Otto-von-Guericke University of Magdeburg

School of Computer Science
Department of Computational Intelligence

Diploma Thesis

The Relevance of Graphics and Sound for the Manipulation of a Video Game Difficulty Using Feature-Based Dynamic Difficulty Adjustment

Author:
Constantin Graf

August 13, 2012

Supervisors:

Prof. Dr. habil. Rudolf Kruse
School of Computer Science
Otto-von-Guericke University
Universitätsplatz 2
39106 Magdeburg, Germany

Pascal Held M.Sc.
School of Computer Science
Otto-von-Guericke University
Universitätsplatz 2
39106 Magdeburg, Germany

Graf, Constantin:
The Relevance of Graphics and Sound for the Manipulation of a Video Game Difficulty Using Feature-Based Dynamic Difficulty Adjustment
Diploma Thesis, Otto-von-Guericke University
Magdeburg, 2012.

Abstract

In this thesis, a modern approach for balancing video games called *Dynamic Difficulty Adjustment* (DDA) is conceptualized and implemented. It utilizes *features*, which serve as a minimal unit for every parameter of a game that helps defining its difficulty. Special emphasis is laid on graphics and sound features, as both scientific studies and commercial game products have mostly neglected their influence on balancing.
In order to be able to verify the results, an accompanying game project called *Dream Runner* is created that serves as a testing environment for the experiments. It is a two-dimensional platformer game that focuses on navigating through a procedurally generated terrain and eliminating enemies to survive. Every element of the game is designed to incorporate the concept of features to offer a versatile and highly dynamic experience. For each experiment, a group of volunteers plays the game and fills in a survey that investigates their perceptions of the game. In addition, data about the behavior of the participants is gathered for the evaluation process.
The results show, that the feature-based DDA manages to adapt the difficulty according to the skills of the player. The graphics and sound features have a measurable impact on the difficulty, although it is much smaller than the influence of gameplay features. On second place, the graphics prove to have a noticeable impact, by manipulating the viewing conditions and creating distractions or by drawing the player's attention to a certain region of interest. The sound features place third and show, that the effects of these features are subtler and unfold their potential easier when combined with features from other areas of the game.
In conclusion, graphics and sound features perform not as well as expected, but they have a measurable impact on the overall difficulty, even when the effect is mostly supplementary. Therefore, these two fields should get more recognition when it comes to video game balancing.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Dynamic Difficulty Adjustment in Games

The balancing of video games is a huge topic in game theory and can greatly affect the overall quality of a game, reaching from giving the player a good and fun challenge to decreasing the player's motivation to continue playing (Bailey and Katchabaw, 2005). In the past, in order to reach the largest possible audience, the developer had to employ vast resources to ensure a well-balanced game (Hunicke and Chapman, 2004). Often, three or more difficulty settings had to be created, which multiplied the time and effort put into this stage of the development (Bailey and Katchabaw, 2005).
In recent years, a new method has emerged, that decreases the resources needed to find the right balance by shifting a large portion of the balancing process into the actual run-time. In the following chapters this method is referred to as *Dynamic Difficulty Adjustment* (DDA).
Dynamic Difficulty Adjustment describes the procedure of customizing the challenge for the player dynamically during run-time. The goal of DDA systems is to enrich the player's experience and enhance his motivation to play the game. In order to reach this goal, the system analyzes the behavior of the player and reacts to critical situations by altering parameters like the speed of enemies, the number of healing items, or the damage taken by attacks (Bailey and Katchabaw, 2005). DDA systems can greatly increase the feeling of joy for the player as it responds to the player's needs for challenge and reward (Yun et al., 2010). On the other hand there are also drawbacks, like the adjustments being too disruptive or the calibrations of the modification not being adapted to the player's skills (Bailey and Katchabaw, 2005).

In this segment, the advantages and disadvantages of DDA systems are discussed and how they should affect the developer's decision for a suitable balancing solution. For this, it is assumed that the DDA system works as intended.
As DDA systems aim to create customized situations based on rules learned online or offline, one of the biggest advantages is that these games can reach a broader audience. Theoretically, the difficulty setting can be neither *too high* or *too low* with DDA, as the dimensions of the difficulty only depend on the configuration of the system. There is also no gap between static difficulty settings like *easy* and *medium*, which can already be too big to satisfy every player (Prabhu, 2010). Using the player's ability as a guideline, the challenges increase in difficulty smoothly and seamlessy. (Bailey and Katchabaw, 2005).
Additionally, by reducing the level of frustration to a minimum, the player's long-term motivation will be increased. Reiterating the same parts of the game after completion still offers a challenge to the player, as the difficulty is dynamically adjusted in relation to the player's skill (Lopes and Bidarra, 2011). From a developer's point of view an advantage is that once the DDA system works as planned, less testing is needed to balance the game, as it will regulate itself. Therefore, the developer does not have to balance a whole game, but only the DDA system. This results in more time for the actual

development (Hunicke and Chapman, 2004).
Risks are, that DDA systems might create a problem as they tend to use a lot of resources for dynamic changes and calculations as the game is running. Depending on the size of the game project and the platform it is designed for, this might take away resources that could have been useful in other areas of the game (Spronck et al., 2006).
Applying regular balancing techniques to create a static learning curve gives the game designer the control to pace the game according to his ideas. He can, for example, add boss fights that are especially hard to win or gameplay forks that change the story according to how well the player played the game. Using a standard DDA system eliminates the possibility to arrange the difficulty curve that way (Hunicke and Chapman, 2004), although the concept could be altered to achieve this, by combining the dynamic adjustments with partly scripted sequences.
For the player, a DDA system that regulates the challenge too much could decrease the player's feeling of accomplishment and, subsequently, the joy of playing and completing challenges. He could even become indifferent about winning the game or not. In this case the DDA is not balanced in itself and will most likely never find the right difficulty for the player (Bailey and Katchabaw, 2005). In addition, another problem could occur, if the DDA system is unable to respond to unpredicted behavior of the player (e.g. if two players with different skills play the game by turns) (Hunicke, 2005).

DDA systems have already been used in well-known games from big developers, like Valve, Capcom, and Nintendo. In this section, the different systems of these games are explained in their function and how they improve the game for the player (Yun et al., 2010).
One of the oldest example for DDA systems are racing games, as they often use a so-called *rubber-band mechanism*. This means, that the AI drivers will be drawn toward the player as if they were tied to him with a rubber-band, making them slower if they are ahead of the player and faster if they are behind. The strength of the effect is proportional to the distance between two drivers. Using this technique ensures that the player will never drive alone and always has the motivation to pass his next opponent, although it could also be interpreted as cheating, as the AI does not follow the same rules as the player (Hunicke, 2005). Games like Mario Kart use this kind of DDA system.
In Resident Evil 5 a combination of traditional difficulty settings and a more dynamic approach is being used. On the surface there are five different settings the player can choose from, but underneath the game separates them even further into ten levels, which are then handled dynamically as the game is played. This way, the system can correct the difficulty if the player does not perform well, moving towards an easier or harder setting progressively without changing the difficulty level the player chose at the beginning.
In Left 4 Dead changes are more drastic and often visible for the player. Every session of the game is supervised by a balancing tool called *AI director*. This system handles spawning enemies, item placement and can even change parts of the level to open new branches and close others. By this, the player gets a new experience every time as he does not know where the enemies will come from and where he has to go to finish the level.

1.2 Motivation

Up to this day, DDA systems still have not reached their full potential on the international games market due to their sparse use in commercial products (Hunicke and Chapman, 2004). Scientific approaches often lack the dynamics to apply them to a wide range of different games and are therefore not very attractive to game developers (Mladenov, 2010).
The potential within this new balancing strategy could have a big impact on the joy of the player. He would not be confused by standardized difficulty settings that only give a vague idea of the actual difficulty anymore (Bailey and Katchabaw, 2005). Additionally, a direct link between the implementation of the system and the perception of the player can be integrated. This means, that by changing parameters of the DDA, the whole balancing of the game can be customized, which makes it easier for

developers to make changes on a global level rather than modifying each element separately.
But to give the DDA system a truly global impact, it has to exceed changing only gameplay parameters like the health of enemies or the item placement (Mladenov, 2010). The influence of graphics and sound effects on the difficulty has not been investigated yet. These fields of a game need to be analyzed to comprehend the potential of DDA systems. Most of the feedback the player receives from a game is based on audio or visual cues, which can have many different forms. So to have the biggest control over the perceived difficulty, the repercussions of visual and audio effects have to be examined.
With this background, the conception and testing of new possible applications for DDA systems becomes a motivational task. For both the economical and the scientific area, finding new ways of keeping the user engaged and satisfied becomes mandatory, as the interactive medium grows in popularity and economic power (Yun et al., 2010). Therefore, the remaining, yet unexplored potential of DDA systems must be uncovered and explored to create reliable and diverse balancing systems in computer games.
For this work, the focus is especially set on the interplay between the three distinct fields of a game: gameplay, visuals, and audio. This way, an investigation of the impact of dynamic changes of graphics and sound effects on the overall difficulty can be extracted and then examined. Additionally, detailed changes can be monitored separately in order to get a new insight on how visuals and sounds can shape the learning curve of the player and what obstacles should be avoided in the process.

1.3 Goals

In order to make the results gathered in this thesis measurable, four goals have been constructed that mark the foundations of its scientific approach. In the following passage, each goal is presented and its meaning for this work explained.
The data gathered by the DDA system concerning the difficulty concurs by 80% with the surveys filled in by the players.
To evaluate the data collected by the DDA system, each test player is asked to fill in a detailed survey containing information about the experience with the game and general feedback. This information then is used to compare subjective impressions of the player with the objective data of the system. This makes it possible to give a statement about the dependability of the system, as its data should generally concur to those gathered in the survey. This also means, that by meeting the above goal the program succeeds in reflecting the actual difficulty for the player.
The game project designed for the thesis manages to keep the player motivated for at least 70% of the time.
As explained in Chapter 2.1, one of the biggest achievements gained through the use of DDA systems is to keep the player motivated for an extended amount of time, so that he is challenged enough to keep playing without being frustrated or bored (Bailey and Katchabaw, 2005). The survey filled in by the test players asks questions about their mood and feelings while playing and how they perceived the changes in the game. By evaluating this information it is possible to determine the level of immersion experienced by the player and when he felt motivated the most.
The surveillance and modification of the game through the DDA system takes place in a non-disruptive manner by 80% and does not interfere with the gaming experience.
An important aspect about games in general is to give the player the feeling that he is in control of his actions at all times. Although there may be a different set of rules than those the player knows from the real world, these rules should never be illogical and he should never have to ask himself why the game reacted as it did. If the game should take away the control, the player might feel cheated and lose interest in the game (Hunicke, 2005). That is why the DDA system created in this thesis works in the background where it cannot be noticed by the player. Every modification only affects elements that are not yet on the screen or even the game. And most important, the dynamic system will never alter the abilities or parameters of the player character.
The modification of graphics and sound effects has a measurable impact on the difficulty and 40% of the players name it a game-defining element.

