Hoogen, Wijnand Ijsselsteijn, W A Ijsselsteijn@tue, Yvonne A W Nl, and Yvonne De Kort. Effects of sensory immersion on behavioural indicators of player experience: Movement synchrony and controller pressure. 01 2009.

[28] Ashwaq Alhargan, Neil Cooke, and Tareq Binjammaz. Multimodal affect recognition in an interactive gaming environment using eye tracking and speech signals. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ICMI 2017, pages 479–486, New York, NY, USA, 2017. ACM.

[29] Wouter M Van Den Hoogen, Eelco Braad, and Wijnand Ijsselsteijn. Pressure at play: Measuring player approach and avoidance behaviour through the keyboard. 08 2014.

[30] Ana Paula Soares, Ana P Pinheiro, Ana Teresa Perez Costa, Carla Sofia Frade, Montserrat Comesaña, and Rita Pureza. Affective auditory stimuli: adaptation of the international affective digitized sounds (iads-2) for european portuguese. *Behavior research methods*, 45 4:1168–81, 2013.

[31] Karolien Poels, Wouter van den Hoogen, Wijnand Ijsselsteijn, and Yvonne De Kort. Pleasure to play, arousal to stay: The effect of player emotions on digital game preferences and playing time. 15:1–6, 08 2011.

[32] Guillaume Chanel, Cyril Rebetez, Mireille Bétrancourt, and Thierry Pun. Boredom, engagement and anxiety as indicators for adaptation to difficulty in games. In *Proceedings of the 12th International Conference on Entertainment and Media in the Ubiquitous Era*, MindTrek '08, pages 13–17, New York, NY, USA, 2008. ACM.

[33] Alessandro Canossa, Anders Drachen, Janus Rau, and Møller Sørensen. Arrgghh!!! -blending quantitative and qualitative methods to detect player frustration (pre-print). 06 2011.

[34] Sebastian C. Müller and Thomas Fritz. Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress. In *Proceedings of the 37th International Conference on Software Engineering - Volume 1*, ICSE '15, pages 688–699, Piscataway, NJ, USA, 2015. IEEE Press.

[35] Wenlu Yang, Maria Rifqi, Christophe Marsala, and Andrea Pinna. Towards better understanding of player's game experience. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, ICMR '18, pages 442–449, New York, NY, USA, 2018. ACM.

[36] ScpToolkit windows driver and xinput wrapper for sony dualshock 3/4 controllers. `https://github.com/nefarius/ScpToolkit`. Accessed: 2019-01-20.

[37] Henry Fernandez, Koji Mikami, and Kunio Kondo. Perception of difficulty in 2d platformers using graph grammars. pages 1–6, 09 2017.

[38] Kusno Prasetya and Zheng da Wu. Artificial neural network for bot detection system in mmogs. In *Proceedings of the 9th Annual Workshop on Network and Systems Support for Games*, NetGames '10, pages 16:1–16:2, 2010.

[39] Xun Li and Risto Miikkulainen. Opponent modeling and exploitation in poker using evolved recurrent neural networks. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 189–196, 2018.

[40] Hwanjo Yu, Jiong Yang, and Jiawei Han. Classifying large data sets using svms with hierarchical clusters. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 306–315, 2003.

[41] Julian Frommel, Claudia Schrader, and Michael Weber. Towards emotion-based adaptive games: Emotion recognition via input and performance features. In *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play*, CHI PLAY '18, pages 173–185, 2018.

[42] David Sharek and Eric Wiebe. Measuring video game engagement through the cognitive and affective dimensions. *Simulation & Gaming*, 45:569–592, 01 2014.

[43] David Sharek and Eric Wiebe. Investigating real-time predictors of engagement: Implications for adaptive videogames and online training. *Int. J. Gaming Comput. Mediat. Simul.*, 7(1):20–37, January 2015.

[44] Claude Sammut and Geoffrey I. Webb, editors. *Leave-One-Out Cross-Validation*, pages 600–601. Springer US, Boston, MA, 2010.

