Michael Steurer, BSc.

A Framework for Player Profiling and Adaptation in a Jump'n'Run Game

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Computer Science

submitted to

Graz University of Technology

Supervisor

Ass. Prof. Dipl.-Ing. Dr.techn. Johanna Pirker, BSc

Institute for Interactive Systems and Data Science
Head: Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt

Graz, May 2022

Affidavit

Abstract

The dynamic nature of differences between players cannot be mirrored by static game behaviour. There is a need to balance out these disparities. Taking into account the players' cultural differences, personality variations, or personal impairments, it is possible to widen the audience of games. In addition, games with a personal touch may make players feel more connected to these experiences. Personalization can benefit from technology's increasing prevalence since it aids in better understanding of each individual.

The goal of this project is to develop a framework for analysing player behaviour and creating player profiles. The game created for this purpose has a parameterized procedural level generation that allows for different play styles to be adopted. As part of the game, the *smart enemy* is aimed at challenging the player. It is trained using self-play reinforcement learning. The relationship between player behaviour in the game, personality, and game experience is examined. Personality prediction and the capacity to extract player profiles directly from data enables for a categorization into different player types. This work not only constructs player models, but also provides insight into methods for game adaptation based on these models.

In order to acquire behavioural data, a user study was conducted. Given the small sample size, this paper only gives a preliminary analysis. As a consequence, the results will need to be confirmed by using a more substantial sample size. The results suggest that the way players interact with the games is related to their personality. Due to data limitations and model complexity, personality estimation has performed poorly. Player profiles were derived directly from the data, and labels could be assigned to these player groups.

Contents

List of Figures

List of Algorithms

1 Introduction

Everybody is unique and differs from one another in behaviour. This makes people exert unique preferences and also have unique styles of play in the game context (Van Lankveld et al., 2011). While some of these differences can come from cultural differences (Bialas, Tekofsky, and Spronck, 2014), or exist due to impairment (Torrente Vigil et al., 2014), there are many models that attempt to describe these differences in general (Yannakakis and Togelius, 2018). Personalization can be aimed at counterbalancing these differences and grant each individual a tailor-made experience, which also makes games more personal (Lopes and Bidarra, 2011). In contrast to the idea of personalization, classical games have been kept relatively static to keep their complexity manageable and less prone to errors (Lopes and Bidarra, 2011). Personalization of games has been shown to have benefits, such as having a positive impact on productivity or satisfaction (Bakkes, Tan, and Pisan, 2012). Personalization craves for personal information and also needs a lot of content adapted to fit various players.

This work aims at constructing a framework to analyse player behaviour in a 2D platformer game. Through behavioural data analysis this work attempts to explore the relationship between in-game behaviour and personality. Additionally, it explores non-psychological methods for player behaviour estimation. From this knowledge, an adaptation of the underlying algorithms of the game is explored.

1.1 Research Goals

The main goal of this work lies in the creation of a game that can detect and construct player profiles through player behaviour. Thus, a framework is developed that applies questionnaires regarding demographics, personality, and experience alongside an implementation of a game. The main contributions of this work are as follows:

- Building a framework that can apply various methodologies to assess people's behaviour while playing.
- Investigation of various methodologies for player profile estimation.
- Exploration of game adaptation based on player profiles.

On top of these contributions the work also contains an outlook on how to adapt the game content of the game developed. These methods can personalize the game to better fit the players' needs and expectations.

1.2 Structure and Methodology

The theoretical background chapter discusses various methods for understanding and modelling player behaviour. Psychological models such as personality and affect are discussed, theoretical game-behavioural models, and the data-based machine learning approach are discussed. Subsequently, a description of game adaptation explains methodologies and application in games and game research. The final element in the behaviour-adaptation pipeline is procedural content generation (PCG). The chapter then outlines how PCG works and explains methodologies that are used to construct various types of game content. The final component in the background chapter discusses the construction of a game component via self-play reinforcement learning.
Chapter 3 specifies the construction of a framework consisting of questionnaires and a 2D platformer game. The game includes a level generation and an analytics module. Furthermore, the chapter discusses the data-server architecture and the data analysis toolkit used.
The preliminary analysis in Chapter 3 explores the results of the user study. It attempts to uncover player profiles from the player data gathered by the developed game.
The adaptation chapter proposes methods for game-based personalisation by using the learnings obtained in the preliminary analysis. It uses player

profiles to adapt the game.
The lessons learned chapter gives an overview of the main lessons learned during each chapter of this work. The second last chapter discusses future directions of research based on the work presented in this thesis. The last chapter concludes this work with an overall overview of the thesis alongside with the implications of the results.

2 Background and Related Work

As the most lucrative entertainment industry, the gaming industry is generating a high interest in game development (Richter, 2021). Alongside with the market, the number of players in 2021 has risen to 2.9 million players across the globe (Wijman, 2021). To stay relevant, game companies need to explore how to make their games more interesting, accessible and fun to their audience.
The COVID-19 pandemic also contributed its fair share to an increase of people playing games, as they were forced to stay home during lockdowns. The pandemic contributed to a 36% increase among internet users worldwide (Statista, 2021). With more players coming online, game audiences continue to grow in size and diversity of player types. New players contribute their own play style, interests, along with their cultural and linguistic qualities. Personalization acts as a way of making games accessible to this new audience. The need for personalization is further emphasized by the fact that among cultures, there are different requirements and assumptions that need to be met. For example, colours can differ significantly in their meanings (Hupka et al., 1997). To cope with this need for variety, methods like Procedural Content Generation (PCG) and player modelling allow game developers to create more variable game content. This type of content can be uniquely tailored to fit each player.
The rising omnipresence of the internet and exposure to social media facilitates the sharing of personal information beyond one's own intent. Furthermore, as technology becomes more ubiquitous, the possibility for data collection expands into new areas, such as the human body. These two phenomena give rise to collecting enough data to create systems that are able to understand and mimic human interaction through affect, personality, and it's context (Vinciarelli and Mohammadi, 2014).

Out of this, a two-step process of assessing the player and then adapting the game to better fit the player arises. Assessment of the player can be based on a psychological basis like personality or affect. Two of which are discussed in the initial section. Consequently, the discussion picks up game-based methodologies to give an overview on approaches utilized in game research. Together, player modelling opens up the possibility to adapt games by applying the inferred behavioural information of the player. Procedural content generation then bridges to the generation of various content in games. Whereas the emphasis lies on building a 2D platformer. Finally, the last section gives a brief overview into reinforcement learning, where the main focus of discussion lies on training an opponent for the game.

2.1 Psychology-Based Player Modelling

The psychological basis of human decision-making in games has different roots. This section investigates different origins of these influences on human choices. It is structured into an initial overview describing the theoretical aspects, followed by an overview of game-based literature. First, personality psychology focuses on the inherent qualities that stay relatively consistent throughout a lifetime. Second, affect describes the aspects that can appear temporarily and thus also have an impact on how the players interact with the game. Lastly, the game itself can have an effect on the players, which ensues influence in their decisions.

2.1.1 Personality Theory

Personality theorists concern themselves with the idea of human uniqueness. They investigate what characteristics make humans similar to or, on the contrary, dissimilar to each other. The objective of a definition of personality lies in capturing these similar characteristics and to specify a relationship between a person's actions and them. D. Schultz and S. Schultz (2016) define Personality as *"the unique, relatively enduring internal and external aspects of a person's character that influence behaviour in different situations"*. This definition grounds the causal relationship between personality and a person's

behaviour. But on the other hand, as Vinciarelli and Mohammadi (2014) remark, the inverse connection also exists and allows for prediction of future behaviour of people. Due to its attribution of stability, the explanatory power stays consistent across the human lifespan.
A combination of one's surrounding environment alongside inherited genetical components give rise to personality. The environmental effect is supported by observing the differences in siblings growing up in the same family, and the heretical influence has been corroborated through twin studies (Caspi, Roberts, and Shiner, 2005).

Over the course of history, different theories establishing an image of how personality emerges were developed. Sigmund Freud formulated a primal theory of personality while observing his patient's behaviour in relation to the unconscious mind, originating the *Psychoanalytic Theory*. Although Freud formulated his theory earlier on, the formal beginning of the personality field of study is considered the work of Allport (1937). The *Behavioural Theory* sees personality emerge from the interaction of an individual with its environment. The overt behaviour is explained by investigating the link to external stimuli. Each individual learns a conforming mapping between these actions and their trigger and adapts one's behaviour accordingly. The *Humanistic Theory* emerged in contrast to the Freudian concept. The humanists share a positive image of humans, guided by free will they strive for self-actualization. The *Cognitive Theory* emphasizes the influence of all cognitive processes on personality. Therefore, emotions, needs and drives are considered components of personality. In the *Social Learning Theory*, homologous to the behavioural theory, they learn by anticipating rewards through observing behaviour from other individuals (D. Schultz and S. Schultz, 2016). *Trait Theory* takes a different framework for defining personality. It is concerned with the terminology that is used to describe what humans are like. It examines these terms to derive a personality description. By looking at the similarity of their meanings, these terms are combined and grouped into a set of distinct categories (Matthews, Deary, and Whiteman, 2003). Trait theory stands out as the most prominent one among these theories. Moreover, it is also the most accurate theory for predicting human behaviour (Vinciarelli and Mohammadi, 2014).
The act of grouping of individuals with similar dispositions into various classes has a long history. In Ancient Greece, where Hippocrates and Galen

used the fluids of the human body to infer the four temperaments sanguine, melancholic, choleric and phlegmatic to people (Dumont, 2010).

Personality exerts on many parts of the human life, for instance career choice (Tett and Burnett, 2003) and burnout (Swider and Zimmerman, 2010). For evaluation of an officer's performance in the military (McCormack and Mellor, 2002). Moreover, personality correlates to success of athletes in sports (Allen, Greenlees, and M. Jones, 2013), a person's choice of food (Keller and Siegrist, 2015), behaviour and usage of social media platforms such as Facebook (K. Moore and McElroy, 2012), the academic performance of students (Poropat, 2009), and playing patterns (Worth and Book, 2015) and preferences (Peever, Johnson, and Gardner, 2012) in video games and also in board games (Bar and Otterbring, 2021).
Finally, these examples underline how understanding one's personality offers many possibilities for predicting future behaviour, ranging from big decisions in life to smaller aspects like playing video games. As a result, if measured correctly, personality presents a solid predictor of prospective overt behaviour.

Personality Assessment

Personality assessment encompasses various methods for the determination of a person's personality. Aside from the various methodologies that have been created by psychologists, two conditions must be met for new tests to be considered persuasive. Reliability postulates that the result of two subsequent tests, taken a week apart, have to be consistent. Additionally, for them to be valid, the assessed items need to match the correct theoretical constructs that were intended (D. Schultz and S. Schultz, 2016).

- **Self-Reporting**: In Self-Reporting, individuals are given a series of questions about their emotions or actions that they have to answer themselves. The results of self-reporting tests can be inaccurate, considering that they rely on the accurate and thorough elaboration of the presented questionnaire. The participants may also possess a self-image differing from their true personality or wilfully attempt to distort the outcome to fit their own envisioned intentions (D. Schultz and S. Schultz, 2016).

- **Projective Techniques**: These methods explore the unconscious mind to deduce personality. The assessed individual is presented with ambiguous stimuli, for instance the images of the Rorschach Inkblot Test elicit an expression of personality (Boyle, Matthews, and Saklofske, 2008).
- **Behavioural Assessment**: Behaviour in a situational context is depending on one's personality and thus, it can be used as evidence on inferring the personality of a person (D. Schultz and S. Schultz, 2016).
- **Data Mining**: Besides these theoretical and corroborated methods based on psychology, there are a number of approaches that utilize data mining. These new possibilities arise from the increased availability of personal information available, as well as the pervasiveness of personal devices (Vinciarelli and Mohammadi, 2014). For instance, in a video game context it has been shown that gameplay (Van Lankveld et al., 2011), game attributes (Halim et al., 2017), and play style (Tekofsky et al., 2013; Bean and Groth-Marnat, 2016) are factors correlating to personality. Mohammadi and Vinciarelli (2012) used extracted prosodic features of audio samples to estimate personality of persons, Majumder et al. (2017) investigated the applicability of utilizing plain text, and Subramanian et al. (2016) explored physiological signals and correlated data generated to personality.

Adjacent to a personality test, Psychologists utilize clinical interviews to corroborate the results of the test by consolidating supplementary features like the physical behaviour of a person (D. Schultz and S. Schultz, 2016). Moreover, Boyle, Matthews, and Saklofske (2008) highlight multi-method measurements as another tool to overcome inconsistencies that can materialize amid a singular personality test. Among these methods, self-reporting is the most frequent one. It is used in many of the personality inventories discussed in the subsequent Section.

Personality Inventories

Personality inventories provide categorizations and methods for estimating personality. They are backed by various types or a single questionnaire that usually takes the form of self-reporting or third-party reporting. Different groups of psychologists established various personality inventories, which they based either on lexical studies of words describing personality, human physiology and genetics or emerging from other fields in psychology, for instance psychopathy. A common denominator among many of these models is the consensus of around five factors (Goldberg, 1993).

- **Big Five Inventory**: The Big Five Inventory defines a hierarchical model with five factors on top. Each one of these factors consists of a set of smaller factors, called facets, to support a more in-depth description of a person's personality (Soto and John, 2017b). The most frequent technique used for estimation is called NEO (McCrae, Costa, and Martin, 2005).
- **Sixteen Personality Factor Questionnaire (16PF)**: The 16PF is a multi-level trait measure with 16 primary scales and five global scales that are similar to the Big Five factors. While the global scales provide a more general overview of a person's personality, the primary scales yield a more detailed audit of personality. The 16PF inventory strongly correlates to the Big Five Inventory. 16PF encompasses questions about everyday behaviour in an effort to mitigate distortions and thus making it more accurate (Cattell and Mead, 2008).
- **Eysenck Personality Questionnaire (EPQ)**: The Eysenck Personality Inventory consists of the three superfactors Extraversion, Neuroticism and Psychoticism supported by a further Lie scale. These scales are subdividable into a list of more precise traits. The EPQ-R, the EPQR and the EPQR-A are examples for Questionnaires (Francis, L. Brown, and Philipchalk, 1992).
- **Minnesota Multiphasic Personality Inventory (MMPI/MMPI-2)**: This inventory detects psychopathy alongside with personality. It includes scales for estimating the subject's willingness for cooperation to counteract the deliberate distortion. It has been shown that it is robustness across different languages and cultures (Butcher, 2010).

Name	Objective	Susceptibility assessment	Publication
BFI	Determination of five primary factors and secondary facets.	-	McCrae and Costa Jr (2008)
16PF	Determination of 16 primary factors and five secondary global scales.	Yes	Cattell and Mead (2008)
EPQ	Three superfactors supported by a lie scale.	Yes	H. Eysenck and S. Eysenck (1984)
MMPI or MMPI-2	Detection of Psychopathy and Personality	-	Butcher (2010)
MBTI	Assign one of 16 distinct personality categories.	-	Myers (1962)

Table 2.1: Comparison between the personality inventories listed in this work.

- **Myers Briggs Type Indicator (MBTI)**: The MBTI consists of 16 unique personality categories represented by four-letter-combinations. These combinations consist of four type pairs with four possible values. The MBTI is widely critiqued and should therefore be applied with caution in mind (Pittenger, 2005).

The most popular method of estimating personality is trait theory alongside with the Big Five Factor Inventory, which will be further elaborated in the next Section. Although, the Big Five are the predominant model also in game research, Worth and Book (2015) remark that the HEXACO model has been successfully used for inferring player-behaviour in World of Warcraft (Blizzard Entertainment, 2005).

2.1.2 Five Factor Inventory (FFI)

The Five Factor Inventory is based in trait theory, where personality is described with the help of traits. That is, by collecting words in the human language that are used to describe personal qualities of humans and breaking them down into a few categories of distinct classes. Three preconditions led to the rise of trait theory. Namely, means for systematic data collection, methods for statistical data analysis and techniques to establish and verify these new theories (Matthews, Deary, and Whiteman, 2003). Thereby, lexical studies allowed to condense the list of these terms down into the five categories that formed the Big Five Factors. This theory has distilled as the most effective at predicting behaviour (Vinciarelli and Mohammadi, 2014), although it is not based on a general accepted scientific theory (Matthews, Deary, and Whiteman, 2003).

Sir Francis Galton formulated the lexical hypothesis of trait theory. His work led to a systematic collection of terms describing personality, leading to the eventual discovery of the Big Five factors. Goldberg (1993) mentions Donald Fiske as the accidental discoverer of the Big Five. Through variable analysis, he uncovered five factors across different tests, but never follow up on them. Goldberg (1993) appointed Tupes and Christal as the genuine finders. They analysed studies for the U.S. Air Force, where they likewise observed and these five replicable factors.

Since the inception of trait theory, one of the fundamental questions concerns with the number of items used. Among the different inventories, HEXACO is composed of six dimensions, the MBTI has 16 primary and five global dimensions and Eysenck's Personality Questionnaire features four dimensions. According to Goldberg (1993) and Digman (1990) most of these methods can either be reduced or expanded into a model with five factors and thus the consensus revolves in having five distinct factors. As such, the Five-Factor model of personality (FFM) has established as the prevalent paradigm in describing human personality traits (Costa Jr and McCrae, 1995). The common nomenclature, after Costa Jr and McCrae (2008a), uses the terms Neuroticism, Extraversion, Openness, Agreeableness, Conscientiousness and as high level traits. *Neuroticism* considers as emotional stability or the likelihood of experiencing negative affect (Digman, 1990). *Extraversion* manifests as the enjoyment of social stimulation. *Openness* describes a person's desire for variety, novelty, and curiosity. *Agreeableness* is the willingness to defer one's own interest for the benefit of others. *Conscientiousness* presents as the level of aspiration to reach one's goals (Costa Jr and McCrae, 2008a). Each one of these traits is composed of lower dimensional traits, also known as facets, which allow to assess a more explicit evaluation of personality.

The Big 5 have been found to be consistent across a wide range of countries and cultures globally. They are denominated by McCrae and Costa Jr (1997) as a cross-cultural *"common human structure of personality"*.
Cross-cultural studies have also investigated on gender differences, where they have shown that the more prosperous a nation, the more dissimilar the personalities between genders become. A reason for that is that in less developed and more traditional settings, personalities are restrained by environmental pressures. With new advancements these restraints are eased and personality can flourish (Schmitt et al., 2008).
Longitudinal research has found the traits stay relatively consistent over a lifetime (D. Schultz and S. Schultz, 2016). According to Costa Jr, McCrae, and Löckenhoff (2019) the changes are minimal. Neuroticism and Extraversion decline, and Agreeableness and Conscientiousness increase with age. Openness on the other hand first rises, and later in life decreases again Costa Jr, McCrae, and Löckenhoff (2019).
Both, genetics and the environment have a distinct influence in the formation of the Big Five Factors. Neuroticism, Extraversion, Openness, and Conscientiousness have been found to exhibit a stronger hereditary influence, while Agreeableness shares a higher environmental significance (D. Schultz and S. Schultz, 2016).

Questionnaires

Self-reporting questionnaires are organized with rating scales like the Likert scale (Boyle, Matthews, and Saklofske, 2008). NEO is the most frequent technique of measuring the Big Five factors (D. Schultz and S. Schultz, 2016). The most comprehensive questionnaire is NEO-PI-3 (McCrae, Costa, and Martin, 2005), which consists of 240 items. It measures the Five Factors and 30 further traits. NEO-PI-3 is an advancement of NEO-PI-R (Costa Jr and McCrae, 2008b), thus the results of both are considered equivalent (McCrae, Costa, and Martin, 2005). Both questionnaires take about 30 minutes of time to complete. McCrae and Costa (2007) note that the NEO-FFI-3 is intended as a shorter alternative to NEO-PI-3 consisting of only 60 items and intended solely for assessment of the five primary factors. Moreover, the first half of the NEO-PI-3 (NEO-PI-3FH) can be used as a questionnaire itself, which utilizes 120 items, and promises estimates of the Big Five Factors alongside

Name	Number of Items	Measurements
NEO-PI-R NEO-PI-3	240	Five Factors and 30 facets
NEO-FFI-3	60	Five Factors
NEO-FFI-3FH	120	Five Factors and facets
BFI-44	44	Five Factors
BFI-2	60	Five Factors and facets
BFI-10	10	Five Factors
BFI-2-S	30	Five Factors and facets
BFI-2-XS	15	Five Factors

Table 2.2: Comparison of various Questionnaires for measuring the Big Five factors with number of items and their intended measurements.

with its facets.
The Big Five Inventory (BFI-44) (John, Donahue, and Kentle, 1991) is an alternative to NEO. The BFI-44 is comprised of 44 items and designed for completion in 5 to 10 minutes. Subsequently, the BFI-2 (Soto and John, 2017b) is an advancement, and consists of 60 items and is achievable in 10 minutes or less. The BFI-10 (Rammstedt and John, 2007) condensed the BFI-44 into 10 items and is used for participants who only have a very narrow timeframe for their assessment. It affirms that it takes one minute of time or less. The BFI-2-S and the BFI-2-XS are short versions of the BFI-2, having 30 and 15 items respectively. BFI-2-S can be used to examine facets, while BFI-2-XS is too brief for that (Soto and John, 2017a).
As self-reports, these questionnaires suffer from the previous noted issues of distortion. With game based personality assessment better cost-effectiveness, and higher reliability Tekofsky et al. (2013) can be reached, while also being fun.

FFI in Games Research

The following works have shown that different parts of games can be used for personality profiling. In order to assess the personalities of the players, the researchers utilized self-report questionnaires.
Van Lankveld et al. (2011) probed whether gameplay behavioural data of the

role-playing game Neverwinter Nights (BioWare, Obsidian Entertainment, 2002) infers personality. The game contains NPC conversations alongside storytelling elements. Trough correlational analysis, they identified influences of all Five Factors on the gameplay and found Openness as the most significant factor. The authors conclude that games are adequate for personality profiling.
Tekofsky et al. (2013) analysed play styles in Battlefield 3 and correlated them to the personality traits. They found that Conscientiousness negatively correlates with speed of action. Moreover, Conscientiousness and Extraversion correlates with unlock score per second.
Halim et al. (2017) combined the behavioural data from the Real-Time-Strategy games Star-Craft (Blizzard Entertainment, 1998), Age of Empire II (Ensemble Studios, 1999), and World of Warcraft (Blizzard Entertainment, 2005). They ascertained relationships between attributes of the game and the Big Five Personality Traits. Technology, military, society indicated high Neuroticism, Extraversion, Openness, and low Agreeableness, Conscientiousness. Trade profit, feudal age, and castle age correlated with high Conscientiousness and Agreeableness.
Bean and Groth-Marnat (2016) examined the relationship between the BFI-44 and play styles in the game World of Warcraft (Blizzard Entertainment, 2005), a game played in multiplayer and thus offering social aspects among players. They ascertained that Player versus Player (PVP) players exhibit higher Extraversion and lower Neuroticism scores, Player versus Environment (PVE) less Agreeableness and Openness with increased Conscientiousness, and Role-play (RP) manifests in higher Agreeableness, Neuroticism, Openness and lower Extraversion and Conscientiousness. Although, they found WoW players to be less neurotic, their personality is similar to that of non-players.
Worth and Book (2015) used the HEXACO model for finding correlations between game behaviour and personality in general. Through a principal component analysis of the gameplay data, they uncovered Aggressing, Winning, Creating, and Helping as behavioural factors. They were able to negatively correlate Aggressing with Honesty-Humility, Helping positively with Agreeableness, and Creating negatively with Conscientiousness.

The effects of personality not only expresses itself through behaviour in games, it already starts with the choice of game. Peever, Johnson, and Gardner (2012), Braun et al. (2016), and Johnson and Gardner (2010) have shown

that players choose their type of game in accordance to their personality. Peever, Johnson, and Gardner (2012) found people with a dominant Extraversion trait to be attracted to casual music and party games. People with lower scores of Extraversion enjoy engaging in games that are often times played alone, for instance strategy, role-playing, and MMORPG games. Conscientious people engage in task oriented game genres, for example sport, racing simulation and fighting games. Openness has been associated with exploration games like action adventures.
In the work of Braun et al. (2016), they associated high Extraversion and low Neuroticism with action games, Openness with role-playing games, and high Conscientiousness with simulation games.
(Johnson and Gardner, 2010) found that people with high Emotional Stability are more likely to play violent games.

Besides the static qualities of personality steering a person's decisions and actions, affect is a quality which can have a temporal effect on the course of a game.

2.1.3 Affect and Effect

Affect considers the positive or negative internal reaction that an individual can display, elicited as a response to novel or past cognitions. These can originate as a consequence of old memories, the present environment, or hypothetical situations. Besides cognition, learned concepts and knowledge can play a part. Affect is a generic term encompassing both, emotions and moods, where emotions describe responses to actions, or events (Wyer Jr, Clore, and Isbell, 1999).

Categorizing emotions is the most basic model of emotions. It places each emotion into a distinct category, discarding the full range of emotions by restricting their continuity. Multidimensional approaches go beyond simple categorization by defining emotions as points in a multi-dimensional space, for example Plutchik's wheel of emotions. Again, the model has its drawbacks, namely it lacks the possibility of simultaneous emotions. The hourglass of emotions goes beyond both of the previous approaches and utilizes four affective dimensions alongside with labels for modelling emotions (Cambria, A. Livingstone, and Hussain, 2012). It allows for expression of four simultaneous emotions (Poria et al., 2017).

Affect Detection

Affect serves as a further method to better understand a person's behaviour. Furthermore, it plays a role in games where a positive experience is essential. For instance, in serious games, the attention and active participation is important (Anolli et al., 2010). Through affect detection, emotions like confusion, boredom, or frustration can be mitigated and an improved learning performance is achievable (DeFalco et al., 2018).

In the work of Anolli et al. (2010) a more in-depth review of self-report measures for emotion assessment can be found, two of which are the Profile of Mood States (POMS) or the Expanded Form of the Positive and Negative Affect Schedule (PANAS-X) measure. These types of self-reports can prove as unreliable due to either intentional distortion or inability to express emotional feelings correctly.
In regard to gaming, the Game Experience Questionnaire (IJsselsteijn, Kort, and Poels, 2013) measures positive and negative affect among other features. It will be discussed in detail in a later Section.

Yannakakis and Paiva (2014) take a data-mining-based approach to affect detection. Consequently, they distinguish between objective data, gameplay data and game context as input feature types.

- **Objective data** regards responses of the body as a method for affect detection. Visual features, such as facial expressions, head posture, gaze and body posture, can be recognized via motion tracking and are analysed for emotion expressions (Yannakakis and Paiva, 2014). It is important to also regard the temporal dynamics of these features. The face designates the most expressive cue (Poria et al., 2017). Besides visual features, vocal features convey further manifestations of emotion. It is composed by its linguistic and paralinguistic components. While the latter has evolved further and is less complex in practice, the former, due to the complexity of real-time text extraction and cultural influences, presents more challenges (Zeng et al., 2008). Additional sensors can further deepen the insights into the human body. Physiological signals, for instance respiration, skin conductivity or heart rate, carry further information regarding emotions (Koelstra et al., 2011). These physiological signals are prone to cheating, as fooling them can be as easy as taking a quick jog in advance to playing Gilleade and Dix (2004).

- **Gameplay data** designates the behavioural impact the player has on the game during play. As Yannakakis and Paiva (2014) put it, if game experiences can influence cognitive processes of a player and consequently the cognition influences emotions, then game interactions may correlate to the game input of the player. Thus, any player behaviour and its spatio-temporal properties may infer emotions.
- **Game Context** plays a further important part in the measurement. It has an effect on the emotions expressed by the player. Thus, it is important to view situational features of the game, such as the audiovisual context, player-agent interactions or the gameplay, as an additional factor of influence (Yannakakis and Paiva, 2014).

Generally, Multimodal analysis, the fusion of any of the above-mentioned methods or sensors, increases accuracy and performance of the models (Poria et al., 2017). Apart from that, Anolli et al. (2010) further remark that emotion detection instruments should not influence one's behaviour and adversely affect the results. As well as increase costs or raise privacy concerns (Lester et al., 2013). Thus, obtaining objective sensory data and visual information might be more suitable to certain environments. Then again, with the advance of ubiquitous computing, some sensors become more unobtrusive and data collection feasible. Therefore, technologies like wearables, or even Gamepads (Sykes and S. Brown, 2003; Gilleade and Dix, 2004), can act as new data-sources viable for affect detection.

Besides affect detection, emotions that are elicited during gameplay can have an effect on aspects outside the game context. As a result, the debate that video games stimulate violent behaviour beyond video games arises.

Effects of Games

The discussion of aggression is a stigma in gaming and research on it has so far been divided. Anderson et al. (2010) point out that the evidence suggests that exposure to violent video games raises aggressive behaviour and reduces social behaviour and empathy. But, as Greitemeyer and Mügge (2014) add to this debate, playing violent video games is a factor among many others, of which some are more likely to stimulate violent behaviour. Moreover, by boosting prosocial content and making the game cooperative, violent video games even abate violent behaviour. Greitemeyer, Osswald, and Brauer (2010) suggest that they even increase empathy and reduce gloating. Johnson, Wyeth, et al. (2012) discuss that immersion in the game makes it more likely to elicit real world aggression. In another work, Johnson and Gardner (2010) add that players who enjoy violent games experience less immersion.

