FACULTY OF SCIENCES

Modeling Player Experience in Web-Based Video Games

Yasmine Bogaert
Student number: 01403612

Supervisor: Prof. dr. ir. Francis wyffels
Counsellor: Andreas Verleysen

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in de informatica

Academic year: 2020 - 2021

GHENT UNIVERSITY

Samenvatting

NEDERLANDSE VERTALING

Summary

Game designers nearly always strive to create games in which the difficulty of a game is appropriate for the player's skill level. Even so, designers are usually not able to accommodate every player's skill level, and frustrating mismatches between a player's skill and a game's difficulty are common. Players begin games at different skill levels, and through playing develop their own skill at their own rates. For some players, even the current best-designed games become uninterestingly easy, while frustrating and difficult for others. As a player's skill improves through practice, a well-designed game will present more difficulties so that the player is never bored by overly easy gameplay or frustrated by overly difficult gameplay. Most of the time, this difficulty and the parameters that come along with it, can only be determined at the beginning of the game by selecting a difficulty level, often done by the player himself. Still, this can lead to a frustrating experience for both experienced and inexperienced gamers, as they attempt to follow preselected learning or difficulty curvy composed by game developers.

A proposed answer to this challenge is Dynamic Difficulty Adjustment (DDA), the process of changing parameters in a video game in real-time, based on the player's observed ability, to balance the game difficulty. This is done in order to avoid making the player bored (if the game is too easy) or frustrated (if it is too hard) and better fit the according skill level of the player. DDA is done during execution, tracking the player's performance and adjusting the game to present proper challenges to the player to ensure a good match between a player's skill and the game's difficulty.

To ensure a proper realization of DDA, a good player model is required. A player model will enable us to predict player behavior in new situations that may have never been observed before. This model is then combined with an algorithm to adapt the game content based on those observations. An adaptive game that is constructed on accurate player models, is able to make predictions about future behavior, adapt the game to those changes, and better direct players toward content they are expected to enjoy.

In an attempt to analyze difficulty parameters, model player behavior, and strive towards a system for DDA, we look at the video game Chrome Dino, an endless runner game in which the player controls a running dinosaur that can jump or duck to avoid obstacles. We set up our own version of this game, with various sampled difficulty parameters, and measure the player's response with a ranked questionnaire in order to determine whether there were statistically significant differences in the perception of game play.

Keywords

Dynamic Difficulty Adjustment, Game Difficulty, Flow, Player Experience, Player Modeling, Chrome Dino.

Modeling Player Experience in Web-Based Video Games

Yasmine Bogaert[1], Prof. dr. ir. Francis wyffels[2], Andreas Verleysen[2]

Abstract

The video game industry has been growing quickly over the past years and so has the technologies that come with it. Still, one aspect has remained stagnant, the difficulty balancing. For some players, even the current best-designed games often become uninterestingly easy, while frustrating and difficult for others. A proposed answer to this challenge is Dynamic Difficulty Adjustment (DDA), in which the difficulty of the game is dynamically adapted to match the player's skill level. To ensure good DDA, an good player model is required that represents the relation between the player's enjoyment and the underlying gameplay parameters. An adaptive game that is constructed on accurate player models, is able to make predictions about future behavior, adapt the game to those changes, and better direct players toward content they are expected to enjoy. This article looks at player modeling for the web-based Chrome Dino game, in the context of DDA.

1. Introduction

References

[1] G. N. Yannakakis, J. Togelius, Artificial Intelligence and Games, Springer, 2018, http://gameaibook.org.

[2] Newzoo | Games, Esports & Mobile Market Intelligence, 2020 Global Esports Market Reporthttps://newzoo.com/ (2020).

[1] Faculty of Sciences, Ghent University

[2] Faculty of Engineering and Architecture, Ghent University

May 5, 2021

Modelleren van Spelervaring in Webgebaseerde Videospelen

Yasmine Bogaert[1], Prof. dr. ir. Francis wyffels[2], Andreas Verleysen[2]

Abstract

NEDERLANDSE VERTALING

1. Introductie

References

[1] G. N. Yannakakis, J. Togelius, Artificial Intelligence and Games, Springer, 2018, http://gameaibook.org.

[2] Newzoo | Games, Esports & Mobile Market Intelligence, 2020 Global Esports Market Reporthttps://newzoo.com/ (2020).

[1] Faculteit Wetenschappen, Universiteit Gent

[2] Faculteit Ingenieurswetenschappen en Architectuur, Universiteit Gent

May 5, 2021

Vulgarizing Summary

officieel nodig:
"Geeft de samenvatting van de masterproef in een taal die begrijpbaar is voor een breed wetenschappelijk publiek."
is de samenvatting niet ook al vulgarized genoeg?

Acknowledgement

I would like to express my gratitude to my promotor, Francis wyffels, and counsellor, Andreas Verleysen. Not only for the opportunity I was given to research such an interesting topic in the domain of dynamic difficulty adjustment and player modeling in Video games, I was also given continued motivational support and guidance throughout this project. Without the numerous meetings, supervision of the directions I was taking, proofreading of different iterations of text and multitude of suggestions, I would not have been able to achieve a thesis of the shape that it is in now.

I would also like to thank my friends and family, for the emotional support throughout this long process, as well as fellow students who helped me out when I was in need of specialized advice on certain topics.

Yasmine Bogaert
June 2021

Permission for Usage

The author gives permission to make this master's thesis available for consultation and to copy parts of this master's thesis for personal use. In the case of any other use, the copyright terms have to be respected, in particular with regard to the obligation to state expressly the source when quoting results from this master's thesis.

Yasmine Bogaert
June 2021

Abbreviations

DDA	Dynamic Difficulty Adjustment
UMAP	Uniform Manifold Approximation and Projection
t-SNE	t-distributed stochastic neighbor embedding
4-AFC	4-alternative forced choice
AI	Artificial Intelligence

Contents

1 Introduction

The video game industry has been growing quickly over the past years, and is expected to grow even more in the future. Games begin to take place in more and more parts of our lives. Where they used to only be present in the entertainment industry, games have widely expanded to other markets. One important trend is the concept of gamification, in which concepts from games (such as rewards, rankings, point systems) are used in non-game environments to enhance the user's experience. The use of serious games has also grown, particularly in such sectors as education, defense, aeronautics, science or health. There is a growing demand for learning through video games and little by little, the market is responding to that demand. [6, 9]

One might ask what keeps players entertained and come back to games. Even with concepts such as gamification, creating a game that is able to create enjoyment for many people and is able to retain that, is not an easy task. Players often get bored when a game gets too easy, or frustrated when a game gets too hard. The key factor is balancing the difficulty of a game so that it matches the skill of the player. Gamification and other similar concepts are based on general aspects of psychology that only cover global trends. The gaming audience is wide spread, and so is the individual skill level of players, ranging from new players to players with a lot of experience. Adapting the difficulty to match the skill level of the player, requires a different approach. One that focuses on individual aspects, to improve personalized player experiences.

Nowadays, adapting difficulty often only comes in the form of a static predefined set of difficulties, from which the user often has to make a choice themselves. In the most basic approach, players can choose between easy, medium, and hard mode of game. It does not come as a surprise that these often fall short for most players as these preset difficulties do not match the widespread individual skill and experience of different players. In addition, a player learns during the game, and his/her skills and expectations of the difficulty may change over time. Every player has an individual learning curve and following a predefined learning curve can either be too easy for a player, when the player learns at a faster rate, or too difficult, when the player needs more time to process and get on the same track. This is where Dynamic Difficulty Adjustment comes into play. With DDA, the game will dynamically adapt to fit the learning curve to the player's skill level. It does this by building an underlying model of the player to gain insight into the player's enjoyment, and then makes an informed decision on how to adapt the difficulty to improve upon.