The difficulty of a game is a complex topic as it is perceived differently by every player and many factors that define the overall gaming experience affect it (Mladenov, 2010). In this thesis, it is examined how changes of graphics parameters, audio effects, and gameplay elements define and shape the subjective difficulty experienced by the player. In order to be able to evaluate this goal, special experiments will be conducted that focus on the interaction between these different layers of the game. To analyze the importance of graphics and sound effects for the difficulty, the aforementioned survey will be used to gather information about how the modifications of those diverse features affected the overall difficulty. All these goals serve one common purpose: They aim to allow the analysis of the DDA to be as close to the player's perception as possible while creating a diverse and interesting difficulty for the player. Only when the player feels that he has surmounted the given challenges and that the accomplishments reached in the game are his own, the system will be believable and non-disruptive (Bailey and Katchabaw, 2005).

1.4 Related Work

There have been diverse approaches to automated game balancing, including parameter tweaking, probability functions, and dynamic scripting. Each approach has dealt with the existing problems of DDA systems in different ways and in this section the most interesting concepts will be presented.
In "AI for Dynamic Difficulty Adjustment in Games" (Hunicke and Chapman, 2004) a probabilistic method is introduced to deal with uncertainty in games. The DDA system designed in the paper is called *Hamlet*. It uses the game Half Life to analyze and adjust the supply and demand of items dynamically. Therefore, Hamlet consists of several libraries that monitor parameters of interest, choose and apply modifications, display data and system settings, and generate data from test sessions. During run-time, Hamlet uses statistical measures in order to observe player behavior. This becomes the basis for predicting future states of the player. If an undesired, but avoidable state is being predicted, Hamlet intervenes and changes the system to lead to a more desired state.
By this, the system recognizes when the player is in need for a change of the difficulty. It acts accordingly to decrease the chance of halting progress in the game. The goal is to keep the game constantly in a state between being too hard and too easy. This is achieved by preferring certain states like *exploration* or *battle* and suppressing others.
Hamlet utilizes two different action patterns:

1. *Reactive*: the elements used for modification are already in the game and the player has come in contact with them. These changes are easily integrated and their impact can be estimated well, although they also might irritate the player when he notices the changes.
2. *Proactive*: the modification is concentrated on elements that are inactive and not in the actual game, but are very likely to be integrated into the active game in the future. The impact of the changes is hard to predict, as there is no data gathered from any previous contacts between the element and the player. This might make reactive actions necessary later. On the other hand it is very unlikely that the player feels irritated by sudden changes.

To regulate the kind and rate of the modifications, a cost value is calculated, which is composed of several parameters, like the progress of the player, how often he died (in the whole game and in specific parts), and how often the DDA made changes to this part. From the action rules and the costs a set of modification rules is generated, which then modifies the game according to the current skills of the player.

The paper "Difficulty Scaling of Game AI" (Spronck et al., 2004) is concerned with the artificial intelligence of enemies in a game and is based on the previous work utilizing *dynamic scripting* by (Spronck et al., 2006). It is examined to what extent dynamic scripting can be applied to modify the AI so that there is a balance between player skills and the difficulty.
In the game industry, online learning is rarely used to gather data during run-time and modify the game following certain rules, although it could lead to an enhanced gaming experience. AI scripting is often

used in complex systems to e.g. enable the enemy to utilize a multitude of different behavior rules, but these are mostly static and cannot react to unforeseen changes.
Three expansions of dynamic scripting are being used: *high-fitness penalizing*, *weight clipping*, and *top culling*. Every expansion contains of a basis for rules that derive from the enemy type. The probability for a certain rule is proportional to its weight, which changes according to its success rate. To evaluate the success, a fitness function exists that examines the following parameters: if the team has won or lost, if the player character died or survived, the remaining energy of the player character, and the amount of damage dealt to the enemies. Every weight is being updated after every confrontation.
The different scaling systems are constructed as follows:

- *High-fitness Penalizing*: instead of rewarding a high fitness, a middle-rate value is being aimed for. High fitness values, however, are rated much worse and might even get punished.

- *Weight Clipping*: depending on the maximum weight of a rule, the behavior of the enemy changes drastically. In the case of high values only a small amount of almost ideal scripts are executed. For low values a broader quantity of scripts is being used, most of which are not optimal for the situation. Weight Clipping then adapts the maximum weight for every rule dynamically.

- *Top Culling*: Acts similar to Weight Clipping, but allows rules to surpass the maximum weight, which as a consequence will lead to their deactivation in the rule selection phase, so that scripts that often lead to the player losing the game will not be chosen anymore.

In order to test the three expansions, a simulation was created in a complex role-playing game containing two teams, one of which was using static scripts and the other one dynamic scripting. The static team applies different tactics, where *beginner* proved to be the most interesting one, as it will lead to the failure of the group in an imbalanced game most of the time.
For every tactic, 100 tests were conducted with the three dynamic expansions and also with basic difficulty scaling. Every test contained 150 confrontations between the two teams. The first 50 tests were used to calibrate the initial setup values, for the rest the number of games won was saved. A balanced game required the number of victories to be between 45 and 50.
The results showed that only the top culling expansions managed to guarantee a balanced game in every combination with the other team, closely followed by Weight Clipping.

"Polymorph: Dynamic Difficulty Adjustment Through Level Generation" (Jennings-Teats et al., 2010) takes a different approach by shaping the player experience through modifications inside of the actual level. To do so, it utilizes techniques from level generation algorithms and machine learning and then applies these to the platformer genre, which centers around timing and precision of the player input.
Therefore the paper introduces Polymorph, a level generator for 2D platformer games that builds a level procedurally while it is played. Every time the player reaches the end of the current section, a new one is built. These sections are called *chunks* and they incorporate dynamic changes to match the player's performance.
In order to find a good balance, Polymorph measures the level of difficulty and the skill of the player to create a statistical model of the play session. It then works with hand-created level chunks that are being put together to fit into an action-rhythm pattern that contains rules like jumping, running, or waiting, so that the actual level consists of a chain of level chunks matching these rules. To find the right calibration for each player, a short segment is being played and the player has to fill in a survey afterward. Polymorph monitors the progression of the player and evaluates the difficulty by analyzing the created level in different categories, e.g. the average gap width, the total change of height of the platforms, the width of the largest platform, or the occurrence of two-component adjacencies. Additionally, information about the behavior of the player is collected, like the amount of jumps, how often he died, or the amount of time where he did not move at all or backwards. To further improve the versatility of Polymorph, future additions could increase the number of examined adjacencies to a higher degree in order to evaluate more complex interdependencies between the different categories.

Chapter 2

Requirements

2.1 Flow Theory

A big part of game design centers around the question how to make a game engaging, motivating, and fun. An activity is fun, when the acting person receives a positive response from it. It rewards and challenges him enough to not lose interest or motivation. In a video game, the experience is often dramatically influenced by the direct interaction between the player and the game. If the tasks the game gives to the player are not challenging enough or too difficult to fulfill successfully, the player will not gain the positive feedback he needs to stay motivated (Andrade et al., 2006). This outlines a small area that can be found between frustration and boredom, called *flow* (see fig. 2.1).
The term flow has been characterized by Mihaly Csikszentmihalyi in 1985 (Csikszentmihalyi, 1991). It describes a state of mind where a person loses all conscious perception of his surroundings and is being absorbed entirely in the activity he performs. This is reached by finding the right balance between the challenge of the activity and the abilities of the player.
The concept of flow is composed of eight elements, as defined by Csikszentmihalyi, which are all required to enter this state of mind (Chen, 2007).

1. The activity provides a challenge.
2. The actions and the player's perception of it are merged.
3. There is one or more defined goals.
4. The player gets feedback directly linked to his actions.
5. The player is concentrated on the activity.
6. The player is in control of his actions.
7. There is a loss of self-awareness.
8. The perception of time is transformed.

It is easy to see that the concept of flow can be applied to a wide range of activities. As long as the aforementioned factors can be adapted to this specific area of application, getting into a state of flow is possible and in the case of video game development, very desirable.
As demonstrated in (Csikszentmihalyi, 1991), a lot of playful activities like chess, climbing, or even dancing can lead to complete absorption as the activity is carried out. Video games exert a strong effect on the player by giving him tasks to fulfill and rewarding him with a score, a virtual object, or the next piece of a story (Andrade et al., 2006). This direct feedback creates an emotional value for the player, building up further motivation to keep playing. It is therefore vital for the game to continuously challenge and reward the player as he learns and masters the game more and more (Goetschalckx et al., 2010). The learning curve of the player must be taken into account when designing the challenges for the player, so that he stays in the flow as long as possible (Mladenov, 2010). To achieve this, the whole

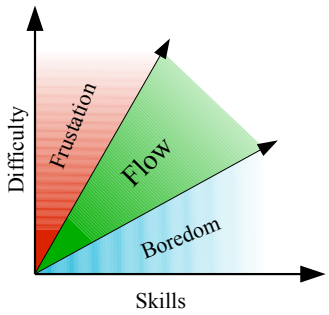

Figure 2.1: The concept of flow, positioned between the two extremes *frustration* and *boredom*.

experience should be designed to guarantee the player never becomes frustrated, because the difficulty of the challenge is higher than his current set of abilities, or bored, because the player has already exceeded the required skill for the challenge (Chen, 2007). In most video games, this is a problem, as the difficulty curve designed by the developer cannot match the learning curve of every player (Bailey and Katchabaw, 2005). In order to still find the right balance to make the game appealing to a wide audience, a dynamic approach has to be incorporated that can react to the player and shape the experience around his personal skills.

2.2 Dynamic Difficulty Adjustment

Dynamic Difficulty Adjustment (DDA; also known as *Dynamic Difficulty Balancing*) is a broader term for all systems that use dynamic modifications at run-time to enhance the playability of a game by altering game elements that have a direct impact on the difficulty. The goal is to find an appealing difficulty setting that is built around the skills and development of the player, rather than giving him a prefabricated difficulty curve (Bailey and Katchabaw, 2005). As the player becomes better (or worse, due to e.g. not playing for a longer period of time (Andrade et al., 2005)), the DDA analyzes changes and either *proactively* or *reactively* readjusts parts of the game. In this thesis, the term *DDA* refers both to the technique of adapting the difficulty dynamically and the actual implementation of the feature-based DDA system.

2.2.1 Structure

The structure of a DDA system is most often divided into three steps that are constructed consecutively (Charles et al., 2005). Each step can have a wide variety of methods executing the necessary functions to adjust the game, but the overall architecture is built as follows.

1. **Monitoring:** This segment is responsible for collecting all the necessary data for the next steps. The gained raw data outlines all the information that is used to compute the modifications to the game, like the number of deaths by falling, the number of defeated monsters, or the amount of health left at the end of the stage.
2. **Analyzing:** The purpose of this step is to convert the raw data sampled from the game into meaningful data that can be processed into rules of modification. This can be e.g. the accuracy of the player, which is determined by the number of hits divided by the number of shots. Therefore a suitable algorithm has to be found that is capable of transforming incoherent data into a statement about the subjective difficulty perception of the player. This is often done by using a challenge function, which results in a number between 0 and 1 to represent the difficulty of this element.

Once this has been computed, the new data is processed to the third segment of the DDA system, where the adjustment takes place.