Frustration is a further negative emotion that can arise during gameplay. While it is an undesirable emotion for players, frustration can be an indicator that the player needs further aid to continue the gameplay, indicating a need for changes in the game. When detected, a game can adjust the game such that equal situations are less likely to occur in future gameplay and thus, avoid player frustration. To further understand frustration, one is obligated to distinguish between at-game and in-game frustration. At-game frustration depicts the player's inability to control the game due to faulty input-devices, lag, overly complex key combinations, or lack of skill. In-game frustration displays itself as the player's inability to defeat challenges due to excessive complexity and size of elements in game design, for instance in the world or in dialogues (Gilleade and Dix, 2004).
Pedersen, Togelius, and Yannakakis (2010) were able to find correlations between player behaviour in Super Mario Bros (Nintendo R&D4, 1985) and frustration. Frustration shows in failing to play the game, for instance continuously falling into gaps. Moreover, frustrated players spend a lot of time standing still and hypothesizing on how to overcome challenges. They also do not exhibit the needed patience for collecting collectibles.

Besides effects perceived as negative, there are also positive ones to consider. Granic, Lobel, and Engels (2014) focus on the positive effects of video

games in their paper. They found evidence that playing games has positive effects on the cognition, behaviour, emotions and social behaviour. Games teach spacial skills, problem-solving, persistence and bolster social skills. Furthermore, they serve as a tool for relaxation and alleviate stress, anxiety and depression.
In conclusion, the effects a video game have on its players relies on the content of the game and therefore it lies in the hands of the video game development studios and their developers (Gentile, 2011).

Affect in Games

Pedersen, Togelius, and Yannakakis (2010) used a parameterized version of the Super Mario Bros (Nintendo R&D4, 1985), that collects parameters of the level generation and gameplay characteristics. The collected data is used to predict the affective states fun, challenge, boringness, frustration, predictability and anxiousness.
A different approach was chosen by Lester et al. (2013). They utilized the Elliot and Pekrun model (Pekrun, Elliot, and Maier, 2009) for inferring emotions in a learning environment.
N. Wang and Marsella (2006) developed a game specifically for studying emotion evoking. They uncovered that in a series of sequentially evoked emotions, the response provoked by prior emotions affected the response of ensuing emotions.

Sundström (2005) proposed the *Affective Loop* as an end-to-end method for emotion recognition and interpretation. The players express their emotion, which is recognized by the system. The system in turn generates a suitable response for the expression, affecting the players' emotions, and eventually another emotional response by the players is elicited.
As a response, the game can conduct adaptation to gain control of the emotions evoked. Adaptation can affect any element of the game content, such as environment and design alongside auditory and visual features, game mechanics, underlying reward systems, game rules or story elements (Yannakakis and Paiva, 2014).
Gilleade and Allanson (2003) developed an SDK to inquire the utilization of an Electrocardiogram (ECG) to create affect-guided adaptation in video

games.
Gilleade and Dix (2004) discuss the susceptibility of the arcade game Tokimeki Memorial: Oshiete Your Heart (Konami, 1997), which uses pulse and sweat level of the player to influence outcome of dates in game.

Beside modelling player behaviour on a psychological basis, there are methods that consider the game-based approach.

2.2 Game-based Player Modelling

In game-based player modelling, there are two distinct methods for inferring behaviour. Model-based considers a top-down approach based on theoretical aspects, either from psychological or game research. Model-free takes the bottom-up approach of data-mining by dissecting behavioural information from player-generated data. Model-Based entails the employment of theoretical models that are either of psychological, physical or a game-behavioural origin (Yannakakis and Togelius, 2018, p.208).
A further distinction can be met regarding the complexity of player modelling. Bakkes, Spronck, and Lankveld (2012) distinguish between action models, tactical models and strategical models. Action models consider the prospective actions of the player for prediction. Tactical and strategical models, consider the short-term and long-term behaviour of the player, respectively. In regard with long-term behaviour, the complexity of the model increases, which makes modelling more computationally more challenging while at the same time also more reliable.

2.2.1 Model-Based Methods

The model-based approach takes a theoretical perspective of player modelling, combining models from psychology and game behavioural research. The psychological models like personality profiling or affect have already been discussed in the preceding Section.
An advantage of the methods listed here are that they have been developed and ascertained by experts to describe behaviour. What appears to be a

strength is also a significant disadvantage. These methods rely on expert knowledge, which if administered improperly, may lead to incorrect deductions (Hooshyar, Yousefi, and Lim, 2018). Furthermore, Yannakakis and Togelius (2018, p.212) remark that caution should be applied when applying these methods, as some of them are not intended for analysing game behaviour or only fit a single type of game genre. Nonetheless, the following models are the most commonly used models used for player modelling.

- Caillois (2001) effected a first classification of player types. He partitioned players into the classes Agon, Alea, Mimicry, Ilinx.
- **Bartle's Player Taxonomy:** It consists of four archetypes Killers, Achievers, Socializers and Explorers. (Bartle, 1996, p.211).
- **Gamer Motivation Profile:** The gamer motivation profile was developed to describe behaviour in MMORPG games. It features Achievement, Relationship, Immersion, Escapism, and Manipulation as measurable dimensions (Yee, 2006)
- **BrainHex:** The BrainHex model combines neurolgical research alongside demographical game models. It presents as the seven seperate player archetypes Seeker, Survivor, Daredevil, Mastermind, Conqueror, Socialiser, Achiever (Nacke, Bateman, and Mandryk, 2014).
- **Player Traits Model:** Developed to improve on a shortcoming of the BrainHex model's inability to model simultaneous motivations. Therefore, the player traits model places each dimension one on a continuous scale, where each player can score on each one of these aspects. The proposed scales are Aesthetic, Narrative, Goal, Social, Challenge orientations, each of which describe players that enjoy these aspects (Tondello et al., 2019).
- **Difficulty:** The difficulty is modelled to keep the player inside the flow (Nakamura and Csikszentmihalyi, 2014). An analogous method for this context is Dynamic Difficulty Adjustment (DDA) (Lopes and Bidarra, 2011).
- **Fun:** The model after Malone (1981) partitioned fun into the contributing factors challenge, curiosity and fantasy. Alternative theories of fun consider Koster (2013) and Lazzaro (2009) four factor model (Yannakakis and Togelius, 2018, p.211).
- **Psychological models:** The Models based on psychology, like personality and affect, which were already discussed in this thesis are likewise

deemed as model based methods.

- Other models include, Magerko, Stensrud, and Holt (2006) who are using curiosity, boredom, frustration, predictability, and anxiety as **motivations for play**. Vahlo and Hamari (2019) found **five reasons for playing games**, namely Relatedness, Competence, Autonomy, Fun, and Immersion.

2.2.2 Model-Free Methods

The Model-Free approach infers a behavioural mapping of the players through data mining. This method differs from the previous one in the construction of appropriate models directly from data, and is thus also referred to as bottom-up approach (Yannakakis and Togelius, 2018, p.212). This alternative process comes with a new set of challenges that need to be taken into account.

- **Complexity:** Game environments become increasingly more complex (Bakkes, Spronck, and Lankveld, 2012), leading to an abundance of data and increasing computation complexity. A separation of tasks into smaller models is a feasible alternative method to reduce this complexity (Hooshyar, Yousefi, and Lim, 2018). Charles et al. (2005) further empathizes that a trade-off between a model that is too plain and one that is too complicated has to be found for optimal performance.
- **Temporal limitations:** The amount of observations during a game session is very narrow, leaving a limited amount of behavioural information to construct models (Bakkes, Spronck, and Lankveld, 2012).
- **Partial observability:** The environment of the game can be only partially observable, reducing the amount of information obtainable (Bakkes, Spronck, and Lankveld, 2012). For instance, not the entire set of opponent parameters is perceivable by the player.
- **Generalization:** Models fail to generalize beyond the scope of the game they have been tested and developed. Furthermore, this impedes the development of models that generalize beyond single types of games (Hooshyar, Yousefi, and Lim, 2018).
- **Expert knowledge:** Model-free methods rely on experts for task definitions and model constructions (Hooshyar, Yousefi, and Lim, 2018).

- **Temporal variation:** Players exhibit play style variations over time, leading to changes in their decisions at different times Valls-Vargas, Ontanón, and J. Zhu (2015).

Besides these challenges, the bottom-up approach is able to forgo certain expert assumptions inherent to the model. It is therefore able to construct tailor-made models for describing behaviour in games. The literature uses a variety of methods for creating model-free player models, a few of which will be discussed adjacently. The discussed methods encompass classification, clustering, and regression as classes of algorithms employed. For a more comprehensive overview on each of these methods alongside with its scientific literature, the reader is referred to the work of Hooshyar, Yousefi, and Lim (2018).

- **Classification**: Pedersen, Togelius, and Yannakakis (2010) used multi-layer perceptrons to build a player model from game behavioural data. Valls-Vargas, Ontanón, and J. Zhu (2015) notes that the model does not consider the temporal dynamics of player behaviour. Additionally, they show that sequential machine learning models deliver better performance due to play styles not being static.
- **Clustering**: In the works of Bauckhage, Drachen, and Sifa (2014) and (Drachen, Thurau, et al., 2014), they compare various clustering methods alongside their up and downsides and outline when to choose which method. Drachen, Sifa, et al. (2012) performed clustering player behaviour in the games Tera (Krafton, 2011), Battlefield: Bad Company 2 (DICE, 2010). Worth and Book (2015) employed PCA on the data supplied by a game behavioural questionnaire instead of utilizing game interaction data. They related these factors aggressing, winning, creating, helping to the HEXACO model.
- **Clustering and Classification**:
  Halim et al. (2017) used behavioural datasets of the games Age of Empires II (Ensemble Studios, 1999), Starcraft (Blizzard Entertainment, 1998), World of Warcraft (Blizzard Entertainment, 2005) and employed clustering techniques revealing two clusters. Consequently, they probed various classification methods to categorize players into one of two categories. By using a support vector machine (SVM) they obtained the best results. Furthermore, by employing feature selec-

tion techniques, they distilled out the primary features. Bunian et al. (2017) used hidden Markov models to extract behavioural differences between players. Consequently, the model was used to classify players by means of their behavioural differences.
- **Clustering and Regression**: Harpstead et al. (2015) uses clustering in combination with regression models to assign profiles to players.

2.2.3 Inputs for Player Modelling

For construction of model-free models, the question arises on what types of input data can be leveraged for them. Yannakakis and Togelius (2011) distinguishes between objective data, gameplay data and the game context. Objective data considers all measurable bodily responses as they may be triggered by emotional responses to the game. Gameplay data encompasses the behaviour during gameplay, which may vary according to the player's emotional state. And lastly, the game context describes the current state of the game, which is another important factor in certain situations (Yannakakis and Togelius, 2011).
Besides these inputs, another way for measurements are game experience questionnaires.

Experience Questionnaires

The literature covers three different types of experience questionnaires.
The Game Experience Questionnaire (GEQ) consists of three parts. The initial part focuses on game experience and measures competence, sensory and imaginative immersion, flow, tension and annoyance, challenge, negative and positive affect. The second part, the social presence module, investigates the social interaction between the player and other entities in the game. The third part surveys the player's sentiment after completion of the game. The questionnaire is carried out after completion of the game (IJsselsteijn, Kort, and Poels, 2013).
The Player Experience and Need Satisfaction (PENS) (Ryan, Rigby, and Przybylski, 2006) is based on self-determination theory. It measures player experience with an emphasis on player immersion (Johnson, Gardner, and Perry,

2018).
The Immersive Experience Questionnaire (IEQ) (Jennett et al., 2008) consists of the five components competence, autonomy, relatedness, immersion and presence, and intuitive controls. Unfortunately it is not applicable for research due to copyright (Johnson, Gardner, and Perry, 2018).

As discussed in this Section, these models provide means for understanding player behaviour on a computational level. Which comprises a necessary step to successfully implement game adaptation and personalize games for players.

2.3 Game Adaptation

Traditional games are built using static game content opposed by a dynamic player. Catching up to the player by transforming the game content into a dynamic and adaptive one can have advantages. Incidentally, higher dynamics result in new challenges due to lower controllability. Moreover, testing also becomes increasingly more complex due to the higher and more variable content that needs to be reviewed for issues and bugs by the game designers (Lopes and Bidarra, 2011).
On the contrary, adaptation in games can help players with different aspects of games. In view of their unique strengths, motivations and goals, they will struggle and triumph over different aspects of the games. Adaptation can act as mediator to tailor game experiences according to the needs of each individual. This is done by adapting the game to their individual style and preferences, or by supporting them during the game. For instance, if players repeatedly fail at the same hindrance in the game, adaptation can alleviate these hindrances such that the player is less likely to become frustrated and stop playing (Charles et al., 2005).
Furthermore, it enables reaching new player demographics outside the traditional target audience by lowering the entry hurdle. Furthermore, it increases their enjoyment and makes the games more accessible for them. But, even if the players can overcome this initial impediment, they will meet another hardship as the game can prove as being considerably more demanding than expected. Again, adaptation can aid the player and ease

these hindrances (Beal et al., 2002).
A further advantage of adaptation lies in the prevention of replayable strategies. Players should not be able to exploit games by repeating past and successful strategies to beat the game with less effort than intended (Charles et al., 2005).

2.3.1 Adaptation Objective

The task of adaptation lies in first identifying why there is a need for an adaptation process. In games, this is done by constructing a measurable parameter that indicates why the player needs the game to change. Subsequently, the game needs to carry knowledge about which game-parameters can be tuned to achieve the desired effect and improving the game in favour of the player (Lopes and Bidarra, 2011). Models such as the notion of fun, described by Malone (1981), are not directly defined as mathematical expressions. They have either been developed for games in general or a specific type of game in mind. Consequently, to utilize these models, they have to be adapted and assumptions have to be made for keeping them precise (Pedersen, Togelius, and Yannakakis, 2010). Upon completion of the definition of a measure, various search algorithms can be applied for the task of optimization. Search methods include exhaustive search if the search space of the problem is narrow enough. Alternatively, heuristic search, gradient search, genetic search and particle swarm optimization present feasible methods inside a larger search space (Yannakakis and Togelius, 2011).
When constructing a measure, the intent of the game must be defined. While players in serious games are learning to improve their knowledge, players in entertainment games desire enjoyment. Their intent behind the measure should mirror this distinction.

Serious Games

In serious games, the adaptation process supports the player's learning process. Besides playing preferences, the individual learning preferences of the players are another important factor that needs to be taken into account (Lopes and Bidarra, 2011). Intelligent tutors that support players when necessary are a common practice in the field. It has been shown that tutoring increases motivational aspects of players and thus also their performance (Beal et al., 2002).

Entertainment Games

In games designed for entertainment, the purpose is different. Instead of favouring learning performance, user entertainment and enjoyment is the prevailing factor driving adaptation. Thus, some of the above discussed player behaviour models have been used for this purpose. Fun or difficulty are a common measures that used for optimization. And thus, models by Malone (1981) or Magerko, Stensrud, and Holt (2006) for instance, are commonly used in this context.
The model of Malone (1981), where he defines fun as the product of challenge and curiosity, is used by Yannakakis and Hallam (2009) to create an adaptive version of the game Bug Smasher, where they try to maximize the entertainment value. Moreover, Togelius, De Nardi, and Lucas (2007) developed a racing game where the entertainment is maximized through track adaptation. They based their approach to modelling on an analysis of their own on what makes games fun. They concluded that a mix of challenge and speed is adequate in both complexity and also sufficient regarding performance of the model.

As a tool for measurement of these parameters, game experience questionnaires are applicable.

2.3.2 Online and Offline Adaptation

The adaptation process can take place either in an online or offline modality, both of which carry their advantages and pitfalls. In offline adaptation, the assessment of the player's preferences is separated from the gameplay. For example, the player answers a questionnaire prior to playing the video game. As an alternative, online adaptation conducts behavioural assessment in real-time during gameplay. A disadvantage of online adaptation lies in data collection being rather limited due to the short nature of gameplay sessions (Bakkes, Tan, and Pisan, 2012).

2.3.3 Adaptable Game Components

Games support different components that can be used to tune the game behaviour. Both, Charles et al. (2005) and Lopes and Bidarra (2011) give a taxonomy of adaptable components of games. Collectively, the result is a division into game generation, AI, story and quests, music, and player matching in multiplayer games.

Game Generation

Game generation considers the world alongside with its objects, like quality of loot drops, variations in amount of collectibles or maps, or visual and structural variations (Charles et al., 2005; Lopes and Bidarra, 2011). Bakkes, Tan, and Pisan (2012) remark that by observing the player, games are able to respond to the player behaviour and make changes in its world. For instance, if a player avoids certain paths, the game might generate more or less realizations of these paths on purpose. These techniques can be found in the games Rogue (A.I. Design, 1980) and Diablo (Blizzard North, 1997). Charles et al. (2005) additionally note, that the game Fable (Big Blue Box, 2009) features an adaptive world design that evaluates how malevolent the player's behaviour is and adapts the landscape of the game accordingly. Left 4 Dead 2 (Valve, 2009) adapts layouts of parts of the level to ease the pathway for less skilled players (Lopes and Bidarra, 2011).
With respect to the scientific literature, Togelius, De Nardi, and Lucas (2007) developed a racing game that evolves in accordance with maximization of the user's entertainment using evolutionary algorithms. Another example marks the work of Pedersen, Togelius, and Yannakakis (2010), where a list of modelled player experience types adapts a parameterized version of Super Mario Bros (Nintendo R&D4, 1985).

Mechanics

The mechanics of the game can also be adapted to improve or impair a player's capabilities. The game Max Payne (Remedy Entertainment, Rockstar Studios, 2001) for instance adapts aim assist to match the player's skill to facilitate it for less-abled players (Lopes and Bidarra, 2011). In Mario Kart 64 (Nintendo EAD, 1996) a rubber-banding mechanic ensures that weak players are not being completely outpaced. In Crash Bandicoot: Wardped (Naughty Dog, 1998) if the player repeatedly fails, the game provides a shield to continue (Charles et al., 2005).

Game AI

The Game AI presents a further adaptable component. The most common method to adapt game AI lies in adaptation of its NPCs. For instance, rubber banding in Mario Kart Wii (Nintendo EAD, 2008) counteracts a stronger player. Pro Evolution Soccer 2008 (Konami, 2007) utilizes strategical and tactical adaptations (Lopes and Bidarra, 2011). In tutoring systems, support systems an also support the player by providing advice to ease the current task in the form of NPCs visual clues or maps (Charles et al., 2005).
Bakkes, Tan, and Pisan (2012) argue that when modelling an AI, it should remain plausible to the player and not striving for maximal efficiency. As a result, game AI incorporated flaws in its decisions.

Narratives

In narrative adaptation, the task is to adapt the story of the game in real-time to fit the preferences and decisions of the player. For instance, PASSAGE (Thue et al., 2007) adapts the narrative through player analysis and their style of play. It uses a current player model for constructing the story from a set of suitable events, which were pre-annotated by a human author (Bakkes, Tan, and Pisan, 2012). For a more detailed view on the various approaches and methods used, the reader is referred to the work of Riedl and Bulitko (2013). The game Left 4 Dead (Turtle Rock Studios, Valve, 2008) adapts its procedural narrative generation according to the player's pace and behaviour. In the game Heavy Rain (Quantic Dream, 2010), the decisions of the player are the driving force behind the decision how the ending of the game's story will conduct (Lopes and Bidarra, 2011).

Quest or Mission Adaptation

Adaptation in quests lies in the completion of quests that exhibit open endings. During gameplay, the behaviour of the player finalizes these endings into concrete storytelling decisions. Furthermore, reaching certain events in a quest can spawn another side quest. The challenging factor in this type of adaptation is the demand to keep enough control for the game designers (Bakkes, Tan, and Pisan, 2012). In the work of Dormans and Bakkes (2011), they cover mission generation with player based adaptation.

Music and Sound Adaptation

Musical adaptation has been a factor responding to the player since early video game development. For instance, games use location changes to change the type of music playing. Music is a powerful instrument in games, considering that it allows real-time induction of emotions and supports immersion. S. Livingstone and A. Brown (2005) implemented a system that switches music in response to the estimated player emotional state (Bakkes, Tan, and Pisan, 2012). Plans and Morelli (2012) explore measuring frustration, challenge, and fun for procedural sound generation in games.

Player Matching

Player matching is important in online multiplayer games. If, on one hand, the opponent is too weak the player will lose interest, and if on the other hand, the opponent is too strong, frustration will result in a bad game experience. Hence, in game environments where real players compete against one another, opponent based skill matching improves the game experience (Bakkes, Tan, and Pisan, 2012). Methods for performing such a matching are the ELO rating system, the Glicko rating system (Glickman and A. Jones, 1999), or Microsoft's TrueSkill and TrueSkill 2 methods (Herbrich, Minka, and Graepel, 2006; Minka, Cleven, and Zaykov, 2018). A discussion with more detail follows in a succeeding Section of this thesis.

As this Section has shown, adaptation addresses how game content can be modified to optimize the experience of the players. The difference to procedural content generation lies in its more generic approach regarding the generation of game content as a whole. In what follows, the emphasis lies exactly on that.

2.4 Procedural Content Generation (PCG)

The purpose of this section is to give a general overview on the methodology used to construct games on an algorithmic basis. This construction encompasses all areas of game content (Togelius, Yannakakis, et al., 2011). Furthermore, this section explores how components are constructed inside the literature and games. In the traditional game-development approach, game studios relied on an increasing amount of developers and designers to produce content manually. At the same time, the studio is restricted by temporal and monetary constraints. PCG methods enable the creation of more content in less time with less resources being spent. While at the same time, the type of work shifts towards maintenance and adjustment of the underlying algorithms (Hendrikx et al., 2013). Furthermore, PCG promises lower memory usage due to the fact that the content is generated instead of prefabricated. Another strength lies in the emerging attractiveness of recurring gameplay, considering that the games are able to offer distinct content every time they are played. Besides that, it also presents a new game mechanic to explore for novel future game genres (Togelius, Yannakakis, et al., 2011).

For PCG to succeed, two detriments need to be overcome. The first lies in the quality of content generation. It is difficult for algorithmic solutions to reach the qualitative standards achieved by manual game designers. Additionally, the generation of certain game content presents as more intricate compared to other types of content. Generating textures is far simpler than creating a complete storyline for an RPG game (Compton and

Mateas, 2006). The second detriment lies in a lack of controllability of the generational algorithms. The task of designing levels and especially testing becomes more tedious as there are way more possibilities that can generate. Therefore, it becomes factually impossible to comprehensively test each and every possibility of content that could be generated (Yannakakis and Togelius, 2011).

PCG Methods

PCG can be partitioned in two types of approaches to generation. While constructive algorithms generate content only once, generate-and-test algorithms pair generation with consequent evaluation. In the latter case, the generation is repeated until all necessary criteria are met and the evaluation transpires with an affirmative result (Togelius, Yannakakis, et al., 2011). There are various approaches to PCG. *Optimization* performs PCG by maximization of a defined measure or evaluation function. *Constraint satisfaction* searches configurations that satisfy predefined constraints. *Grammar* specifications define content production rules used to construct content with help of an algorithm accordingly. The *constructive* approach compiles content from a library of predefined building blocks (Smith, 2014). Besides exerting these methods individually, it is possible to combine them with the optimization approach. For instance, the work by Shaker et al. (2012) fuses evolutionary algorithms with generative grammars. PCG goes beyond these methods, and thus discussing the entire catalogue of methods would go boyond the scope of this work. The interested reader is therefore referred to the work of Yannakakis and Togelius (2018, pp. 165–171), which also covers methods like cellular automata, noise generators, fractals, and machine learning.

Quality Measurement

In PCG the task of evaluation functions lies in estimating the quality of the generated content. In *direct* evaluation, values that are obtained directly from the content inform the quality of the generation (Yannakakis and Togelius, 2011). For a 2D platformer, the amount of obstacles and opponents can serve

as an indicator for assessing quality of generation or difficulty in general.
For the evaluation of *simulation-based* evaluation functions, the game must be played by an artificial agent reflecting human behaviour. Togelius, De Nardi, and Lucas (2007) use an artificial model of a player to assess the track quality and subsequently evolving the tracks to maximize the entertainment value (Yannakakis and Togelius, 2011).
The evaluation of *interactive* evaluation functions can only be performed by a physical player. The information collection is performed either during the game session or in advance through exertion of a questionnaire (Yannakakis and Togelius, 2011).

Content Representation

The representation of content imposes the requirement of specifying parameters to control the generation of content. Upon determination of these parameters and their degree of control, a tradeoff regarding level of abstraction has to be found. Simultaneously, it is a question of the amount of complexity used when modelling. On one hand stands a generation guided by a vector of values that only indicates certain properties like number of enemies. And on the other hand, more direct control can be exerted by controlling each entity's positions alongside further properties inside the parameter vector (Yannakakis and Togelius, 2018, p.158).
Another question regards the initial assignment of these parameter vectors. The decision lies between assigning deterministic values or stochastic ones. Consequently, the game will be static every time it starts or completely randomly generated every time (Togelius, Yannakakis, et al., 2011).

Content Generation

The level of automation at which the generation occurs can differ between experiental chunks, templates, component patterns and subcomponents. *Experiential Chunk* considers the approach where level designers create substantial game building blocks of the game. These are playable by themselves and arranged and played sequentially. *Templates* are less explicitly defined blocks. Although they are still authored by level designers, they contain

variable elements supplemented by an algorithm. *Component patterns* are patterns that are defined by level designers but aren't essential to the player's experience. They are used alongside a level generator to further actualize behaviour in the generation. *Subcomponents* describe the generators that construct levels at the subcomponent level, where they are able to mimic human level designers (Smith, 2014).
Besides level of automation employed, Togelius, Yannakakis, et al. (2011) draw a distinction between necessary content and optional content. Necessary represents the game content that is crucial for completion of the game, and optional content which is not. Levels, puzzles, narration, and mechanics can be vital elements to the game. On the opposed end, buildings, weapons, visuals, and camera controls are dispensable content and not necessary needed.

Online and Offline Generation

Game content generates either before or during the game. Besides that, the hybrid approach of afore generation paired with alterations during the game is also possible (Smith, 2014). The discussion around whether PCG occurs online or offline is also a question of efficiency of the algorithm. For online generation to function, it is necessary that the game is running smoothly without freezing. Thus, the algorithm needs to construct content very fast. When the generation is computationally expensive, offline generation proves as more straightforward than online generation (Togelius, Yannakakis, et al., 2011). Additionally, the requirements of the algorithm are a further differentiator between the methods. If the input relies on information obtained during the game, only online generation is feasible. This is the case, for instance, when the player's behaviour directly influences content generation (Smith, 2014).

Player Interaction and Influence

The amount of influence the player's interactional information exerts on the level generation can differ. Starting with *no interaction*, the interaction happens exclusively with the generated content. The generator itself does

not consider any interactional data. *Parameterized* control implicates a small level of control through parameter tweaking of the generator before the generation takes effect. Another type of control the players can have through *preferences*. Direct critique of the content, performed either directly or indirectly, helps to improve future generations. *Direct manipulation* gives the player direct control of the PCG generation. The parameters and constraints can directly be adapted by the player (Smith, 2014). A less specific classification is given by Yannakakis and Togelius (2018, p.175). They classified the player's influence in *experience driven*, meaning the player influences the generator, and *experience agnostic*, where no influence transpires.
In this context, a feasible option is the utilization of machine learning mechanisms for estimation of player information like preferences (J. Liu et al., 2021).
The inverse relationship of the game having an influence on the player's experience is also relevant. *Indirect* influence exerts no intentional control on the players' experience. It differs in the *compositional* case, as the players' experience arises through the appearance of the components placed. *Experiential* influence considers a generator that is able to induce and control specific player experiences (Smith, 2014).

PCG and Machine Learning

Besides the method where PCG acts autonomous, there is the method where game designer and algorithm cooperate. This mixed-initiative method facilitates the creation and exploration process of the game space by a game designer. Summerville, Snodgrass, et al. (2018) refer to this method as *"auto-complete for game-content"*. In the work of Liapis, Yannakakis, and Togelius (2013), the level designer draws a map that is consequently automatically evaluated and further suggests changes to encourage creativity. Besides the task of autonomous generation of content, machine learning is suitable for alternative tasks. These include methods for content repair, where the algorithm suggests erroneous areas and consequently highlights or fixes them. Similar to this notion is the possibility of critiquing the content. And finally, through technologies like autoencoders data compression becomes a feasible option (Summerville, Snodgrass, et al., 2018).

2.4.1 Game Content Generation

The focus of the remainder of this section lies on the generation process of a 2D platformer game, alongside elements that fit into the genre. The scope of describing exactly what and how PCG is performed in context of all possible individual game elements would go beyond the scope intended in this work. For this task, the works of Hendrikx et al. (2013), Togelius, Yannakakis, et al. (2011), and Yannakakis and Togelius (2018) encompass a closer review of content generation approaches paired with appropriate algorithmic methodologies and further examples.

Narrative Generation

The narrative is an essential part of games, as it helps the players to better immerse themselves into the game and allows influencing the player's affect simultaneously (Yannakakis and Togelius, 2018, p.190). The generation of the narrative revolves around the generation of the subset of stories that make up the narrative and how the game's NPC's react to the player (Togelius, Yannakakis, et al., 2011). Modelling these NPC's further encourages the realism factor of the game (Yannakakis and Togelius, 2018, p.190).
The narrative generates through the application of planning or optimization algorithms. The algorithms are supported by additional annotated information about the game world. Humans perform the annotation, which either happens by hand or is assisted by data-driven methods. While playing, the player's actions inform the algorithms to advance the story through a number of possibilities (Yannakakis and Togelius, 2018, p.190). Dormans (2010) developed another approach, which uses a generative grammar to generate a story graph upfront. Consequently, their implementation constructs the game from that story.