The most important step in constructing a DDA system, is creating the player models models. Player models are based on objective game metrics, combined with labels about the individual subjective player's experience. A player model will enable us to predictplayer behavior in new situations that may have never been observed before. This model is then combined with analgorithm to adapt the game content based on those observations. An adaptive game that is constructed on accurate player models, is able to make predictions about future behavior, adapt the game to those changes, and better direct players toward content they are expected to enjoy.

Artificial Intelligence (AI) is an popular topic at the moment. More researchers than ever work on AI in some form and games are often a popular application area for AI research. Video games have during the last decade increasingly become the domain of choice for testing and showcasing new algorithms. Games not only pose interesting and complex problems for AI to solve, games are also a rare domain where science (problem solving) meets art and interaction: these ingredients have made games a unique and favorite domain for the study of AI.

The purpose of this thesis is to investigate player modeling within the context of DDA for Chrome Dino, a web-based video game. Human player data will be collected to get insight in how player enjoyment relates to the gathered player metrics and a model that represents these will be set up. Extensive research is done on different topics. From what defines player enjoyment from a psychological view, to how to capture it in a statistical reliable way and how to process all the gathered data to construct suitable player models.

The contents of this thesis can be summarized as follows: In chapter two, a literature study discussing dynamic difficulty adjustment, player enjoyment, player modeling, how to measure player experience and ++(what kind of analysis?). This is followed by the methodology chapter, where the different steps of the approach are laid out. Chapter four then deals with the collection and analysis of player data. After this, the architecture of the player modeling system is explained, followed by a chapter discussing the results of several experiments. A conclusion is given in chapter six, which is followed finally by a chapter for future work.

2 Literature

2.1 Dynamic Difficulty Adjustment

Dynamic Difficulty Adjustment (DDA) is a technique used in games to adapt the games difficulty parameters to the players skill level in order to ensure that the player does not get bored (when the game is too easy) or frustrated (when the game is too hard). Te essence of the DDA is to retain the interest of the user throughout the game and to offer a satisfactory challenge level for the player.

++ more

2.2 Player Enjoyment

There have been several psychological studies that try to identify what is 'fun' in a game and what engages people to play video games. These studies include Malone's principles of intrinsic qualitative factors for engaging game play [10], namely challenge, curiosity and fantasy as well as the concept of flow theory by Csikszentmihalyi [4].

The concept of flow defines player states in two dimensions: skill and challenge. It relates the difficulty of a task to the viewers perception of its challenge. The flow channel is a state where a balance between skill and challenge is achieved, often described colloquially as being in "the Zone". When doing an activity that is neither boring nor frustrating, the person becomes immersed, is able to perform longer and keep focused on the task. This is a phenomena that is present in all fields, not only in the scope of video games.

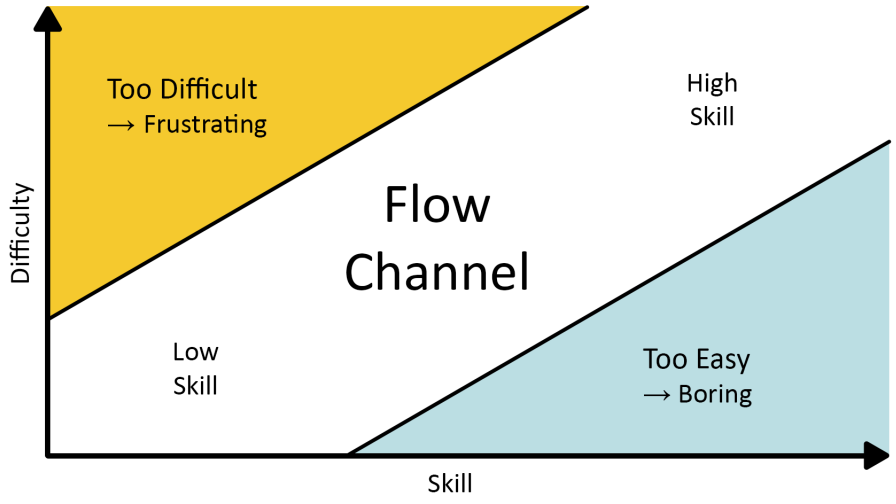


Figure 2.1: Flow Channel.

In figure 2.1, we can see that when the difficulty of the game is higher than the player's skill, the game becomes frustrating, pushing the player into a state of anxiety. On the other hand, when the player's skill is higher than the difficulty, the game is too easy, pushing the player into a state of boredom. Our goal is to keep the player in the flow channel and away from states where the game is far too challenging, or way too easy. One should only adapt the difficulty when a player can be kept in a state of flow. This is the precise task of the DDA system. DDA will enhance difficulty gradually, when it evaluates that the player has had sufficient time to learn and improve so that their skill can conquer new challenges. [8]

To reach the initial state of flow, proper conditions have to be met. First of all, the player must first perform a challenging activity that compels them to improve their skills. Second, the activity should have a clear and easy to achieve goal (levels, missions, high scores, etc.) with immediate feedback on progression. Lastly, the final result of the activity the player is performing should be uncertain, but at the same time, the player has to have a direct impact on its outcome. [5, 2]

In a different study, Malone [10] suggested that if the the three factors of fantasy, challenge and curiosity have need to be in balance at all times during gameplay. With challenge, we mean the uncertainty of achieving a goal (due to e.g., variable difficulty level, multiple level goals, hidden information or randomness). Curiosity refers to the feeling of uncertainty of what will happen next. Lastly, fantasy is the ability of the game to evoke situations that are actually not present. These three dimensions can be quantified and maximized together to achieve a system for optimal player enjoyment.

There have been many interesting studies on what defines 'fun' in video games and many of these models have a tendency to overlap and relate to each other in one way or another. In our following research, we focus on general trends that are present in terms of fun, challenge and boredom.

2.3 Player Modeling

The primary goal of player modeling is to understand how a player's interaction with a game has an effect on the player's individual experience. We try to get an understanding of players' cognitive, affective and behavioral patterns. [11]

We can make out two main factors: player behaviour and player experience. While player behavior and player experience are interwoven concepts, there is a subtle difference between them. Player behavior is about what a player does in a game whereas player experience refers to how a player feels during gameplay. It goes without saying that a player's feelings during a game (the player experience), correlates with what the player does during the game (the player behaviour). To construct an accurate player model, we measure both the player behaviour and the player experience. When we but both together, we create the inputs for our player model.

When measuring player behaviour, we look at gameplay input, also called gameplay metrics. These concern any elements that can be derived from the direct interaction between the player and the game (e.g. number of jumps, time elapsed between milestones, buttons pressed). A major limitation of gameplay metrics is that they are always only indirectly observed. For example, a player who has little interaction with a game might either be really thoughtful and captivated, or just bored and busy doing something else. Gameplay metrics can only be used to approach the likelihood of the existence of certain player experiences. Therefore, one should never attempt to to use pure player metrics to make estimates of player experiences and make the game respond accordingly to those estimates, as they could be inaccurate. It is best to also keep track of the player feedback, and adapt only when these indicate that the player experience was estimated incorrectly.

There are different ways of measuring player experience. One way is to look at the player's emotional responses through physiological measurements. These include things like player's facial expressions, muscle activation, brain waves, posture or speech. The relationship between psychology and these physiological effects has been studied extensively. [11] Monitoring such bodily alterations could help us in constructing the player's model. These signals are usually obtained through motion trackins, electrocardiography (ECG), galvanic skin response (GSR), respiration, electroencephalography, and electromyography (EMG).