3. **Adjusting:** This step uses the processed data to apply fitting modifications to certain parts of the game. It is the only step that actively intervenes with the setup of the game and changes how it is played. For every part that the second step has evaluated as not adjusted appropriately to the needs of the player, a modification is computed and then used on the actual parameters of the game. For example, if the player often failed to overcome a certain obstacle like an enemy, the characteristics of this specific enemy can be readjusted to stimulate a better performance of the player and give him a feeling of accomplishment once he successfully defeated it.

2.2.2 Proactive vs. Reactive Behavior

A DDA can be used to react on changes in the performance of the player. This is called *reactive behavior*, as it only takes information about the actual moment into consideration. A different approach would be to work with the changes in the performance over time and generate a prediction for a future performance, which is then used to adjust the game before the player reaches a critical state. This is called *proactive behavior*.
Both strategies have their advantages and disadvantages. Proactive behavior helps to make the transitions between two different parameter setups more fluent as it extrapolates the performance the player will most likely show and prepares the game accordingly. If the prediction and the actual outcome vary widely, though, there is a high probability that the player will be disrupted by the changes and lose the feeling of the flow.
Reactive behavior, on the other hand, works only with already existing elements of the game, making it more safe to use, as the outcome of the adjustment can be easily predicted. But as it only modifies parameters after a critical threshold has been exceeded, it always has a certain delay that could already be enough to make the player lose his motivation (Andrade et al., 2006).
The best solution for a specific DDA depends on several factors, such as the area of operation, the type of game, or the computational cost.

2.3 Fuzzy Control Engineering

For the analyzing process of the DDA system, a convenient algorithm has to be found, in order to transform the raw data sampled by the monitoring into comprehensible data. It is crucial that this algorithm creates stable and trustworthy results, or else the adjustments will not request the desired modifications, which can lead to a huge number of issues like disrupting the immersion of the player or even making the game unplayable.
In this thesis, an approach has been chosen that is flexible and reliable: *fuzzy controllers*. To understand the concept of fuzzy controllers and their area of application, a brief introduction follows (Kruse et al., 2002).

2.3.1 Fuzzy Sets

In classic mathematics, to every equation a truth value can be assigned, therefore it can be *true* or *false*. This makes it very easy to evaluate statements. Machines that use binary code to compute complex solutions to mathematical problems are built that way as they mostly need accurate and precise information. Humans on the other hand do not think that way. A human often uses vague and contextual expressions to exchange information, using inexact words like *almost*, *around*, *small* or *very much*. All these expressions do not refer to an actual value, they can contain a multitude of interpretations and are often used in completely different contexts. The understanding of vague and

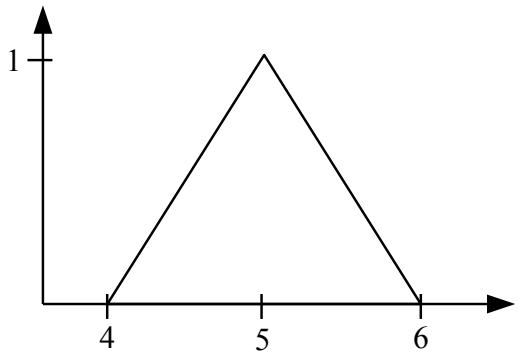

Figure 2.2: Fuzzy representation of the linguistic term *around 5*.

imperfect information depends on the human giving the information as well as on the receiver of the message, as they might, for example, have a different concept of what it means to meet at around 9 p.m. *Fuzzy logic* addresses this problem by introducing linguistic terms into mathematics. In order to allow the correct interpretation and handling of expressions like *slower* or *almost 5*, classic mathematics and their binary evaluation of statements are expanded. In fuzzy mathematics, an expression cannot only be one or the other, but also everything in between. This is realized by defining a membership function that provides information about the degree of affiliation of an input value to the linguistic term (Kruse et al., 2002, p. 1 ff).
We define a fuzzy set μ of the underlying set $X \in R$ as a transformation $\mu : X \to [0, 1]$, so that for every $x \in X$ there is a membership value $\mu(x)$. As every element of X is a real value, μ then is a real function with values in the unit interval $[0, 1]$. For illustration purposes, these functions can be represented as graphs (fig. 2.2). As the task for fuzzy sets used in this thesis is to work as an algorithm to find the correct modification rule for a game parameter, these fuzzy sets are convex, which means they are monotonically increasing up to one point and after that monotonically decreasing. Figure 2.3 shows three different graphs of fuzzy sets as an example of how different forms can be used to describe specific contexts (Kruse et al., 2002, p. 6 ff).
The most basic form, shaped like a triangle, is defined as

$$\Lambda_{a,b,c} : R \to [0, 1], x \mapsto \begin{cases} \frac{x-a}{b-a} & \text{if} \quad a \leq x \leq b \\ \frac{c-x}{c-b} & \text{if} \quad b \leq x \leq c \\ 0 & \text{else.} \end{cases}$$

It can be used to outline linguistic phrases like *around 7* or *more or less 2*. The second form is displayed as a trapeze and enhances the triangle form to describe intervals of arbitrary length. It is defined as

$$\Pi_{a',b',c',d'} : R \to [0, 1], x \mapsto \begin{cases} \frac{x-a'}{b'-a'} & \text{if} \quad a' \leq x \leq b' \\ 1 & \text{if} \quad b' \leq x \leq c' \\ \frac{d'-x}{d'-c'} & \text{if} \quad c' \leq x \leq d' \\ 0 & \text{else} \end{cases}$$

and allows linguistic expressions like *mostly between 1 and 4* or *fast, but not very fast.* These two forms are often enough to characterize the needed behavior, although there are also bell-shaped curves possible, which allow a more detailed modeling of the expression. They are defined as

$$\Omega_{m,s} : R \to [0, 1], x \mapsto \exp\left(\frac{-(x-m)^2}{s^2}\right).$$

For this thesis, the concept of fuzzy sets is used to control the behavior of the DDA system. To accomplish this, a suitable approach for the evaluation and manipulation of incoming and outcoming parameter values is needed. To achieve this, *control engineering* is introduced and extended to allow *fuzzy controllers.*

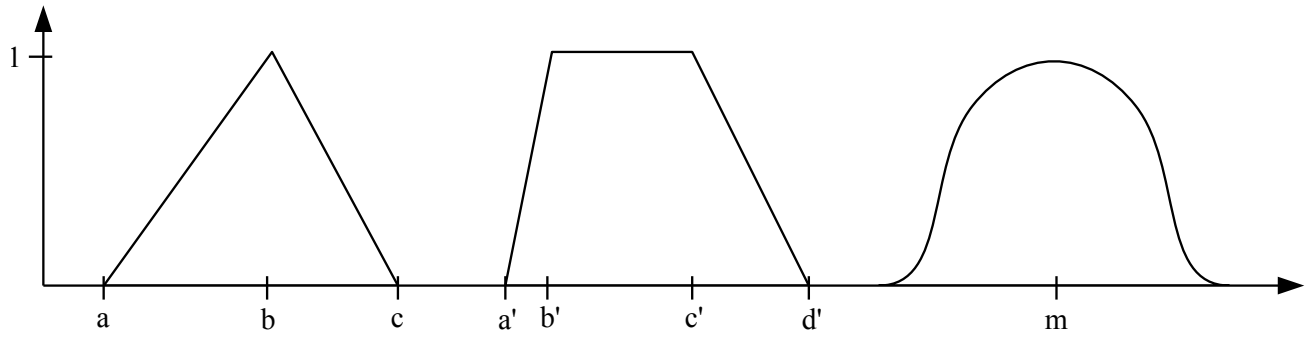

Figure 2.3: Three types of fuzzy sets: triangular ($\Lambda_{a,b,c}$), trapezoid ($\Pi_{a',b',c',d'}$), and bell-shaped ($\Omega_{m,s}$)

2.3.2 Control Engineering

Control engineering is concerned with the manipulation of systems that should show a desired behavior over time. The area of application includes mainly the regulation of machines, but it can be applied in every environment comprised of a dynamic system that can be altered in order to get a different output. The standard control engineering setup is composed of two main components: the system and the controller (Kruse et al., 2002, p. 59 ff). The system stands for the natural behavior of the environment and is object to external modifications. External means, that the system itself is not changed in any way, only the data put into it and as a result its behavior. In this thesis, several requirements are defined for the system, which are as follows (Kruse et al., 2002, p. 73 ff).

1. **Time-Invariant:** the parameters of the system are constant over time.
2. **Continuous:** the course of the signals is given for every moment in time.
3. **Concentrated:** the system consists of a finite number of input values.

The controller is a collection of rules and functions that can be applied to the input values of the system to regulate its behavior. The mechanisms behind this are called *control variables*. The output values of the controller are compiled by computing the difference between the *reference value*, which represents the desired state of the system, and the output value of the previous step. This consequential error value is called *control deviation* and then used to determine the scale of the necessary intervention. After altering the input values, they are transmitted to the system, where new output values are calculated. This process is repeated in every step. Like the system, the controller needs to satisfy certain rules that are as follows (Kruse et al., 2002, p. 62 ff).

1. **Stable:** the transient response after altering the input of the system is eliminated over time, i.e. reaching the reference value after correcting the control deviation is possible.
2. **Precise:** the control deviation after the transient response is as small as possible.
3. **Fast:** after changing the command variable the control deviation needs to be eliminated quickly.

The concept of control engineering can be expanded by integrating fuzzy sets into the process. This is desirable when the controller needs to work with vague data e.g. coming from a human user. The difference between classic control engineering and fuzzy control engineering is the strategy to model the controller. To design a classic controller, first a model of the system is being created and in a second step, the controller is being built on top of this information. With fuzzy controllers, only the controller is being modeled intuitively, having merely an approximate idea of the properties of the system (Kruse et al., 2002, p. 239 ff).

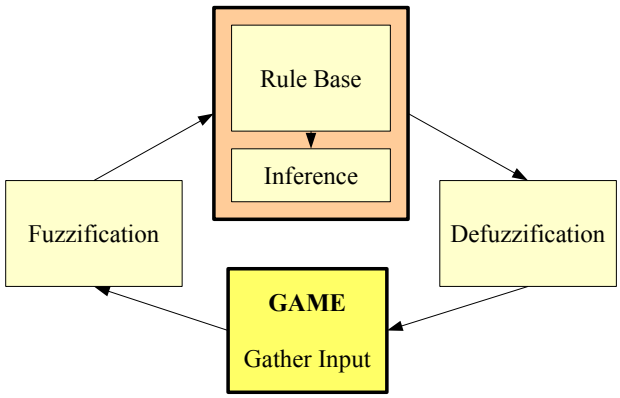

Figure 2.4: A block diagram showing the data flow between the components of a fuzzy controller.

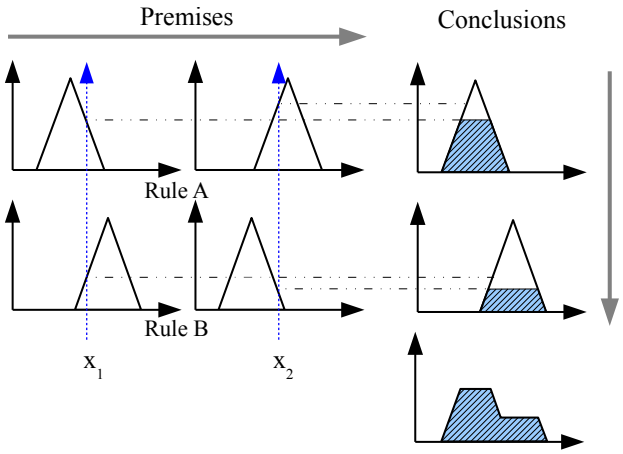

Figure 2.5: The process of evalutating the input fuzzy sets with a Mamdani-controller.