[45] Hunick Robin. The case for dynamic difficulty adjustment in games. In *Proceedings of ACE '05*, pages 429–433. ACM New York, NY, USA ©2005, June 2005.

[46] Andrew Rollings and Ernest. Adams. *Andrew Rollings and Ernest Adams on Game Design*. New Riders, 201 West 103rd Street, Indianapolis, Indiana 46290 An Imprint of Pearson Education, 2003.

[47] Nacke Lennart and Lindley Craig A. Flow and immersion in first-person shooters: measuring the player's gameplay experience. In *Proceedings of Future Play '08*, pages 81–88. ACM New York, NY, USA ©2008, November 2008.

[48] Cox Anna L., Cairns Paul, Shah Pari, and Carroll Michael. Not doing but thinking: The role of challenge in the gaming experience. In *Proceedings of CHI '12*, pages 5–10. ACM New York, NY, USA ©2012, May 2012.

[49] Nordin A. Imran, Ali Jaron, Animashaun Aishat, Asch Josh, Adams Josh, and Cairns Paul. Attention, time perception and immersion in games. In *Proceedings of CHI EA '13*, pages 1089–1094. ACM New York, NY, USA ©2013, April 2013.

[50] Janet Murray. *Hamlet on the Holodeck: The Future of Narrative in Cyberspace*. Cambridge, MA: The MIT Press, 1997.

[51] Lennart E. Nacke, Anders Drachen, Kai Kuikkaniemi, Joerg Niesenhaus, Hannu J. Korhonen, Wouter M. van den Hoogen, Karolien Poels, Wijnand A. IJsselsteijn, and Yvonne A. W. de Kort. Playability and player experience research. In *Proceedings of DiGRA '09*, 2009.

[52] Christopher Perdersen, Julian Togelius, and Georgios N. Yannakakis. Modeling player experience for content creation. *Computational Intelligence and AI in Games, IEEE Transactions*, 2(1):54–67, 2010.

[53] Gillian Smith, Mike Treanor, Jim Whitehead, and Michael Mateas. Rhythm based level generation for 2d platformers. In *Proceedings of FDG '09*, pages 175–182. ACM New York, NY, USA ©2009, April 2009.

[54] Paul Coulton, Carlos García Wylie, and Will Bamford. Brain interaction for mobile games. In *Proceedings of MindTrek '11*, pages 37–44. ACM New York, NY, USA ©2011, September 2011.

[55] Kenneth Chan, Koji Mikami, and Kunio Kondo. Measuring interest in linear single player fps games. In *Proceedings of SA '10*. ACM New York, NY, USA ©2010, December 2010.

[56] Rina R. Wehbe, Dennis L. Kappen, David Rojas, Matthias Klauser, Bill Kapralos, and Lennart E. Nacke. Eeg-based assessment of video and in-game learning. In *Proceedings of CHI EA '13*, pages 667–672. ACM New York, NY, USA ©2013, April 2013.

[57] Fabien Lotte. Brain-computer interfaces for 3d games: hype or hope? In *Proceedings of FDG '11*, pages 325–327. ACM New York, NY, USA ©2011, June 2011.

[58] Kyrgyzov Olexiy and Souloumiac Antoine. Adaptive eeg artifact rejection for cognitive games. In *Proceedings of ICMI '12*, pages 567–570. ACM New York, NY, USA ©2010, October 2012.

[59] Johnson Daniel, Wyeth Peta, Clark Madison, and Watling Christopher. Cooperative game play with avatars and agents: Differences in brain activity and the experience of play. In *Proceedings of CHI '15*, pages 3721–3730. ACM New York, NY, USA ©2015, April 2015.

[60] Hou Xiyuan and Sourina Olga. Emotion-enabled haptic-based serious game for post stroke rehabilitation. In *Proceedings of VRST '13*, pages 31–34. ACM New York, NY, USA ©2013, October 2013.

[61] Inc. NeuroSky. Neurosky.com, (2015). biosensors and algorithms — ecg — eeg — neurosky, 2015. [Online; accessed 26-January-2016].