On the other hand, labeling player experience can also be done through manual annotation. Either done by the player himself or by a third-person. These assigned labels ideally need to be as close to the ground truth as possible. The accuracy of that estimation is regularly questioned as there are numerous factors contributing to a deviation between a label and the actual underlying player experience. The most common technique of labeling experiences is through a questionnaire. Manually annotating players and their gameplay is a challenge in its own right and arises many questions, such as: Will the labeling be dong by the player himself or a third person? Will the labeling consist of discrete states or instead involve a continuous representation? Should the responses be queried in an absolute or relative fashion?. We will discuss the details on how to measure player experiences further in the following section 2.4.

2.4 Measuring Player Experience

In this section, we focus on measuring player experience through manual annotation. We take a look at free vs forced responses, the different types of forms that we can use to represent our data, how to structure questionnaires and how to measure the player experience.

2.4.1 Free vs. Forced Responses

Player experiences can be evaluated either with free responses (e.g. thinking out loud) or forced responses (e.g. questionnaires). Free response contain more detailed information, but are often unstructured, chaotic and therefore become hard to analyze. On the other hand, forcing players to self-report their experiences using directed questions constrains them to specific items. In the remainder of this section, we will focus on forced responses as these are easier to analyze and are far more appropriate for data analysis and player modeling.

2.4.2 Forms of Data

There are three types of data for structuring player experiences from which we can pick: ratings, classes and ranks. Firstly, with ratings a player is asked to pick between a scalar value or a vector of values to indicate his opinion on given statement. Examples are "Did you find the game challenging?" or "Did you lose track of time?". Nowadays, ratings are unarguably the most widely used approach for assessing different aspects of a user's experience. The most popular rating-based questionnaires make use of questions in the format of a Likert scale, which asks the user to indicate their level of agreement with (or disagreement) with a given statement, within a range of predefined points. See figure 2.2a for an example.

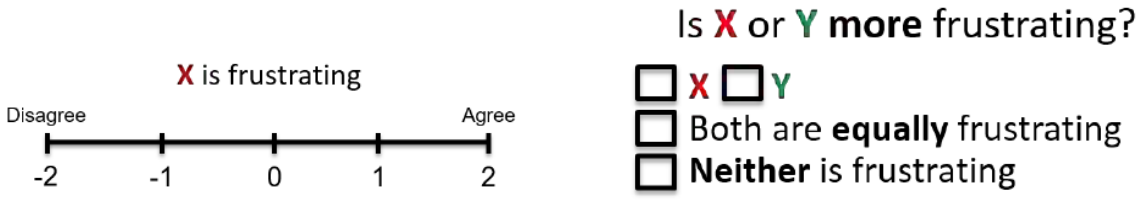
Rating-based questioning is inherent to limitations, which are often overlooked. Firstly, ratings are often analyzed by comparing their values across all participants. This neglects the existence of inter-personal differences as the meaning of each level on a rating scale may differ across participants. For example, two participants assessing the difficulty of a level may assess it as exactly the same difficulty, but then one rates it as "very easy to play" and the other as "extremely easy to play". Ratings also struggle with primacy and recency effects, in which information at the beginning and at the end are retained better than information in the middle. Ratings are also often converted into ordinal values. In most questionnaires, labels are represented as pictures or adjectives (such as "moderately", "fairly" and "extremely"). These labels are often converted to integer numbers, violating basic axioms of statistics. Since ratings suffer from these shortcomings, we take a look at other options.

++ refs

The second data type is the class-based format. Classes ask players to select from a set of two or more options. Examples of questions are "Was that game level frustrating or not?" or "Which level was the most stressful?". Classes have less statistical limitations compared to ratings, but do not reach the same level of detail.

Finally, rank-based ask the player to make a preference ranking between two or more options, such as different gameplay sessions. In the case of pairwise preferences, the player is asked to compare two options and specifies which one is preferred. With more than two options, the participants are asked to provide a ranking of some or all the options. An example of a ranked-base questionnaire is 4-alternative forced choice (see figure 2.2b).

++ why rank better, does not suffer from ...



(a) Ratings on a 5-point Likert Scale

(b) Rankings through 4-alternative forced choice

Figure 2.2: Examples of ratings vs rankings.

++ make own version of images (copyright things?)

2.4.3 Questionnaire

Questionnaires can contain elements of player experience (e.g. the Game Experience Questionnaire [3]), demographic data and/or personality traits. Similar research questionnaires often ask players to rank games of each game pair with respect to fun, challenge, boredom, frustration, excitement, anxiety and relaxation. The selection of these seven states is based on their relevance to computer game-playing and player experience. [12] Smaller, initial investigations only focus on three emotions: fun, challenge and frustration. [7]

2.4.4 How to measure (++different title?)

Annotation can happen either within particular time intervals or continuously. The process of annotation however, appears to require a higher amount of cognitive load. Therefore, One should aim for short annotation at intervals where a change in the player's state occurs. [11] Annotation can also happen pre, during or post gameplay. It should be taken into consideration that self-reports fade over time and the experience felt near the end of a session is often stronger, this effect is called the peak-end rule.

Evaluating player experience can either be done by the player himself or by a third-person. However, the player's annotations should normally be closer to their actual experience (the ground truth), compared to a third person. Self-evaluation, however, may suffer from self-deception and memory limitations (memory-experience gap).

2.5 Analysis

++ predictions, classification, ...
++ player personas

3 Methodology

In what follows is an explanation of the methodology followed to analyze difficulty parameters, model player behavior, and strive towards a system for DDA.

3.1 Game Selection

Before being able to gather information about player experience, a game has to be selected that is suitable for the context for this thesis. Due to the current circumstances of COVID-19, real-life experiments are out of the question and this search was limited to remote evaluations.

To select thegame, the following parameters have been taken into consideration:

1. **Duration.** Shorter games are desirable. Since the player's experience will be evaluated in between games to avoid loss of memory, ++ and ++ (like explored in section **??**), games should be as short as possible.
2. **Number of game elements.** Too few elements are not enough to make enough variation in difficulties. On the other hand, too much elements can overload the complexity and make it too hard to discern a proper amount of difficulties. A game with a good balance in amount of elements, not too much, but also not too little, would be an ideal fit.
3. **Easy to implement.** To set up our application, an open source model of the game should be available that can be adjusted to our own likes, or it is doable to start from scratch and remake the game within realizable time considering the time frame of this thesis.
4. **Expected popularity.** Taking into account the current circumstances, a big switch has been made to online activities. An online game is expected to gain popularity. We also look for a game the will attract attention with the biggest target audience.

To find a game that fits these parameters, a collection of games was gathered that fit at least one or more of aforementioned parameters. These games include: Chrome Dino, Pacman, Flappy Bird, Snake, Space Invaders, Pong, Super Mario Bors. and Tetris. See figure A.1 for a visual overview of these games. To make a decision about which of these game fit our requirements the best, each game was assigned a score for each of the parameters from 0 to 5, and globally compared. These scores are of course annotated manually, and are subjective to personal interpretation, but effort has been put into making them as close to the ground truth as possible.

From these scorings, there are three games that come out on top: Chrome Dino, Pacman and Flappy Bird. See figure 3.1 for a radar chart with the individual scores of these games. Herein, we see that Chrome Dino is the game which meets our requirements the most. See appendix A.2 for an overview of the analysis of all games.