Sound Generation

Music in games has two tasks. First, it defines the ambience of the game. Secondly, audio feedback is important to give the players feedback as they perform actions in the game (Hendrikx et al., 2013). For performing sound and music generation in games, there are different approaches. Either combining samples through compositional rules, sequencing musical samples after a precompiled database, stochastic approaches, or generative ones are used. The game Spore (Maxis, 2008) contains a stochastic generator implemented in the programming language Puredata (Plans and Morelli, 2012). In the work of Plans and Morelli (2012), their generative music engine is experience driven and uses genetic algorithms to evolve the generated music.

Weapons

Weapons for a further component that can be generated in games. Although they are necessary for the sake of being able to complete some games, the sheer abundance of weapons in many games makes them an optional component. The game Borderlands (Gearbox Software, 2009) employs PCG for weapon generation. The game generates its weapons at random and deduces their power from the player's current level (Togelius, Yannakakis, et al., 2011)

Puzzles

A frequent type of components that is encompassed in many types of games are puzzles. Their generation presents a task that is more difficult due to their necessity of being strictly solvable. Furthermore, they can exhibit repeating patterns, causing players to become bored while playing the game (De Kegel and Haahr, 2019). The types of games that consist of sequential puzzles that have to be solved in order to advance are prone to said detriment (Hendrikx et al., 2013). Methods for generating puzzles include genetic algorithms with an evaluation function to balance the level of challenge (Togelius, Yannakakis, et al., 2011). Another method is presented by Kartal, Sohre, and Guy (2016). They generate levels for the game Sokoban using Monte Carlo tree search. Moreover, De Kegel and Haahr (2019) provide a closer review of methods suitable for generation of various types of puzzles.

Level generation

The level generation for 2D platformer can be partitioned into methods that learn from examples and generators which create novel types of levels of their own. By learning from other levels, generators are often limited in their ability to create novel elements, that aren't present in the training dataset (Summerville and Mateas, 2016). An advantage of the method is that level can be generated very fast once the training is complete. Thus, the training phase of the algorithm yields the highest computational expense. Fortunately, the training can be performed already beforehand (Khalifa et al., 2020). In the following a few methods used for level generation are discussed.

- Supervised sequence learning methods like **Markov Chains** (Snodgrass and Ontañón, 2014) and **LSTM's** (Summerville and Mateas, 2016) have also been shown as feasible for replication of 2D platformer levels. Summerville and Mateas (2016) note that the quality of the generation hinges on the information included in the training set. As such, they were able to increase playability of the levels and simultaneous also the similarity to the human designed ones through incorporation of player path information.
- Khalifa et al. (2020) utilize **reinforcement learning** for the generation. An advantage of the method is that it does not rely on training data. Furthermore, the incremental nature of the algorithm enables its usage as a mixed-initiative game design tool.
- **Grammatical Evolution** takes a different approach by evolving the underlying grammar using genetical algorithms, the level is adapted to specific metrics (Shaker et al., 2012).

2.4.2 Rhythm-based Level Generation

Rhythm-based level generation is a method for level generation that does not depend on a set of training samples. It is based on the idea that players enjoy rhythmic patterns (Dewsbury et al., 2016; Smith, Whitehead, et al., 2011). With a objective of gaining a deeper understanding of the 2D platformer genre. Smith, Cha, and Whitehead (2008) dissect games of the genre into their categorical components. Furthermore, they give a vocabulary and analytical framework for structuring levels. The subdivision starts by partitioning the game into levels, which consist of sequences of cells and portals. Cells are sections of linear gameplay connected by portals. Cells can be further subdivided into rhythmic groups, which are patterns that translate into elements where the player has to perform periodical actions while playing. Rhythmic groups give the game their enjoyment and difficulty. They are composed of different categories of components, namely platforms, obstacles, movement aids, collectible items and triggers. All of them are briefly discussed subsequently (Smith, Cha, and Whitehead, 2008).

- **Platforms** are elements where the player can walk on without falling off.
- **Obstacles** are components that can hurt the player. They can consist of static elements such as spikes or enemies.
- **Movement Aids** assist the player traversing the level. Such elements include ladders and movable boxes.
- Any **collectible item** that can be picked up by the player that can serve as reward for risks undertaken or for guidance through the level. For instance, while weapons and extra lifes pose compensations, coins are suited for directing the player.
- By interacting with **triggers**, the player can alter the state of the level. These can realize as temporary or persistent changes in the game's mechanics.

On top of this specification, non-linearity gives the players further possibilities to traverse the game by generating branches. At the same time, it forces them to evaluate risks linked to potential rewards of both paths (Compton and Mateas, 2006).
Another important element of the game is the avatar. It represents the playable character in the game. Depending on the game, its abilities like movement or the wielding of weapons can vary (Smith, Cha, and Whitehead, 2008).
Through this representation of levels, the construction of a compact generator for new levels is possible. Furthermore, rhythmic patterns help to get the player into the flow state and facilitate to hit the correct timing of the jumps (Compton and Mateas, 2006). Geometrical realizations that force

the player to play along these patterns generate and sequentially arrange (Dewsbury et al., 2016; Smith, Whitehead, et al., 2011).

Rhythm and Action generation

Rhythmic groups contain duration, a type, and a density values, which allow a description of how difficult such a group is. Generating structures for these rhythms entails resolving them into sequences of player actions like run and jump that are arranged accordingly. The level generator then uses these sequences of actions to generate the level's geometry. The generation forces the player to press the buttons in a rhythm according to the previous definition of duration, type, and density. The various rhythmic groups connect together through player jumps (Smith, Whitehead, et al., 2011).

Geometry generation

By employing a grammar, game generation allows to represent a wide variety of possible interpretations in a compact form. Every terminal symbol contained in the grammar has a geometrical counterpart that a game designer created. These components are enclosed in a database where they are mapped to a terminal symbol (Dewsbury et al., 2016). The elements created must be tested against the mechanics and physical constraints of the game engine to ensure that they are playable (Smith, Whitehead, et al., 2011). Besides that, in the design of each of the element the game designer can further implement more variety. For example, equal building blocks can differ in their visual representations.
Rhythm-based level generation yields too many permutations, and therefore it is advised to filter out unwanted the generations. Thus, critics detect and subtract unwanted generations through application of measures (Smith, Whitehead, et al., 2011).

Rewards

Rewards can generate as part of the level generation. They guide the players through the level and compensate their actions with an equivalent reward to the risk taken. Rewards can be different elements like coins, extra weapons, extra lifes, or collectibles (Smith, Cha, and Whitehead, 2008).

2.4.3 Experience-based PCG

In experience-based PCG a model of the player's experience influences the content generation of the game. In the work of Hartsook et al. (2011), the authors examine the prospect of generating role-playing games using PCG based on a list of player styles and preferences. RPG games are complex in terms of generation as they are heavily reliant on the plot and contain further elements like NPC's, riddles, hidden items, and also boss battles. The story generation adapts an existing pre-authored story to the players preferences. From the generated story, the algorithm generates the world. An optimization process using genetic algorithms guides the underlying adaptation process.
Shaker et al. (2012) utilize a different approach of optimization in pair with a generational grammar. A genetic algorithm optimizes the generation as given by measurements of engagement, frustration, and challenge.
Plans and Morelli (2012) explore experience-based musical generation. The idea behind the approach is to detect levels of frustration, challenge, and fun during playing and then generate the game's music in consequence. The authors use genetic algorithms as method for optimization of the music respectively to these parameters.
In the work of Shu, J. Liu, and Yannakakis (2021), they developed a framework that generates levels for a Super Mario Bros (Nintendo R&D4, 1985) game. The game is then optimized towards achieving the highest amount of fun. A GAN algorithm to generate segments of the level, which are sequentially ordered by a reinforcement learning algorithm. Additionally, they test and adapt the levels by using an evolutionary algorithm.

Besides the optimization process in PCG, reinforcement learning allows training opponents automatically. Through repeated competition against each other, the opponent learns to play against a simulated player. Thus, the game content can be supplemented with another type of opponent who achieves very well-versed performance while playing against a human player.

2.5 Reinforcement Learning

The origin of Reinforcement Learning (RL) can be backtracked to three independent series of research topics. The oldest topic dates back to the work of Thorndike et al. (1912), who analysed the psychology of animals and deduced trial and error learning from their behaviour. They examined the interplay between animals and their environment and concluded that animals condition their behaviour in compliance with the quality of reward delivered by their environment. Alan Turing further elaborated on the idea and became the first individual detailing a *"pleasure-and-pain-system"* in conjunction with computers. The second topic that fostered RL is called dynamic programming. Bellman (1957) studied methods for resolving optimal control problems. His methods later became known as *dynamic programming*. The last topic contributing to RL is called Temporal-difference learning (TD-learning), which was first outlined by Samuel (1959) as a learning method for the game Checkers. The algorithm quantifies the difference between two consecutive predictions to encourage its learning process Tesauro (1995). The merger between trial-and-error and TD-learning can be credited to Klopf (1972). Witten (1977) then combined trial-and-error learning with dynamic programming. And Watkins (1989), with his work on Q-learning, managed to combine all three of these concepts into one (Sutton and Barto, 2018).

RL agents are goal oriented entities who are situated in an environment governed by uncertainty. Their intention lies in finding the best strategy for goal maximization through interaction with their surrounding environment. It describes a paradigm alongside supervised and unsupervised learning in the field of artificial intelligence, as there are differences in the application of the methods (Jordan and Mitchell, 2015).
In supervised learning, a labelled set of data samples is used to train

the algorithm. The algorithm can then estimate the most likely label for samples that were not included in the training set, and hence samples that the algorithm has never seen before (LeCun, Bengio, and Hinton, 2015). Unsupervised learning emerged through the requirement for constructing feature detectors without relying on labelled data samples (LeCun, Bengio, and Hinton, 2015). The method is used to uncover hidden structures in a data. RL, unlike supervised learning, does not learn from labelled data samples (Kaelbling, Littman, and A. Moore, 1996). Moreover, in RL the agent is expected to learn from itself in unfamiliar situations, and it is simply beyond achievable to obtain examples for all potential situations. Hence, supervised learning is not suitable for carrying out the task of RL. Unsupervised learning provides means to detect structure in data which is not consistent with the objective of reward maximization (Sutton and Barto, 2018). A further distinction presented by Kaelbling, Littman, and A. Moore (1996) lies in the concurrency of learning and evaluation of the system at the same time. Hence, both of the discussed methods are unsuitable for the task of RL, and it corroborates the equation of RL as a paradigm among supervised and unsupervised learning.
There are also differences in the training process for a RL system. Without a prefabricated dataset, the process of training a RL agent becomes more tedious than in both of the other discussed methods. Consequently, it is beneficial to choose a suitable testing environment.

Testing and Applications

Testing environments in RL are particularly important because they are used to evaluate and progress new ideas and methodologies (Juliani et al., 2018). The Arcade Learning Environment (Bellemare et al., 2013) presents 2600 emulated Atari games for that circumstance. More example environments include the work of Beattie et al. (2016) with *Quake 3* (id Software, 1999), Kempka et al. (2016) with *Doom* (id Software, 1993) and Matthew et al. (2016) with *Minecraft* (Mojang Studios, 2011). Furthermore, the OpenAI Gym (Brockman et al., 2016) presents a list of test environments with an array of diverse tasks such as algorithmic tasks, board games and 2D and 3D robotics tasks. Juliani et al. (2018) remark that due to RL having reached superhuman skill in most of these environments in consequence of the rise of deep learning, many environments are no longer demanding enough. Instead of employing pre-developed environments, they propose utilizing game engines, given that they allow the developers to create a perfectly tailored setting for themselves and thus evaluating new and more challenging techniques.

Besides evaluation, RL also encompasses a wide range of applications in a diverse area of fields. In *robotics* RL is employed for tasks such as visual navigation and planning (Y. Zhu et al., 2017) or robotic manipulation (Gu, Holly, et al., 2017). *Computer Vision* is a topic closely related to robotics, in consequence to both of them are employing visual sensors. Research examples enclose visual tracking of objects (Yun et al., 2017), dexterous manipulation (Marcin et al., 2020) and solving the Rubik's cube (Akkaya et al., 2019). *Autonomous driving* is likewise comprised of robotics and computer vision tasks, for example in an end to end approach for autonomous driving (Kendall et al., 2019). In *Natural Language Processing* (NLP) it is used for transfer learning in language aware agents (Bahdanau et al., 2018) and translation from natural language to SQL (Zhong, Xiong, and Socher, 2017). In the field of *Gaming* it is employed for conducting and evaluating research. It has been the testbed for a few landmark results such as a playing Atari on performing on par with human players (Mnih, Kavukcuoglu, Silver, Graves, et al., 2013) by learning only from raw pixel values as input. Furthermore, it achieved human expert performance in Go (Silver, Huang, et al., 2016; Silver, Schrittwieser, et al., 2017), Vinyals et al. (2019) reached the Grandmaster level in Starcraft 2 (Blizzard Entertainment, 2010) and Berner et al. (2019) defeated the world champions of the esports game Dota 2 (Valve, 2013). Besides realizing an in-game adversary, RL in games can also be used for level generation (Khalifa et al., 2020) or reaching better generalization by employing PCG (Risi and Togelius, 2020). A further approach employs two symbiotic agents in a generator and solver architecture, alternating between generation an evaluation of a level. The method aims to reach better generalization and encourages the generation of novel environments (Gisslén et al., 2021).

The following two sub-chapters elaborate on the topics of single-agent RL, where alongside the fundamental definitions and two different methods the PPO algorithm is discussed. Furthermore, a theoretical overview of methods and difficulties regarding the multi-agent case of RL are discussed in Section 2.5.2. The final sub-chapter concludes by discussing self-play and the utilization of rating systems in Multi-Agent Reinforcement Learning.

2.5.1 Single-Agent Reinforcement Learning

A RL system consists of two elemental components, specifically an agent and its environment. The agent features a policy that assigns an action to each state (Kaelbling, Littman, and A. Moore, 1996). In practice, the policy is often modelled as a lookup table where each state carries an appropriate action, or in more complex methods a search process is employed. The reward signal indicates to the agent whether the behaviour is beneficial. At each timestep, the action and the state of the system results in a determination of the signal. The policy is constructed on the basis of observing these rewards (Sutton and Barto, 2018).

The model maintains a depiction of the environment that the agent can use to plan its strategy. Like the environment, the model can be subject to changes over time, to which the agent can thus react. When regarding model-based learning types, a further distinction between model-free and model-based approaches is necessary. While model-based methods keep a model to estimate the impact of an action on the environment, model-free methods explicitly forego keeping a model and directly learn by trial and error (Sutton and Barto, 2018).
RL problems are formulated by means of the *Markov decision process* (MDP) that is constructed of a set of states $\mathcal{S}$, actions $\mathcal{A}$, a transition probability function $\mathcal{P}$ and a reward function $\mathcal{R}$ (Kaelbling, Littman, and A. Moore, 1996). $\mathcal{P}$ defines the reward for transitioning from state $s \in \mathcal{S}$ to state $s' \in \mathcal{S}$, where $\mathcal{P} : \mathcal{P} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$ for any action $a \in \mathcal{A}$. Additionally, a reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ yields the immediate reward signal for the transitioning agent. The discount factor $\gamma \in [0, 1)$ balances the influence of immediate and future rewards.
At timestep t the agent observes the state s_t and decides to perform the action a_t in order to reach the state s_{t+1} The task of solving the MDP lies in finding a policy $\pi : \mathcal{S} \rightarrow \Delta\mathcal{A}$ such that the discounted cumulative reward, shown in Formula 2.1, is maximal (Zhang, Z. Yang, and Başar, 2021).

$$\mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s_0 \right] \tag{2.1}$$

The exploration vs exploitation trade-off is a fundamental dilemma in RL. The agent must rely on the knowledge of the connection between actions and rewards observed in his past to obtain the best reward. At the same time, he is obliged to try and identify new action sequences that yield an overall higher reward (Li, 2018). The discount factor is used to regulate the amount of exploration and exploitation utilized in the algorithms.

Two approaches for quantifying a solution for this problem are value-based approaches and estimation via the policy gradient (Li, 2018). Both of which will be discussed in Section 2.5.1 and Section 2.5.1 respectively.

Value Methods

The *value function's* focus lies in capturing the long-term reward rather than the direct and intrinsic one of the reward signal. The value of a state describes the desirability of reaching a definite state by anticipating the state's attainable future reward. Thus, agents are able to reckon whether an environmental state should be avoided, as they know what future states can succeed. The *value function* can be used for this task and is defined in Formula 2.2 (Sutton and Barto, 2018).

$$V_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s \right] \tag{2.2}$$

$$v_* = \max_\pi v_\pi(s), \forall s \in \mathcal{S} \tag{2.3}$$

In order to find an optimal policy π_*, the *optimal value function v_** requires deduction via maximization, as is indicated in Formula 2.3. The Bellman equation then grants a recursive reconstruction for both functions, and thus enables adopting a dynamic programming paradigm (Li, 2018). Reminiscing back to the intrinsic reward signal, it becomes apparent that values are

more important in the decision-making process. However, as they must be approximated by an agent, they are therefore also harder to estimate (Sutton and Barto, 2018).
Finding a solution in RL is no simple task, considering that there are multiple approaches to quantify a solution. First, if the model of the environment is known, the solution can be calculated via dynamic programming and policy iteration or value evaluation, both of which methods are closer described in the work of Li (2018). If, on the other hand, the model must be observed by the agent and is thus not fully known, the problem can be solved via a variety of algorithms such as Temporal difference learning (Tesauro, 1995), Q-Learning (JCH and Peter, 1992), Deep Q-Learning (Gu, Lillicrap, et al., 2016), SARSA (Rummery and Niranjan, 1994) or Monte-Carlo learning (Sutton and Barto, 2018).

Policy Optimization

Contrary to value methods, policy optimization methods use an estimator of the policy gradient $\pi(a|s,\theta)$ and perform gradient ascent on it (Schulman, Wolski, et al., 2017). During optimization, the parameters θ are updated iteratively. The policy gradient (Formula 2.4) can be expressed by means of the expectation (Li, 2018).

$$\nabla_{\theta\pi}(a|s;\theta) = \hat{\mathbb{E}}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s;\theta) Q^{\pi_\theta}(s,a)\right] \tag{2.4}$$

Policy-based methods are characterized by their better convergence guarantees (Zhang, Z. Yang, and Başar, 2021). Apart from that, they exhibit issues such as ceasing a local instead of a global optimum, various inefficiencies and variance issues. Methods for policy optimization include the REINFORCE algorithm (Williams, 1992), actor-critic algorithms and policy iteration algorithms (Li, 2018).

Proximal Policy Optimization (PPO)

As discussed before, the ML-Agents framework (Juliani et al., 2018) provides a powerful method for developing and testing RL in a game engine. It offers

two differing algorithms for RL. The first one being, the work of Schulman, Wolski, et al. (2017) with Proximal Policy Optimization (PPO) and secondly, Soft Actor-Critic (SAC) by Haarnoja et al. (2018). However, due to the non-stationary dynamics in the multi-agent setting, it is recommended to use PPO with self-play (Foerster et al., 2017; Juliani et al., 2018)[1].
PPO is based off of the TRPO (Schulman, Levine, et al., 2015) algorithm, both of which utilize a clipped surrogate function for optimizing the policy gradient. PPO proposes a new clipped surrogate objective function termed L^{CLIP} (Formula 2.5), where the additional parameter ϵ ensures that the probability ratio $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ does not result in a policy update too substantial. Combined with the min operator, the effect achieved ensures that the change of $r_t(\theta)$ is ignored only if the objective is worsened (Schulman, Wolski, et al., 2017).

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \tag{2.5}$$

PPO utilizes the algorithm outlined in Algorithm 1 by Schulman, Wolski, et al. (2017), which switches between sampling data and performing gradient ascent with minibatches on the loss function constructed during the sampling (Schulman, Wolski, et al., 2017). In practice, PPO was employed by Berner et al. (2019) for training the *OpenAI Five* Dota 2 (Valve, 2013) agent that defeated the esport's acting world champions.

Algorithm 1: PPO, as described in Schulman, Wolski, et al. (2017)

```
for iteration=1,2,... do
    for actor=1,2,...,N do
        Run policy $\pi_{\theta_{old}}$ in environment for T timesteps;
        Compute advantage estimates $\hat{A}_1, ..., \hat{A}_T$;
    Optimize surrogate L w.r.t. $\theta$, with K epochs and minibatch size M $\leq$ NT;
    $\theta_{old} \leftarrow \theta$;
```

[1] `github.com/Unity-Technologies/ml-agents/blob/main/docs/ML-Agents-Overview.md#training-in-competitive-multi-agent-environments-with-self-play`, 2021

Deep Learning

Recent advances in the field of deep learning (DL) have also affected RL. The advent of neural networks enables agents to directly learn from high dimensional inputs such as raw pixel values. Convolutional neural networks in particular, enable the network to further extract structure from the input data and perform better inference (Mnih, Kavukcuoglu, Silver, Graves, et al., 2013). Mnih, Kavukcuoglu, Silver, Rusu, et al. (2015) state in their work, that the link between RL and DL is based on their evaluation of human cognition. According to the researchers, humans and animals both exercise a combination between RL and hierarchical sensory processing systems themselves. Ultimately, DL enables algorithms to achieve a performance on a par with human performance by supplying a high dimensional input to an agent with practically no precognitions. DL lays the groundwork for more complex agents and in conclusion contributed to the revived interest of multi-agent learning.

2.5.2 Multi-Agent Reinforcement Learning (MARL)

In Multi-Agent Reinforcement Learning the setting is expanded to multiple agents. Each one of the agents is maximizing their own reward, but in MARL state and reward hinge on the decisions of all other agents in the system, further complicating the setting.
In MARL, there exist two classes of methods to model the multi-agent problem, depending on the setting needed. The first class are *Markov games*, where the agents observe the system's state and execute an action before being rewarded by the system and proceeding into the next system-state. An important factor lies in the complete observability of the game to the agent. Markov games are a multi-agent extension to the MDP described in Section 2.5.1 (Zhang, Z. Yang, and Başar, 2021). *Extensive form games* on the other hand, exercise a tree-representation for modelling the temporal flow of games. They allow to model agents that only have access to partial information about their opponents and environment (Shoham and Leyton-Brown, 2008).

AlphaGo and its subsequent evolutions (Silver, Huang, et al., 2016; Silver, Schrittwieser, et al., 2017; Silver, Hubert, et al., 2018) are amongst the most well-known model projects in the field, having made remarkable progress by training an agent through self-play and defeating a human expert of the game. Subsequently, researchers at DeepMind were able to demonstrate, that games with even higher complexities, such as *Starcraft 2* (Blizzard Entertainment, 2010), can be mastered as well. Their algorithm AlphaStar (Vinyals et al., 2019) was able to reach the Grandmaster level and was ranked higher than 99.8% of human players playing the game. A further milestone was accomplished by Berner et al. (2019). Their system, *OpenAI Five*, defeated the reigning world champions in the esport game *Dota 2* (Valve, 2013), further fortifying the hypothesis that superhuman performance indeed is possible.

However, alongside the new accomplishments, a series of new challenges emerge. The benefits and challenges will be discussed in a subsequent section. The field of MARL is vast, making it impossible to give an overview of all methods contrived. Instead, this sub-chapter serves as a general overview of the topic.

Behaviours

In MARL, there are three different sorts of behaviors: *collaboration*, *competition*, and a *mixed* approach that combines both of the previous methods. In the cooperative setting, multiple agents cooperate on progressing a collective goal. Collective MARL is utilized in unmanned aerial vehicles for establishing means of wireless communication (X. Liu, Y. Liu, and Chen, 2019). In the competitive case, two opposing sides try to defeat the other side towards reaching a collective goal first. Examples for this case are the games Go, Chess, Shogi (Silver, Hubert, et al., 2018) and StarCraft 2 (Blizzard Entertainment, 2010) by Vinyals et al. (2019). Lastly, a combination of both types (Zhang, Z. Yang, and Başar, 2021) was used for mastering Dota 2 (Valve, 2013) by Berner et al. (2019), a game which features two opposing teams of agents.
For a more extensive examination of the algorithms and methods used in

the various behavioural types, the reader is referred to the work of Zhang, Z. Yang, and Başar (2021).

Communication

Communication covers the information exchange between agents. Zhang, Z. Yang, and Başar (2021) discuss three contrasting structures of information exchange in MARL systems. Firstly, they present a *fully centralized* method which employs a central controller that congregates the joint information such as actions, rewards and design policies for distribution among the agents. Secondly, agents that rely on their own observations and share information among each other *peer to peer*. And lastly, a *fully decentralized* setting where no communication between agents is permitted and thus no exchange of information occurs (Zhang, Z. Yang, and Başar, 2021). Communication can take place via communication protocols such as direct messaging or shared memory (Hernandez-Leal, Kartal, and Taylor, 2019).

Benefits and Challenges

Although MARL is can be computationally much more extensive than ordinary RL, MARL can benefit from a speed-up through parallelization. Furthermore, two more favourable properties are its robustness and scalability. MARL systems excel by their ability to replace agents if one of them drops out or malfunctions. Also, the architecture of MARL systems usually allows for an easy adjustment of the number of agents appointed. Moreover, it introduces new ways for agents to share their experiences through communication, instruction, and imitation (Buşoniu, Babuška, and De Schutter, 2010).
Alongside these benefits, MARL also suffers from novel types of challenges regarding complexity, stability, performance and design.

- **Multi-Dimensionality and Learning Goals**: In the multi-agent context, it can become much harder to define a goal due to its multi-dimensionality (Zhang, Z. Yang, and Başar, 2021). Agents generally pursue their own goals, which can be unique or correlate with the goal of other agents (Buşoniu, Babuška, and De Schutter, 2010). In terms of convergence, a method for finding a solution for the multidimensional system lies in ascertaining the Nash Equilibrium, a fixed point where no agent deviates once reached (Y. Yang and J. Wang, 2020). The question of how to model specific goals sparked new areas of interest such as managing communication learning, averting agent overfitting and robust learning to hinder malicious agents (Zhang, Z. Yang, and Başar, 2021).

- **Non-Stationarity**: As all agents learn in parallel and continuously want to reach their own goal, they construct a non-stationary environment. The Agents cannot affirm that the reward distributed was earned due to their own action or due to the actions of other agents (Y. Yang and J. Wang, 2020). Since the reward function of an agent becomes dependent upon the rewards of all other agents, the Markov property gets invalidated (Zhang, Z. Yang, and Başar, 2021). Regardless of neglecting the Markov property, the method of *independent learners* successfully applies algorithms used in the single-agent case for MARL (Hernandez-Leal, Kartal, and Taylor, 2019). On the other hand, a case where the Markov property can be neglected are *identical-interest* games, seeing that all agents share one collective goal they can optimize without regarding the other agent's policies, the setting becomes stationary.
- **Scalability**: As the number of agents rises, their joint-action-space grows exponentially, and hence rapidly requiring more performance along with new issues amidst scaling up (Zhang, Z. Yang, and Başar, 2021). Most MARL algorithms are devised as two-player zero-sum games, making general-sum games with a higher amount of players an ongoing challenge in current research.
- **Communication Structure**: Agents will have a particular level of knowledge about other agents and the system, or, depending on the type of communication structure used, none at all. If the agents must rely exclusively on the local observations available to them and communication between them is prohibited, their decisions can lead to globally sub-optimal decisions. This predicament, when combined with the previously mentioned concerns of stationarity and convergence, can accumulate and further worsen the total performance. Mitigating these issues can lead to a change of the communication

structure by facilitating information exchange and encouraging communication for better inference (Zhang, Z. Yang, and Başar, 2021). As Buşoniu, Babuška, and De Schutter (2010) believe, communication is a critical component in collaborative systems and can also be advantageous in a competitive environment.

Self-play

Self-play enables agents to learn from their environment and their adversaries by competing against each other. It paves the way for training highly complex agents that are able to learn a perfect curriculum (Bansal et al., 2017). Zhang, Z. Yang, and Başar (2021) argue that the outcome of self-play training itself is not optimal. As a result, more variation during training is beneficial for obtaining better results. But, as a consequence, the computational workload will also increase.
The method of self-play isn't novel, as it's been examined before by Samuel (1959) with the game Checkers and later by Tesauro (1995) with TD-Gammon and Schraudolph, Dayan, and Sejnowski (1994) with Go. AlphaGo is a more recent example where self-play contributed to its successful achievement. The method consists of a mix of deep learning combined with tree search. It utilizes supervised policy learning based on the moves of human experts and is subsequently enhanced by RL via self-play to estimate the move probabilities. A second deep neural network is used for position evaluation. Both algorithms are mutually used in a Monte Carlo tree search to evaluate the best move achievable. Subsequently to AlphaGo, Silver, Schrittwieser, et al. (2017) devised AlphaGo Zero which no longer relies on human knowledge. Indeed, the training process is encapsulated solely by learning through self-play. As follows, it can be corroborated that *"AlphaGo becomes its own teacher"* as is stated in its paper by Silver, Schrittwieser, et al. (2017). AlphaGo Zero was able to acquire strategies and methods already familiar to expert Go players, however it also successfully uncovered novel ones that were unidentified beforehand (Silver, Schrittwieser, et al., 2017). In such zero-sum games, the reward becomes ineffective for tracking the learning progress, and thus a different rating scheme, for instance ELO (Glickman and A. Jones, 1999), is required.