In order to integrate the functionality to work with fuzzy sets, the process of evaluating the input and output data has to be adapted. As the block diagram in figure 2.4 shows, four components have to be introduced, that control the handling of the input and output data.

1. *Fuzzyfication*: performs a conversion of the crisp input values into a number of fuzzy membership values, by determining the degree of affiliation to every fuzzy set (as seen in Chapter 2.3.1). This enables the representation of parameter values as a fuzzy truth value to a linguistic expression.
2. *Rule Base*: provides a set of rules for the evaluation of membership values. The rules are based on intuition and expert knowledge and serve as a database for the inference engine.
3. *Inference Engine*: consists of a number of if-then-inquiries. The most common form is the min-max-inference, which contains one or more if-statements that are evaluated using the Mamdani-implication, which is defined as $\mu_{A\to B}(x,y) = \min\{\mu_A(x), \mu_B(x)\}$, and one conclusion. In the case of the min-max-inference, it evaluates the statements by choosing the minimum membership value of every involved fuzzy set and the maximum of the agglomerated results, as seen in figure 2.5.
4. *Defuzzification*: uses the results of the inference engine to transform the fuzzy membership values back into one single output value. This value is processed to perform the manipulation of the game system. There is a multitude of algorithms to achieve this conversion, like choosing the maximum value of all input fuzzy sets or computing the center of gravity of the agglomerated output fuzzy sets.

Chapter 3

Dream Runner & Feature-based Dynamic Difficulty Adjustment

3.1 Characteristics

In Chapter 1 four goals have been stated as a way to encourage an efficient and stable approach to construct the feature-based DDA and to review its success after its completion. The goals are as follows:

1. *The data gathered by the DDA system concerning the difficulty concurs by 80% with the surveys filled in by the players.*
2. *The game project designed for the thesis manages to keep the player motivated for at least 70% of the time.*
3. *The surveillance and modification of the game through the DDA system takes place in a non-disruptive manner by 80% and does not interfere with the gaming experience.*
4. *The modification of graphics and sound effects has a measurable impact on the difficulty and 40% of the players name it a game-defining element.*

In order to incorporate these goals into the final product, several requirements for the DDA have been defined. First of all, to be able to work with the different fields of the game world, meaning gameplay, graphics, and sound, they have to be designed to allow describing them using parametric attributes, like *height* or *volume*. This leads directly to the next requirement: a modular approach. An important characteristic of the DDA is the ability to express every aspect important for the shaping of the difficulty as a minimal unit, called *feature*. This gives two big benefits: every element can be treated the same way, meaning that their layout is identical, and, having a parametric representation of the attributes, they also can be compared to one another regarding their impact on the difficulty. This allows the DDA to work non-restrictive. No matter what element the feature represents, whether it is a gameplay, graphics, or sound component, the procedure is always the same.
One requirement already mentioned in Chapter 1 is the non-disruptive nature of the DDA. This can be achieved by applying changes in a subtle way, avoiding sudden changes like adjusting speed parameters for enemies with high values. This also means that the parameters of the player character himself are never to be altered, as these changes would be too obvious. Alongside with this convention comes the policy to use the DDA as rarely as possible and as often as necessary to support a balanced experience for the player. To employ this, the DDA used in this thesis is restricted to reactive behavior as it has been described in Chapter 2.2.2.
Additionally there are some guidelines concerning the impression of the DDA toward the player. First of all, the DDA needs to be able to identify and react on situations where the player finds himself in a

critical state. This means, that he may be about to lose interest in the game by either being frustrated or bored. Therefore, threshold values must exist that determine the state the player is in and then find the correct solution to the problem. To be able to judge a situation, a clear distinction between different difficulty setups must be guaranteed.
In addition to that, the intention of the DDA is to challenge the player and stimulate him to become better at the game, not to give him a challenge he cannot overcome without intense training. The other extreme of providing no challenge at all should also be avoided, as the lack of difficulty results in the player not feeling any kind of reward when he accomplishes to overcome an obstacle. Without the positive feedback his motivation will steadily decrease until he stops playing the game. The most desirable challenge-reward ratio is achieved when the difficulty lies slightly above the skills of the player. This way, the player has to use his skills to overcome each new challenge by getting a little bit better. It is not impossible to fail from time to time, but the player will likely have the feeling that he can still succeed if he tries again with the knowledge he already accumulated in the previous attempts. This results in a cycle of demanded concentration and given rewards, creating a permanent and powerful motivation for the player.

Apart from the requirements, there are also areas in the development of the DDA that have been neglected in order to shorten the development time or to simplify the handling.
As the DDA in this thesis performs its actions only when it is called by the game (as opposed to adjusting the game in every step), there is no need to spend much time on the optimization of the performance. The only action that has to be performed in every step is the monitoring, as it has to react on a multitude of input commands executed either by the player or the program itself. This is generally possible in a very performance-friendly way, as the monitoring consists only of if-then queries that change specific monitoring parameters. The actual adjustment takes place between levels, while the player is not playing the game, but waiting for the next level to start, making it redundant to retain real-time computation.
It is often advisable to differentiate between players and derive player clusters from the data available. These clusters are called player profiles. They serve the purpose of gathering interesting information about the player to apply a more individual setup that is tailored around his needs (as seen in Spronck et al. (2004)). The downside to this approach is that it is likely to become very complex the more data the player profile is comprised of. Additional knowledge about player behavior has to be collected and processed into data comprehensible for the game to be able to categorize each player into the right profile. In this thesis there are no player profiles, there is only one trivial class called *player*. That means, that there are also no different setups for different player types. The DDA has to be capable of creating a unique and balanced experience for every kind of player despite their history with video games and without an in-depth analysis of the style of play. It is designed this way to keep the complexity of the project as small as possible, but also because it would turn out to be another error source.

3.2 Methodology

3.2.1 Setup of Dream Runner

Dream Runner is the name of the accompanying game project (as seen in fig. 3.1). It serves as a testing environment for the DDA, with its main purpose being the utilization for experiments using real test persons. These experiments are built to be conducted over the internet. Therefore, the game client has to fulfill certain requirements: it has to be *small*, so that every participant can download the game regardless of his internet speed, *user-friendly*, due to the fact that the experiments are not supervised by a human, and *performant*, as the game needs to run flawlessly even on older machines. Taking all these requirements into account, the choice for the game project came down to a two-dimensional platforming game, as these are highly accessible and resource-efficient.

Figure 3.1: An illustration of the main components of Dream Runner.

Controls

In the design of the game, an important factor proves to be the balance between allowing the player to instantaneously grasp the game and its handling and incorporating enough complexity to be able to create a diverse and adjustable experience. The input values garnered from the player have to give enough information about his state and how he perceives the game to find the correct adjustment values. Therefore, the control scheme consists only of a very small number of input keys, containing both the mouse and the keyboard.
There are two keys on the keyboard representing the actions *jump* and *duck*. They give the player the possibility to maneuver the character through the levels. In order to increase the complexity, a second function to each of these actions is added. These are called *double jump* and *jump roll* and can be performed in mid-air to give the ability to control how the character moves while not on the ground.
The mouse is used for another gameplay element: shooting. To further increase the amount of interactivity with the game world, the player is able to move a cursor over the screen and shoot in the direction it aims from the player's position by clicking with the left mouse button. This mechanism is used to destroy hostile obstacles, which are from now on called *enemies*.
These are, apart from purely meta-functional input keys like pausing, the only ways the player can manipulate the game and yet a lot of information can be derived from them just by looking at the context in which they are used. The aiming is the only high-level interaction as it asks the player not only to react on different situations by pressing a button but to comprehend the 2D space as a whole and direct the motion of the cursor so that the aiming vector from the player character to the cursor points in the correct direction. But as most players are used to navigate a mouse in other programs, this requirement is easily fulfilled and adds to the customizability of the game.

Game Elements

A brief description of every major game element follows, giving an outline of the experimental environment used in this thesis.
Player: the player is the instance responsible for all the interaction with the game world. It consists of the player character and the mouse cursor. The player character runs from left to right as long as there is no obstacle in the way. It can take damage from the surrounding enemies, fall into gaps and get stuck on abrupt height changes in the terrain. It is the player's task to avoid this and reach the end of the level. It can also collect supporting objects, called *power-ups*. The mouse cursor controls the arm of the player character for shooting. Shots can be fired in 360° and they hit their target instantaneously.

Enemies: the enemies in the game are separated into three distinct categories.
Flying: these enemies fly horizontally over the screen and shoot projectiles in a specific time interval. They do not damage the player directly, but the fired projectiles do. Flying enemies can be modified in their size, the speed in which they fly over the screen, the shooting rate, the number of projectiles fired, and their health. They can only be destroyed by shooting them.
Projectiles: projectiles are fired by the flying enemies and deal damage to the player on impact. They are destroyed either by shooting them or by hitting a part of the terrain. The direction they fly in is dependent of the vector starting from the flying enemy's position to the player's position in the moment when they are shot. They never change their direction afterward. They can be modified in their size, their speed, the damage they deal, and their health.
Traps: traps are, in contrast to the flying enemies, stationary. They lie on the ground and detonate if the player comes too close. They consist of two components, the body and the sensor. The body is the vulnerable part that can be shot by the player. The sensor can be touched by the player character, which activates the detonation after a short delay. The damage dealt by the explosion depends on the distance between the player character and the trap body. It is also possible to activate a trap without taking any damage by moving away quickly after touching the sensor. After the detonation, both the body and the sensor are removed from the game. Traps can be modified in the size of the sensor, the damage they deal, and their health.
Level Eraser: the level eraser is a black barrier that reaches vertically across the whole level. It moves in the same direction as the player and destroys every part of the terrain it touches, creating the urge to keep moving forward for the player. If not set on the highest difficulty, the level eraser moves slower than the player, giving him the chance to increase their distance to one another, although the eraser is set to the same speed as the player once it leaves the screen to retain the sense of danger. The level eraser does not damage the player directly and it cannot be destroyed. It can only be modified in its speed.
Level Structure: each level is characterized by two opposing components, which are called *gaps* and *terrain*. Terrain is the ground that the player walks upon. It can have different heights, but overall it only consists of one block object which is placed sequentially on a horizontal line to form plateaus of various length. The other component, the gap, is automatically created in places where there is no terrain. Naturally it can also have different lengths. The level structure can be modified in its overall length, the minimum and maximum width and height of the plateaus, the minimum and maximum width of the gaps, and the distribution rates for enemies and power-ups.
Power-ups: power-ups are collectible objects that give the player positive effects that can help him to overcome the obstacles in the level. They are collected on touch. The player can also shoot them to cycle through their different effects. There are three power-ups:
Health: if the player collects this power-up, a part of his health is replenished. It can only be modified by the amount of restored health.
Power: this power-up increases the damage dealt by the player's shots for a short amount of time, making it easier to destroy enemies. The exact amount of time can be modified.
Time: once the player touches this power-up, time is slowed down by one third, giving him more time to react on dangerous situations. This effect neutralizes after a short period of time. As with the power power-up, the amount of time before the effect stops can be modified.
Checkpoint: the checkpoint marks the end of every level. It is, similar to the level eraser, a vertical rectangle spanning over the entire level. It places itself according to the length of the level. When the player touches it, the values monitored throughout the level are processed into the DDA and the adjustments take place, starting the level over with the new values once it is finished.