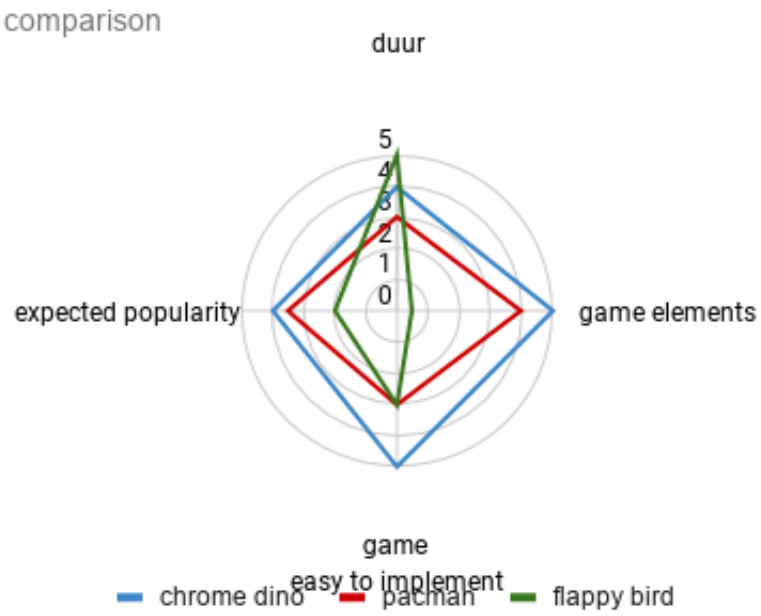


Figure 3.1: Comparison of Chrome Dino, Pacman and Flappy Bird.

3.2 Game Mechanics of Chrome Dino

Chrome Dino, also known as the Dinosaur Game, T-Rex Game, or Dino Runner, is a built-in browser game in the Google Chrome web browser. A user can encounter this game while browsing and a connection to the internet was not able to be established.

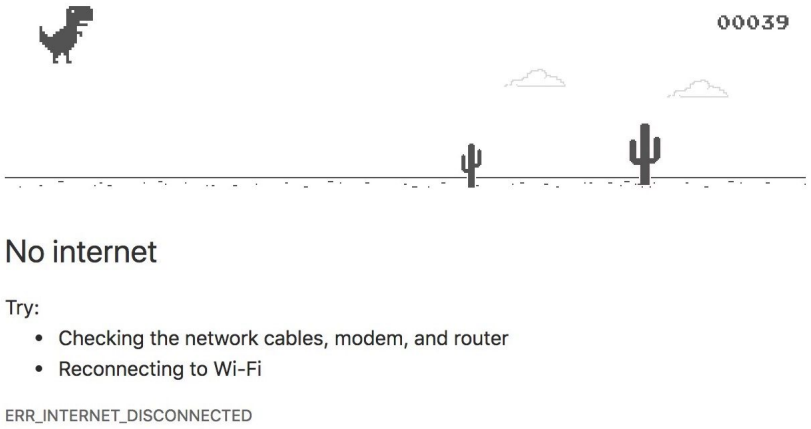


Figure 3.2: Chrome Dino game in the Google Chrome browser.

The game can be started by tapping the dinosaur (on mobile) or pressing the space or arrow-up keys (on desktop). The player controls the running dinosaur by jumping (tap, space, or arrow-up) or ducking (arrow-down) to avoid obstacles, which include cacti and pterodactyls. When a player collides with any of these, the game is over. A score is accumulated for the distance traveled and the goal of the game is to stay alive as long as possible, hence getting a score as high as possible. A new game can be started by pressing the space or arrow-up keys again.

When the player reaches 700 points, the game switches from black graphics on a white background (representing daytime) to white graphics on a dark background (representing nighttime). Reaching the score of 900 (200 points further ahead) will switch the color scheme back to daytime and the next switches will occur at consecutive multiples of the milestones. For example, 1400 points for the next nighttime and 1600 for the corresponding switch to daytime.

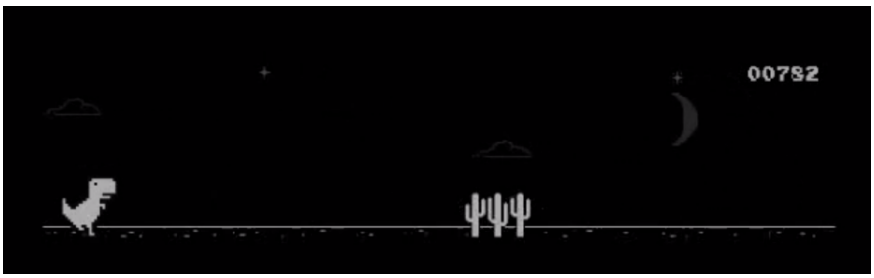


Figure 3.3: Chrome Dino: Night Mode.

At the beginning of the game, a period of time occurs when no obstacles are spawned yet, to let the player adapt to the pace of game. The game starts at a moderate speed and slowly accelerates over the distance passed.

Obstacles consist of: large cacti, small cacti and pterodactyl. Both large and small cacti can occur in different width sizes, up to a maximum of three. Each one making it harder to jump over. Pterodactyl can occur at three different heights. While a pterodactyl that flies low over the ground requires the player to jump over, a pterodactyl that flies in the middle can either be ducked under or jumped over. A pterodactyl that that is positioned at the highest point does not per se require a duck. The dinosaur fits right under it while running regularly. A player can still duck underneath it, but it is not necessary. See figure 3.4 for an overview.

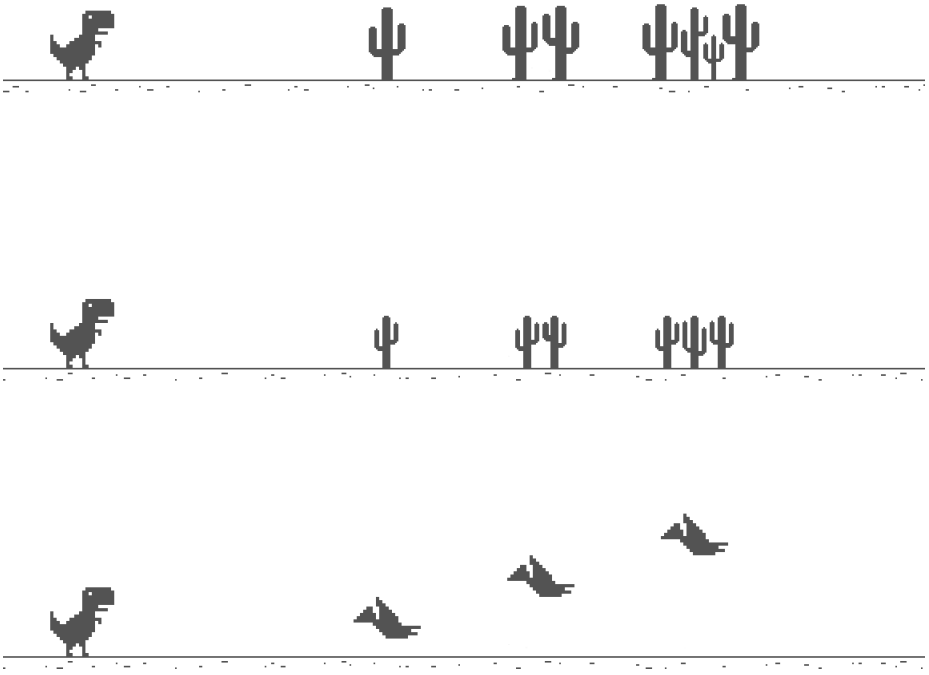


Figure 3.4: Obstacles in Chrome Dino.
Large cacti, small cacti, and pterodactyl, in their available sizes and heights.

3.3 Data Collection and Analysis

To evaluate player experience and analyze it in the scope of DDA, player game data has to be collected. This is done with the goal of giving insight into how the underlying game metrics correlate to the game experience. Two types of data will be recorded. Game metrics, that log events during gameplay (such as the number of jumps, the achieved score), as well as the experience responses will be gathered through questionnaires in between games.