ELO Rating System

The ELO rating system was proposed for rating players in chess and is also used for ranking players in a multiplayer setting or MARL. Game ratings are utilized for identifying players with the same skill set in order to deliver a balanced game experience in multiplayer games. Furthermore, it is used as an entry barrier for tournaments to ensure that the players that are competing exhibit a certain base level of skill (Herbrich, Minka, and Graepel, 2006). In MARL, the ELO rating system is used for estimating the performance of the learning process, as the cumulative reward becomes non-descriptive for monitoring the training progress in zero-sum games.
It was used by Silver, Huang, et al. (2016), Silver, Schrittwieser, et al. (2017) for evaluation of their Go learning agent. Beyond that, it is used in educational systems (Neumann et al., 2011). The ELO rating system was developed explicitly for chess, therefore it possesses characteristics that are unfavourable in other settings. As such, Glickman and A. Jones (1999) proposed the Glicko rating system, which poses an improvement to the ELO rating system for chess. A further alternative is is presented by TrueSkill (Herbrich, Minka, and Graepel, 2006) and it's progression TrueSkill 2 (Minka, Cleven, and Zaykov, 2018), which were developed by Microsoft and proposed for measuring skill for online multiplayer based games with more than two players.

2.6 Summary

Understanding the players and the causes of their overt behaviour opens up new possibilities of personalization in games. By making them more accessible to players, new demographics evolve. Furthermore, by increasing the enjoyment factor in entertainment games, they have the potential to become more beneficial in coping with matters like stress or depression (Russoniello, O'Brien, and Parks, 2009). The same has been observed with knowledge transfer in serious games, which boosts the efficacy of the learning process (Lopes and Bidarra, 2011).
Player modelling is a method for gaining such an understanding of the players. As the first task at hand, various models need to be tested and

corroborated for usage in a game context. Therefore, correlations between behavioural recordings of gameplay sessions and player models need to be examined and verified.

From a psychological perspective, the Big Five Factors have transpired as the best description of relatively stable human behaviour (Vinciarelli and Mohammadi, 2014). Besides them, the person's current affect and also the effects provoked by the game play another important role (Yannakakis and Togelius, 2011). Considering that the usage of this methodology lies in the creation of a description of general human behaviour, they may turn out as too general in a game context (Yannakakis and Togelius, 2018).
Besides psychological models, there are methods with a game research based origin. Once more the question of suitability arises. This comes from the fact, that a number of the models listed were developed primarily for studying specific game genres Yannakakis and Togelius (2018, p.212). From a computational perspective, instead of relying on prefabricated models that may not wholly meet the necessary requirements, data-learning is another feasible method. Various machine learning methods manifest as suitable methods in this case, although for any application an expert is indispensable for construction and evaluation (Hooshyar, Yousefi, and Lim, 2018).
The next predicament lies in the input of these models. Previous work (Van Lankveld et al., 2011) has shown that estimations of personality can be inferred purely from interactional data. Nevertheless, it is possible to ameliorate them with additional data sources. One type regards the signals of the body as a further source. Game context provides another important cue that can help in the distinction between similar affects for instance (Yannakakis and Paiva, 2014).

After fruition of developing a player model, the resulting acquirements allow guidance of the adaptation process. An adaptation objective in combination with optimization algorithms informs the game on how it should be adapted (Lopes and Bidarra, 2011). As outlined in this chapter, different parts of games can be adapted. The level, its mechanics, and game AI, the narration, and the game's atmosphere have all been adapted in various scientific papers.

While adaptation addresses the problem of modifying content to better fit the player, PCG takes a more generic view by consideration of the generation

as a whole. Thus, both topics are closely related. PCG is a vast area which is gaining increasingly more popularity among games due to it's memory and expressive benefits (Togelius, Yannakakis, et al., 2011). The techniques applied, vary from the type of content generated (Smith, 2014). Moreover, the generation is either more straightforward or difficult depending on the type of content (Yannakakis and Togelius, 2011).
With an emphasis on level generation of a 2D platformer, the suitability of methods has to be taken into account. Learning from examples limits the expressive range by the constraints specified by the training data (Summerville and Mateas, 2016). As such, it is challenging to expand the game by adding new game elements to the generation. A grammar generation on the other hand, gives the game designer more control of the content generated inside a level. A drawback to this method is that its generations can seem more generic. As Compton and Mateas (2006) note, a formidable generation with grammars is still very challenging to achieve.

3 Design and Implementation

This chapter covers the design and implementation of a game that analyses player behaviour in order to inform adaptation of the same game. The fundamental motivation behind this is to create a self-personalizing game that better meets the expectations of the players. These can range from entertainment games that provide fun and enjoyment to serious games that improve learning performance. Thus, a framework that allows the game to learn from its players and to adapt itself is discussed. Games nowadays contain a large amount of elements and mechanics that players can interact with. Different types of players will enjoy different aspects of games. As a consequence, the game applies adaptations to its content to better suit what players esteem. For instance, gamers that appreciate puzzles will encounter them more frequently while playing the adapted game. Player analysis, modelling, and game adaptation play a crucial role in this process. From a grand scheme of things, the task of this work thus consists of two core parts.

1. Construction of a game that identifies player profiles.
2. Utilization of player profiles to adapt the game's content.

The first item focuses on the creation of player profiles in general. It explores whether psychological scales, such as the BIG-5 are applicable for this task. For this purpose, the work performs a study that validates the suitability of various methodologies from the literature. For the latter task, procedural content generation provides the necessary adaptability of content on a probabilistic basis.

The twofold separation of the task opens up to address different types of target groups of this work. The first step serves as behavioural analysis of the players. In combination with the second part and through game adaptation, the game is able to also address the players themselves.

- **Players:** As an end-to-end approach, the framework enables the game to better support the players' expectations.
- **Game researchers and developers**: The functionality of the framework developed can be transferred to other types of games. It outlines how to implement player based profiling with adaption in games.
- **Game analysts and behavioural scientists:** The game provides an alternative method for behavioural estimation. An advantage being that the profiling process can take place at any place, and anytime. Therefore, the subjects can participate in a familiar and relaxed environment. Leading to fewer inaccuracies and making deliberate distortion of the test less likely.

A minimum viable function set assures that the game can reach and satisfy these stakeholders. The next section outlines these functionalities.

3.1 Requirements

For the project to succeed, a list of preliminary requirements are identified. These functional requirements define what a system must be able to do in order for it to function. Non-functional requirements on the other hand refer to performance and qualitative objectives (Glinz, 2007).

3.1.1 Functional Requirements

1. The level must be fully traversable from start to finish.
2. The game commences with a tutorial.
3. The game is won as soon as the player visits the house at the end of the level.
4. Portals serve as link between paths.
5. The player can have multiple lives.
6. When players die, they respawn at the most recently passed checkpoint.
7. The player can shoot bullets.
8. Enemies can be killed with a bullet or by jumping on top of them.
9. The smart enemy can move, jump, and shoot bullets.

10. Upon approaching NPCs, a dialogue window appears.
11. NPCs can be killed.
12. Players can choose different actions in dialogues.

3.1.2 Non-functional Requirements

1. The gameplay should be fluid.
2. The movement and behaviour of the player should feel natural.
3. The amount of time spent playing should not exceed what feels pleasant to players.
4. The players make active and passive decisions in the game.
5. Parts of the game should appeal to different kinds of players. Therefore, the following categories should be contained.
   a) Players should be challenged with various types of enemies.
   b) The game should contain challenging elements like puzzles and moving platforms.
   c) The level can be traversed more rapidly with shorter paths.
   d) Collectibles address players that like to maximize exploration and collection of tokens.
6. The game should work on any modern computer and run on multiple platforms.
7. The game's level generation should be extensible.
8. Elements of the game should follow well-known patterns of the genre.
9. The game's key bindings should be similar to those used in other games in the genre.
10. The level generation must be parametrizable.
11. The game must be able to record user behaviour.
12. Player data recordings should suffice for identification of player profiles.
13. The smart enemy should resemble the player in behaviour and appearance.

The remainder of this chapter discusses how to put these requirements into practice. The conceptual architecture proposes how different components collectively contribute to a system that adapts a game to the players' behaviour.

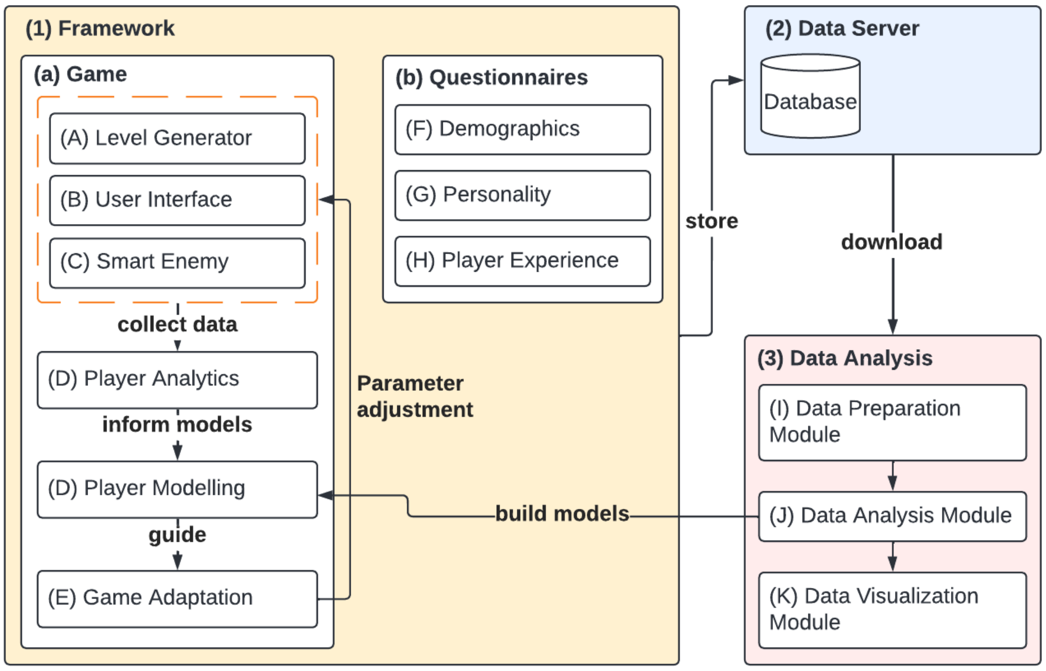


Figure 3.1: The coneptual architecture of the project consists of the core framework, a data server, and a data analysis component.

3.2 Conceptual Architecture

The conceptual architecture outlines the main components with their dependencies. Overall the project consists of the three main components *Framework*, *Data Server*, and *Data Analysis*. Figure 3.1 shows each of these components along with their subordinate components. The framework consists of another segmentation into the two components *Game* and *Questionnaires*. The player analytics combined with the results of the questionnaires is stored on an external data server. The *Data Analytics* component downloads the data and analyses it. The analysis serves as guide to create the player modelling in the game component of the *Framework*. Once the player modelling is established, the player analytics inside the framework directly inform the player models. The player models guide the adaption proces, which in turn adapts parameters of the game.

1. **Framework:** The framework presents the core part of the project. It consists of the game with all its content and the questionnaires.
   a) **Game:** The game consists of the generation, analysis, and adaptation processes. It is divided into six modules.
      (A) **Level Generator:** The level generation module procedurally builds the levels of the game.
      (B) **User Interface:** Graphical interface visible to the players during the game. It displays relevant information, such as dialogues, or number of lives left, during the game.
      (C) **Smart Enemy:** The task of the Smart Enemy module lies in the development of an alternative opponent that challenges the player.
      (D) **Player Analytics:** The module collects the players' behavioural information used for profiling.
      (E) **Player Modelling:** The task lies in creation of player profiles.
      (F) **Game Adaptation:** The adaptation identifies the parameters of the game that need to be altered.
   b) **Questionnaire:** Supplies Questionnaires to the player
      (G) **Demographics:** Demographical questions are posed to players.
      (H) **Personality:** The BFI-10 questionnaire queries the true personality of the players.
      (I) **Player Experience:** The questionnaire inquires the players' experience after completion of the game.
2. **Data Server:** A database collects the behavioural player data.
3. **Data Analysis:** The data analysis component combines player data and analyses it. It downloads the player data from the data server. Subsequently, the task lies in exploration of player profiles from the data. This part consists of the tree subcomponents extraction, analysis, and visualization. Finally, it informs the player modelling inside the framework.
   (I) **Preparation/Extraction Module:** Extracts additional meta-information from the user generated information.
   (J) **Analysis Module:** Performs analysis on the extracted data.
   (K) **Visualization Module:** Visualizes the information collected and ascertained during analysis.

The remainder of this chapter expands on the components that were outlined in this section.

3.3 Framework

The framework in its core consists of the game and questionnaires. The framework is implemented using the game development platform *Unity* version 2019.4.13f1. The platform grants more frugal development of games due to its wide variety of APIs available for development. Moreover, Unity enables the game to run the game on various platforms via WebGL. The strength of WebGL is that it can be executed in any modern web browser that supports WebGL execution. Moreover, the game can be hosted on a web server like *GitHub Pages*[1] facilitating its distribution. The game developed in this work is a 2D Jump'n'Run game. Because the genre allows a wide range of content to be integrated, it is ideal for investigating the suitability of various components.

3.3.1 Jump'n'Run Template

This work builds on top of the Unity *Platformer Microgame*[2] template. The base functionality of the template consists of a basic 2D platformer game, with a single level constructed using tile maps. The tile maps allow constructing levels by painting tiles onto a 2D map inside the tile map editor. Additionally, the template encloses basic functionalities like game physics and mechanics. The preconstructed level also contains collectible tokens. Moreover, the game already features a static and a moving type of enemies. The moving enemy can be controlled through adjustment of its patrol path. The template encompasses a list of tutorials [3] and assets to extend it. Besides assets, additional precompiled Unity Gameobjects from the template can be placed in the game.

[1]https://pages.github.com

[2]https://assetstore.unity.com/packages/templates/platformer-microgame-151055

[3]https://learn.unity.com/project/2d-platformer-template

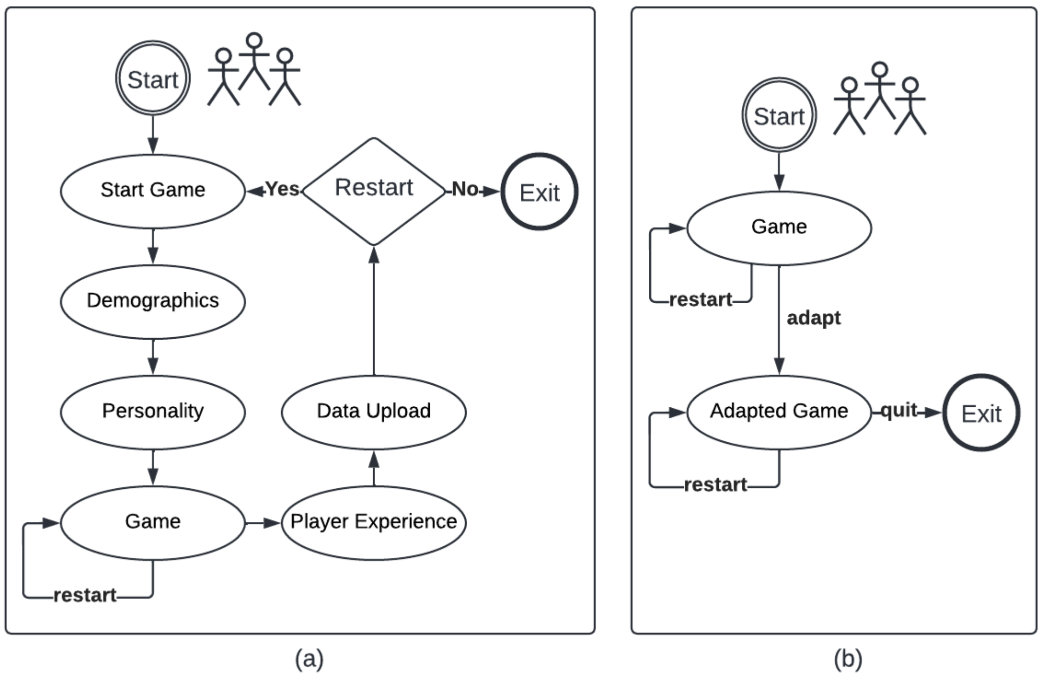
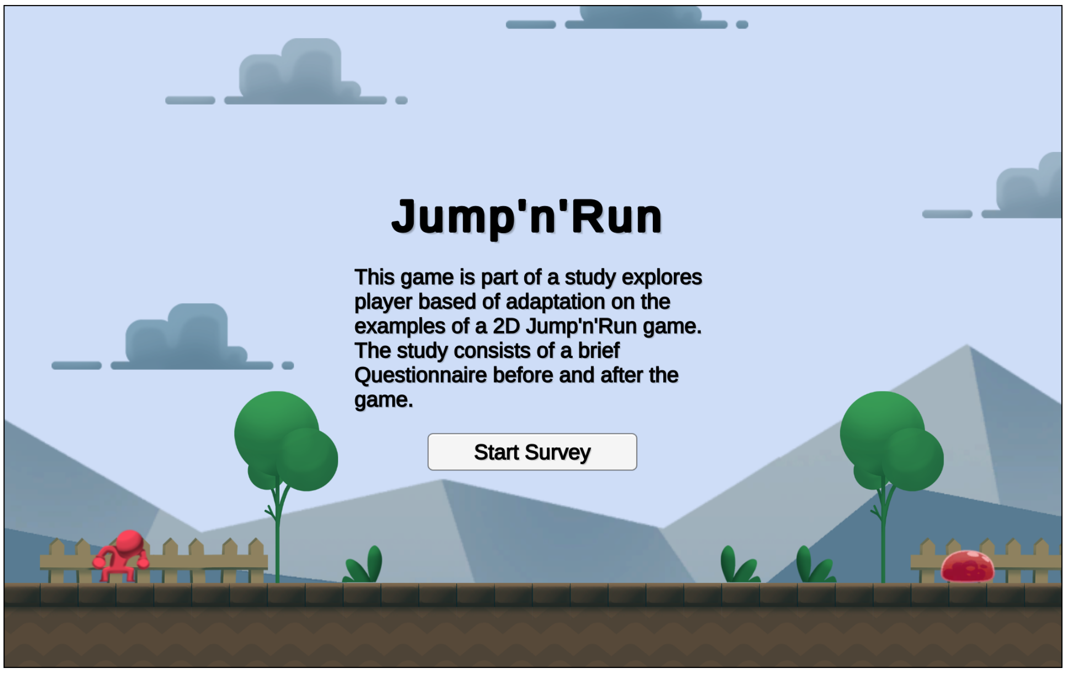


Figure 3.2: The framework consists of two procedures. Procedure (a) outlines the study, and procedure (b) outlines the game that combines analysis and adaption.

During the development of the game, it was extended with novel elements and structures. These additions consist of moving platforms, NPCs, the Smart Enemy, portals, movable boxes, extra lives, and player weapons. In addition, the level contains new puzzles and multiple paths. The entire set of these components will be covered in a later section.

3.3.2 Structure of the Framework

The game component consists of six modules divided into seven game scenes administered to the players in succession. Figure 3.2 (a) outlines the sequence of the complete study, while (b) outlines the idea of the final version of the game. The game aims at fusing player analytics and adaptation in a single game instance. The following segments describe the study's structure and components.



Jump'n'Run

This game is part of a study explores player based of adaptation on the examples of a 2D Jump'n'Run game. The study consists of a brief Questionnaire before and after the game.

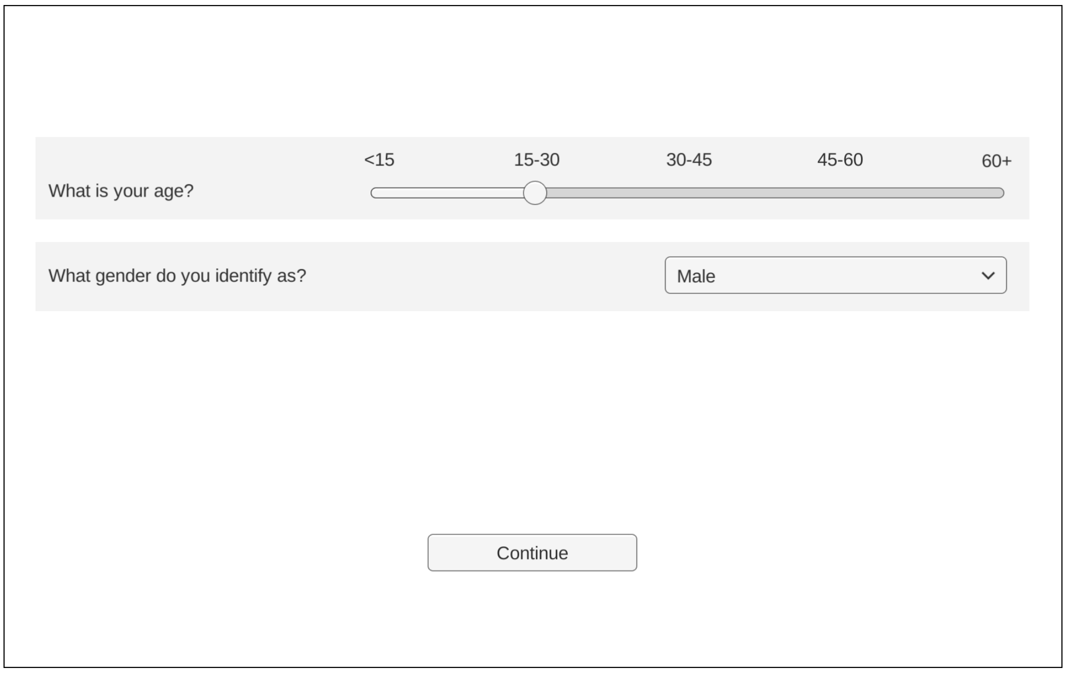
Figure 3.3: The Start-Game Scene explains the content and procedure of the study to the participant.

Start-Game Scene

The initial *Start-Game* Scene explains the intent of the study and contains a start button that guides the participants to the first questionnaire of the study. The scene, as shown in Figure 3.3, embodies a custom game scene as backdrop.

Demographics Scene

In the *Demographics* scene, the players have to complete demographical information concerning their age and gender. The scene is shown in Figure 3.4. The players are required to enter their age in a text field. The gender is selectable from a dropdown menu, where the players select between male, female, other, and prefer not to disclose. The continue button of the questionnaire appears after expiration of a two-second timer.

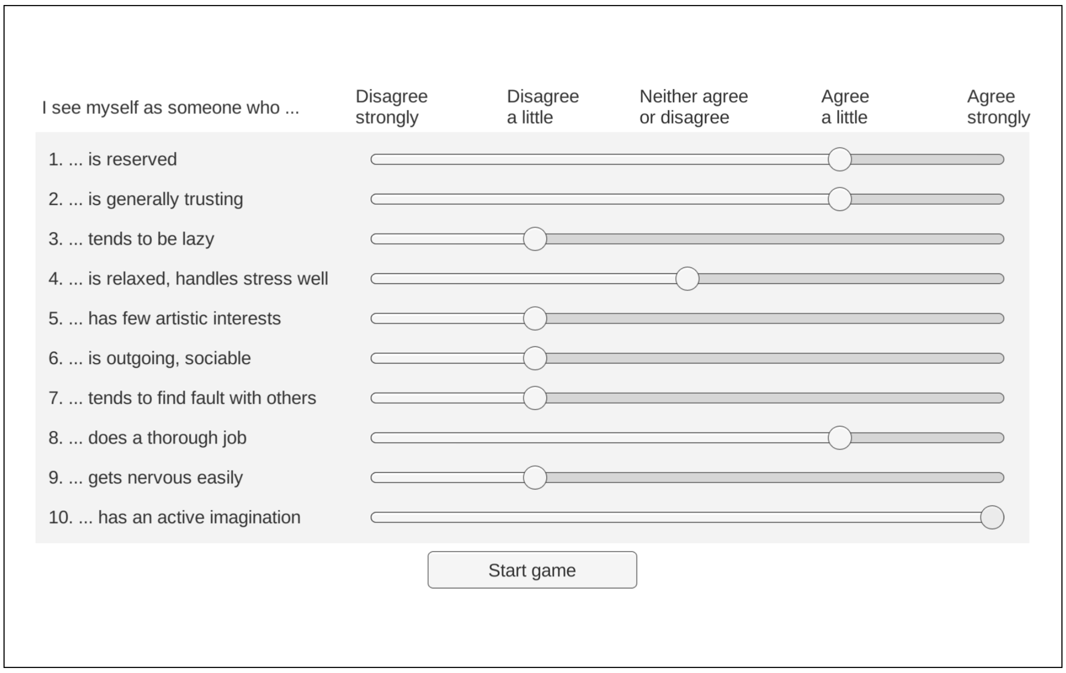

What is your age?

<15 15-30 30-45 45-60 60+

What gender do you identify as?

Male

Figure 3.4: The Demographics Scene consists of questions regarding age and gender identification of participants.



I see myself as someone who ...	Disagree strongly	Disagree a little	Neither agree or disagree	Agree a little	Agree strongly
1. ... is reserved				●	
2. ... is generally trusting				●	
3. ... tends to be lazy		●			
4. ... is relaxed, handles stress well			●		
5. ... has few artistic interests		●			
6. ... is outgoing, sociable		●			
7. ... tends to find fault with others		●			
8. ... does a thorough job				●	
9. ... gets nervous easily		●			
10. ... has an active imagination					●

Figure 3.5: The Personality Assessment Scene estimates the true personality of the participants through application of the BFI-10 questionnaire.

Personality Scene

The second questionnaire scene presents the *BFI-10* (Rammstedt and John, 2007) questionnaire. As is shown in Figure 3.5, it consists of ten items that are rated using a Likert scale. The questionnaire assesses the ground truth of the participants' personality. The players choose the level of approval regarding each item in the list through adjustment of sliders. The continue button of the questionnaire is hidden on purpose. It presents itself after expiration of an eight-second timer, or as soon as the player has adjusted more than two sliders.

Game Scene

The *Game* Scene, shown in Figure 3.6, denotes an important part of the study. The players play the Jump'n'Run game. Consequently, the game records the

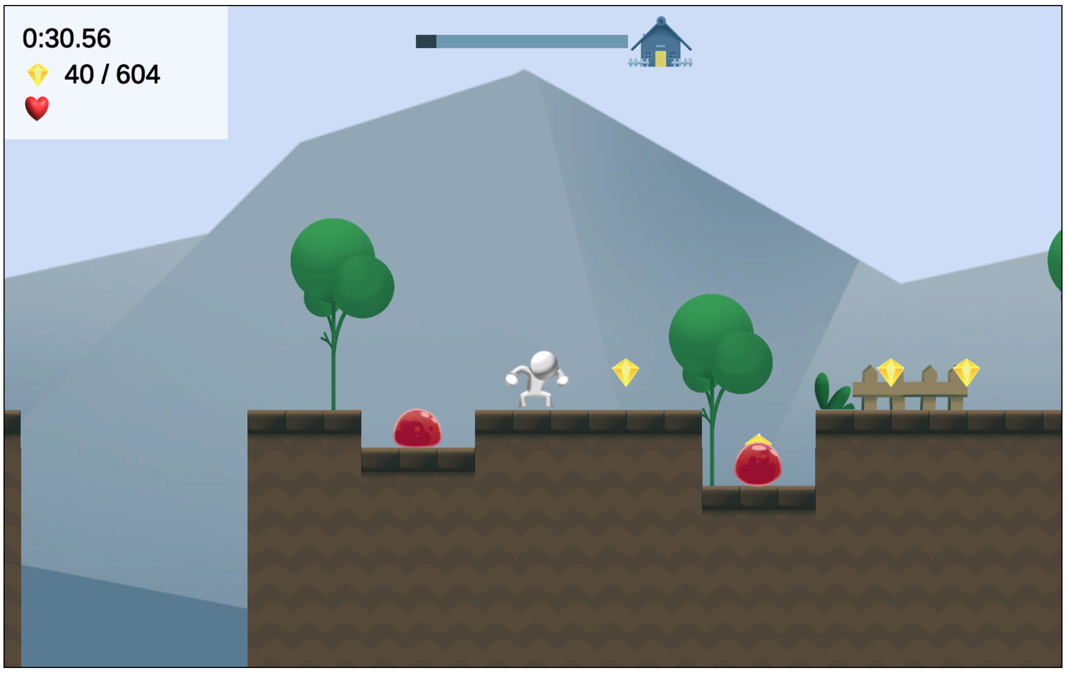
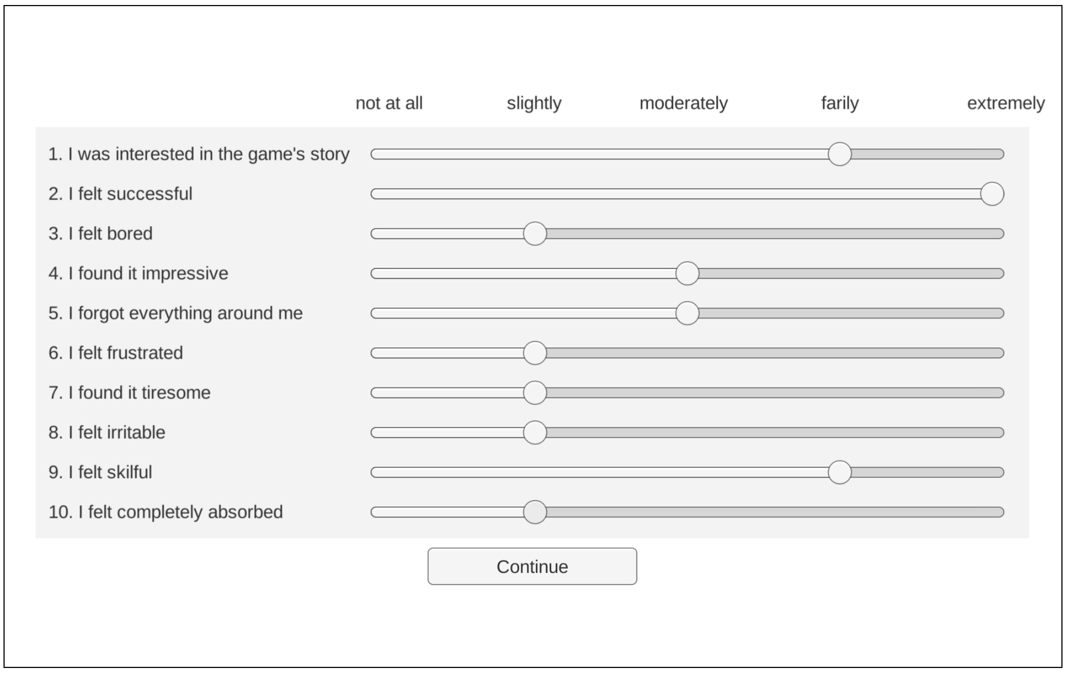


Figure 3.6: The Game Scene consists of the Jump'n'Run game that documents the players' behaviour while playing.

players' interaction. A dissection of the game along with all of its elements and the player data collected is discussed in a later part of this chapter. After completion of the game, the game automatically progresses to the next questionnaire scene.