Graphics

As with the gameplay, the graphics used in Dream Runner are completely two-dimensional. There are three distinct types of graphics, which are called *sprites*, *backgrounds* and *effects*. Each type handles the illustration of different game elements and they are defined as follows.

Sprites are used for everything that has to be altered in every step and is interactive. This can be enemies, which move in one direction, the terrain which can be destroyed, or the player character that has many animation frames. Backgrounds (and foregrounds) are used for elements that do not have to be interactive and are big in size. They can consist of a multitude of elements to create more depth. In this thesis, a *parallax scrolling* effect is added, which means that each background layer moves at a speed that is dependent of its z-depth, making it seem like a landscape that is drifting by. Effects like particle systems are necessary when there is a need for a lot of small objects that do not have to be interactive, but still have to show a specific behavior. They are mostly used to give visual feedback to the player. All these different types of graphics help to shape the difficulty of the game when they are utilized in the correct situation. Sprites and particle effects, for example, facilitate the use of many elements to increase the amount of movement on screen. The parallax effect in the background enhances this effect additionally.
In order to have a strong differentiation between game elements that need the player's attention and less urgent elements, a very specific visualization technique is utilized, mostly consisting of different shades of gray or even monochromatic black and white colors. There are a few objects like projectiles and traps using color, mainly to emphasize regions of interest. In addition to that, feedback effects that occur when the player was hit by a projectile or explosion or when his health is low are typically red. Flying enemies also become more and more red the lower their health is to signalize a higher priority, as their threat can be eliminated faster. This visual feedback is crucial for the player, as he can distinguish much better between the important and the neglectable elements and increase his efficiency.
The same approach has been applied to the shapes of different objects. There are mostly geometric shapes like rectangles, but also finer structures like trees. These more complex shapes have two tasks. One is to make the game more pleasant for the eye, the other is to enhance the difficulty by adding more structures to the screen. When there are many complex shapes, the difficulty increases, when there are very few complex shapes, it decreases.
Another way to generate attention is to use motion by applying animations to the objects. If the player character would be an inanimate rectangle, it would be much easier to overlook it. But as it changes its form with each new animation frame, its position and motion become clearer and the player does not have to spend much time on finding the important objects on the screen.
To realize the needed dynamic behavior of the graphics, suitable parameters have to be utilized. These parameters include the amount of parallax background and foreground layers, the player hit feedback (global) and enemy hit feedback (local), the amount of filler objects, and the strength of visual effects.

Sound

The third subject in the creation of Dream Runner was the audio department. This includes both background music and sound effects, which are important to set the mood of the game and help perceiving the game as a whole. The main purpose of the sound effects is to give feedback to the player. Whenever something interacts with one another, a sound is played. There are three tasks for sound effects.

1. Hit detection: these sounds are played whenever something is hit. This can be the player himself, the terrain, a flying enemy, a trap, or a power-up.
2. Approaching danger: these sounds give steady feedback for a dangerous state that might be reached in the near future, like the crumbling of the terrain behind the player.
3. Actions: these sounds underline the actions of the player, giving feedback that the game recognized his input correctly, e.g. when the player jumps or shoots.

These sounds are abstract 8-bit compressed audio files due to time limitations, but they are designed to be easily distinguishable and emphasize the meaning of the action. The range of their effect can be described as local, as they are only played under certain conditions.
The background music has a global effect on the game as it is played the whole time and as long as

the player needs for the level. Therefore it repeats itself once it reaches the end. Its main purpose is to build the atmosphere and to control the amount of stress the player is feeling. There is only one music track in the whole game, but it is divided into five stages with different intensities, ranging from mysterious to enthralling. These stages are realized by adding more instruments and they are necessary to incorporate the concept of the DDA and make the music dynamic.
In order to make the sounds of the game compatible with the design of the DDA, several parameters have to be set that change between levels to help define the difficulty settings. These are the volume and the bluntness for sound effects and the volume, the intensity, the speed, and the amount of volume reduction when the player is hit for the background music.

Procedure

The process of playing through one segment has to be very flexible and controllable at the same time, which makes it a challenge to find the right balance for the player between action and reaction. To get a better understanding of the progression through a level, a short description follows.
After the actual level is created, the player character instantly starts running automatically to the right. Shortly thereafter, first obstacles in the form of gaps, height changes and enemies appear from the right and move to the left, crossing the way of the player. It is now his task to avoid these gaps, while also jumping over heightened ground and evading or shooting upcoming flying enemies, projectiles, and traps, without being caught by the level eraser, all at the same time. This massive amount of impressions and demands would be overwhelming if executed too aggressively and without surveillance.

Level Generation

Originally a complex level generation tool based on the same concept of features as used in the DDA was planned, including a layer-based approach to create interesting and diverse levels. The DDA should have had as much control over the level generation as possible. In the end, the undertaking was rejected, because it proved to be too complex for this thesis and not necessary.
Instead, the plan was shifted to a partly dynamic level generation, which takes a preexisting level template and adjusts those parts that are important for the game randomly within given thresholds. Therefore, at the beginning of each level, several control objects that have been placed by hand generate a unique landscape by removing terrain to create gaps and by heightening and flattening terrain blocks. Every control object has a small area of effect to ensure that the level created in the process is still playable. After forming the landscape, interactive objects like flying enemies, traps, and power-ups are distributed over the length of the level, which completes the level creation process.

3.2.2 Implementation of the feature-based DDA

The conception and realization of the feature-based DDA has been the main concern for this thesis, as it can lead to very different results depending on the configuration of the parameters. A trivial DDA setup would be a game that adjusts itself automatically without the need of the three steps monitoring, analyzing, and adjusting. This could be a similar game like Dream Runner, where running into a wall could automatically lead to a decreased running speed. It would be self-adjusting, does not need any knowledge about the state of the game, and the dynamic behavior comes only as a direct result of the player behavior. This is not the intention of this thesis, as it would not generate enough information about the player to create a dynamic and personalized experience.
Therefore, the concept of the DDA is centered around the idea, that the impact on the difficulty of every element can be converted into a scale between 0 and 1. This scale can be adjusted to influence the player's success. In order to achieve this, information about two major components has to be given: the

player and the attributes of the game. The actions of the player are the basis for each decision made by the DDA. There are no modifications made without the inclusion of monitored actions. The attributes of the game on the other hand are the tool to determine and to alter the current difficulty level. They need to be comprehensible and accurate to allow utilizing the data that the game was built upon. As a result, Dream Runner was created to provide as much needed information as possible about both the player and the state of the game, making the application of a DDA system realizable.

Features

To be able to transform every attribute of the game that is involved in the difficulty settings into a representative scale, the concept of features is introduced. A *feature* is a comparative value in the interval of $[0, 1]$ and it serves as the minimal unit of the difficulty regulation. Its value describes the level of difficulty the particular element of the game attains. Every element that can be targeted for adjustments is called feature. All features are independent from one another and they do not affect each other in any way, only the data monitored by the DDA has an impact on them.
Features can be found in every field of Dream Runner. One example for a gameplay feature is the *terrain height*, which determines the magnitude of height changes from the standard height. By this, the layout of the level is dramatically changed. A graphics feature is the thickness of the fog, which is created through the use of a number of *foreground layers*. When the feature value exceeds a threshold, an additional layer is drawn, further occluding the screen. An example for a sound feature is the *intensity of the music*. This is determined by the instrumentation of the background music, which changes when the feature value is altered.
All these different components and their respective effects combine to define the overall difficulty of the game.

Difficulty

The control over the difficulty of Dream Runner is the main goal of this thesis. Therefore it is important to find an appropriate measure for it, that has the ability to represent the whole range of difficulty settings for as many players as possible (see fig. 3.2). Trivial difficulty settings would be a level that the player cannot lose, in this case a straight level with no gaps or obstacles, or win, which would be a level with a gap that cannot be crossed or something similarly insurmountable. The right balance naturally lies somewhere in between and this is where the difficulty for Dream Runner should be located. It is obvious, that the objective difficulty determined by the game does not necessarily have to match with the subjective perception of the player, making it unfavorable to use a global difficulty scale. That is why the applied technique works on a local level, taking only information of the current state of the player into account, and adjusting from this point on.

Figure 3.2: Three different difficulty settings in Dream Runner: low feature values, medium feature values, and high feature values. (f.l.t.r.)

As the state of flow described in Chapter 2.1 is best reached by creating a challenge for the player, this is defined as the neutral zone, meaning that whenever the DDA detects that the player is challenged but also able to finish the level successfully in a given time window, no changes are made. In conclusion, the best case of the current difficulty is that there is always a challenge. As long as this challenge is present, there is no need to intervene.
Having a permanent challenge increases the possibility of making mistakes and failing to reach the end of the level. As a result, the challenges given in the game are designed to stimulate the motivation to keep playing even after failing. Whenever the player loses, the level is instantaneously reloaded with the same set of feature values, but a different layout due to the partly random level generation. This way, the player never has the same experience, even if he fails. The instant restarting of the level also minimizes waiting times, which could frustrate players with little patience if too long.
The procedure of starting the one level included in the game over and over again, either by failing and restarting or by succeeding and restarting with a different set of feature values, implies that the game has no ending. There is no guideline or story that distracts from the challenge given to the player. This lack of distraction facilitates reaching the state of flow, as it satisfies all eight requirements as listed in Chapter 2.1.
In order to eliminate the danger of getting stuck in one level without ever reaching the end, a *damping function* has been constructed in addition to the normal DDA. This function lowers every feature by a preset amount multiplied by the progress the player made in the level whenever he fails to complete it. The adjustment is very small and only meant as a last resort to support the player and prevent the experience from becoming too frustrating to keep playing.