For each consecutive pairs of two games A and B, subjects will report their preference for several emotional states (fun, frustrating and challenging) using a 4-alternative forced choice (4-AFC) protocol.

We will collect human data through a constructed variant of the Chrome Dino game, made available on a public website. Every new instance of the game will generate a new random version of the game, with different underlying game parameters that are randomly sampled from fixed ranges to make every game change in difficulty. The details of how this application was set up, will be explained in the data collection chapter (see chapter 4).

3.4 Analysing Results

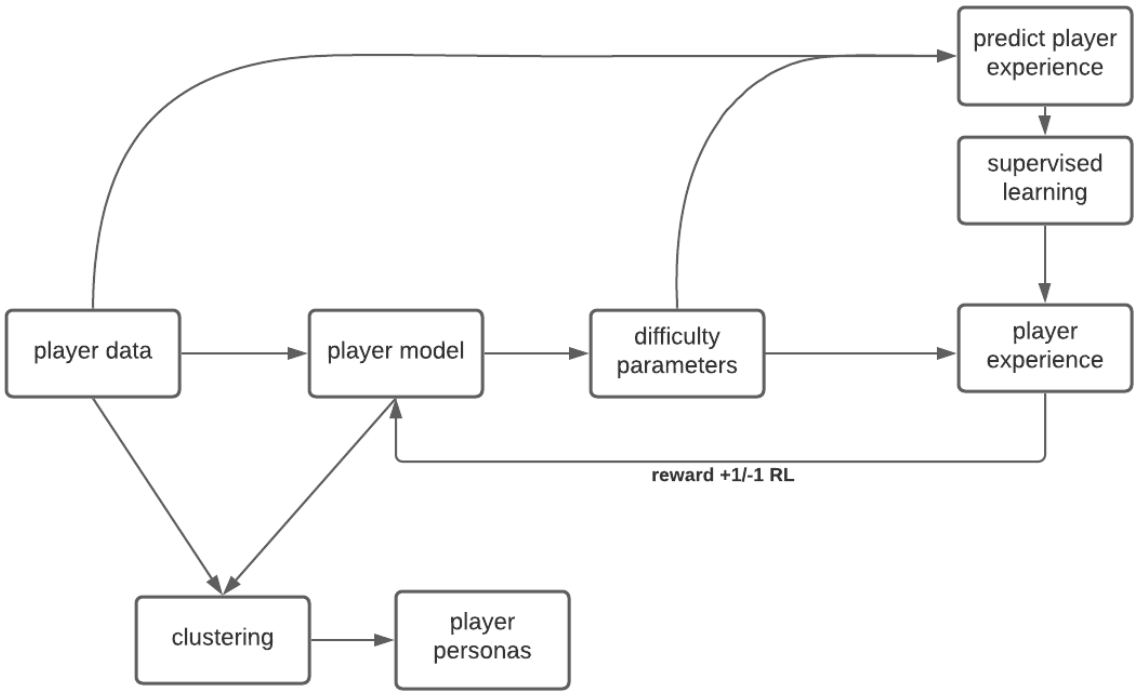


Figure 3.5: Diagram.

++ explain fig

To get a global overview of the gathered data, a descriptive analysis can be done to gain further insight into possible trends. To discern different clusters, clustering techniques or dimensionality reduction methods like UMAP or t-SNE can be applied.

4 Data Collection

In order to evaluate player experience in Chrome Dino, player data has to be collected. The goal of this collection is to find possible relationships between the evaluated player experience and collected records from gameplays.

4.1 Types of Data

Before any modeling could take place, we needed to collect data. We collected three types of data:

1. **Difficulty Parameters.** Underlying game parameters that give each new game a different difficulty. For example, the begin speed or different obstacles generated. See table 1 for a full overview of all the difficulty parameters.
2. **Game Metrics.** Upon occurrence of different events, game log are captured live during the game to encapsulate player behaviour. Examples are keyboard or mouse logs, the total number of jumps or the final score. A summary of all the different game metrics that are logged is given below in table 2.
3. **Player's Experience.** The player's experience is evaluated through a questionnaire.

At first, our goal was to create underlying game parameters based on predetermined groups of difficulty settings (e.g. easy, medium, hard). When trying to construct those kind of difficulties, we realised that this was a harder task than we though at fist. For example, an easy game could intuitively be thought of as a game with a relatively low start speed or only one type of obstacle. However, when testing out these settings ourselves, these games were sometimes very tedious and became harder to focus on. We decided not to make any presumptions about which parameters correlated to which difficulties. After all, putting more degrees of freedom towards these parameters allows us to later make a more free analysis about our data, without grouping them under 'easy' , 'medium' or 'hard. After all, this is the exact goals of our research, to find out which parameters contribute to a feeling of difficulty for the players. Since Chrome Dino is a rather simple game, the total amount of underlying parameters is fairly compact. We therefore chose to capture every possible underlying parameter as much as we can. Each game's difficulty is altered by randomly sampling the underlying difficulty parameters to get a wide spread on presented game difficulties and corresponding experiences.

	Difficulty Parameter	Type	Description
1	`speed`	integer	start speed value between `5` and `10`
2	`acceleration`	float	acceleration added to `speed` in increments value between `0.002` and `0.01`
3	`max_speed`	integer	acceleration stops when `max_speed` is reached value between `speed` and `15`
4	`obstacle_types`	array	obstacle types that will be generated any combination of `cactus_large`, `cactus_small`, and `pterodactyl`
5	`obstacle_types_chances`	dictionary	chance of every obstacle type to occur chances assigned to `obstacle_types`, together 100%
6	`max_obstacle_length`	integer	the max width of small and big cacti `1`, `2`, or `3`
7	`pterodactyl_height`	array	pterodactyl can fly over ground, in the sky or in between any combination of `50`, `75`, and `100`
8	`clear_time`	integer	time before the first obstacle will occur value between `1000` and `5000`
9	`min_gap`	integer	minimum gap between obstacles value between `250` and `400`
10	`max_gap`	integer	maximum gap between obstacles value between `min_gap` and `400`
11	`night_mode_enabled`	boolean	is the switch between night- and daytime enabled `true` or `false`
12	`night_mode_distance`	integer	the distance after which will be switched to nighttime value between `100` and `1000` with steps of `100`

Table 1: Difficulty Parameters

While selecting the game metrics, we tried to pick those that would be most meaningful in relation to the player's experience. However, the same reasoning as with game parameters as mentioned before can be applied so we decided not to make any assumptions about whether or not which parameters contribute to a player's experience. Therefore, we chose to capture as much metrics (or game logs) as possible. To test this, we set up a replay functionality. Because of the deterministic nature of the game, this is possible and allows to look deeper into individual games in the future that exhibit interesting features. It also validates the logging system as replayed and played games must have the same outcome.

	Game Metric	Type	Description
1	`score`	integer	achieved score at the end of a game
2	`collision_obstacle`	dictionary	obstacle upon which the dinosaur collided and caused death `large_cactus`, `small_cactus` or `pterodactyl`, with corresponding configuration parameters (`width` and `height`)
3	`inverted_gameover`	boolean	did the player die while in night mode
4	`nr_jumps`	integer	the total number of jumps done by the dinosaur
5	`key_logs`	array	all key and mouse events

Table 2: Game Metrics

When a game ends, the game parameters and game metrics are stored together as a single record in the gameplays database.

After playing two (or more) consecutive games A and B, players are presented with a form structured in a 4-AFC way. Players can choose whether they experienced game A or be to be more fun, challenging, and frustrating.