Player Experience Scene

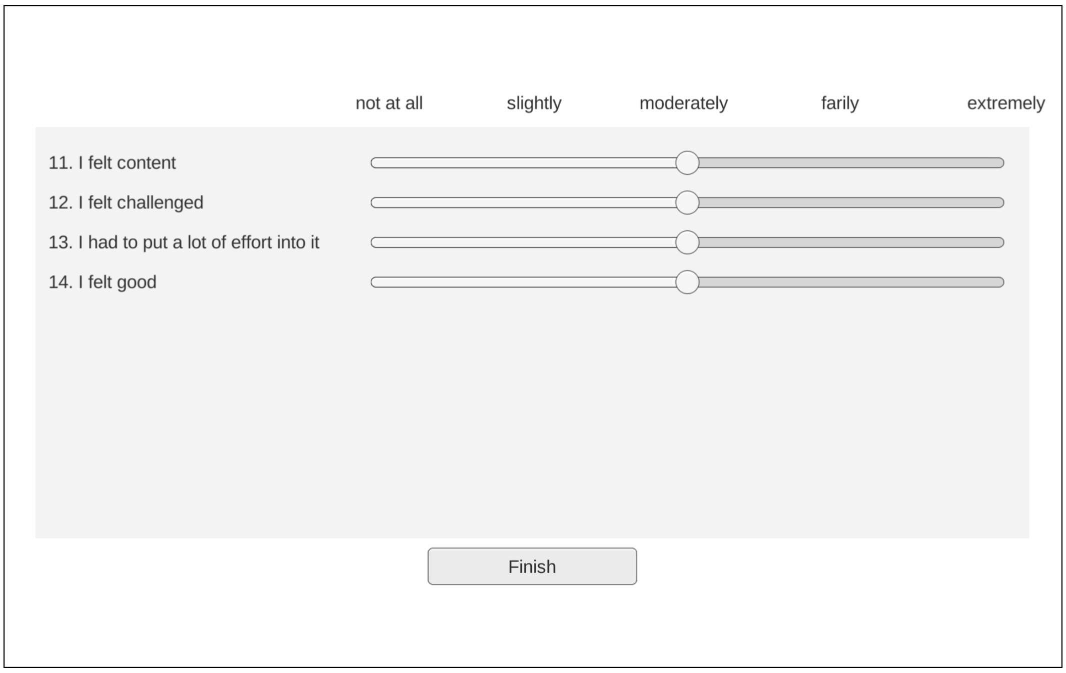
Post-game, the *GEQ* (IJsselsteijn, Kort, and Poels, 2013) is applied. Due to its length, two scenes present in parts of the questionnaire. These scenes are illustrated in the Figures 3.7 and 3.8. The study uses the much shorter *in-game* module of the GEQ, which consists of 14 items. Like the BFI-10, the questionnaire uses a Likert scale where the players can select how much they can identify with each item by adjusting a slider. Both scenes apply small delays to prevent players from skipping the questionnaire.



not at all | slightly | moderately | farily | extremely

1. I was interested in the game's story
2. I felt successful
3. I felt bored
4. I found it impressive
5. I forgot everything around me
6. I felt frustrated
7. I found it tiresome
8. I felt irritable
9. I felt skilful
10. I felt completely absorbed

Figure 3.7: The first GEQ Scene presents the first then items of the GEQ questionnare to players.



11. I felt content
12. I felt challenged
13. I had to put a lot of effort into it
14. I felt good

Figure 3.8: The second GEQ Scene presents the last four items of the GEQ questionnare to players.

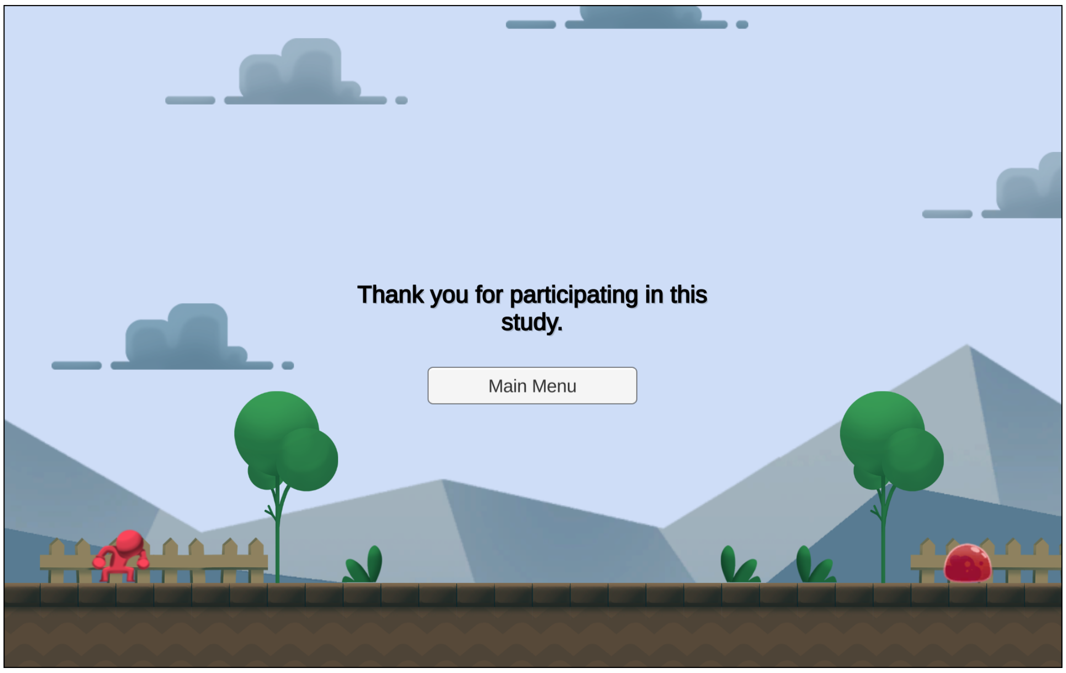


Figure 3.9: The Final Scene thanks players for their participation and gives the option to restart the game or the study from anew.

Final Scene

The *Final* scene transmits the player data to the data collection server. As is shown in Figure 3.9, it consists of a graphical representation of the game as backdrop along with a message that thanks the players for their participation in the study. After successful upload, the game presents players with the option to either restart the game, or the complete study from the beginning.

After having established a general depiction of all scenes of the framework, the next section will go more into detail of the functionalities of the game scene.

3.4 Game UI

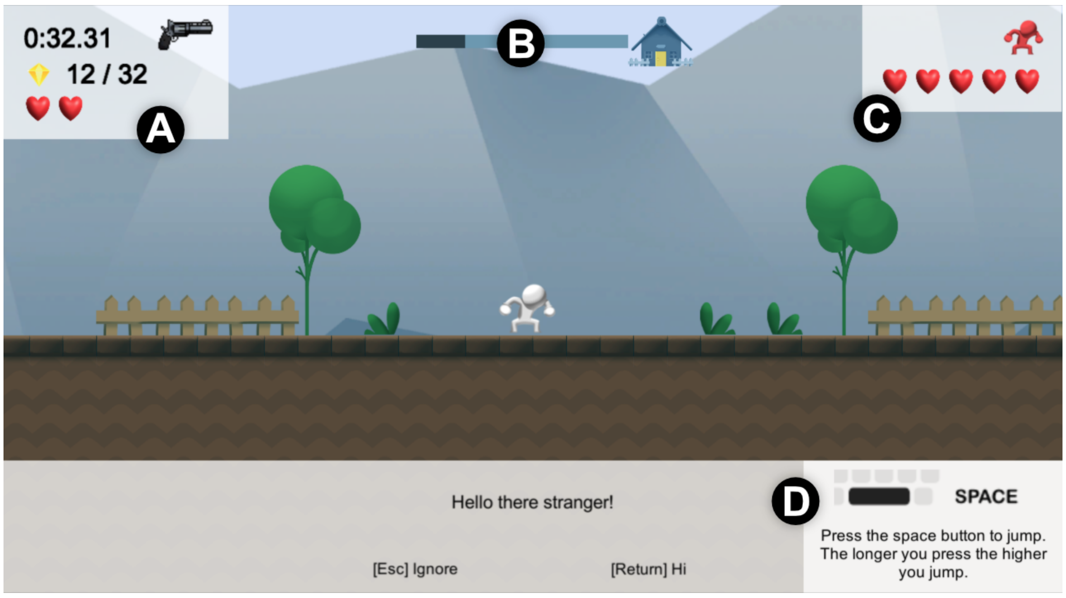
The game UI consists of four principal components that serve as indicator of additional information whenever necessary. Figure 3.10 shows the UI with all its elements.

(A) The player UI in the top-left of the UI shows the players' time since the game started. Furthermore, it shows the number of tokens collected and the number of lives left. The weapon tutorial graphic becomes visible when the player collects a weapon token in the game.
(B) The indicator in the top-center of the UI shows the progress the players have towards completion of the level.
(C) When the players get in the close vicinity of a smart enemy, the UI in the top-right corner appears. It visualizes the amount of lives of the smart enemy the player has encountered.
(D) The bottom part of the UI consists of the dialogue and tutorial system. It consists of a text message with two options, along with an optional tutorial graphic.

The dialogue system denominates the component with the highest complexity among these elements. Thus, a more detailed breakdown ensues.

3.4.1 Dialogue and NPCs

The dialogue system uses binary graphs that structures the procedure of the dialogues. The dialogue UI consists of a text message and two answers the players can choose from. As players select either option, the graph advances along that edge to display the next dialogue item. If there is no further item along the selected edge, the dialogue interface disappears and the dialogue concludes. Each dialogue item has additional configuration parameters. For instance, a timeout controls the automatic disappearance of dialogue items, an attached portal can appear or vanish, or it can restrict player movement. Various dialogues are presented to the player after the game start, during the tutorial, to warn players from the smart enemy, or when a player approaches a NPC. In NPC encounters, the choices made by players can yield different effects in the game. Each NPC gets an assignment



Hello there stranger!

[Esc] Ignore

[Return] Hi

Press the space button to jump. The longer you press the higher you jump.

Figure 3.10: The user intefrace visible during the game. It consists of the game statistics, the level progress, the smart enemy status, and the NPC dialogues with a tutorial graphic.

out of one of the four possible story assignments *Coins*, *Enemies*, *Puzzle*, and *Short*. NPCs are optional game content and intended as a type of social affinity detector in the game.

3.4.2 Tutorial Dialogues

Tutorials dialogues are specific types of dialogues layered on top of normal dialogues. The game has tutorial dialogues for movement, jumping, and gun. Tutorial dialogues possess different set of key bindings to advance the dialogue graph. For instance, in the case of the movement tutorial dialogues, the player automatically advances the dialogue by pressing of the correct movement keys. The same applies for the jumping tutorial, where the correct jumping keys advance the dialogue. Another distinction between standard dialogues and tutorial dialogues, lies in the tutorial's ability to show an additional image. The image shows specific instructions of the correct controls. These images appear after a expiration of a short 3.5-second delay as an additional source of help.
Tutorial dialogues appear grouped together in two different places in the game. At the onset of the game, the movement and jumping tutorial sequentially show after the introductory story message. The second one appears upon the initial collection of the weapon token.

3.5 Level Generation

This section discusses how the level generator generates the level. It starts by describing the overall structuring of the game, and subsequently diving into the a detailed discussion of the level generator employed.

3.5.1 Paths

The level consists of paths the players can walk on. They have to traverse from the leftmost point to the rightmost point of the level. The game discerns between the main and secondary paths. The main path exists as a single instance, while the secondary ones can generate multiple times. The generation algorithm compares bounding boxes of paths in order to ensure that they will not overlap. Special zones called *KillzoneOutOfGame* precede and follow paths in their generation. They prevent the players from jumping off the path, while simultaneously working as a detector of said event.

Paths contain four probabilities P_{coin}, P_{enemy}, P_{puzzle}, P_{length}, used to specify the generational amount of various types of content. In the main path, these values are assigned invariable. In secondary paths, the level generator assigns these variables probabilistically during the generation. The impact of each one of these probabilities on the path generation of all types of paths is outlined as below:

- P_{coin}: The variable emphasizes the generation of a large amount of collectible tokens. They generate an evenly distributed amount of tokens on the generated platforms.
- P_{enemy}: The game varies the amount of opponents that impair the players.
- P_{puzzle}: The players must overcome higher amounts of moving platforms. Furthermore, the generator spawns sequential moving platforms more often.
- P_{length}: The length of the path varies among secondary paths and kept fixed regarding the main path.

Main Path

The main path starts with an introductory tutorial structure. The structure originates with an initial dialogue combined with the game's tutorial. After the tutorial dialogues, players need to move beyond two obstacles. They cross a NPC giving an explanation about the purpose of checkpoints. The tutorial proceeds with another NPC explaining opponents to the players. The last element in the tutorial consists of another NPC that explains how NPCs behave in the game. While conversing, the NPC spawns a portal, that teleports the player to the beginning of the game. Once the player steps through the portal, the game's clock starts counting.
The tutorial structure is followed by path generation consisting of 17 states generated with the rhythm based generator. The main path can generate

secondary paths during its generation. At the end of the main path the generator generates a final obstacle ensued by the game's *VictoryZone*. The last obstacle poses a final threat to the player by generating a smart enemy with two lives.

Secondary Paths

Secondary paths serve a different kind of purpose in the game. The path probability with the highest value determines the type of path. With a variety of types, the game aims to target a diverse range of players who will venture on the various paths. Paths are intended for players to silently express their preferences on the game.
The start structure is identical for all paths, regardless their type. A portal teleports the players onto a platform with a checkpoint at the beginning of the secondary path. Hence, if the players die on a secondary path, they will always spawn on the same path. Subsequently, the rhythm generator generates the path. The variable P_{length} determines the length of the generated path, and generates between one and nine states. In secondary paths, the generator cannot generate novel secondary paths. The ending structure of the path furthermore follows a predefined pattern that depends on the path type. As the first element, a unique structure generates. Subsequently, an optional reward in the form of an additional life or a collectible weapon generates. The final element is a portal leading back onto the main path. The return position of the portal lies next to the portal of origin on the main path. Upon player contact, the portal resets the respawn position of the player back onto the main path.
The main differences between secondary paths are as follows:

- The paths of type **Coins** generate a high number of tokens, due to a high value of P_{coin}. Furthermore, the path ends with an additional life as reward.
- Due to a high P_{enemy} probability, the type **Enemies** features many opponents in its generation. Just ahead of the return portal, the path generates a smart enemy with a single life. The smart enemy is modelled in form and behaviour after the player. Prior to the portal, the

players can collect a weapon token. Collecting the token gives them the ability to shoot projectiles afterwards.

- Paths of type **Puzzle** generate puzzles as a final element of challenge in the path. The generator selects one out of three possible puzzle generations. All of which are puzzles that require the players to move two boxes for advancement. After the players' prevailing of the puzzle, they can collect an additional life as reward.
- The type **Short** is an exception to the other types of paths. The probability P_{length} indicates a short path, and thus the rhythm generator generates a single state only. Prior to the portal, neither a reward nor a special structure generates. Another difference lies in the return position of the portal. It lies at the next possible checkpoint generated after the portal of origin. Thus, short paths give the players an advantage in completing the level in less time. The only predicament being the necessity of interacting with NPCs.

3.5.2 Rhythm-Based Path Generation

Rhythmic level generation consists of two separate, coherent steps. The *RhythmGenerator* generates rhythmic sequences that are passed to the *LevelGenerator*, which in turn creates a geometrical representations. The implementation is based on the previous works of Dewsbury et al. (2016) and Smith, Whitehead, et al. (2011).
The *RhythmGenerator* class manages the rhythm-based path generation. The implementation is additionally is composed of the three class definitions *RhythmGenerator*, *RhythmGroup*, and *Rhythm*. Adjacently, an outline of the purpose of each of these classes follows.

- **RhythmGenerator:** The number of states passed to the *RhythmGenerator* indicates the amount of rhythms that is generated. It constructs a *RhythmGroup* object and initializes its lists of Rhythms.
- **RhythmGroup:** The *RhythmGroup* class stores a list of Rhythms. The *getStates* method converts the list of rhythms into a list of States. States contain a type property discerning them between pause and movement types. Furthermore, a time and duration property specify the temporal

constraints of the state. During the level generation, the generator converts these into sequences of movements, pauses, and jumps.

- **Rhythm:** The *RhythmGenerator* generates and configures the parameters of each rhythm randomly. The type of rhythm can be regular or random. The duration lies in the range of values between 5 and 11 seconds. The density of jumps in the rhythm varies between low, medium, or high. Between the rhythms, the generator inserts a pause element with a duration between zero and five seconds. The *generate-Jumps* method in the *Rhythm* class calculates the timings of each jump accordingly to these properties.

LevelPath Generation

The *LevelPath* initializes the *RhytmGenerator* and loops through the list of states and jumps obtained from it. The algorithm audits the type of state and uses a generational grammar to generate geometrical representations. A dissection of geometry grammar utilized follows next.

3.5.3 Geometry Grammar

The level generator uses the geometry grammar to convert the list of States into geometrical representations. Its task consists of mapping the *States* to symbols and subsequently onto a list of grammatical terminal and non-terminal symbols.
The final geometrical generations consist of platfoms and jumps with different elements placed on top. Platforms describe structures that support players walking on them. Jumps on the other hand, depict structures where players cannot walk on, but instead falls down and dies. To detect a player falling down, the generator creates an element termed as *DeathZone*. A description of each item depicted in the grammar of Table 3.1 follows.

move	$\rightarrow$	**Platform**
pause	$\rightarrow$	**Moving_platform** | **Path_change** | **Checkpoint**
jumping	$\rightarrow$	**Flat_Gap** | gap_action | enemy
enemy	$\rightarrow$	**Enemy_static** | **Enemy_moving**
gap_action	$\rightarrow$	**Jump_up** | **Jump_down** | **Bouncepad**

Table 3.1: Level generation grammar consists of a mapping into a list of non-terminal symbols and terminal symbols.

move $\rightarrow$ **Platform**

The move symbol directly translates into platforms with length as defined by the rhythm state.

pause $\rightarrow$ **Moving_platform** | **Path_change** | **Checkpoint**

The pause state has three possible realizations. It can generate either as a moving platform, a NPC, or a checkpoint. The generation of these elements can be omitted by the generator in case certain conditions apply.

- **Moving_platforms** are platforms that are moving between two fixed points. Furthermore, players can walk on top of these type of platforms. The probability $P = P_{progress} * P_{platforms}$ determines the amount of sequential platforms generating. It is chosen between one and four instances. The value $P_{progress}$ stands for the player's advancement through the level, and $P_{platforms}$ depicts the path's assigned probability of moving platforms. Figure 3.11 shows two sequential moving platforms. $P_{progress}$ singularly defines the movement speed of moving platforms. For multiple sequential platforms, the speed slightly increases for each platform due to playability reasons. Clouds generate as backdrop to moving platforms to give players a sense of movement on platforms.
- The **Path_change** symbol consists of the generation of a NPC, a portal, and a secondary path. Figure 3.12 displays the NPC with its portal that serves the purpose of teleporting the player onto the secondary path. Path changes and thus also NPC encounters can only appear on the main path. The players have no obligation to interact with NPCs.

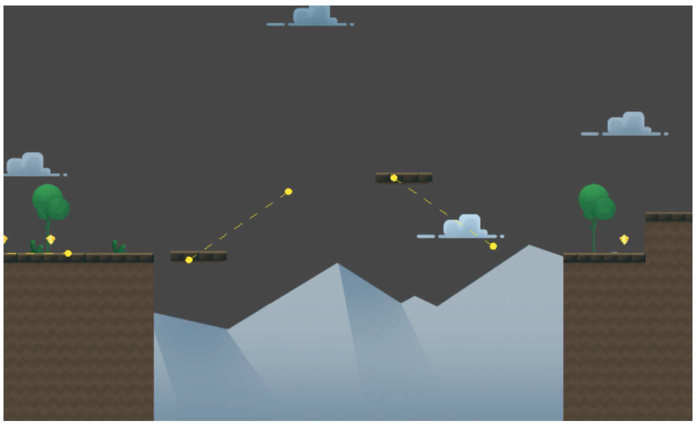


Figure 3.11: Two sequential moving platforms that move along the outlined path.

Instead, the NPC discloses the type of path to the players during the interaction. Furthermore, the interaction is necessary for the portal to appear and will perish if players interrupt the dialogue. The path type derives from the set of random probabilities assigned during creation. The type also specifies the appearance of the NPC. Figure 3.13 illustrates all types of NPC in the game.

- **Checkpoints** are elements that alter the players' existent respawn location to the position of the checkpoint. The generator places checkpoints only if the distance to the preceding instance exceeds 25. Figure 3.14 shows an instance of a checkpoint.

jumping $\rightarrow$ **Flat_Gap** | gap_action | enemy

The symbol jumping resolves to the terminal symbol *Flat_gap* or into the non-terminals *gap_action*, or enemy. The level generator makes the decision on a probabilistic basis. The subsequent items explain each one of the non-terminal symbols further.

The **Flat_gap** symbol poses the generation of a gap, the players ought to bypass by jumping across. The size of the gap is informed by the duration value that was defined in the rhythm generation. The level progress $P_{progress}$ is used to favour shorter jumps in the early game.

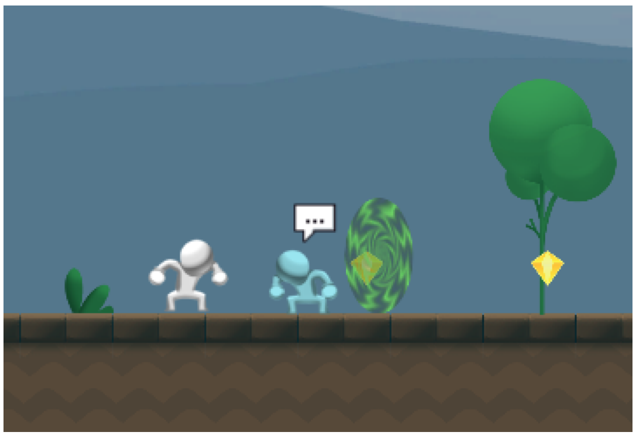


Figure 3.12: The path change structure consists of a NPC and its portal. The portal serves as teleporter onto the secondary path.

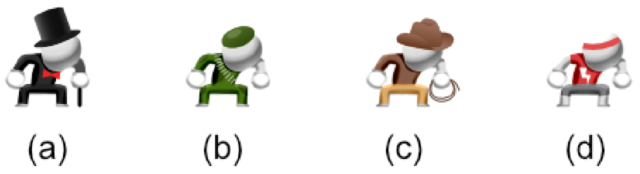


Figure 3.13: The type of NPCs consist of (a) coin, (b) enemy, (c) puzzle, and (d) short. They are placed with every path change element and hint to the type of path their portal leads.

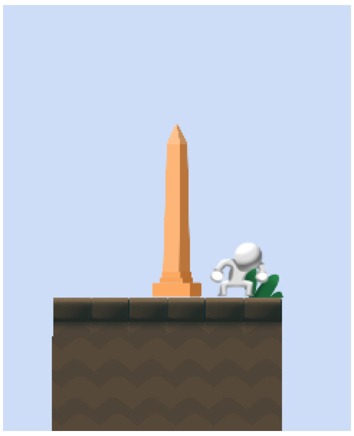


Figure 3.14: The checkpoint serve as respawn points for players in case of their death.

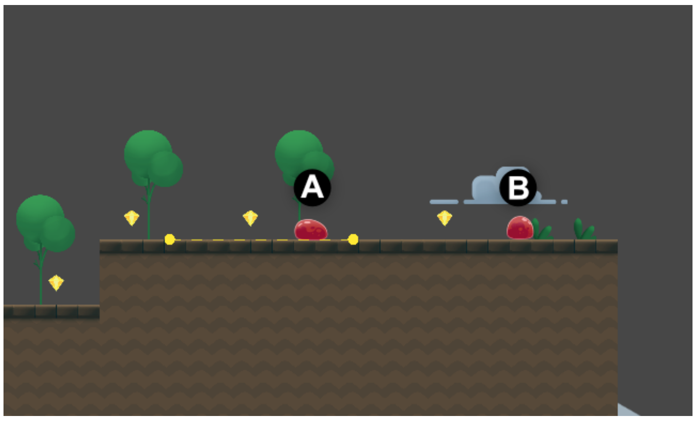


Figure 3.15: The game generates (A) moving and (B) static enemies

enemy $\rightarrow$ **Enemy_static** | **Enemy_moving**

The non-terminal symbol *enemy* resolves in generation of an enemies with differing adversity. The probability $P_{progress}$ defines the level of difficulty encountered by players. In the early game the generator tends to place enemies on a lowered platform. Through this measure, players know the range of enemies instantly. Furthermore, $P_{progress}$ makes the generator to prefer generation of static enemies early, and moving enemies later in the game. The difference between these types of enemies is outlined in the following. Figure 3.15 displays both (A) moving, and (B) static enemies.

- **Enemy_static** defines enemies without movement. They have a fixed position and occasionally perform jumps.
- **Enemy_moving** are enemies that patrol alongside paths. The path consists of two points with distance as defined by the duration of the state given. Moreover, the level advancement probability $P_{progress}$ contributes to the enemy's speed, which leads to a difficulty increase over time.

gap_action $\rightarrow$ **Jump_up** | **Jump_down** | **Bouncepad**

The symbol *gap_action* resolves into one of three possible realizations chosen probabilistically. The entire set of cases requires players to perform a jump.

- **Jump_up** lets the generator generate an obstacle where the players must jump up to overcome it. It is defined of a fixed height offset where players elevate themselves into a higher position.
- The **Jump_down** symbol generates a fall of random height the players need to traverse. The generation varies as conditioned on the level of the height chosen. If the height surpasses a certain threshold, a sign enclosing an arrow directed downwards and coins generate alongside the downward path. Both items serve as indicators that the players need to drop to a lower platform that can be too low to see.
  Treasure rooms are another type of element that may generate if the height threshold is crossed. They contain large amounts of collectible tokens. Treasure rooms generate in backward direction at the same height as the platform reached after the drop. Hence, the players need to drop down, turn around and walk backwards to enter the room. Figure 3.16 shows a geometrical representation of the state along with a generated coin path, sign, and treasure.
- The **Bounce pad** terminal symbol tells the generator to place a bounce pad object. Bounce pads give the players a jump boost that exceeds their intrinsic jump force. A gap of random with and height succeeds bounce pads. Furthermore, the generated jumps by no means take up the maximal jumping distance possible. Figure 3.17 shows an instance of a bounce pad.

Difficulty Adjustment

The game contains difficulty adjustments to keep the game challenging over time. A few different game components adjust as the player advances through the game. These include the speed opponents are moving in, the maximal length of jumps, and the amount and speed of moving platforms.

Outside the rhythmic level generation, there are further elements that generate statically. One of which, being the smart enemy, poses another challenge to players.

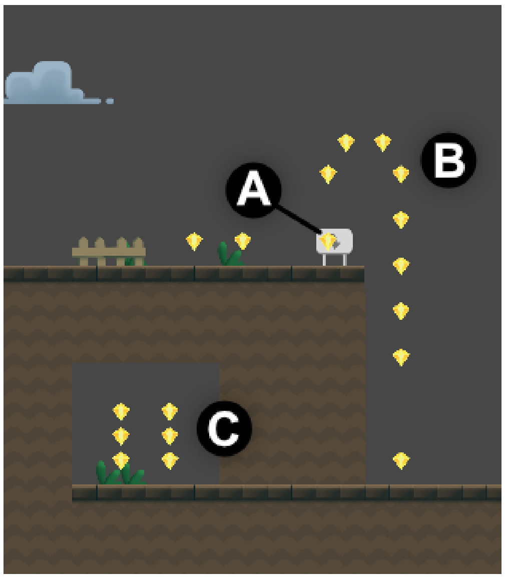


Figure 3.16: The jump_down structure can generate a downward directed coin path, along with a sign and an optional treasure.

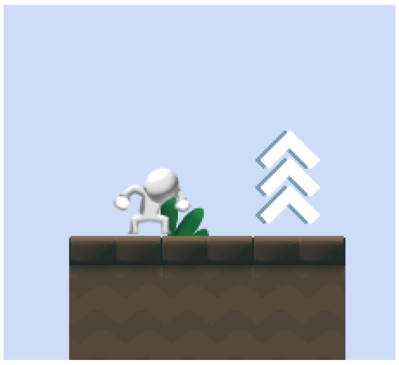


Figure 3.17: Bouncepad propell the players onto platforms that are beyond reach with the players' inherent jumping force.