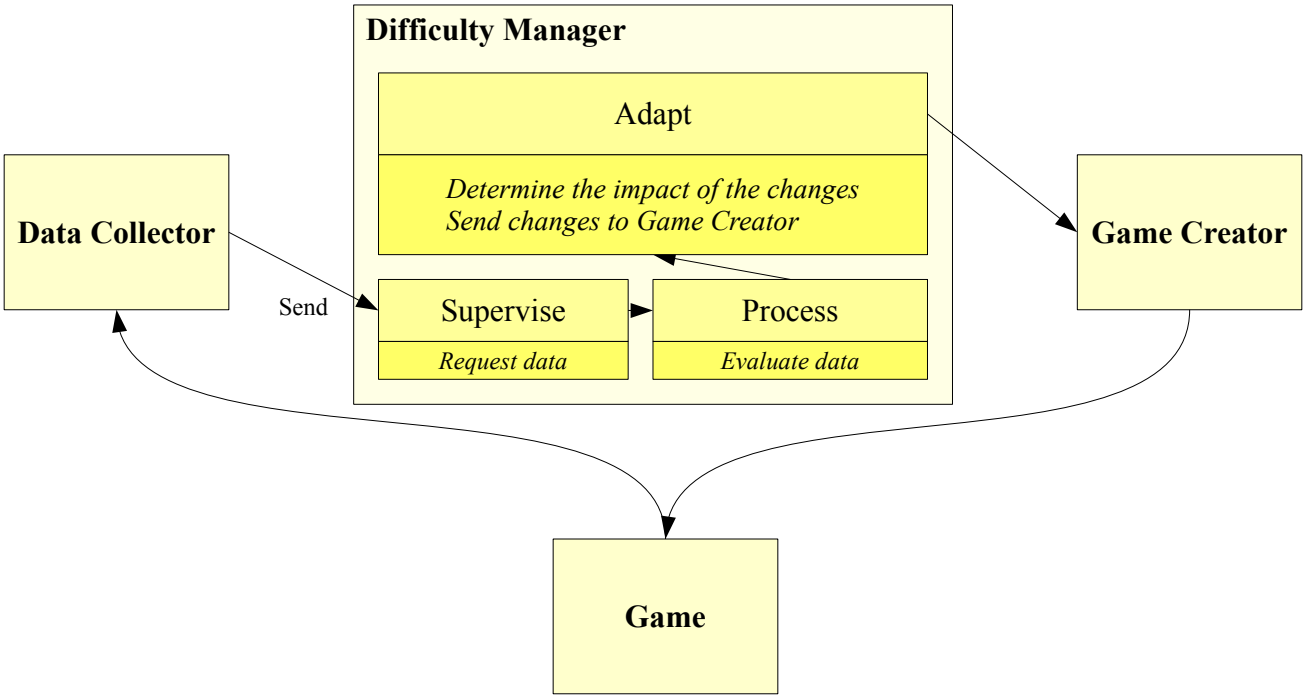

Figure 3.3: The structure of the game project, with the DDA being an interconnected module that contains all the necessary steps for the adjustment.

Structure of the DDA

After designing the conceptual blueprint for the DDA, a practical implementation is needed. The overall structure of the program is the key to build a stable and comprehensible algorithm. To illustrate this development, a brief description of the different parts of the DDA follows.
The DDA is constructed as a group of objects, which are active when their functionality is needed. The overall structure of the whole system can be seen in figure 3.3, which describes the data flow.

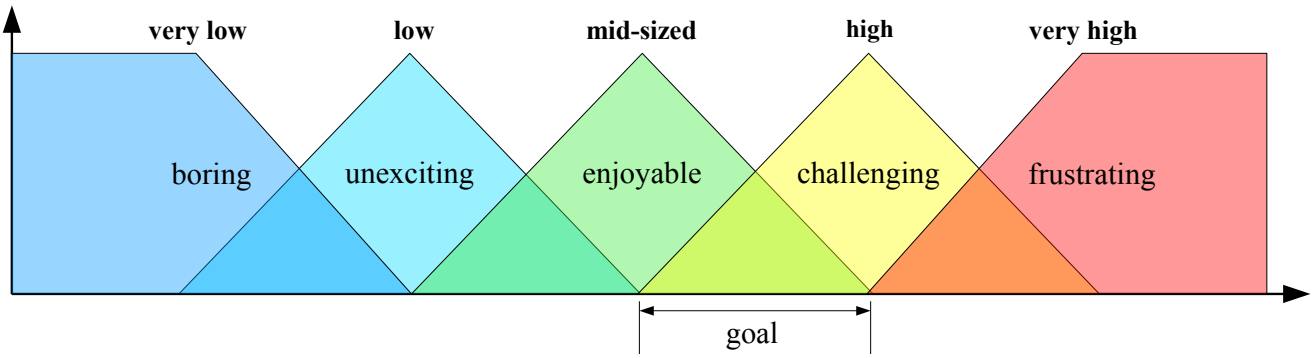

Figure 3.4: The five different difficulty groups and how they are interpreted. The *goal* shows the desired state.

The first component is the *data collector*. It stores all the information necessary for the adjustment process. This data is called *monitoring values*. After completing a level, the *difficulty manager* requests the data from the data collector and copies its values to a list for an easier management. Then this list is processed and the data evaluated as follows.
First, every monitoring value is normed in the interval of $[0, 1]$. This ensures that the values are comparable and mathematical functions like the mean value can be computed.
The second step is the *fuzzification*. As the values are already normed, one single transformation mask can be used to turn the crisp monitoring values into fuzzy sets. It consists of five triangular functions defining the membership for every value. They range from *very low* to *very high* (see fig. 3.4). These linguistic terms refer to thresholds that have been chosen on the base of intuition. Figure 3.5 illustrates that every value can only be a member of one or two linguistic terms and the added memberships are always one.
As an intermediate step, all fuzzy sets are converted to have the same direction. The goal is to guarantee that for every fuzzy set applies: the higher the membership values on the left side, the easier this element is for the player. Respectively, higher values on the right side refer to a harder difficulty. This is achieved by simply mirroring the membership entries where this does not apply.
Next the adjustment rules are applied. Therefore, for every feature either one or two fuzzy sets are processed through a number of if-then-assignments. These rules are the same for every feature, as they only evaluate their quantitative values. They cover every combination of fuzzy set memberships. In the case that only one monitoring value influences the particular feature, the application of the rules equals a simple transformation from one fuzzy space into another using one rule for every membership function. When two monitoring values are being processed, a rule base of 25 rules is used, utilizing

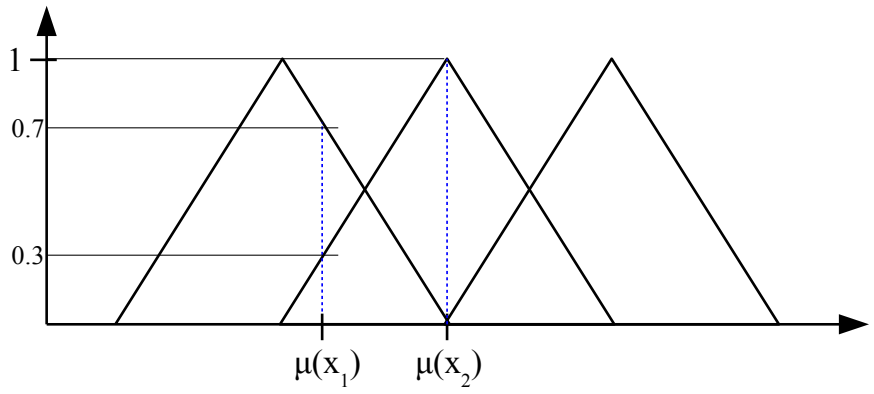

Figure 3.5: Every input value can belong to no more than two membership functions and the sum of its membership values is always 1.

every combination possible in the form of

$$\mu\left(a_{\text{very low}}\right)\times\mu\left(b_{\text{very low}}\right),\mu\left(a_{\text{very low}}\right)\times\mu\left(b_{\text{low}}\right),\ldots,\mu\left(a_{\text{very high}}\right)\times\mu\left(b_{\text{high}}\right),\mu\left(a_{\text{very high}}\right)\times\mu\left(b_{\text{very high}}\right).$$

For the results of each combination, see table 3.1.
The output fuzzy set is generated by taking the minimum of the input fuzzy sets and writing this value into the fitting triangle of the output. An output fuzzy set consists of seven triangular functions, labeled with *much lower*, *lower*, *unchanged*, *slightly bigger*, *bigger*, *much bigger*, and *very much bigger*. The triangle for *unchanged* not being in the center of the fuzzy set is a result of the policy that there should always be a challenge for the player.
The following step is the *defuzzification*, which reverts the effect of the fuzzification by converting the fuzzy output set into one single value. This value describes the amount of change for the associated feature. In order to get this value, the weighted average of the membership values for each output triangle is computed. This is defined as

$$\text{output} = \frac{\sum_{i=0}^{6}\mu(x_i)\cdot c_i}{\sum_{i=0}^{6}\mu(x_i)}.$$

The last step is to change the feature values for the next iteration of the level by adding the change values to the current feature value. After this, the process of adjusting the game is complete and the level is loaded with the new set of features. Additionally, all the monitoring values are set back to zero to get a result of the current level only.

	Low	**low**	**Mid**	**big**	**Big**
Low	$\ggg\!>$	$\ggg$	$\gg$	$\gg$	$>$
low	$\ggg$	$\gg$	$\gg$	$\gg$	$=$
Mid	$\gg$	$\gg$	$\gg$	$>$	$=$
big	$\gg$	$\gg$	$>$	$=$	$<$
Big	$>$	$=$	$=$	$<$	$<\!\lll$

Table 3.1: Results in the rule base for every possible combination of two input fuzzy sets (ranging from very low to very high). The more symbols, the higher is the change for this feature ($<$ means decrease, $>$ means increase).

3.3 Discussion

3.3.1 Scope of the work

In order to achieve the goals set for this thesis, a strict plan for the scope of the examined topic has to be created, giving a basic idea what should be expected in the end.
Having to implement a whole game with features in the three departments gameplay, graphics, and sound, it was obvious from the beginning that only a very compact game design comes into question. Therefore, a two-dimensional jumping game is the ideal choice, as it is easy to implement, allows a wide range of challenges through parameter changes, and is very well-known among gamers. Nonetheless, the concepts and ideas generated in this thesis should not only stand for themselves, but allow an easy transformation, reorganization, and adaption for other dynamic game projects in different genres. This means, that the Dream Runner project itself could be small in its scope, while also retaining a huge range due to its expandability and versatility.

3.3.2 Why Dream Runner?

Designing and building a complete game project as a testing environment for the DDA means a lot of time and thought has to be put into the process of creating graphical assets, sound effects, background music, and gameplay elements. Hence, every decision for the game has to take into account the repercussions for the development of the DDA and the time invested.
As the DDA cannot produce results on its own without input data, some kind of input value generator is necessary. It would be simple to employ artificial data, using thresholds to simulate different kinds of players, but then the means to verify the results would be absent, as there would not exist any information from the surveys to confirm the evaluation data of the DDA. Additionally, the interpretation of subjective perceptions would not be possible, which would also mean, that no statement about the impact of graphics and sound on the difficulty could be made. For this reason, the existence of Dream Runner is necessary and allows an extensive analysis of the topic.
The decision for a 2D platformer is based on its simplicity in the implementation and handling for the player. Although the interaction with the game world is composed of only two actions (*jump* and *duck*), each with a second function in mid-air (*double jump* and *jump roll*), and a more complex third mechanic (*shoot*), the amount of data derived from a play session is enough to create a sophisticated and adaptable system for dynamic adjustments. From a player perspective, having a minimal number of input keys makes it easier to immerse themselves into the activity.
Apart from the controls, other areas of the game also show a minimalist approach. The level terrain is comprised of only one block object placed over and over again to represent the landscape, making it very easy to construct and test levels. There are only two types of enemies, the flying enemy shooting projectiles and the trap exploding at a touch. In a static game, this would soon begin to frustrate or bore most players, but as every element has a number of parameters that can be adjusted, the game seems to have much more content. And these adjustments are not only cosmetic changes, they dramatically alter the challenge for the player.