Questions are structured as follows:

`Which game was more X?`

- `A`
- `B`
- `Both`
- `Neither`

Where A is the previous game and B is the current game and X is the emotional state in question: fun, challenging or frustrating. Upon submitting a form, the answers are stored as a single record in the form database, with references to the id's of the corresponding games A and B.

4.2 Website

In this section, we explain how the website was set up, how we tracket progress during the data collection and which challenges we had to overcome.

4.2.1 Game

A variant of the Chrome Dino was constructed based on a version of the original source code. [1] This version was extracted from the offline error-page that is built-in in the chromium browser, an open source project that is the base for the original Google Chrome browser.

The main components of the application exist of the Javascript frontend that runs the game, and a backend that runs a MongoDB database for storage of gameplay and questionnaire records. MongoDB was chosen for it's simplicity. The website was put online at `chromedino.ml`.

Upon opening the website, the player is presented with a screen as in figure 4.1. Instructions on how to start the game are visible on the bottom of the playing field. Upon performing one of the start actions, the game will begin. The horizon will expand and the dinosaur will start running. After the initial clear time, the first obstacle will be generated.

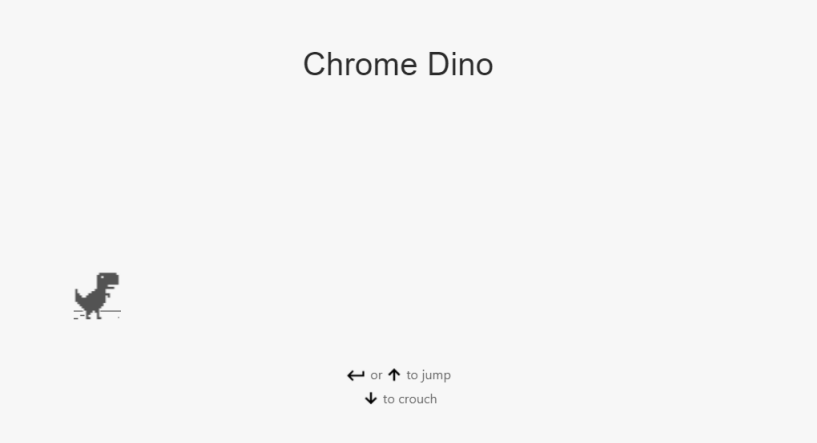


Figure 4.1: Chrome Dino Startscreen.

Upon death, the difficulty parameters and game metrics (logs) are saved in the database, together with a snapshot of the gameover screen. This snapshot will be used in the questionnaire to display the played game.

4.2.2 Questionnaire

After playing two or more games, the player will be presented with the questionnaire form that evaluates the player's experience.

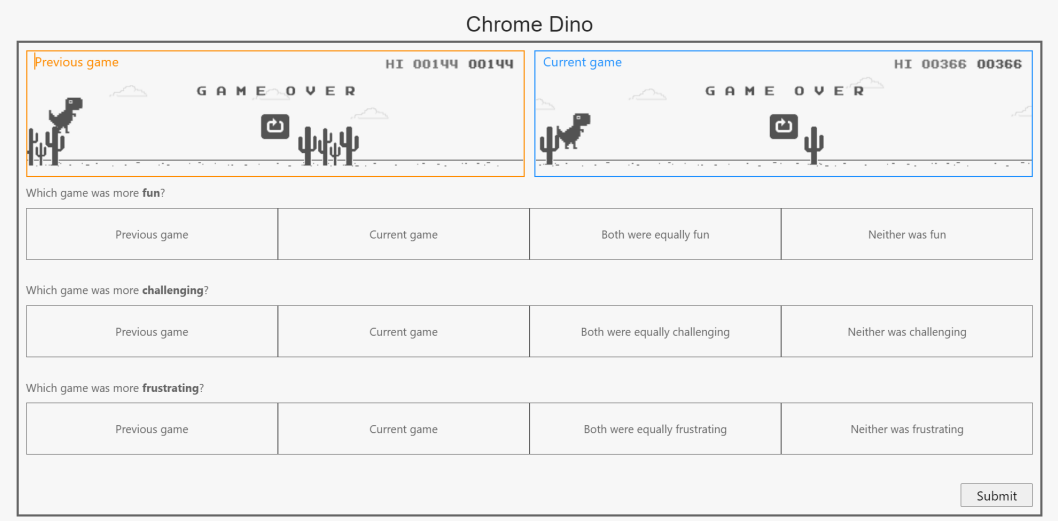


Figure 4.2: Chrome Dino Form.

4.2.3 Dashboard

To keep track of incoming data, a dashboard has been composed to give a quick overview of the most important data. The dashboard contains:

1. **Record counters.** The amount of total accumulated games played and filled in questionnaires are displayed.
2. **Descriptive graphs.** Small descriptive graphs are made to get a feeling of global statistics. These graphs give a head start for further analysis and will be examined further in chapter 4.4.
3. **Gameplays overview.** An tabular overview of the entries in the gameplay database, with their most relevant parameters. Each row in the table is clickable and connects to the replay functionality for that instance.

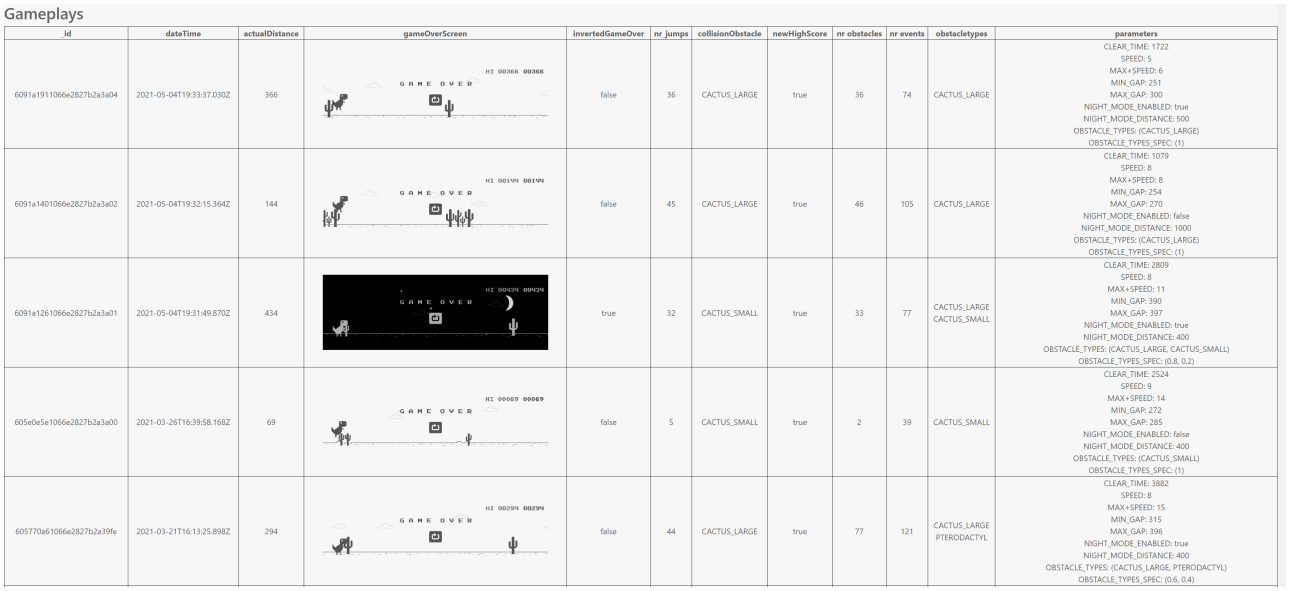


Figure 4.3: Chrome Dino Dashboard.