3.6 Smart Enemy

The game features an opponent trained thought self-play RL. The Unity ML-Agents framework (Juliani et al., 2018) provides the necessary tools to facilitate the training of these types of agents. As already addressed, the generator generates the smart enemy in specific predefined locations. These locations consist of the ending of paths with predominant enemy probability, and on the main path as final hindrance before completing the game.

Actions and Observations

The player model of the 2D platformer template serves as the basis for development of the self learning agent. The adaptation of the model consists of implementing of two functions. The first of which receives a list of action values that are mapped to the agent's executable actions. These variables lie on a continuous scale and steer the agent's decisions regarding movement, jumping, and shooting.

The second function collects observations from the agent's environment. Using these observations, the agent decides the best course of action for the current state. The observations collected in this implementation consist of the following items:

- The distance between the player and the agent as x and y coordinates.
- The grounding state of the agent itself.
- The distance of the closest bullet emitted by the opponent as x and y coordinates.
- The movement bounds of the agent.
- Velocity of the opponent player.
- The grounding state of the opponent.

Constraints

The implementation of the agent utilizes constraints to simplify the learning task. Moreover, it keeps the agent within the bounds of what is perceived as reasonable strong for opponents. The bounds force the agent to not leave the area of movement that is bounded by two points. Furthermore, the agent can only fire in the direction of his opponent. Finally, all constraints that apply for players, also apply to the agent. For instance, both have a weapon cooldown.

Rewards

The reward distribution in self-play is binary. The victor of each episode obtains a reward value of one, while the looser receives a reward of zero.

Training

The ML-Agents uses its own the tool called *mlagents-learn* for learning. A yaml file configures the training hyperparameters. The ML-Agent tennis example serves as initial configuration of these parameters used for the learning. The example performs a similar task to the one that is required for the agent in this work. The hyperparameters were adapted during training according to the ML-Agent guidelines[4] and trial and error. The underlying neural network uses two layers with 80 hidden units each. The value of the β parameter selected steadily decreases the entropy during training.

The *Tensorboard* utility gives statistics during training. These are used to observe the learning progress of an agent and also indicate whether changes to the hyperparameters should be made. The ELO rating system monitors the progress of the agent during training. Due to the zero-sum rewards, the output of the reward graph is insignificant.

During training, two agents face one another on a single platform and try to kill each other, as is shown in Figure 3.18. Each episode, they spawn at distinct random positions. The episode ends when one player dies. They can die either by a bullet, or if the opponent jumps on top of the agent. The training fuses two different modalities, that alternate during training.

[4] `https://github.com/llSourcell/Unity_ML_Agents/blob/master/docs/best-practices-ppo.md`

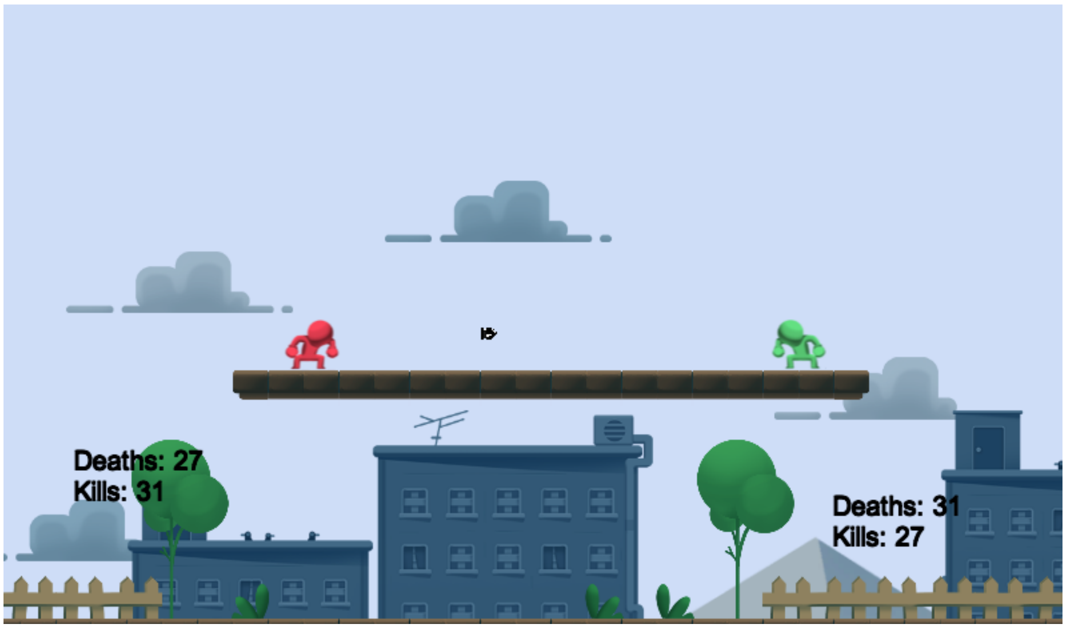


Figure 3.18: The training scene opposes two agents facing one another and trains them through self-play.

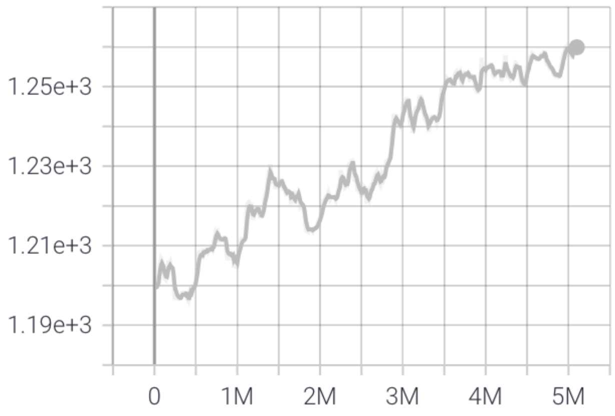


Figure 3.19: The ELO training progress of the final trained model.

- The first modality enforces that the agent can shoot only when he is grounded. Unfortunately, this has the effect that the agent ceases to jump over time. Which has the undesired effect, that the agent can no longer deal with a jumping opponent.
- In the second modality, the agent must be off-ground to shoot. The negative effect of this latter modality, lies in an agent jumping all the time. Which makes the opponent less believable to players. Simultaneously, the opponent is less capable fighting grounded opponents.

The training of the final model took a total time of 2 days and 15 hours. Figure 3.19 shows the learning progress of the ELO score over time.

With the Smart Enemy, the level generation task completes. The next major component of this work describes how the framework records individual behavioural information.

3.7 Player Analytics

The game collects player analytics in the background during gameplay inside a shared instance of the *AnalyticsController* class. The class contains three types of information. It stores player statistics, gameplay events, and meta information concerning the level generation.

3.7.1 Game Parameters and Statistics

The analytics file contains the items regarding the game's parameters and player statistics. The game parameters contain the values that define the level generation. The variable *n_states* indicates the amount of states generated on the main path. The *seed* specifies the initialization vector of the random generator used. The parameter *n_paths* describes the amount of secondary paths generated. Whereas, the parameters *numberPathShort*, *numberPathEnemy*, *numberPathCoin*, *numberPathPuzzle* contain the number of paths of each type generated.
Further relevant items are the variables *numberDeathZones*, and *numberOutOfGame*. They keep count of the amount of in- and out-of-game death zones.

The number of victory zones is unimportant, as per definition only one can exist per level.
Besides the parameters that influence the generator, the analytics contain a number of variables that count the amount of items generated. The amount of collectible tokens are stored in the variables *numberTokens*, *numberHealth*, and *numberWeapons*. Furthermore, the parameters *numberTokensSeen*, *numberTokensCollected*, *numberHealthSeen*, *numberHealthCollected*, *numberWeaponsSeen*, *numberWeaponsCollected* store the number of collected and appeared items for each type of collectible respectively.
Another type of objects generated are non-collectible items. These are counted in the following items inside the analytics.

- The variable *numberCheckpoints* indicates the amount of checkpoints generated.
- The variable *numberMovingPlatforms* indicates the amount of moving platforms generated.
- The variable *numberPortals* indicates the amount of portals generated.
- The variable *numberTreasures* indicates the amount of treasures generated.
- The variable *numberBouncePads* indicates the amount of bouncepads generated.

A further class of elements generated are the game's enemies. The player encounters normal and smart enemies in the game. The variable *numberEnemies* stores number of normal enemies generated, while *numberEnemiesSeen* indicates the amount of normal enemies and seen. The same holds for smart enemies and the items *numberSmartEnemies*, and *numberSmargEnemiesSeen* respectively.
The variables *distance* and *distance_back* indicate the distance travelled in forward and backward direction by the player respectively. Furthermore, the overall *duration* of the game is stored.
The analytics file furthermore holds the players specified responses of the questionnaires, alongside with the time that it took them to complete them before advancing. The variables *key* and *gender* store the demographical information requested in the first questionnaire. A list of numbers stores the ten BFI-10 items and another list consisting of 17 items records the values set in the GEQ in-game questionnaire.

Name	*Type*	*Description*
Event type	*EventType* enum	Event identification
Timestamp	float	Time
Coordinates	Vector3	Position of the element
Additional value	float	Additional information like element IDs
Path information	Object	Identfication of path, and path probabilities

Table 3.2: Analytics event variable with corresponding types.

3.7.2 Game Events

Events add a temporal context to the players' actions. They are triggered in specific situations throughout the game. For instance, firing a bullet provokes the release of an event. Each event contains a number of properties aimed at storing additional information. The base configuration of each event consists of a *type* definition and *timestamp* indicating its temporal occurrence. The full set of items is listed in Table 3.2.

Besides the primary information enclosed, events can infer additional information about the player's actions during the game. For instance, the coordinates of player deaths allow the deduction of possible relations between them. Such information can serve as an indicator that a player struggled multiple times at the same obstacle. A description of each event and their relations follows.

Enemy Events

Enemy events describe all events that occur while interacting with normal enemies. They consist of events that arise when player and enemy or enemy and bullet collide.

- **EnemyCollision:** The event indicates that an enemy collided with the player. Such an event entails either the death of the player or the enemy.

- **DeathEnemyTopJump:** This event serves as further indication that a player jumping on top of an enemy induced its death.
- **DeathEnemyBullet:** A player bullet killed an enemy.

Smart-Enemy Events

Smart enemies differ from conventional enemies in their abilities and danger posed to the player. Thus, also their repertoire of events that can occur differs.

- **SmartEnemyBulletFired:** Smart enemies have the ability to fire bullets. The event indicates that the smart enemy launched a bullet.
- **SmartEnemyTopJump, SmartEnemyShot:** The smart enemy can be hurt either by being shot with a player based bullet or if the player jumps on top of it. The events that arise in these cases are listed respectively.
- **HealthSmartEnemyLost:** The smart enemy lost a life, in the case of damage and possession of more than a single life. The event ensues the prior listed event.
- **SmartEnemyDeath:** This event arises when a smart enemy dies. It occurs either when a bullet hits or the player jumps on its top of the enemy.
- **SmartEnemyShown, SmartEnemyHidden:** Both of these events capture the moment the UI of the smart enemy appears or disappears on the screen of the players. They give an estimate of the time players were aware of the smart enemy.

Player Events

Player events revolve around the behaviour of players. They give insights into the players' actions by recording movements and interactions with certain elements of the game. Furthermore, they record the cause of death of players.

- **GameStarted, Spawned, PlayerDeath, Victory:** These events give an estimation of the time the player was alive.

- **DeathKillzone, DeathKillzoneOutsideGame:** The player fell off a platform and died due to a collision with a killzone. The latter event serves as a distinction between areas where players accidentally fell off a platform and the ones where they did not. And thus, it serves as detector of the players' explorative movement by going where they aren't supposed to.
- **PlayerDeathEnemy:** An enemy killed the player.
- **PlayerBulletCollision, PlayerDeathBullet:** Player got hit by a bullet and player died from a bullet respectively. These types of bullets can only originate from the smart enemy.
- **BulletFired, BulletNotFired, BulletNotFiredNoWeapon:** Player tried to fire a bullet, before or after the cooldown time fully elapsed respectively. The last event indicates that the player tried to fire a bullet, but does not yet yield an active weapon.
- **DirectionChange:** The player switched the direction of movement.
- **KeysArrow, KeysWASD:** These events map to the two possible keyboard layouts the game supports.
- **MovingButtonsDown, MovingButtonsUp:** The timings of pressing the movement buttons give insight into the time spent moving or standing still.
- **Jumped:** The player jumped.
- **JumpPad, Jumped:** Time between the player touching a jump pad and releasing the jump button. The player cancelled the bouncepad jump
- **TouchMovPlatformStart, TouchMovPlatformFinish:** The players' time spent on moving platforms.
- **Checkpoint:** A player collided with a checkpoint, and thus the respawn position changed.
- **Victory:** Player reached the victory zone, meaning the player finished the game.

Game Events

Further events are recorded as the players play. They do not have any impact on the game, but are important for analysis.

- **GamePauseStarted, GamePauseFinished** indicates whether the player has opened the pause menu during game. These events needed to find the correct timespans between events. It is needed as the time does not stop counting while the game is paused.

Collectible Events

Collectibles emit events whenever a player interacts with them. They contains the exact position of the collectible token collected.

- **TokenCollected, PlayerWentDownBack:** The player collected tokens, or found a treasure leading to the collection of even more tokens.
- **HealthCollected:** The player gained an additional life by collecting a heart token.
- **WeaponCollected:** The player collected a weapon token, and is henceforth able to shoot bullets.

Puzzle Events

Puzzle events are related to the puzzle that is generated at the end of secondary paths of type puzzle.

- **EnteredPuzzle, FinishedPuzzle:** These events describe that the player set foot into, or completed a puzzle. From it the time spent solving the puzzle is inferred.
- **TouchBoxStart, TouchBoxFinished:** The player interacted with a box inside the puzzle. It is used for the calculation of the time the player pushes boxes.
- **TouchBoxReachedVoid:** The player pushed a box into a DeathZone. As direct consequence, and in order to prevent the player from being stuck, the box respawns after a short delay.

Dialogue and NPC Events

Temporal durations and interactions of the players with the games dialogues and NPCs. Besides an initial dialogue at the beginning, dialogues appear when approaching an NPC, or a smart enemy.

- **NPCDialogueSet and NpcDialogueFinished**: The events record the total time of dialogue
- **NpcDialogueUiShown and NpcDialogueUiHidden:** These events indicate the visibility of the conversational user interface to the players.
- **NpcDialogueEnter and NpcDialogueEscape:** The player advances or terminates the Dialogue.
- **DeathNPC:** An NPC died from a bullet with player based origin.
- **PortalAppeared, PortalDisappeared, PortalTeleport:** Time the portal was visible before the player either uses it or it is hidden again.

Tutorial Events

While players play the tutorial, the game emits tutorial events.

- **TutorialStart and TutorialFinished:** Both events serve to measure the players' total time spent in the tutorial.
- **TutorialShow and TutorialHide:** The events enable measuring the time the player has spent interacting with each segment of the tutorial process.
- **TutorialAllowedKeysPressed and TutorialNotAallowedKeysPressed:** The player interacts with the tutorial and either advances or terminates it.

Post completion of the game, the *JsonUtility* API serializes the analytical data and prepares it for transmission to the data server. The data server component gathers the complete set of the players' behavioural data in a single instance and prepares it for the analysis step.

3.8 Data Server

A server stores the players' behavioural data. The data is serialized into the JSON format and compressed to lower memory usage and enable faster transmission. The implementation utilizes Microsoft's Azure Blobs for storage. It uses the StorageServices[5] framework in combination with the RESTClient[6] framework to communicate with the servers. Both are non-Microsoft third party frameworks that work with Unity, and enable communication to Microsoft's Azure Blobs. The data server serves as storage before analysing the player data.

3.9 Data Analytics

The *Data Preparation Module* decompresses and parses the players' behavioural data. It prepares the data and metadata from the analytics as outlined in the prior section. Once completed, the *Data Analysis Module* analyses all variables through correlational measures, and attempts personality prediction and to create of player profiles. The results of the analysis informs the player modelling task inside the framework. As a final task, the *Data Visualization Module* visualizes the behavioural information.
The player profiles established by the *Data Analysis Module* drives the adaption process of the game.

3.10 Adaptation

The profiles extracted from the players' interactional data can offer insights into what components of the game appeal to players. The game possesses various methods that can drive adaptation in the game:

- The level generation parameters P_{coin}, P_{enemy}, P_{puzzle}, P_{length} tune the type of content generated.

[5]`https://github.com/Unity3dAzure/StorageServices`
[6]`https://github.com/Unity3dAzure/RESTClient`

- Parameters of game content alter properties inherent to entities in the game, such as the speed of enemies, or the speed of moving platforms.
- Player parameters change the behaviour of the player character. They include items such as speed and jump force.
- Physics parameters of the game alter the behaviour of all items and entities of the game.
- The behaviour of the AI of the smart enemy can be altered through parameters or a model swap.
- The focus of the player can be shifted toward various game goals. The UI needs to mirror these changes.

3.11 Summary

This chapter proposed the development of a 2D Jump'n'Run game that reacts on the players' behaviour. In order to build the game, the *Platformer Microgame Template* was extended with additional content. This process has the intention of expanding the amount of elements to better encourage and support different play styles. Furthermore, it discussed the overall architecture and liaison between the components of the system. As part of the study, the game administers questionnaires regarding player demographics, personality (BFI-10), and game experience (GEQ).
The game features a UI component that indicates important information such as play time, the amount of tokens collected, the type of weapon wielded, the number of lives left and the amount of lives left for opponents. Furthermore, it contains a dialogue and tutorial system.
As part of additional content added to the game, a novel opponent was trained with reinforcement learning. This chapter discussed its components and implementation using the ML-Agents framework. The opponent, labelled as *Smart Enemy*, adds another comparatively challenging opponent to the game.
The levels of the game generate procedurally, and are tunable on a probabilistic basis. This serves the purpose of tuning the game to the expectations of the players during adaption. During gameplay sessions, player data is collected. After the completion of the game, the data is sent to the data collection server. The analytics section dived into each and every statistic

and event collected. Additionally, it discusses the information contributed by various combination of events.
The task of the data analysis component lies in combining player data. Afterward, the exploration of player data based segmentation into profiles is probed. Its task also lies in correlating and visualizing the information collected. Moreover, it explores usage of machine learning techniques for this task. The comprehensive analysis of the player behavioural data follows in the next chapter of this thesis.

4 Preliminary Analysis

The analysis of this work is strictly preliminary. Therefore, the validity of the results must be further investigated using more data samples. The chapter goes into detail on the dissection of the data collected and used by the analysis tool for player analysis. The main task lies in the exploration of establishing and verifying player profiles from behavioural information.

4.1 Methodology

The methodology goes into detail on the goals of this thesis. Consequently, it outlines the composition of the participants who took part in the study. The materials considers the questionnaires and game data that were collected during the study. The study procedure itself describes how the participation procedure transpired. And finally, methods used for the investigation of the research goals are presented.

4.1.1 Research Goals

The aim of this work lies in uncovering relationships between player behaviour and content with the players' true personality and their game experience. On this premise it attempts to make a preliminary estimation of player profiles based solely on behavioural interaction information of players of a game.

(I) Is there a direct relationship between the big 5 personality items and the players' game behaviour?

(II) Do players with certain personality traits interact with different elements of the game?
(III) Can behavioural data be used for personality prediction?
(IV) Can behavioural player profiles be inferred from the data?

Additionally, the game experience is examined in this work. Due to the presence of a very strong opponent as the final obstacle in the game, frustration may emerge as a consequence.

(V) How did game behaviour influence the players' experience?
(VI) How was the players' experience when interacting with different elements?

4.1.2 Participants

The study was conducted between 13 April 2022 and 27 April 2022 with 30 participations. The participants were either approached through the platform *Discord* or asked directly for their participation. With 23 individuals, male participants dominated the study and only 6 *female* participants and a single participant who identified themselves as other. The age range lies between 21 and 56 years ($\mu = 30.3$; $\sigma = 9.2$).
According to the results of the *GEQ*, shown in Figure 4.1, players were more likely to experience positive rather then negative affect while playing. In addition, the game did not feel very tense for the players. It was slightly above average challenging and competent.

4.1.3 Materials

During the study, participants received three questionnaires and had to play a Jump'n'Run game that collects their behavioural interaction data in the background.

1. The first questionnaire collects the *demographical* information from participants. More precisely, it collects their age and gender of identification.

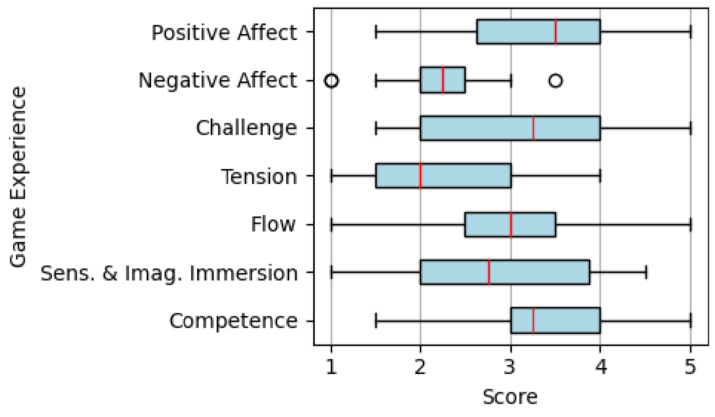


Figure 4.1: The players' rated game experience as a result of the GEQ questionnaire.

2. The secondary questionnaire evaluates the true *personality* of participants. It uses the BFI-10 questionnaire, which estimates personality by assigning each one of the five factors a value between one and five. The factors are Neuroticism, Extraversion, Openness, Agreeableness, Conscientiousness.
3. The last questionnaire evaluates the players' experience with the game. It consists of the factors competence, sensory and imaginative immersion, flow, tension and annoyance, challenge, negative affect, and positive affect.
4. The game stores a list of statistics and game events that contain further meta-information. Using this data, the analysis tool extracts behavioural measurements of players.

4.1.4 Procedure

The study is performed by participants on a voluntary basis and proceeds as follows. Initially, participants were asked to open the study website[1] using a web browser. The study consists of three major parts, consisting of an

[1]https://mich312.github.io/master/

assessment of the players' personality, the game, and the resulting player experience. As participants opened the link they received, a short description of the purpose of the study was presented to them before starting. As they continued, they had to answer demographic questions. Subsequently, the BFI-10 questionnaire evaluated their true personality, as according to the five factor model. After the first set of questionnaires, participants proceed with playing the game. While playing, the game compiled behavioural data collected by game analytics in the form of statistics and events. After the game, the GEQ questionnaire evaluated the payers' experience playing the game. With completion of this final questionnaire, participants completed their session. As a final step, they had to wait until the collected data were successfully transmitted and stored on the data storage server. The participant then could decide to replay the game or to end their participation.

4.1.5 Methods

For investigation of the research goals the three methods correlation, regression, and clustering were used.
Correlation was used to investigate the direct relationship between the measurements and player behaviour. For this task, *Pearson's Correlation Coefficient* was used. The method yields the value r as a result. The value of r must be distinguished between strong correlation ($r > 0.5$), medium correlation ($r > 0.3$) and weak correlation ($r > 0.1$) is met (J. Cohen and P. Cohen, 1975, p. 56). Moreover, with a p-value less than 5%, r is deemed as statistically significant.
For parameter prediction, regression models were used. They were combined with feature selection methods to reduce the dimensionality of the data and to attempt in overcoming the curse of dimensionality. As quality measurements, the MSE, MAE, R^2 scores were used.
Clustering was used as unsupervised method to find distinct groups of data inside the behavioural data. The algorithm *KMeans* was used to identify n Clusters $C_1...C_k$ from the data. KMeans is suitable for finding models of player behaviour (Bauckhage, Drachen, and Sifa, 2014). The silhouette coefficient in combination with a silhouette analysis was used to estimate the quality of the clustering model obtained.

	Strong Corr.		*Medium Corr.*		*Weak Corr.*	
Factor	Pos.	Neg.	Pos.	Neg.	Pos.	Neg.
Extraversion	-	-	2	7	-	-
Agreeableness	1	-	1	2	-	-
Conscientiousness	1	-	1	-	-	-
Neuroticism	-	1	-	-	-	-
Openness	-	-	2	-	-	-

Table 4.1: The number of strong, medium, and weak correlations for each item of the big five factors with game behaviour.

4.2 Results

As part of the results the data used in an attempt to give a first insight to the research goal is listed and discussed. Due to the data limitations the validity of the results has to be re-evaluated using a larger scale study.

4.2.1 Research Goals

(I) **Is there a direct relationship between the big 5 personality items and the players' game behaviour?**

The direct relationship was investigated with the Pearson's Correlation Coefficient. An overview of all types of correlations with each item of the BFI-10 questionnaire are shown in Table 4.1. The factor *Extraversion* had two positive and seven negative medium correlations. *Agreeableness* had one positive strong correlation, one positive medium correlation, and two negative medium correlations. *Conscientiousness* had one strong and one weak positive correlation. *Neuroticism* had one negative strong correlation and *Openness* one positive medium correlation.

- Extraverted players were less likely to take puzzle paths (r=-0.491; p=0.008) and jumped less often (r=-0.464; p=0.013). Additionally, they were less often hit by bullets from the smart enemy (r=-0.443; p=0.018), and thus died less often by a bullet (r=-0.423; p=0.025). They used the same bounce pad more than once (r=0.379; p=0.046).

- Players with higher *agreeableness* were more likely to push boxes off platforms (r=0.679; p=0.0). They had a lower likelihood of dying from falling off platforms (r=-0.403; p=0.033). They were more likely to kill smart enemies by jumping on their top (r=0.484; p=0.009) and less likely to kill them by shooting them (r=-0.487; p=0.009).
- *Conscientious* players spent more time in the tutorial (r=0.405; p=0.033) and were more successful in doing damage to the smart enemy by jumping on top (r=0.5; p=0.007).
- *Neurotic* players did not kill the smart enemy mainly by jumping on top (r=-0.545; p=0.003)
- Players that scored as more *open* tried to collect the weapon placed after the smart enemy first, then killed the smart enemy and finally went through the portal (r=0.478; p=0.01). As a consequence, they had to use more jumps to defeat the smart enemy (r=0.488; p=0.008).

(II) **Do players with certain personality traits interact with different elements of the game?**

For the examination of the direct relationships between personality and different content of the game correlation was used. Only two factors had correlations with game elements. *Extroverted* players were found to skip puzzle paths (r=-0.491; p=0.008) and more likely to use bounce pads multiple times (r=0.379; p=0.046). Players with higher *Agreeableness* were more likely to push boxes off platforms (r=0.679; p=0.0).

(III) **Can behavioural data be used for personality prediction?**

In order to predict personality, each item of the Big five personality inventory estimated using regression models. Due to the high dimensionality of the behavioural data feature selection methods were applied.
The data was initially split into a training set and a test set. The training set was used to train a model with cross-validation using a 3-way split between the training and the validation set. During cross-validation, the models were unable to reach a consistent positive R^2 score. Thus, the difference of variance between the prediction and the true labels was poor. The same applies to the R^2 score on the test set. Therefore, no model of personality

n Clusters	Model	Silhouette Coeff. (SC)	N Features
2	No feature selection	0.24	61
	Some feature selection	0.254	52
	Strong feature selection	0.362	16
3	No feature selection	0.199	61
	Some feature selection	0.219	52
	Strong feature selection	0.395	16
4	No feature selection	0.203	61
	Some feature selection	0.221	52
	Strong feature selection	0.4	16
5	No feature selection	0.178	61
	Some feature selection	0.2	52
	Strong feature selection	0.393	16

Table 4.2: The number of obtained clusters during clustering with *KMeans* with their respective silhouette score with various levels of feature selection.

prediction from the players' behavioural information could successfully be inferred.

(IV) Can behavioural player profiles be inferred from the data?

For analysis of obtaining player profiles, the clustering method KMeans was used. Table 4.2 illustrates the effect of feature selection on the behavioural dataset and the silhouette coefficient. Furthermore, Figure 4.2 shows the silhouette analysis for KMeans using between one and five clusters without feature selection. Overall the average silhouette score μ_{SC} is low for all models. According to the silhouette graphs, the performance of the models decreases as the number of clusters increases.

(V) How did game behaviour influence the players' experience?