3.3.3 Why modular?

The decision for a modular structure of the DDA derives from the assumption that every part of a game can be used to shape the difficulty. If this is true, then it makes sense to build the structure so that every element can be treated the same. Having no differentiation between gameplay, graphics, and sound features allows for a number of mathematical and logical functions like computing differences and comparing the (normed) parameters of every single feature with one another. Features also become easily exchangeable, which makes the development much more dynamic.
A second reason for a modular approach is the possibility to abstract the game logic and project it onto other forms of games. A DDA can be build for every kind of game that provides a challenge to the player. Dream Runner is only one small example of the capabilities of a feature-based DDA and the whole concept could be expanded to other genres like racing games or fighting games. Due to the modular approach, adding other game mechanics into the library of compatible genres would not need a whole reconstruction of the system, but merely a new definition of features based on the parameters available in this game. This makes the DDA independent of the type of game, as long as its elements have a parametric representation.

3.3.4 Why graphics and sound?

One focus of this thesis, apart from the modular approach, is the examination of the impact of graphics and sound elements on the difficulty. The research done regarding other DDA systems shows, that these two departments of a game have been neglected in studies before, concentrating entirely on gameplay mechanics. The same holds true for most games published on the international market deploying

dynamic adjustments. Though there are examples like Left 4 Dead, which go beyond altering parameters by also adjusting parts of the level and sound cues to attract the attention of the player, the impact of graphics and sound on the difficulty is largely unexplored.
The basic idea in this thesis is to look at the game in its entirety and figure out, if the graphics and sound are relevant for players concerning the perception of the difficulty. As the balance of a game is highly influential for the amount of joy felt by the player, the impact of graphics and sound and what they mean for the fun experienced in a game session is an important part of this work.
In order to construct an environment that allows the investigation of the interplay between game elements that have little in common at first appearance, a parametric representation of attributes (like the *size*, *brightness*, or *volume* of a game element) is advantageous. This way, every parameter can be changed during run-time simply by altering its value. Combined with the modular approach, it is possible to treat every game element the same and have a very dynamic testing environment.

3.3.5 Why fuzzy controllers?

In the conception and execution of the DDA, a fuzzy controller gives a wide range of possibilities to adapt and customize the behavior of the system. Based on intuitive design decisions, the results gathered mostly depend on a logical and complete setup, with only a few basic rules to incorporate. As long as the fuzzy set interval used in the fuzzification process is complete, so that for every value x at least one $\mu(x) > 0$ exists, a stable conversion from crisp input values to fuzzy sets can be guaranteed. Another requirement is, that for every fuzzy set one rule or a combination of rules is defined to create the output fuzzy set. This asks for a complete rule base, although not every combination of premises must necessarily be available, if they, for example, contradict each other. These two rules are both satisfied and allow for a stable and secure DDA structure.
Fuzzy controllers can provide benefits for the developer creating the DDA as well as the player experiencing the effects of the adjustments. From a designer perspective, the intuitive approach facilitates the transformation from the more abstract concept to the actual implementation. This is accomplished by removing the barrier between the natural-linguistic information processing of the human mind and the analytic and precise calculation of the computer. The layout of fuzzy controllers can be understood as a convergence of these largely different ways of understanding, resulting in a clearer structure that leaves more time for the actual design and calibration of the various modules. This way, a developer can easily incorporate his own knowledge into the game, find thresholds for difficulty parameters, and adjust the rule base to his needs. For the player the advantage lies in the possibility to immediately analyze his behavior at a low computation cost to minimize waiting times. The player's actions can be interpreted during run-time and give a precise impression of his achievements.

3.3.6 Conclusion

All the advantages stated in this section revolve around one basic design keystone, which is the *customizability*. This design choice can be found in several different layers of the conception of the DDA, which are as follows:

1. *Feature*: having the same structure for all the features in the game, the replacement of unnecessary features or the addition of further features becomes very easy to conduct.
2. *Game element*: as there is no differentiation between gameplay, graphics or sound components, every game element becomes exchangeable.
3. *DDA*: the features in Dream Runner are completely realized with parametric values, so that the only requirement for the DDA algorithm is to readjust these values. The input and its manipulation is open for changes and could be done by a range of other algorithms, like neural networks or heuristics.

The DDA constructed in this thesis is meant to give a compact outlook on the possibilities for graphics and sound regarding their impact on the difficulty. Adding a customizable and stable work environment enables the further development and enhancement of the system and the playing experience and prevents a one-sided observation of the subject.

Chapter 4

Experiments

4.1 Prerequisites

Conducting experiments as a means to gather results for the evaluation has been an important part of the concept of the DDA from the very beginning. There are several goals based on experiments that directly influence the design process. First of all, one of the most important intentions is to find interesting features for the adjustment that either have a strong impact on the difficulty or shape the game experience in other ways. By evaluating the data collected through the experiments and looking for critical values, these features can be detected and refined.
Gathering both information about the player behavior with the monitoring tool and his subjective perception by the use of in-game surveys helps to verify the data acquired during play sessions, which is why this method is applied in every experiment. Another design decision regarding the experiments is to conduct them even in early stages of the development. The game should not only be an instrument to collect data, it should also grow with the experiments, using every available feedback to improve its elements successively. That is why the experimental phase is spanned over a long period of time, although the DDA has not been implemented up to this point.
Before conducting the experiments, an efficient and clear setup is required to gather sufficient and correct data for the evaluation process. Every experiment is built upon the results of the preceding tests, meaning that the game has been completed to an extent that makes experiments possible. Then, a special version of the game is sent to two test groups, one consisting of people accustomed to playing games in their free time (from now on called *core group*), the other composed of a more mixed audience of casual gamers and people from the games industry (from now on called *mixed group*).
The game contains a tutorial to give every test person the opportunity to learn the controls and practice the different situations in the game before starting the actual test. After that, the main level is loaded, which leads the test person through the course of the experiment with instructions. After each level, each part of the test, and after the experiment, a survey interrupts the game and asks the test person questions about the difficulty.
When the game is over, a file is created with the name of the player, the time he played, the monitored data, the setup of the features, and the answers from the survey. This file is required for the evaluation of the experiments. After each experiment, all the files sent in are analyzed and their values stored in one big table.

4.2 Execution & Evaluation

This section gives an explanation for every experiment conducted in this thesis, starting with a preparation experiment for calibration purposes. First, the goals and the setup of the particular experiment are

expounded. After that, the results are presented, including a brief discussion of the impact of the experiment on the thesis. Please note that only the last experiment makes use of the dynamic adjustment of features.

4.2.1 Preparation Experiment

The goal of the preparation experiment is to verify, if the data sampled by the monitoring tool is congruent to the apprehended difficulty of the player. This is done by playing three consecutive levels with three predefined feature value sets and asking the player to fill in surveys with questions about the perceived difficulty afterward. The first level contains a very easy difficulty setting, the second level employs medium settings and the third one tests difficult settings.
Additionally, good minimum and maximum difficulty thresholds shall be found in order to make sure that the perception of the difficulty is in accord with the difficulty model of the DDA. It is impossible to create a difficulty interval that has the same meaning for every player, but it should give participants without practice the possibility to learn and expand their skills, while also enabling good players with a lot of experience to have a decent challenge that stretches them to their limits. This experiment is conducted completely by the *mixed group*, as it consists of people with very different gaming backgrounds, allowing a detailed look on the difficulty.

	Shots	Enemy Hits	Projectile Hits	Projectile Damage	Trap Damage
Level 1	37.44	16.11	1.67	0.78	0.78
Level 2	185.00	96.00	8.11	10.22	3.33
Level 3	1164.33	635.44	77.00	127.56	19.56
	Trap Activation	Jumps	Double Jumps	Fall Deaths	Shot Deaths
Level 1	2.44	16.00	5.44	0.00	0.00
Level 2	14.11	48.00	22.56	3.00	0.00
Level 3	100.00	299.44	162.56	34.78	8.22

Table 4.1: Monitoring values for various game elements. Level 1: easy settings; Level 2: medium settings; Level 3: hard settings

Results

The results of this test show, that the data sampled by the DDA concurs with the answers given in the survey. The mean values over the results of all the participants indicate, that the monitoring values from the first level are generally low, giving the player not much of a challenge. This becomes particularly clear when looking at the values *fall death rate* and *shot death rate*, which are both zero, meaning that none of the players died in the first level. In addition, this makes it obvious that all the players had understood the mechanics of the game and could navigate through the level without problems.
The second level, set up to give the player a mild challenge, shows values that are constantly higher. Especially the *shot rate* and the *jump rate* show a big increase, while the *fall death rate* is at three deaths per player, indicating that the player had to interact more with the game and that he also failed at times, creating a challenge-reward situation that is necessary for entering the state of flow.
The third and hardest level shows *fall death rate* values increased by 1100%, which is due to most of the participants having massive problems to finish this level. The *shot death rate*, which was 0 in both previous levels is now over 8, showing the impact of increased *projectile* and *trap damage*. The values for the *shot rate*, the *jump rate*, the *projectile damage rate*, and the *trap activation rate* are also highly increased, expressing the difficulties the players had (see table 4.1).
The results of the survey reflect this development of the difficulty settings. In the first level, the

participants rated the overall difficulty with a mean value of 4.22 of 10, which is slightly below the average. It is likely that the fact, that the participants played the game for the first time, increased the evaluation to some degree, as they still needed to learn the skills necessary to succeed. 89% of the participants also stated that they enjoyed playing this level. The second level with medium settings was rated with 6.78 of 10 concerning the difficulty, indicating that there was a challenge and that it was possible to overcome it. Again, 89% thought that the level was fun to play. In the third level with the highest difficulty settings, the overall difficulty was rated with 9.56 of 10, showing that the level was perceived as too hard to complete. Only one of the participants reached the end of the level. This time, only 11% of the participants enjoyed to play the level (see table 4.2). But as the purpose of this test is to find good thresholds for the difficulty, this can be evaluated as a success. In other words, the third level was not meant to be completed. The one person that did manage to reach the goal needed over 40 attempts, which would not be tolerable once the DDA has been implemented.
In conclusion, the boundaries of the difficulty are set in a way that enables most participants to enjoy the game and have a decent challenge. This allows the DDA to work in an area suitable for adjustments that help to motivate the player to keep trying. Consequently, the foundation for the following experiments is set.

	Diff.	Enjoym.
Player 1	3.00	1.00
Player 2	6.00	1.00
Player 3	5.00	1.00
Player 4	7.00	1.00
Player 5	5.00	1.00
Player 6	2.00	1.00
Player 7	3.00	1.00
Player 8	0.00	1.00
Player 9	7.00	0.00
Average	**4.22**	**0.89**

Level 1

	Diff.	Enjoym.
Player 1	5.00	1.00
Player 2	8.00	1.00
Player 3	7.00	1.00
Player 4	6.00	1.00
Player 5	8.00	1.00
Player 6	5.00	1.00
Player 7	7.00	1.00
Player 8	7.00	0.00
Player 9	8.00	1.00
Average	**6.78**	**0.89**

Level 2

	Diff.	Enjoym.
Player 1	7.00	0.00
Player 2	10.00	0.00
Player 3	10.00	0.00
Player 4	10.00	1.00
Player 5	10.00	0.00
Player 6	9.00	0.00
Player 7	10.00	0.00
Player 8	10.00	0.00
Player 9	10.00	0.00
Average	**9.56**	**0.11**

Level 3

Table 4.2: Evaluation of the difficulty and enjoyment in the preparation experiment.