4.2.4 Challenges

When designing the form, it initially had the exact structure as mentioned in 4.1. The responses to the questions were: A, B, both were equally X and neither was X. With X being fun, frustrating or challenging. The gameover screens of the corresponding games were given the labels 'Game A' and 'Game B'. From an outsider's perspective, however, this can be confusing. It is hard to discern which game corresponds to A and which game corresponds to B. If it was the game hat had just been played or the game before that, and in which order. In an attempt to make the form more unambiguous, the labels were changed to 'previous game' and 'current game' as can be seen in figure 4.3. We also made sure that both games appeared in chronological order from left to right. To make it even more clear, colors were added. The previous game was given an orange color, and the current game was given a blue color. The responses 'both were equally fun' and 'neither was fun' were also given the colors red and green correspondingly, to signal a negative and positive feeling.

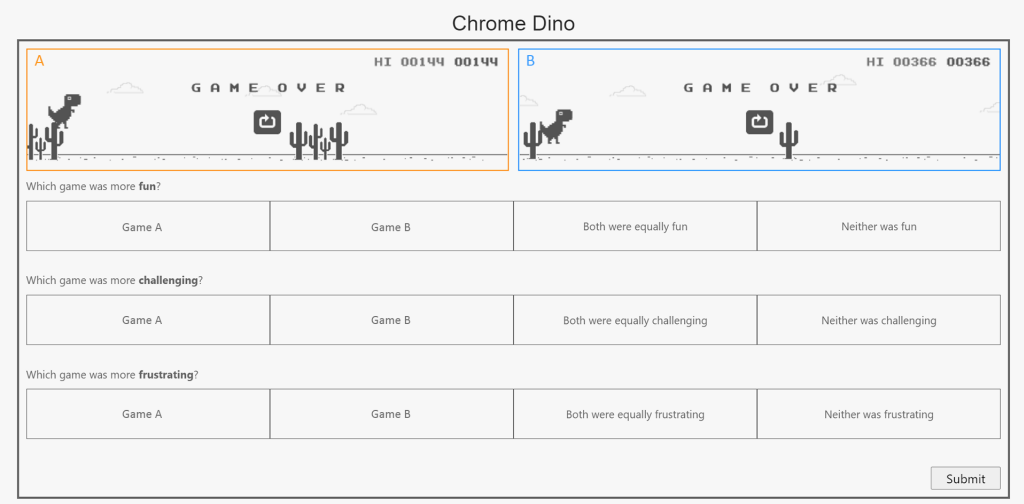


Figure 4.4: Chrome Dino Form with the old format.

The original Chrome Dino game does not contain instructions on how to begin the game, i.e. "`enter` or `up` to jump, `down` to crouch". However, a few subjects reported that it was unclear on how to precisely to play the game, presumable subjects who have never played the Chrome Dino game before. This being the case, we decided to add the instructions to the start screen. The instructions also stay beneath the playing field when the game has started.

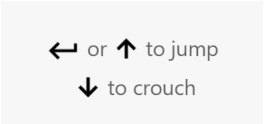


Figure 4.5: Chrome Dino Instructions.

Due to the deterministic nature of the game, we were sure to be able to set up the replay functionality by just replaying the jumps and ducks of the dinosaur, combined with giving the games the same generated difficulty parameters. However, key and mouse events are not generated per time tick, but are timestamped. Timestamps are unfortunately very system dependent and hard to get right. Due to this trait, a lot of precaution had to be taken in making the system act according to the replay events. We added a parameter for individual adapting of system time dependencies, to compensate for incorrect gameplays as much as possible, but we will never be able to make all replays foolproof.

4.3 Data Collection

Data is collected over the Internet. Subjects are recruited via social media and networks of colleagues, friends and family.

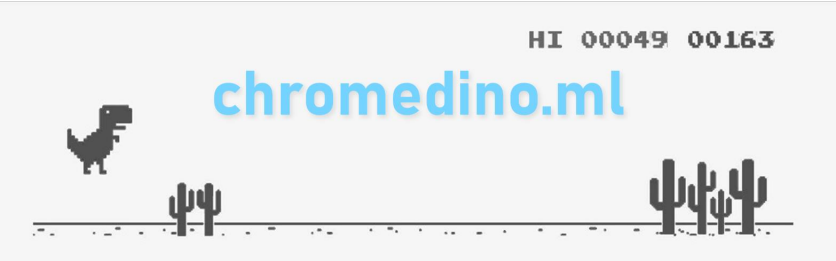


Figure 4.6: Image used to advertise on the internet.

4.4 Analysis

In total, 1074 gameplays were recorded, together with 524 answers to the questionnaire. In a first step, some preprocessing is done to the datasets, after which a descriptive analysis is made.

4.4.1 Preprocessing

4.4.2 Descriptive Analysis

4.5 Conclusion

5 Player Modeling

Here comes some text. This text makes use of 1.5 line spacing.

5.1 A

5.2 B

5.3 Challenges

++ ranked based

6 Conclusions and Future Work

Here comes some text. This text makes use of 1.5 line spacing.

References

[1] P.W.D. Charles. t-rex-runner. https://github.com/wayou/t-rex-runner, 2013.

[2] Dagmara Dziedzic and Wojciech Włodarczyk. Approaches to measuring the difficulty of games in dynamic difficulty adjustment systems. *International Journal of Human–Computer Interaction*, 34(8):707–715, 2018.

[3] IJsselsteijn, W. A., de Kort, Y. A. W., & Poels, K. The Game Experience Questionnaire. 2013.

[4] M. Csikszentmihalyi. Flow: the Psychology of Optimal Experience. Harper Collins.

[5] Nacke, L. E. Flow in games: Proposing a flow experience model. Proceedings of the workshop on conceptualising, operationalising and measuring the player experience in videogames at fun and games. page 104–108, 2012.

[6] Newzoo | Games, Esports & Mobile Market Intelligence. 2020 Global Esports Market Report. 2020. https://newzoo.com/.

[7] Chris Pedersen, Julian Togelius, and Georgios N. Yannakakis. Modeling player experience in super mario bros. In *2009 IEEE Symposium on Computational Intelligence and Games*, pages 132–139, 2009.

[8] Robin Hunicke, Vernell Chapman . AI for Dynamic Difficulty Adjustment in Games .

[9] Kalyani Sonawane. Serious Games Market - Global Opportunity Analysis and Industry Forecast, 2016-2023. 2015.

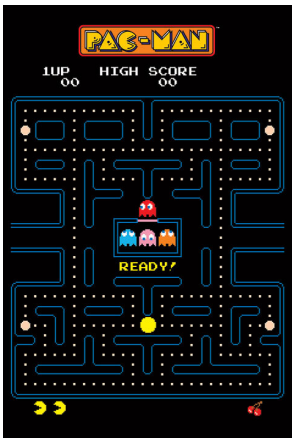
[10] Thomas W.Malone. What makes computer games fun? page 258–277, r 1981.

[11] Georgios N. Yannakakis and Julian Togelius. *Artificial Intelligence and Games*. Springer, 2018. http://gameaibook.org.

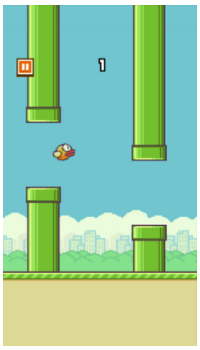
[12] Yannakakis G.N., Hallam J. Ranking vs. Preference: A Comparative Study of Self-reporting. *Lecture Notes in Computer Science*, 6974, 2011.