There was a medium correlation between players that experienced tension and the number of times that they died through a bullet from the smart enemy (r=0.393; p=0.038). Furthermore, tension had a medium negative

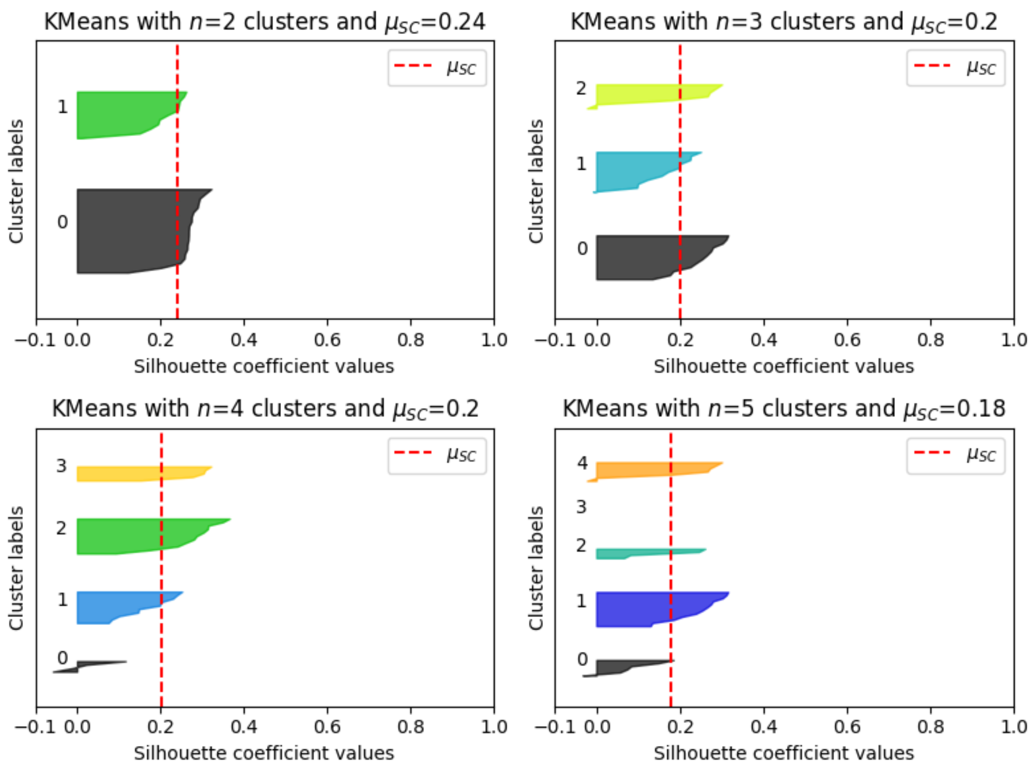


Figure 4.2: The silhouette plots for KMeans with two, three, four, and five clusters.

correlation with average time alive (r=-0.418; p=0.027).
Competence had a medium negative correlation with their average duration of stationarity (r=-0.454; p=0.015).
Players that experienced a high amount of flow, did less jumping while fighting the smart enemy (r=-0.375; p=0.049).
Challenge was experienced by players who waited for the game controls to appear during the tutorial (r=0.438; p=0.02). The number of interactions with boxes (r=0.46; p=0.014) and solving a puzzle (r=0.482; p=0.009) also had medium positive correlations with challenge. Less challenge was experienced by players that moved faster (r=-0.377; p=0.048) and jumped less often (r=-0.482; p=0.009).

(VI) **How was the players' experience when interacting with different elements?**

The *Challenge* score had the most notable correlations with game elements. It was higher for players who spent more time in puzzles (r=0.482; p=0.009) and also on moving platforms (r=0.461; p=0.014). Players who did not interact with NPC dialogues also experienced more challenge (r=0.445; p=0.018). For the opposite case of higher interaction in dialogues, the challenge was lower (r=-0.445; p=0.018). In addition, a longer average dialogue length (r=-0.377; p=0.048) correlated negatively with challenge.

4.3 Discussion

Due to the small sample size of data from this preliminary study, the results must be considered with this limitation in mind. Some questions could be answered better than others.

4.3.1 Game Behaviour Correlations

A portion of the correlations fell short due to the data quality and data sample size. For example, extraversion had a medium correlation with

P(treasures visited) (r=-0.4; p=0.035). A closer investigation of the relationship revealed that only one person did not collect treasures. Coincidentally, this person had also scored the highest value on the extroversion scale. However, there have been relationships that could successfully be related to items of the big five personality inventory.

- *Extraverted* players skipped puzzles, and did less jumping. They also failed to avoid enemy bullets, and died more often by them. Hence, higly extraverted players may take more risks than introverted ones. Which, is in accordance with the literature (Nicholson et al., 2005).
- *Agreeableness* on the other hand had fewer deaths that were caused by falling off platforms. This behaviour might be described as considerate, which is related to Agreeableness (Caspi, Roberts, and Shiner, 2005).
- *Conscientious* players were found to spend more time in the tutorial, which may be an indication of carefulness. They killed the smart enemy by with a precise jump on its top. Both are aspects linked to the trait Conscientiousness (D. Schultz and S. Schultz, 2016).
- *Neurotic* players were less likely to kill the smart enemy by jumping on top. Players might lack the confidence to undertake such an endeavour. Coincidentally, Neuroticism additionally negatively correlated with competence, which also is consistent with the literature (Caspi, Roberts, and Shiner, 2005).
- Players with a high *Openness* cleverly collected the weapon before fighting the smart enemy. As a consequence, they had to do more jumping. This can be seen as risk taking (Nicholson et al., 2005) and daring (D. Schultz and S. Schultz, 2016), which both are associated with the openness trait.

Besides these relationships, there have been a few presumptions that yielded no results. The game was built with the expectation of having a relationship between extraversion and NPC interactions. Unfortunately, this relationship could not be corroborated. The same holds for openness and the amount of exploration of the game. These shortcomings could be associated with bad level design, but also with wrong expectations of how these traits express themselves in the game.

Player experience overall had better correlations with player behaviour. It primarily emphasised the more challenging elements of the game. The longer players spent in places like the puzzle and on moving platforms and the higher the amount of deaths, the more challenging they perceived the game. The correlations with tension indicates a relationship with the smart enemy. Players who died more often through a bullet from the smart enemy experienced more tension, and hence the smart enemy could be a source of tension in the game. In order to corroborate these presumptions, a further analysis is needed.

4.3.2 Regression

The regression task was unable to obtain a stable and positive R^2 score. With each run, the value fluctuated strongly. This may be due to the high dimensionality and low data sample size. Thus, the regression overfits and would need a further investigation with more data.

4.3.3 Clustering

The best result, with enough interpretability, that was obtained during clustering used a a cluster size of two, no feature selection, and a silhouette score of $SC = 0.243$. The low silhouette score may be a fragment of high-dimensional data and a small sample size. Nevertheless, the manual analysis of these clusters yielded promising results.

In order to obtain descriptions of each cluster, the cluster centers for the models between two and five clusters were compared to one another. The feature selection was discarded due to being too aggressive and impeding the interpretation of the clusters.

For $k = 2$ clusters, the main differences between the centers all related to behaviour associated with the enemy path. Thus, one cluster indicates players who decided to go on the enemy path, whereas the other cluster consists of players who did not.

The same holds for $k = 3$ clusters. There is one cluster center that differs from the other two in behavioural measures related to the enemy path. The difference between the other two centers consists of measures of all other types of paths. Hence, the first cluster is related to players that mainly went on the enemy path. The second cluster consists of players who walked the entire set of paths, whereas they had a lower probability of selecting the enemy path. The last group consists of players that are less likely to go on any path at all and do not fully explore the level. They also had a harder time because of a high accuracy of the smart enemy at the end of the level.

For $k = 4$ clusters, the cluster differences differ in comparison to $k = 3$. There still remains a cluster that indicates players that mainly went on the enemy path. Another cluster specifies players who jump more frequently and are less likely to visit any path. The final two clusters consist of players who went on all paths, except the enemy one. The difference between these lies in their application of movement keys and the average duration of interacting with elements like moving platforms, puzzles, or dialogues.

The resulting clusters for $k = 5$ are less distinctive than with $k = 4$ clusters. An attempted description of the clusters is as follows:

1. Players that primarily went on the enemies path, and killed the smart enemy by jumping on top.
2. Players that likewise primarily went on the enemies path, but fought the smart enemy by shooting.
3. Players that went on all paths but the enemy one. Since they collected no weapons, they defeated the final smart enemy by jumping.
4. Players that mainly went on all paths but the enemy one. According to the behavioural data, they died more often and needed more time on moving platforms or puzzles. Thus, they might be more inexperienced either with the game or gaming in general.
5. Players that just walked the main path and did a lot of jumping.

The profiles obtained by clustering need to be further verified. A direct relationship between the clusters and personality has not been found in this work. Additionally, a trained classifier may be useful in assigning players a group and driving adaptation on these grounds. The next chapter covers how these obtained models and correlations for player behaviour could be used to drive adaptation.

5 Adaptation

Based on the research conducted in this thesis, this chapter provides an overview of approaches for driving adaptation. The effects of the proposed adjustments are purely hypothetical and should be investigated further in a separate study.
There are two main modalities to consider for adaptation. Both, online and offline adaptation are applicable for this task. Both strategies are comparable to the affective loop proposed by Sundström (2005), except instead of modelling affection, they model personality or the game experience.

1. In the *online adaptation*, the model is estimated during the gameplay, and directly informs the adaptation algorithm.
2. *Offline adaptation* builds the player model only after completing a level. Consequently, while creating a new level, the generator only takes this player model into account.

Adaptation can be driven by modification of the level generation parameters and path probabilities. The speed of elements such as enemies, bullets, and moving platform may be adjusted further. The mechanics of the player and the physics of the game provide players with different capabilities. The AI of the smart enemy can also be adjusted or its model replaced with another one. Another way to adapt the game is by changing its presentation and goals.

5.1 Personality

For personality-based adaptation to work, there must be a measure that can estimate the players' personality. The following list gives an overview of adaptations based on each personality item.

- Due to the fact that *extraverted* players were less likely to visit puzzle paths, the generator may cease or reduce their generation. Furthermore, they had a hard time evading bullets, which might be due to their risk taking disposition. In order to make the game less difficult for them, adaptions regarding the enemy path and the smart enemy may be implemented. The aim of the smart enemy and the cooldown of its weapon may be modified to give players more time to react to bullets. A more frequent generation of additional life tokens would also decrease difficulty. In view of risk taking, they may enjoy elements that forces them to take risks, which in turn yields positive rewards.
- *Agreeable* players were more successful in avoiding bullets. Because this behaviour is similar to the behaviour of extraverted players, the smart enemy may be altered to increase difficulty. Furthermore, the smart enemy may be spawned more frequently.
- *Conscientiousness* in players surfaced as carefulness, and preciseness. As a result, they could enjoy the informative value of game dialogues or game elements where they need to perform a sequence of accurate jumps. In addition, the gun can be taken out of the game to force them to defeat the smart enemy by jumping on their top.
- *Neurotic* players lack the confidence for jumping on top of the smart enemy. They are less likely to get into that predicament by giving them the gun earlier. Additionally, more live tokens can help to overcome their lack of confidence and help suppress their negative emotions because they are less likely to die.
- *Openness* helped player playing clever and find creative solutions to problems. By collecting the weapon prior to overcoming the smart enemy, they were more effective. Moreover, due to their risk taking, they may enjoy elements that force them to take a chance and yield positive rewards.

5.2 Game Experience

Game experience prediction might be used as a detector to determine whether adaptation is required to avoid an unpleasant gaming experience. Challenge has correlations with the time spent during the tutorial and with

elements such as moving platforms or the puzzle. One probable explanation may be that these players have less gaming experience. On these grounds, the game may attempt to alleviate challenge by adjustment of other components. For example, the speed of moving platforms may be decreased, as could the size of gaps between platforms. As a consequence, the number of deaths caused by falling off platforms may be reduced.
The relationship between tension and the amount of player deaths caused by the smart enemy may suggest that its efficacy needs to be reduced. Lower bullet speeds and longer weapon cooldowns may give the player more time to react and respond.

5.3 Player Profiles

Because they are based on real-world behaviour, player profiles extracted from player data have a high relevance. In the following, this thesis attempts to give some notions on how to adapt the game for five different player types from the clustering process.

- For the two groups of players who mostly travelled on the enemy path, the enemy probability P_{enemy} may be increased for all paths. As a result, more enemies generate on all paths. Furthermore, the smart enemy might be generated more frequently. However, in that case, there should be more behavioural variance between them.
- Players that choose to travel on all paths might enjoy fewer enemies. Hence, they may benefit from a decreased enemy probability P_{enemy} or outright omission of enemy elements.
- There is one group that seems to have less gaming expertise. To prevent them from being mentally overstrained, challenging elements may be transformed into less difficult versions.
- The group of players who choose not to go on any path may be uninterested in exploration of the game. For them, the goal of the game may be shifted towards speed-running the main path. With changes to UI components, this purpose may be emphasised even more. The playing time could be made the most prominent element,

and a leaderboard after completion of a level might underline this purpose.

6 Lessons Learned

This chapter reviews the lessons learned while working on the theoretical, implementation, and analysis chapters.

6.1 Literature Research

There is no agreed upon definition of personality in the literature. Rather, there are a variety of techniques aimed at describing relatively static behavioural differences between individuals. The big five personality model has become the predominantly used inventory among these. With this inventory, relationships between gameplay, play styles, and parameters have been found. Affect on the other hand, is much more difficult to discern. Due to its temporal range and its compound quality, it is also more complex.
There is a trade-off to be considered when choosing between model-based and model-free methods. While the first type of strategy relies on expert knowledge of the model, the second is more difficult to generalize and interpret.
In the adaptation process, modelling is combined with adaptation methods. By maximization a given objective is optimized. This objective is contingent on what quality of the game must be improved through adaptation.
PCG is already used in games, and the level of complexity varies depending on the type of content that needs generating. Many techniques to level generation learn from given sets of examples to produce new variants of the same type of content. This restricts the genrator's expressive range, making it harder to adopt novel content into the generation. Experience based PCG bridges the gap between game modelling and adaptation.
Self-play using RL has shown to be as an effective method for training an opponent for a variety of games. Highly complex games such as Starcraft 2

(Blizzard Entertainment, 2010) and Dota 2 (Valve, 2013) have been mastered by using this method.

6.2 Implementation

The advantage of Jump'n'Run games is that they are readily extendable, which is useful for testing various types of components. On the other hand, the data collection can quickly become quite complex. As a consequence, the development and the testing complexity rises, especially in combination with PCG. During the development of the game, certain features have been omitted due to a too high effort of associated with their implementation. In addition, questions about game design came up during development. For instance, regarding the frequency and location of the generated tokens. Performance was another issue during the developmental phase. Colliders, a significant tile map complexity, and the RL agent consumed a lot of system resources. The lag issue concerning the smart enemy got addressed by short temporal movement restriction and the presentation of a dialogue.
The reward function employed in self-play learning is straightforward to define, since the reward payout is binary and symmetrical. Due to the large amount of time it took to train the agent, the complexity of the action space was reduced into a smaller subset. Despite this, the training required still took a considerable amount of time.
The amount of analytics collected grows in accordance with the game's increasing complexity. The final version uses 65 different sorts of events that the players can elicit. As a result of this high complexity, the debugging process and analysis become more time consuming.
There is no official framework for communicating with the Azure Blobs service. Neither Microsoft nor Unity offer a framework that is specifically made for this purpose. Instead, an out-of-date and no longer maintained framework had to be used.

6.3 Analysis

The lack of data makes it difficult to draw firm conclusions regarding the study's objectives. Therefore, it is suggested to further investigate these findings through a larger-scale study.

During the correlation task, there were some quality concerns due to the number of samples available. There have been occasions where the relationship between variables was not clearly visible or rather distorted due to an outlier. For example, a single person who did not collect the treasure and scored the maximal possible value for extraversion was sufficient to yield a correlation.
Some player interactions were less frequent due to the game's design. As a consequence, the efficacy of behavioural measures in these cases severely limited. Adjusting the level generation to boost the generation of these elements and hence encourage more interaction might be a solution to this problem.
Correlations do not necessarily indicate causation. The existence of a relationship between experience and behavioural variables, does not indicate that one caused the other. Thus, another study investigating these relationships would be reasonable.
Due to game design, some decisions may hinge on others. If players take the puzzle path, they will automatically be exposed to a higher amount of box interactions.
Some expected relationships that were formed during the game's creation did not yield any results. Both, Extraversion and NPCs, as well as openness and level exploration were found to be unrelated. This might be due to the fact that the game is insufficient to provoke these personality-related behaviours. Some game components, such as the enemy path or the smart enemy, were better catalysts in revealing the player's behavioural nature. Contrary to that, other components, such as the token paths and normal enemy interactions, had no meaningful relationships.

The curse of dimensionality became evident during the prediction of the players' personality, and resulted in the fact that no predictive model could be obtained. Therefore, a future study should delve deeper into this issue.

By increasing the amount of data or reducing the game's complexity, an improvement should be possible.

Because of the low clustering coefficient, the clustering may not be ideal. An interpretation of the results is very tedious due to the high dimensionality. Through manual comparison between the cluster centers promising results were obtained. As previously stated, these results should be further verified.

7 Future Work

This chapter outlines different future research directions. It explains how the preliminary study of this work may be improved. Furthermore, more improvements to the level generation would beneficial. Other areas of future research include online and offline adaptation, as well as an end-to-end approach for adaptation.

7.1 Extensive Study

The most important direction of future work, lies in conducting a comprehensive study to corroborate the findings of this work. The validity of the results can be improved by increasing the sample size or reduction of the complexity. Furthermore, this study did not evaluate randomized level configurations to obtain results free of certain presumptions. For instance, the sequence of paths generated could have an influence on what type of paths players decide to go along.

7.2 Game Improvements

The quality of game experience should be improved. In addition to the components added to the game, additional elements could be investigated. These parts could be studied individually or as part of a comprehensive larger study. Additionally, the elements added for this study necessitate further improvements.

- The visual diversity of generated structures such as puzzles should be increased.
- NPCs should have other tasks besides the tutorial and portal spawning. They might obtain a more vital role as part of the game's story.
- The story of the game could be enhanced and be made a more prominent part of the game.
- The smart enemy AI could be parameterized.
- Smaller secondary paths might enhance the amount of path-related decisions in the game.
- Overall design and sound improvements would be beneficial.

7.3 Online Modelling and Adaptation

Future research might also investigate inferring player profiles online. An investigation concerning the duration it takes to obtain a reliable estimates for player profiles would be of interest. On that basis, levels could evolve while the players play. As the behavioural models of players gain accuracy over time, the level should improve to better fit players.

7.4 Optimization Objective

Instead of performing personality prediction, and ensuing construction of adaptation methods, a single factor could be used for direct end-to-end optimization of the game. The work of Pedersen, Togelius, and Yannakakis (2010) employs estimations that quantify satisfaction or fun for adaptation. Through optimization, the game experience is then adapted to maximize these quantities.

8 Conclusion

Every human being has a distinct personality and set of interests. Static games are often unable to accommodate this level of individual variability. Personalization can act as a counterbalance to a multitude of these variations, ranging from cultural differences to personality and personal impairments. Hence, players of games would benefit from dynamic elements in games. Dynamic adjustments help players identify with games more easily. Given the fact that the ubiquitous use of technology has increased the quantity of personal data available about individuals, personalization can benefit from it by better fitting the players' preferences.

The goal of this project was to create a framework that can assess various player information. It gathered player demographics, the players' true personality, game behaviour, and game experiences for storage on a data collection server. The game is a Jump'n'Run game with parameterized grammar-based procedural level generation. The generation can be modified with various probabilities to support different play styles. The smart enemy in the game was created via self-play reinforcement learning and poses a difficult obstacle to players. The data collected were then processed using the analysis tool that attempts to establish relationships between player behaviour, personality, and player experience.

The user study conducted did not reach the needed number of participants to be fully conclusive. Nevertheless, the analysis was able to establish relationships between game behaviour and components with personality. These relationships have been successfully connected to the description of personality traits. Due to the high dimensionality of the data and the large number of data samples, a predictive model for personality could not be inferred. Learning from the data on the other hand, player profiles that are separated by behaviour could be established. Components of increased tension and challenge were identified by analysing the players' experience while playing the game. Some expectations that had been made in advance could not be confirmed by the results. As a result, the game may not provide enough ground for players to effectively express the full variety of their personality traits. Based on the findings of this work, there are recommendations on how to integrate game adaptation based on personality, experience, or player profiles into the game. Finally, this research should be extended to corroborate and reinforce the relationships discovered in this study.

Appendix

Bibliography

Akkaya, Ilge et al. (2019). "Solving rubik's cube with a robot hand". In: *arXiv preprint arXiv:1910.07113* (cit. on p. 44).

Allen, Mark S, Iain Greenlees, and Marc Jones (2013). "Personality in sport: A comprehensive review". In: *International Review of Sport and Exercise Psychology* 6.1, pp. 184–208 (cit. on p. 7).

Allport, Gordon Willard (1937). "Personality: A psychological interpretation." In: (cit. on p. 6).

Anderson, Craig A et al. (2010). "Violent video game effects on aggression, empathy, and prosocial behavior in eastern and western countries: a meta-analytic review." In: *Psychological bulletin* 136.2, p. 151 (cit. on p. 18).

Anolli, Luigi et al. (2010). "Emotions in serious games: From experience to assessment". In: *International Journal of Emerging Technologies in Learning (iJET)* 5.2010 (cit. on pp. 16, 17).

Bahdanau, Dzmitry et al. (2018). "Learning to understand goal specifications by modelling reward". In: *arXiv preprint arXiv:1806.01946* (cit. on p. 44).

Bakkes, Sander, Pieter Spronck, and Giel van Lankveld (2012). "Player behavioural modelling for video games". In: *Entertainment Computing* 3.3, pp. 71–79 (cit. on pp. 20, 22).

Bakkes, Sander, Chek Tien Tan, and Yusuf Pisan (2012). "Personalised gaming: a motivation and overview of literature". In: *Proceedings of the 8th Australasian Conference on Interactive Entertainment: Playing the System*, pp. 1–10 (cit. on pp. 1, 28–30).

Bansal, Trapit et al. (2017). "Emergent complexity via multi-agent competition". In: *arXiv preprint arXiv:1710.03748* (cit. on p. 53).

Bar, Amit and Tobias Otterbring (2021). "The role of culture and personality traits in board game habits and attitudes: Cross-cultural comparison between Denmark, Germany, and USA". In: *Journal of Retailing and Consumer Services* 61, p. 102506 (cit. on p. 7).

Bartle, Richard (1996). "Hearts, clubs, diamonds, spades: Players who suit MUDs". In: *Journal of MUD research* 1.1, p. 19 (cit. on p. 21).

Bauckhage, Christian, Anders Drachen, and Rafet Sifa (2014). "Clustering game behavior data". In: *IEEE Transactions on Computational Intelligence and AI in Games* 7.3, pp. 266–278 (cit. on pp. 23, 100).

Beal, Carole et al. (2002). "Intelligent modeling of the user in interactive entertainment". In: *AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment*, pp. 8–12 (cit. on pp. 26, 27).

Bean, Anthony and Gary Groth-Marnat (2016). "Video gamers and personality: A five-factor model to understand game playing style." In: *Psychology of Popular Media Culture* 5.1, p. 27 (cit. on pp. 8, 14).

Beattie, Charles et al. (2016). "Deepmind lab". In: *arXiv preprint arXiv:1612.03801* (cit. on p. 43).

Bellemare, Marc G et al. (2013). "The arcade learning environment: An evaluation platform for general agents". In: *Journal of Artificial Intelligence Research* 47, pp. 253–279 (cit. on p. 43).

Bellman, Richard (1957). "A Markovian decision process". In: *Journal of mathematics and mechanics* 6.5, pp. 679–684 (cit. on p. 42).

Berner, Christopher et al. (2019). "Dota 2 with large scale deep reinforcement learning". In: *arXiv preprint arXiv:1912.06680* (cit. on pp. 44, 48, 50).

Bialas, Mateusz, Shoshannah Tekofsky, and Pieter Spronck (2014). "Cultural influences on play style". In: *2014 IEEE Conference on Computational Intelligence and Games*. IEEE, pp. 1–7 (cit. on p. 1).

Boyle, Gregory J, Gerald Matthews, and Donald H Saklofske (2008). "Personality measurement and testing: An overview". In: *The SAGE handbook of personality theory and assessment* 2, pp. 1–26 (cit. on pp. 8, 12).

Braun, Beate et al. (2016). "Personality and video gaming: Comparing regular gamers, non-gamers, and gaming addicts and differentiating between game genres". In: *Computers in Human Behavior* 55, pp. 406–412 (cit. on pp. 14, 15).

Brockman, Greg et al. (2016). *OpenAI Gym*. eprint: `arXiv:1606.01540` (cit. on p. 43).

Bunian, Sara et al. (2017). "Modeling individual differences in game behavior using HMM". In: *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference* (cit. on p. 24).

Buşoniu, Lucian, Robert Babuška, and Bart De Schutter (2010). "Multi-agent reinforcement learning: An overview". In: *Innovations in multi-agent systems and applications-1*, pp. 183–221 (cit. on pp. 51–53).

Butcher, James N (2010). "Minnesota multiphasic personality inventory". In: *The Corsini Encyclopedia of Psychology*, pp. 1–3 (cit. on pp. 9, 10).

Caillois, Roger (2001). *Man, play, and games*. University of Illinois press (cit. on p. 21).

Cambria, Erik, Andrew Livingstone, and Amir Hussain (2012). "The hourglass of emotions". In: *Cognitive behavioural systems*. Springer, pp. 144–157 (cit. on p. 16).

Caspi, Avshalom, Brent W Roberts, and Rebecca L Shiner (2005). "Personality development: Stability and change". In: *Annu. Rev. Psychol.* 56, pp. 453–484 (cit. on pp. 6, 106).

Cattell, Heather EP and Alan D Mead (2008). "The Sixteen Personality Factor Questionnaire (16PF)." In: (cit. on pp. 9, 10).

Charles, Darryl et al. (2005). "Player-centred game design: Player modelling and adaptive digital games". In: (cit. on pp. 22, 25, 26, 28, 29).

Cohen, J and P Cohen (1975). "Applied multiple regressional/correlation analysis for the behavioral sciences". In: (cit. on p. 100).

Compton, Kate and Michael Mateas (2006). "Procedural Level Design for Platform Games." In: *AIIDE*, pp. 109–111 (cit. on pp. 31, 39, 56).

Costa Jr, Paul T and Robert R McCrae (1995). "Domains and facets: Hierarchical personality assessment using the Revised NEO Personality Inventory". In: *Journal of personality assessment* 64.1, pp. 21–50 (cit. on p. 11).

Costa Jr, Paul T and Robert R McCrae (2008a). *The Revised NEO Personality Inventory (NEO-PI-R).* Sage Publications, Inc (cit. on p. 11).

Costa Jr, Paul T and Robert R McCrae (2008b). *The Revised Neo Personality Inventory (neo-pi-r).* Sage Publications, Inc (cit. on p. 12).

Costa Jr, Paul T, Robert R McCrae, and Corinna E Löckenhoff (2019). "Personality across the life span". In: *Annual review of psychology* 70, pp. 423–448 (cit. on p. 12).

De Kegel, Barbara and Mads Haahr (2019). "Procedural puzzle generation: a survey". In: *IEEE Transactions on Games* 12.1, pp. 21–40 (cit. on p. 37).

DeFalco, Jeanine A et al. (2018). "Detecting and addressing frustration in a serious game for military training". In: *International Journal of Artificial Intelligence in Education* 28.2, pp. 152–193 (cit. on p. 16).

Dewsbury, Neall et al. (2016). "Scalable level generation for 2d platforming games". In: *Proceedings of the FDG workshop on Procedural Content Generation* (cit. on pp. 38, 40, 76).

Digman, J M (1990). "Personality Structure: Emergence of the Five-Factor Model". In: *Annual Review of Psychology* 41.1, pp. 417–440. DOI: 10.1146/annurev.ps.41.020190.002221. eprint: https://doi.org/10.1146/annurev.ps.41.020190.002221. URL: https://doi.org/10.1146/annurev.ps.41.020190.002221 (cit. on p. 11).

Dormans, Joris (2010). "Adventures in level design: generating missions and spaces for action adventure games". In: *Proceedings of the 2010 workshop on procedural content generation in games*, pp. 1–8 (cit. on p. 36).

Dormans, Joris and Sander Bakkes (2011). "Generating missions and spaces for adaptable play experiences". In: *IEEE Transactions on Computational Intelligence and AI in Games* 3.3, pp. 216–228 (cit. on p. 30).

Drachen, Anders, Rafet Sifa, et al. (2012). "Guns, swords and data: Clustering of player behavior in computer games in the wild". In: *2012 IEEE conference on Computational Intelligence and Games (CIG)*. IEEE, pp. 163–170 (cit. on p. 23).

Drachen, Anders, Christian Thurau, et al. (2014). "A comparison of methods for player clustering via behavioral telemetry". In: *arXiv preprint arXiv:1407.3950* (cit. on p. 23).

Dumont, Frank (2010). *A history of personality psychology: Theory, science, and research from Hellenism to the twenty-first century*. Cambridge university press (cit. on p. 7).

Eysenck, HJ and SBG Eysenck (1984). "Eysenck personality questionnaire-revised". In: (cit. on p. 10).

Foerster, Jakob et al. (2017). "Stabilising experience replay for deep multi-agent reinforcement learning". In: *International conference on machine learning*. PMLR, pp. 1146–1155 (cit. on p. 48).

Francis, Leslie, Laurence Brown, and Ronald Philipchalk (1992). "The development of an abbreviated form of the Revised Eysenck Personality Questionnaire (EPQR-A): Its use among students in England, Canada, the USA and Australia". In: *Personality and individual differences* 13.4, pp. 443–449 (cit. on p. 9).

Gentile, Douglas A (2011). "The multiple dimensions of video game effects". In: *Child development perspectives* 5.2, pp. 75–81 (cit. on p. 19).

Gilleade, Kiel and Jen Allanson (2003). "A toolkit for exploring affective interface adaptation in videogames". In: *HCI International 2003*, pp. 370–374 (cit. on p. 19).

Gilleade, Kiel and Alan Dix (2004). "Using frustration in the design of adaptive videogames". In: *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pp. 228–232 (cit. on pp. 17, 18, 20).

Gisslén, Linus et al. (2021). "Adversarial reinforcement learning for procedural content generation". In: *arXiv preprint arXiv:2103.04847* (cit. on p. 44).