[62] Grierson Mick and Kiefer Chris. Better brain interfacing for the masses: progress in event-related potential detection using commercial brain computer interfaces. In *Proceedings of CHI EA '11*, pages 1681–1686. ACM New York, NY, USA ©2011, May 2011.

[63] Genaro Rebolledo-Mendez, Ian Dunwell, Erika A. Martinez-Miron, Maria Dolores Vargas-Cerdan, Sara Freitas, Fotis Liarokapis, and Alma R. Gacria-Gaona.

Assessing neurosky's usability to detect attention levels in an assessment exercise. In *Proceedings of HCI Part I '09*, pages 149–158. Springer-Verlag Berlin, Heidelberg ©2009, July 2009.

[64] Afergan Daniel, Peck Evan M., Solovey Erin T., Jenkins Andrew, Hincks Samuel W., Brown Eli T., Chang Remco, and Jacob Robert J.K. Dynamic difficulty using brain metrics of workload. In *Proceedings of CHI '14*, pages 3797–3806. ACM New York, NY, USA ©2014, April 2014.

[65] Noor Shaker, Julian Togelius, and Mark J. Nelson. *Procedural Content Generation in Games*. Springer International Publishing, Switzerland, 2016.

[66] Julian Togelius and Georgios N. Yannakakis. Experience-driven procedural content generation. *IEEE Transactions on Affective Computing*, 2(3):147–161, 2011.

[67] Wikipedia. Dynamic game difficulty balancing, 2012. [Online; accessed 18-January-2016].

[68] Blanchard Gilles and Blankertz Benjamin. Bci competition 2003-data set iia: spatial patterns of self-controlled brain rhythm modulations. *Biomedical Engineering, IEEE Transactions*, 51(6):1062–1066, 2004.

[69] P. J. McCullagh, M. P. Ware, and G. Lightbody. Brain computer interfaces for inclusion. In *Proceedings of AH '10*. ACM New York, NY, USA ©2013, April 2010.

[70] Nintendo. Super Mario Bros, 1985.

[71] Adrian Moran and Shane M. Murphy. *The Oxford Handbook of Sport and Performance Psychology: Concentration: Attention and Performance*. Oxford University Press, 198 Madison Avenue, New York, NY 10016, 2012.

[72] David Adams. *Automatic Generation of Dungeons for Computer Games*. University of Sheffield, UK, 2002.

[73] Roland Van Der Linden, Ricardo Lopes, and Rafael Bidarra. Procedural generation of dungeons. In *IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES 2014*, pages 78–79, 2014.

[74] Josep Valls-Vargas, Jichen Zhu, and Santiago Ontanon. Graph grammar-based controllable generation of puzzles for a learning game about parallel programming. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*. ACM New York, NY, USA ©2017, August 2017.

[75] Joris Dormans. Adventures in level design: generating missions and spaces for action adventure games. In *PCGames '10 Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, June 2010.

[76] Joris Dormans and Sander Bakkes. Generating missions and spaces for adaptable play experiences. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(3):216–228, 2011.

[77] Roland Van Der Linden, Ricardo Lopes, and Rafael Bidarra. Designing procedurally generated levels. In *2013 AIIDE Workshop*, pages 78–79, 2014.

[78] Ernestn Adams. *Fundamentals of Game Design, Second Edition*. Pearson Education Inc, New Riders 1249 Eighth Street Berkeley, CA 94710, 2010.

[79] Mohammad M. Khajah, Brett D. Roads, and Robert V. Lindsey. Designing engaging games using bayesian optimization. In *CHI '16 Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 5571–5582, 2016.

[80] María V. Aponte, Guillaume Levieux, and Stephane Natkin. Measuring the level of difficulty in single player video games. *Entertainment Computing, 2011*, 2(4):205–213, 2011.

[81] James Fraser, Michael Katchabaw, and Robert E. Mercer. A methodological approach to identifying and quantifying video game difficulty factors. *Entertainment Computing, 2014*, 2014.