A Game Selection Analysis

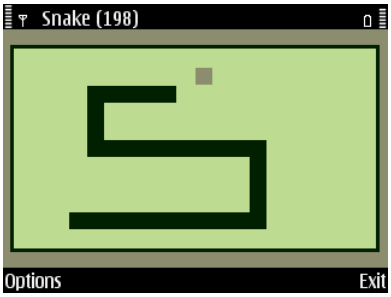
(a) Chrome Dino



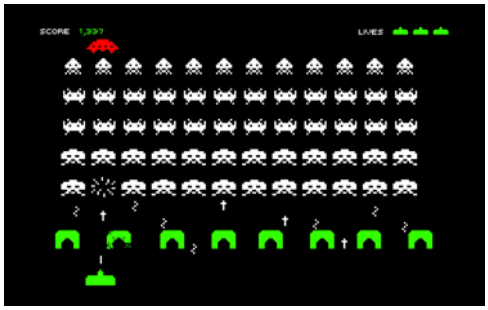
(b) Pacman



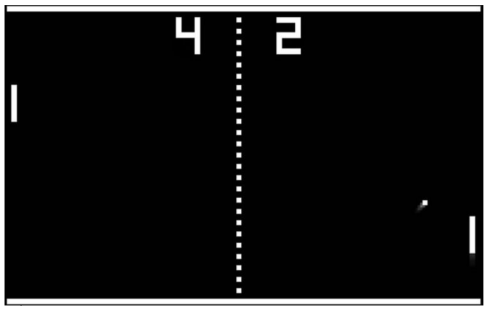
(c) Flappy Bird



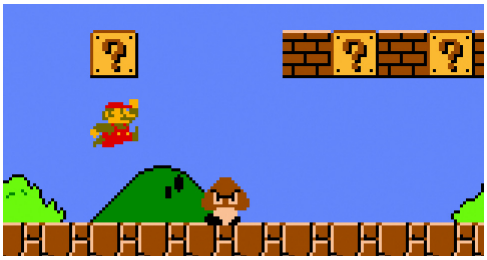
(d) Snake



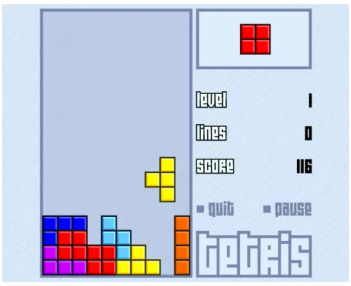
(e) Space Invaders



(f) Pong

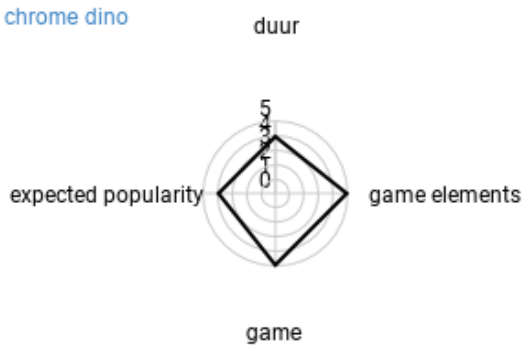


(g) Super Mario Bros

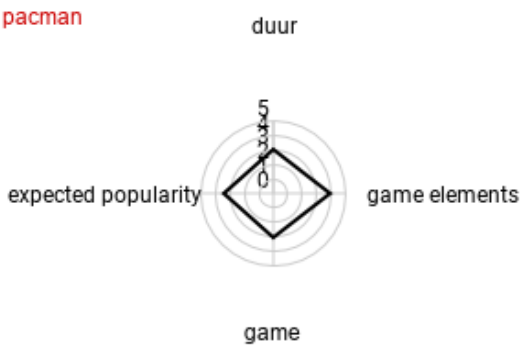


(h) Tetris

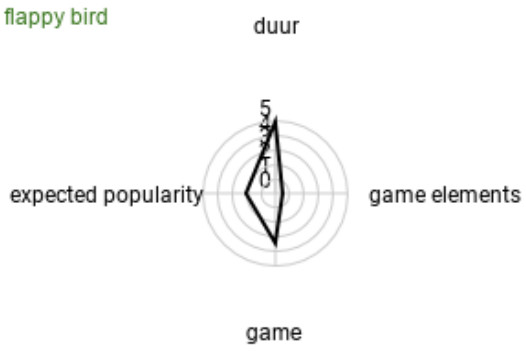
Figure A.1: Video Games considered for selection.



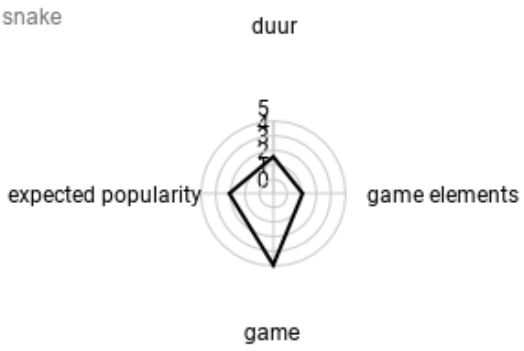
(a) Chrome Dino



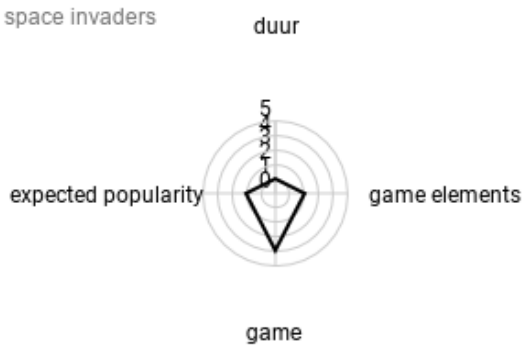
(b) Pacman



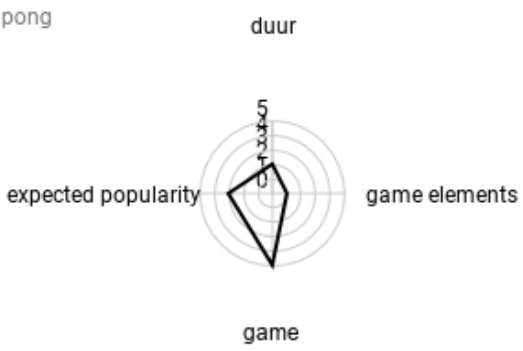
(c) Flappy Bird



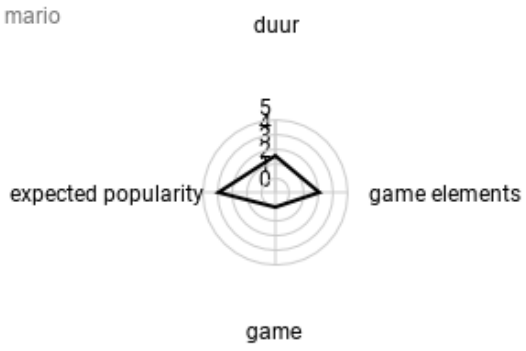
(d) Snake



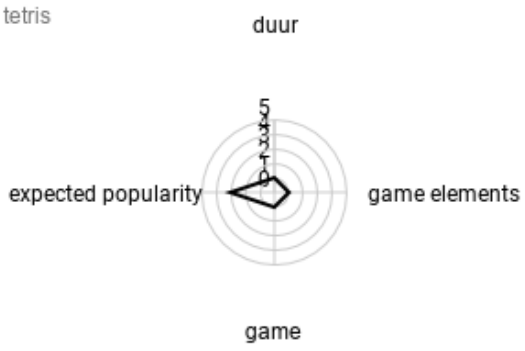
(e) Space Invaders



(f) Pong



(g) Super Mario Bros



(h) Tetris

Figure A.2: Analysis of the selected video games.

B APPENDIX B

Here comes some text. This text makes use of 1.5 line spacing.