Glickman, Mark E and Albyn Jones (1999). "Rating the chess rating system". In: *CHANCE-BERLIN THEN NEW YORK-* 12, pp. 21–28 (cit. on pp. 30, 53, 54).

Glinz, Martin (2007). "On non-functional requirements". In: *15th IEEE international requirements engineering conference (RE 2007)*. IEEE, pp. 21–26 (cit. on p. 58).

Goldberg, Lewis R (1993). "The structure of phenotypic personality traits." In: *American psychologist* 48.1, p. 26 (cit. on pp. 9, 11).

Granic, Isabela, Adam Lobel, and Rutger CME Engels (2014). "The benefits of playing video games." In: *American psychologist* 69.1, p. 66 (cit. on p. 18).

Greitemeyer, Tobias and Dirk O Mügge (2014). "Video games do affect social outcomes: A meta-analytic review of the effects of violent and prosocial video game play". In: *Personality and social psychology bulletin* 40.5, pp. 578–589 (cit. on p. 18).

Greitemeyer, Tobias, Silvia Osswald, and Markus Brauer (2010). "Playing prosocial video games increases empathy and decreases schadenfreude." In: *Emotion* 10.6, p. 796 (cit. on p. 18).

Gu, Shixiang, Ethan Holly, et al. (2017). "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 3389–3396 (cit. on p. 44).

Gu, Shixiang, Timothy Lillicrap, et al. (2016). "Continuous deep q-learning with model-based acceleration". In: *International conference on machine learning*. PMLR, pp. 2829–2838 (cit. on p. 47).

Haarnoja, Tuomas et al. (2018). "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor". In: *International Conference on Machine Learning*. PMLR, pp. 1861–1870 (cit. on p. 48).

Halim, Zahid et al. (2017). "Profiling players using real-world datasets: Clustering the data and correlating the results with the big-five personality traits". In: *IEEE transactions on affective computing* 10.4, pp. 568–584 (cit. on pp. 8, 14, 23).

Harpstead, Erik et al. (2015). "What drives people: Creating engagement profiles of players from game log data". In: *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, pp. 369–379 (cit. on p. 24).

Hartsook, Ken et al. (2011). "Toward supporting stories with procedurally generated game worlds". In: *2011 IEEE Conference on Computational Intelligence and Games (CIG'11)*. IEEE, pp. 297–304 (cit. on p. 41).

Hendrikx, Mark et al. (2013). "Procedural content generation for games: A survey". In: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 9.1, pp. 1–22 (cit. on pp. 31, 36, 37).

Herbrich, Ralf, Tom Minka, and Thore Graepel (2006). "Trueskill™: A Bayesian skill rating system". In: *Proceedings of the 19th international conference on neural information processing systems*, pp. 569–576 (cit. on pp. 31, 54).

Hernandez-Leal, Pablo, Bilal Kartal, and Matthew E Taylor (2019). "A survey and critique of multiagent deep reinforcement learning". In: *Autonomous Agents and Multi-Agent Systems* 33.6, pp. 750–797 (cit. on pp. 51, 52).

Hooshyar, Danial, Moslem Yousefi, and Heuiseok Lim (2018). "Data-driven approaches to game player modeling: a systematic literature review". In: *ACM Computing Surveys (CSUR)* 50.6, pp. 1–19 (cit. on pp. 21–23, 55).

Hupka, Ralph B et al. (1997). "The colors of anger, envy, fear, and jealousy: A cross-cultural study". In: *Journal of cross-cultural psychology* 28.2, pp. 156–171 (cit. on p. 4).

IJsselsteijn, Wijnand A, Yvonne AW de Kort, and Karolien Poels (2013). "The game experience questionnaire". In: *Eindhoven: Technische Universiteit Eindhoven* 46.1 (cit. on pp. 16, 24, 67).

JCH, Watkins Christopher and Dayan Peter (1992). "Q-learning". In: *Machine learning* 8.3-4, pp. 279–292 (cit. on p. 47).

Jennett, Charlene et al. (2008). "Measuring and defining the experience of immersion in games". In: *International journal of human-computer studies* 66.9, pp. 641–661 (cit. on p. 25).

John, Oliver P, Eileen M Donahue, and Robert L Kentle (1991). "Big five inventory". In: *Journal of Personality and Social Psychology* (cit. on p. 13).

Johnson, Daniel and John Gardner (2010). "Personality, motivation and video games". In: *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction*, pp. 276–279 (cit. on pp. 14, 15, 18).

Johnson, Daniel, John Gardner, and Ryan Perry (2018). "Validation of two game experience scales: the player experience of need satisfaction (PENS) and game experience questionnaire (GEQ)". In: *International Journal of Human-Computer Studies* 118, pp. 38–46 (cit. on pp. 24, 25).

Johnson, Daniel, Peta Wyeth, et al. (2012). "Personality, genre and videogame play experience". In: *Proceedings of the 4th International Conference on Fun and Games*, pp. 117–120 (cit. on p. 18).

Jordan, Michael I and Tom M Mitchell (2015). "Machine learning: Trends, perspectives, and prospects". In: *Science* 349.6245, pp. 255–260 (cit. on p. 42).

Juliani, Arthur et al. (2018). "Unity: A general platform for intelligent agents". In: *arXiv preprint arXiv:1809.02627* (cit. on pp. 43, 47, 48, 84).

Kaelbling, Leslie Pack, Michael L Littman, and Andrew Moore (1996). "Reinforcement learning: A survey". In: *Journal of artificial intelligence research* 4, pp. 237–285 (cit. on pp. 43, 45).

Kartal, Bilal, Nick Sohre, and Stephen J Guy (2016). "Data driven Sokoban puzzle generation with Monte Carlo tree search". In: *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference* (cit. on p. 37).

Keller, Carmen and Michael Siegrist (2015). "Does personality influence eating styles and food choices? Direct and indirect effects". In: *Appetite* 84, pp. 128–138 (cit. on p. 7).

Kempka, Michał et al. (2016). "Vizdoom: A doom-based ai research platform for visual reinforcement learning". In: *2016 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, pp. 1–8 (cit. on p. 43).

Kendall, Alex et al. (2019). "Learning to drive in a day". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 8248–8254 (cit. on p. 44).

Khalifa, Ahmed et al. (2020). "Pcgrl: Procedural content generation via reinforcement learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*. Vol. 16. 1, pp. 95–101 (cit. on pp. 38, 44).

Klopf, A Harry (1972). *Brain function and adaptive systems: a heterostatic theory*. 133. Air Force Cambridge Research Laboratories, Air Force Systems Command, United ... (cit. on p. 42).

Koelstra, Sander et al. (2011). “Deap: A database for emotion analysis; using physiological signals”. In: *IEEE transactions on affective computing* 3.1, pp. 18–31 (cit. on p. 17).

Koster, Raph (2013). *Theory of fun for game design*. " O’Reilly Media, Inc." (cit. on p. 21).

Lazzaro, Nicole (2009). “Why we play: affect and the fun of games”. In: *Human-computer interaction: Designing for diverse users and domains* 155, pp. 679–700 (cit. on p. 21).

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). “Deep learning”. In: *nature* 521.7553, pp. 436–444 (cit. on p. 43).

Lester, James C et al. (2013). “Serious games get smart: Intelligent game-based learning environments”. In: *AI Magazine* 34.4, pp. 31–45 (cit. on pp. 17, 19).

Li, Yuxi (2018). “Deep reinforcement learning”. In: *arXiv preprint arXiv:1810.06339* (cit. on pp. 46, 47).

Liapis, Antonios, Georgios Yannakakis, and Julian Togelius (2013). “Sentient sketchbook: computer-assisted game level authoring”. In: (cit. on p. 35).

Liu, Jialin et al. (2021). “Deep learning for procedural content generation”. In: *Neural Computing and Applications* 33.1, pp. 19–37 (cit. on p. 35).

Liu, Xiao, Yuanwei Liu, and Yue Chen (2019). “Reinforcement learning in multiple-UAV networks: Deployment and movement design”. In: *IEEE Transactions on Vehicular Technology* 68.8, pp. 8036–8049 (cit. on p. 50).

Livingstone, Steven and Andrew Brown (2005). “Dynamic response: real-time adaptation for music emotion”. In: *ACM International Conference Proceeding Series*. Vol. 123, pp. 105–111 (cit. on p. 30).

Lopes, Ricardo and Rafael Bidarra (2011). “Adaptivity challenges in games and simulations: a survey”. In: *IEEE Transactions on Computational Intelligence and AI in Games* 3.2, pp. 85–99 (cit. on pp. 1, 21, 25, 26, 28–30, 54, 55).

Magerko, Brian, Brian S Stensrud, and Lisa S Holt (2006). *Bringing the schoolhouse inside the box-a tool for engaging, individualized training*. Tech. rep. SOAR TECHNOLOGY INC ANN ARBOR MI (cit. on pp. 22, 27).

Majumder, Navonil et al. (2017). "Deep learning-based document modeling for personality detection from text". In: *IEEE Intelligent Systems* 32.2, pp. 74–79 (cit. on p. 8).

Malone, Thomas W (1981). "Toward a theory of intrinsically motivating instruction". In: *Cognitive science* 5.4, pp. 333–369 (cit. on pp. 21, 26, 27).

Marcin, Andrychowicz OpenAI: et al. (2020). "Learning dexterous in-hand manipulation". In: *The International Journal of Robotics Research* 39.1, pp. 3–20 (cit. on p. 44).

Matthew, Johnson et al. (2016). "The Malmo Platform for Artificial Intelligence Experimentation." In: *IJCAI*. Citeseer, pp. 4246–4247 (cit. on p. 43).

Matthews, Gerald, Ian J Deary, and Martha C Whiteman (2003). *Personality traits*. Cambridge University Press (cit. on pp. 6, 11).

McCormack, Luke and David Mellor (2002). "The role of personality in leadership: An application of the five-factor model in the Australian military". In: *Military psychology* 14.3, pp. 179–197 (cit. on p. 7).

McCrae, Robert R and Paul T Costa Jr (2007). "Brief versions of the NEO-PI-3". In: *Journal of individual differences* 28.3, pp. 116–128 (cit. on p. 12).

McCrae, Robert R, Paul T Costa Jr, and Thomas A Martin (2005). "The NEO–PI–3: A more readable revised NEO personality inventory". In: *Journal of personality assessment* 84.3, pp. 261–270 (cit. on pp. 9, 12).

McCrae, Robert R and Paul T Costa Jr (1997). "Personality trait structure as a human universal." In: *American psychologist* 52.5, p. 509 (cit. on p. 12).

McCrae, Robert R and Paul T Costa Jr (2008). "The five-factor theory of personality." In: (cit. on p. 10).

Minka, Tom, Ryan Cleven, and Yordan Zaykov (2018). "TrueSkill 2: An improved Bayesian skill rating system". In: *Tech. Rep.* (cit. on pp. 31, 54).

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, et al. (2013). "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602* (cit. on pp. 44, 49).

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A Rusu, et al. (2015). "Human-level control through deep reinforcement learning". In: *nature* 518.7540, pp. 529–533 (cit. on p. 49).

Mohammadi, Gelareh and Alessandro Vinciarelli (2012). "Automatic personality perception: Prediction of trait attribution based on prosodic features". In: *IEEE Transactions on Affective Computing* 3.3, pp. 273–284 (cit. on p. 8).

Moore, Kelly and James C McElroy (2012). "The influence of personality on Facebook usage, wall postings, and regret". In: *Computers in human behavior* 28.1, pp. 267–274 (cit. on p. 7).

Myers, Isabel Briggs (1962). "The Myers-Briggs Type Indicator: Manual (1962)." In: (cit. on p. 10).

Nacke, Lennart E, Chris Bateman, and Regan L Mandryk (2014). "BrainHex: A neurobiological gamer typology survey". In: *Entertainment computing* 5.1, pp. 55–62 (cit. on p. 21).

Nakamura, Jeanne and Mihaly Csikszentmihalyi (2014). "The concept of flow". In: *Flow and the foundations of positive psychology*. Springer, pp. 239–263 (cit. on p. 21).

Neumann, Christof et al. (2011). "Assessing dominance hierarchies: validation and advantages of progressive evaluation with Elo-rating". In: *Animal Behaviour* 82.4, pp. 911–921 (cit. on p. 54).

Nicholson, Nigel et al. (2005). "Personality and domain-specific risk taking". In: *Journal of Risk Research* 8.2, pp. 157–176 (cit. on p. 106).

Pedersen, Christopher, Julian Togelius, and Georgios Yannakakis (2010). "Modeling player experience for content creation". In: *IEEE Transactions on Computational Intelligence and AI in Games* 2.1, pp. 54–67 (cit. on pp. 18, 19, 23, 26, 28, 118).

Peever, Nicole, Daniel Johnson, and John Gardner (2012). "Personality & video game genre preferences". In: *Proceedings of the 8th australasian conference on interactive entertainment: Playing the system*, pp. 1–3 (cit. on pp. 7, 14, 15).

Pekrun, Reinhard, Andrew J Elliot, and Markus A Maier (2009). "Achievement goals and achievement emotions: Testing a model of their joint relations with academic performance." In: *Journal of educational Psychology* 101.1, p. 115 (cit. on p. 19).

Pittenger, David J (2005). "Cautionary comments regarding the Myers-Briggs type indicator." In: *Consulting Psychology Journal: Practice and Research* 57.3, p. 210 (cit. on p. 10).

Plans, David and Davide Morelli (2012). "Experience-driven procedural music generation for games". In: *IEEE Transactions on Computational Intelligence and AI in Games* 4.3, pp. 192–198 (cit. on pp. 30, 37, 41).

Poria, Soujanya et al. (2017). "A review of affective computing: From unimodal analysis to multimodal fusion". In: *Information Fusion* 37, pp. 98–125 (cit. on pp. 16, 17).

Poropat, Arthur E (2009). "A meta-analysis of the five-factor model of personality and academic performance." In: *Psychological bulletin* 135.2, p. 322 (cit. on p. 7).

Rammstedt, Beatrice and Oliver P John (2007). "Measuring personality in one minute or less: A 10-item short version of the Big Five Inventory in English and German". In: *Journal of research in Personality* 41.1, pp. 203–212 (cit. on pp. 13, 66).

Richter, Felix (2021). *Gaming: The Most Lucrative Entertainment Industry By Far*. URL: `https://www.statista.com/chart/22392/global-revenue-of-selected-entertainment-industry-sectors/` (visited on 10/06/2021) (cit. on p. 4).

Riedl, Mark Owen and Vadim Bulitko (2013). "Interactive narrative: An intelligent systems approach". In: *Ai Magazine* 34.1, pp. 67–67 (cit. on p. 29).

Risi, Sebastian and Julian Togelius (2020). "Increasing generality in machine learning through procedural content generation". In: *Nature Machine Intelligence* 2.8, pp. 428–436 (cit. on p. 44).

Rummery, Gavin A and Mahesan Niranjan (1994). *On-line Q-learning using connectionist systems*. Vol. 37. Citeseer (cit. on p. 47).

Russoniello, Carmen V, Kevin O'Brien, and Jennifer M Parks (2009). "The effectiveness of casual video games in improving mood and decreasing stress". In: *Journal of CyberTherapy & Rehabilitation* 2.1, pp. 53–66 (cit. on p. 54).

Ryan, Richard M, C Scott Rigby, and Andrew Przybylski (2006). "The motivational pull of video games: A self-determination theory approach". In: *Motivation and emotion* 30.4, pp. 344–360 (cit. on p. 24).

Samuel, Arthur L (1959). "Some studies in machine learning using the game of checkers". In: *IBM Journal of research and development* 3.3, pp. 210–229 (cit. on pp. 42, 53).

Schmitt, David P et al. (2008). "Why can't a man be more like a woman? Sex differences in Big Five personality traits across 55 cultures." In: *Journal of personality and social psychology* 94.1, p. 168 (cit. on p. 12).

Schraudolph, Nicol N, Peter Dayan, and Terrence J Sejnowski (1994). "Temporal difference learning of position evaluation in the game of Go". In: *Advances in Neural Information Processing Systems*, pp. 817–817 (cit. on p. 53).

Schulman, John, Sergey Levine, et al. (2015). "Trust region policy optimization". In: *International conference on machine learning*. PMLR, pp. 1889–1897 (cit. on p. 48).

Schulman, John, Filip Wolski, et al. (2017). "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (cit. on pp. viii, 47, 48).

Schultz, Duane and Sydney Schultz (2016). *Theories of personality*. Cengage Learning (cit. on pp. 5–8, 12, 106).

Shaker, Noor et al. (2012). "Evolving levels for super mario bros using grammatical evolution". In: *2012 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, pp. 304–311 (cit. on pp. 32, 38, 41).

Shoham, Yoav and Kevin Leyton-Brown (2008). *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press (cit. on p. 49).

Shu, Tianye, Jialin Liu, and Georgios Yannakakis (2021). "Experience-driven PCG via reinforcement learning: A Super Mario Bros study". In: *arXiv preprint arXiv:2106.15877* (cit. on p. 41).

Silver, David, Aja Huang, et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587, pp. 484–489 (cit. on pp. 44, 50, 54).

Silver, David, Thomas Hubert, et al. (2018). "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". In: *Science* 362.6419, pp. 1140–1144 (cit. on p. 50).

Silver, David, Julian Schrittwieser, et al. (2017). "Mastering the game of go without human knowledge". In: *nature* 550.7676, pp. 354–359 (cit. on pp. 44, 50, 53, 54).

Smith, Gillian (2014). "Understanding procedural content generation: a design-centric analysis of the role of PCG in games". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 917–926 (cit. on pp. 32, 34, 35, 56).

Smith, Gillian, Mee Cha, and Jim Whitehead (2008). "A Framework for Analysis of 2D Platformer Levels". In: *Proceedings of the 2008 ACM SIGGRAPH Symposium on Video Games*. Sandbox '08. Los Angeles, California: Association for Computing Machinery, pp. 75–80. ISBN: 9781605581736. DOI: 10.1145/1401843.1401858. URL: https://doi.org/10.1145/1401843.1401858 (cit. on pp. 38, 39, 41).

Smith, Gillian, Jim Whitehead, et al. (2011). "Launchpad: A Rhythm-Based Level Generator for 2-D Platformers". In: *IEEE Transactions on Computa-*

tional Intelligence and AI in Games 3.1, pp. 1–16. DOI: 10.1109/TCIAIG.2010.2095855 (cit. on pp. 38, 40, 76).

Snodgrass, Sam and Santiago Ontañón (2014). "Experiments in map generation using Markov chains." In: *FDG* (cit. on p. 38).

Soto, Christopher J and Oliver P John (2017a). "Short and extra-short forms of the Big Five Inventory–2: The BFI-2-S and BFI-2-XS". In: *Journal of Research in Personality* 68, pp. 69–81 (cit. on p. 13).

Soto, Christopher J and Oliver P John (2017b). "The next Big Five Inventory (BFI-2): Developing and assessing a hierarchical model with 15 facets to enhance bandwidth, fidelity, and predictive power." In: *Journal of personality and social psychology* 113.1, p. 117 (cit. on pp. 9, 13).

Statista (2021). *Coronavirus: impact on the gaming industry worldwide*. URL: https://www.statista.com/study/72150/coronavirus-impact-on-the-video-game-industry-worldwide/ (visited on 10/06/2021) (cit. on p. 4).

Subramanian, Ramanathan et al. (2016). "ASCERTAIN: Emotion and personality recognition using commercial sensors". In: *IEEE Transactions on Affective Computing* 9.2, pp. 147–160 (cit. on p. 8).

Summerville, Adam and Michael Mateas (2016). "Super mario as a string: Platformer level generation via lstms". In: *arXiv preprint arXiv:1603.00930* (cit. on pp. 38, 56).

Summerville, Adam, Sam Snodgrass, et al. (2018). "Procedural content generation via machine learning (PCGML)". In: *IEEE Transactions on Games* 10.3, pp. 257–270 (cit. on p. 35).

Sundström, Petra (2005). "Exploring the affective loop". PhD thesis (cit. on pp. 19, 109).

Sutton, Richard S and Andrew G Barto (2018). *Reinforcement learning: An introduction*. MIT press (cit. on pp. 42, 43, 45–47).

Swider, Brian W and Ryan D Zimmerman (2010). "Born to burnout: A meta-analytic path model of personality, job burnout, and work outcomes". In: *Journal of Vocational behavior* 76.3, pp. 487–506 (cit. on p. 7).

Sykes, Jonathan and Simon Brown (2003). "Affective gaming: measuring emotion through the gamepad". In: *CHI'03 extended abstracts on Human factors in computing systems*, pp. 732–733 (cit. on p. 17).

Tekofsky, Shoshannah et al. (2013). "Psyops: Personality assessment through gaming behavior". In: *In Proceedings of the International Conference on the Foundations of Digital Games*. Citeseer (cit. on pp. 8, 13, 14).

Tesauro, Gerald (1995). "Temporal difference learning and TD-Gammon". In: *Communications of the ACM* 38.3, pp. 58–68 (cit. on pp. 42, 47, 53).

Tett, Robert P and Dawn D Burnett (2003). "A personality trait-based interactionist model of job performance." In: *Journal of Applied psychology* 88.3, p. 500 (cit. on p. 7).

Thorndike, Edward L et al. (1912). *Animal intelligence. Experimental studies* (cit. on p. 42).

Thue, David et al. (2007). "Interactive Storytelling: A Player Modelling Approach." In: *AIIDE*, pp. 43–48 (cit. on p. 29).

Togelius, Julian, Renzo De Nardi, and Simon M Lucas (2007). "Towards automatic personalised content creation for racing games". In: *2007 IEEE Symposium on Computational Intelligence and Games*. IEEE, pp. 252–259 (cit. on pp. 27, 28, 33).

Togelius, Julian, Georgios Yannakakis, et al. (2011). "Search-based procedural content generation: A taxonomy and survey". In: *IEEE Transactions on Computational Intelligence and AI in Games* 3.3, pp. 172–186 (cit. on pp. 31–34, 36, 37, 56).

Tondello, Gustavo F et al. (2019). ""I don't fit into a single type": A Trait Model and Scale of Game Playing Preferences". In: *IFIP Conference on Human-Computer Interaction*. Springer, pp. 375–395 (cit. on p. 21).

Torrente Vigil, Javier et al. (2014). "Towards a low cost adaptation of educational games for people with disabilities". In: *Computer Science and Information Systems* (cit. on p. 1).

Vahlo, Jukka and Juho Hamari (2019). "Five-factor inventory of intrinsic motivations to gameplay (IMG)". In: *Proceedings of the 52nd Hawaii International Conference on System Sciences, Hawaii, USA, 2019.* HICSS (cit. on p. 22).

Valls-Vargas, Josep, Santiago Ontanón, and Jichen Zhu (2015). "Exploring player trace segmentation for dynamic play style prediction". In: *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference* (cit. on p. 23).

Van Lankveld, Giel et al. (2011). "Games as personality profiling tools". In: *2011 IEEE Conference on Computational Intelligence and Games (CIG'11)*. IEEE, pp. 197–202 (cit. on pp. 1, 8, 13, 55).

Vinciarelli, Alessandro and Gelareh Mohammadi (2014). "A survey of personality computing". In: *IEEE Transactions on Affective Computing* 5.3, pp. 273–291 (cit. on pp. 4, 6, 8, 11, 55).

Vinyals, Oriol et al. (2019). "Grandmaster level in StarCraft II using multi-agent reinforcement learning". In: *Nature* 575.7782, pp. 350–354 (cit. on pp. 44, 50).

Wang, Ning and Stacy Marsella (2006). "Introducing EVG: An emotion evoking game". In: *International Workshop on Intelligent Virtual Agents*. Springer, pp. 282–291 (cit. on p. 19).

Watkins, Christopher John Cornish Hellaby (1989). "Learning from delayed rewards". In: (cit. on p. 42).

Wijman, Tom (2021). *Global Games Market to Generate $175.8 Billion in 2021; Despite a Slight Decline, the Market Is on Track to Surpass $200 Billion in 2023*. URL: https://newzoo.com/insights/articles/global-games-market-to-generate-175-8-billion-in-2021-despite-a-slight-decline-the-market-is-on-track-to-surpass-200-billion-in-2023/ (visited on 10/06/2021) (cit. on p. 4).

Williams, Ronald J (1992). "Simple statistical gradient-following algorithms for connectionist reinforcement learning". In: *Machine learning* 8.3, pp. 229–256 (cit. on p. 47).

Witten, Ian H (1977). "An adaptive optimal controller for discrete-time Markov environments". In: *Information and control* 34.4, pp. 286–295 (cit. on p. 42).

Worth, Narnia C and Angela S Book (2015). "Dimensions of video game behavior and their relationships with personality". In: *Computers in Human Behavior* 50, pp. 132–140 (cit. on pp. 7, 10, 14, 23).

Wyer Jr, Robert S, Gerald L Clore, and Linda M Isbell (1999). "Affect and information processing". In: *Advances in experimental social psychology*. Vol. 31. Elsevier, pp. 1–77 (cit. on p. 15).

Yang, Yaodong and Jun Wang (2020). "An overview of multi-agent reinforcement learning from game theoretical perspective". In: *arXiv preprint arXiv:2011.00583* (cit. on p. 52).

Yannakakis, Georgios and John Hallam (2009). "Real-time game adaptation for optimizing player satisfaction". In: *IEEE Transactions on Computational Intelligence and AI in Games* 1.2, pp. 121–133 (cit. on p. 27).

Yannakakis, Georgios and Ana Paiva (2014). "Emotion in games". In: *Handbook on affective computing* 2014, pp. 459–471 (cit. on pp. 16, 17, 19, 55).

Yannakakis, Georgios and Julian Togelius (2011). "Experience-driven procedural content generation". In: *IEEE Transactions on Affective Computing* 2.3, pp. 147–161 (cit. on pp. 24, 26, 32, 33, 55, 56).

Yannakakis, Georgios and Julian Togelius (2018). *Artificial intelligence and games*. Springer (cit. on pp. 1, 20–22, 32, 33, 35, 36, 55).

Yee, Nick (2006). "The demographics, motivations, and derived experiences of users of massively multi-user online graphical environments". In: *Presence: Teleoperators and virtual environments* 15.3, pp. 309–329 (cit. on p. 21).

Yun, Sangdoo et al. (2017). "Action-decision networks for visual tracking with deep reinforcement learning". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2711–2720 (cit. on p. 44).

Zeng, Zhihong et al. (2008). "A survey of affect recognition methods: Audio, visual, and spontaneous expressions". In: *IEEE transactions on pattern analysis and machine intelligence* 31.1, pp. 39–58 (cit. on p. 17).

Zhang, Kaiqing, Zhuoran Yang, and Tamer Başar (2021). "Multi-agent reinforcement learning: A selective overview of theories and algorithms". In: *Handbook of Reinforcement Learning and Control*, pp. 321–384 (cit. on pp. 45, 47, 49–53).

Zhong, Victor, Caiming Xiong, and Richard Socher (2017). "Seq2sql: Generating structured queries from natural language using reinforcement learning". In: *arXiv preprint arXiv:1709.00103* (cit. on p. 44).

Zhu, Yuke et al. (2017). "Target-driven visual navigation in indoor scenes using deep reinforcement learning". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 3357–3364 (cit. on p. 44).

Ludography

A.I. Design (1980). *Rogue* (cit. on p. 28).
Big Blue Box (Sept. 2009). *Fable* (cit. on p. 28).
BioWare, Obsidian Entertainment (June 2002). *Neverwinter Nights* (cit. on p. 14).
Blizzard Entertainment (Mar. 1998). *StarCraft* (cit. on pp. 14, 23).
Blizzard Entertainment (Feb. 2005). *World of Warcraft* (cit. on pp. 10, 14, 23).
Blizzard Entertainment (July 2010). *StarCraft II: Wings of Liberty* (cit. on pp. 44, 50, 114).
Blizzard North (Jan. 1997). *Diablo* (cit. on p. 28).
DICE (Mar. 2010). *Battlefield: Bad Company 2* (cit. on p. 23).
Ensemble Studios (Sept. 1999). *Age of Empires II* (cit. on pp. 14, 23).
Gearbox Software (Oct. 2009). *Borderlands* (cit. on p. 37).
id Software (Dec. 1993). *Doom* (cit. on p. 43).
id Software (Dec. 1999). *Quake 3 Arena* (cit. on p. 43).
Konami (1997). *Tokimeki Memorial Oshiete Your Heart* (cit. on p. 20).
Konami (Oct. 2007). *Pro Evolution Soccer 2008* (cit. on p. 29).
Krafton (Jan. 2011). *TERA* (cit. on p. 23).
Maxis (Sept. 2008). *Spore* (cit. on p. 37).
Mojang Studios (Nov. 2011). *Minecraft* (cit. on p. 43).
Naughty Dog (Oct. 1998). *Crash Bandicoot: Warped* (cit. on p. 29).
Nintendo EAD (Dec. 1996). *Mario Kart 64* (cit. on p. 29).
Nintendo EAD (Apr. 2008). *Mario Kart Wii* (cit. on p. 29).
Nintendo R&D4 (Sept. 1985). *Super Mario Bros.* (Cit. on pp. 18, 19, 28, 41).
Quantic Dream (Feb. 2010). *Heavy Rain* (cit. on p. 30).
Remedy Entertainment, Rockstar Studios (June 2001). *Max Payne* (cit. on p. 29).
Turtle Rock Studios, Valve (Nov. 2008). *Left 4 Dead* (cit. on p. 29).
Valve (Nov. 2009). *Left 4 Dead 2* (cit. on p. 28).
Valve (July 2013). *Dota 2* (cit. on pp. 44, 48, 50, 114).