4.2.2 Experiment A: Gameplay, Graphics & Sound

The first official experiment aims to examine the impact of gameplay, graphical, and sound adjustments on the overall difficulty and the experience of the player. Therefore, the experiment is split into three parts, each analyzing one of the aforementioned fields.
Part *A-1* adjusts only gameplay elements. It has four levels, the first one having the purpose of accustoming new participants to the game, starting with only low feature values. The three following levels each increase major elements defining the gameplay of Dream Runner. Level two has long gaps, level three many enemies that shoot a lot of projectiles, and level four many traps that deal high damage.
Part *A-2* addresses the graphical elements of the game. It has three levels, each consecutively increasing the graphical settings like the number of background and foreground image layers, the number of trees and bushes, the graphical feedback when the player gets hit, and the brightness of the terrain.
Part *A-3* focuses on the sound settings. It also has three levels, which vary in the volume of the sound and background music, the bluntness of sound effects, and the instrumentation of the music. In the first level, the volume is low and the music calm. The volume is increased and the music enthralling in the second level. Finally, in the third level the volume reaches its maximum and the music sounds hectic.
After each part of the experiment, a survey is shown to the player asking him about the elements that he liked the best and the ones he liked the least in this particular part. Additionally, an estimation has

to be given by every participant at the end of the experiment about the impact of each field and which element has the biggest influence on the difficulty.

Level Type	Shots	Enemy Hits	Jumps	Projectile Damage	Fall Deaths
Normal	36.00	16.92	18.46	1.42	0.38
Many Gaps	240.58	139.00	77.71	16.25	7.21
Many Enemies	202.92	130.75	51.04	22.75	2.63
Many Traps	109.79	63.33	31.71	7.21	1.00
Graphics Low	31.38	18.04	19.08	1.25	0.25
Graphics Medium	33.54	17.38	18.00	1.13	0.13
Graphics High	31.67	16.04	17.25	1.25	0.17
Sound Low	34.96	17.13	18.21	1.29	0.25
Sound Medium	31.67	17.33	17.71	0.83	0.08
Sound High	36.38	17.46	18.83	1.13	0.17

Table 4.3: Monitoring values for experiment A: red is the highest value, orange the second highest and yellow the third highest value.

Results

The data gathered for this experiment indicates, that most of the monitored elements of the game are in fact not influenced by the adjustment of graphics and sound features. The differences between the monitored values are very small and most probably due to fluctuations because of the random structure of the levels. This is true for a great number of features, including the *shot rate*, the *jump rate*, the *hit rate*, the *item pickup rate*, the *fall death rate*, and the *shot death rate*. The accuracy on the other hand shows small variations. It has the highest values on the medium graphics settings. This could be due to the poor sight in the high graphics settings, which makes it harder to aim. The *health* of the player is lowered by 3-6% in the graphics segment. This is also related to the viewing conditions in those levels. The gameplay levels on the other hand show significant changes in the monitored values. The peak values can be found exclusively in those three levels dedicated to gameplay changes. And even in this part of the test, there is a strong order concerning the highest values. 65% of the monitoring elements show that the level with the longest gaps has the highest impact on the difficulty, closely followed by the level with many enemies. On the third place is the third gameplay level with many traps and high trap damage, although this level has the smallest impact and is often closer to the results of the other levels. There are even some values that show no difference for the third level, like the *item pickup rate*, the *item hit rate*, and the *overheat rate*. The second level that has a high number of enemies has its peak values in the categories *projectile damage*, *overheat rate*, *shot death rate*, *level skip rate*, and *player health*. The magnitude of these values is only logical, as these are all elements that are closely related to the enemies of the game. The high *level skip rate* is most likely due to the fact that the players failed to complete this level several times. The only reason why the *shot rate* is not the highest in this level is that the first level had to be repeated even more often due to the high *fall death rate*. The *fall death rate* in level two is very high because one of the players fell 25 times before completing the level (see table 4.3).
The results of the survey filled in by the players display a similar, but not identical outcome. The player evaluation of the difficulty of each field places sound adjustments on a third place, not even remotely close to the other two departments, reaching only an overall ranking of 3.1 of 10. The peak value of 7 of 10 is only chosen once, marking it as only of little importance for the difficulty. Gameplay and graphics on the other hand show a very similar outcome, both rated with 5.9 of 10. The evaluation of the gameplay includes only one 0 and one 3, with all the other values at least higher than 5, meaning that the players thought of the gameplay changes as having a noticeable impact on the diffculty. The graphical adjustments are rated with a 1 and all the other values above 5, which means that these

changes are considered to be equally effective for defining the difficulty (see table 4.4). It is interesting to see, that the evaluation of the graphical changes are rated much higher than the monitoring values might suggest. This might be due to the changes being more visible to the player. As he sees that there is a change, he connects it with the difficulty, although it does not have a big impact on it. Sound changes are much more subtle and seem to work only as an addition.
The rating of the different elements of the game shows, that 57% of the participants name the fog as a positive addition to the game, while 40% perceive it as negative. Either way, it is the most obvious graphical change. When it comes to the sound, 64% like the use of music, with 21% preferring the dramatic music over the calm music. 20% complain about the volume of it, though. Concerning the gameplay elements, the effects of the three different power-ups are rated as positive by 50% of the players, followed by the shooting mechanics with 33%. The jumping on the other hand is rated mostly negative by 69% due to the high learning curve and the fact, that some normal jumps were detected by the game as double jumps, leading to the player falling into a gap.
The game element with the highest impact on the difficulty shows similar results, as 40% of the participants voted the jumping mechanics on first place, followed by the gaps with 20% and both the automatic running of the player character and the enemies with 13%. These are almost exclusively gameplay elements, further showing that gameplay changes are the most obvious and perceivable.

	Gameplay	Graphics	Sound
Player 1	10	5	6
Player 2	6	5	2
Player 3	7	8	7
Player 4	5	7	0
Player 5	5	1	1
Player 6	7	7	2
Player 7	3	5	0
Player 8	10	10	5
Player 9	0	5	5
Average	**5.89**	**5.89**	**3.11**

Table 4.4: Survey results of the player evaluation for the impact of the different fields using a 0 to 10 scale.

4.2.3 Experiment B: Features & Interactions

In the second experiment, a more detailed look is taken at the impact of features and their interactions with one another. The goal is to find features that have a big influence on the difficulty, both as single features and in combination with other significant features. This experiment is divided into two separate parts.
The first part *B-1* concentrates completely on single features. There are two levels per examined feature, which test low and high feature values. The following features were chosen by intuition for this part:

- **Sound**: *music speed* and *sound bluntness*
- **Graphics**: *background layers*, *object layers*, and *global hit feedback*
- **Gameplay**: *terrain height*, *level eraser speed*, and *projectile speed*

The order, in which the participant plays through the feature segments is randomized to minimize the interference between different features.
In *B-2*, two features from different fields are adjusted at the same time, creating the possibility to look if the impact both have on the difficulty is different from their influence as single features. This time,

there are two levels per pairing and three feature combinations in total. The examined pairs are the music speed with the amount of background layers, the terrain height with the amount of object layers (like trees and bushes), and the global hit feedback with the projectile speed.
The survey is also split into two parts, each focusing on either *B-1* or *B-2*. After every feature segment in *B-1*, the player is informed about which feature has been the focus of the segment and asked how he perceived the changes. After that, he has to evaluate the impact of this feature on the difficulty. It is the same for *B-2*, but this time the participant is notified about the two features that were changed and asked how he judges the interaction between the features and their impact on the difficulty. After completing the whole experiment, the survey also inquires if there was an element in the game that generally frustrated or bored the player. This experiment proves to be the longest, as it includes 23 levels, each played with different feature settings.

Feature	Shots	Enemy Hits	Jumps	Projectile Damage	Fall Deaths
Music Speed[1)]	−5.15	−3.46	−2.31	−0.69	−0.08
Sound Hollow	3.54	−1.62	−0.23	−0.62	−0.38
Background Layer[2)]	−8.38	−3.08	−4.00	−0.85	−0.23
Object Layer[3)]	2.62	1.31	2.08	0.46	−0.08
Global Feedback[4)]	9.00	4.08	3.62	0.54	0.54
Terrain Height[5)]	−0.54	−1.77	2.38	0.77	−0.15
Eraser Speed	−12.15	−7.69	−2.54	−0.85	0.00
Projectile Speed[6)]	−2.62	−0.15	0.23	0.15	0.08
1) & 2)	7.23	4.69	2.08	1.38	0.15
3) & 5)	5.77	4.92	7.08	1.08	0.85
4) & 6)	3.77	2.54	−1.15	−1.38	−0.31

Table 4.5: Changes between low feature values and high feature values for each segment.

Results

The influence of the single features on the monitored data gives a hint whether this element is important for the shape of the difficulty. Therefore, the magnitude of the changes between a level with low feature values and a level with high feature values is an indicator for its relevance. For the following examination, only the peak values of every observed feature concerning the monitoring data has been included into the evaluation. A positive change means that the monitored value was higher in the level with high feature values than in the level with low feature values, a negative change means the opposite (see table 4.5, for additional values see appendix 5).
The feature regulating the music speed shows almost no significance, as its only impact is on the amount of activated traps, which decreased after the increase of the music speed. The bluntness of the sound effects has a much bigger impact, although it only comes into effect when the player is hit by many projectiles or traps. It has a negative impact on the *fall death rate* and a positive impact on the *projectile hit rate*, the *overheat rate*, the *trap hit rate*, the *trap destroyed rate*, and the *player health*, rating it as one of the most relevant features of this experiment, as it shifts the awareness of the player to possible dangers.
The amount of background layers also shows a big impact on many monitoring values, as it decreased the *jump rate*, the *jump success rate*, the *trap damage rate*, the *enemy destroyed rate*, the *item health taken rate*, the *item power taken rate*, and the *time needed*. All of these monitored data, except for the *time needed* and the *trap damage rate*, can lead to the conclusion, that the amount of background layers decreases the ability to perceive single objects and makes the player act increasingly careful, while also having a bigger difficulty to target enemies.
The object layer represented by trees and bushes that stand on the terrain shows a much smaller relevance for the difficulty, as it only affected the *item hit rate* and the *overheat rate* in a positive way. The

same is true for the global hit feedback, which increased the *shot rate* and the *enemy destroyed rate* the most, which indicates that the visual feedback of the player getting hit makes him try harder to destroy the enemies.
The terrain height has the highest negative impact on the *trap hit rate* and the *trap destroyed rate*, which are both closely related. A positive change can be noted for the *item time taken rate*, meaning that the player either wanted to slow down time to be able to plan the next jump or collected the power-up unintentionally.
The speed of the level eraser chasing the player influenced a wide range of monitoring values. It decreased the *shot rate*, the *enemy hit rate*, the *projectile hit rate*, the *overheat rate*, the *double jump rate*, the *item health taken rate*, and the *item time taken rate*. As the level eraser has a high impact on the time the player has to make decisions while navigating the different terrain, this outcome leads to the conclusion that a faster level eraser shifts the concentration of the player away from shooting and collecting power-ups and more to finding the quickest way through the level.
The projectile speed shows the biggest negative impact for the *item hit rate* and the *overheat rate*, indicating that faster projectiles decrease the motivation to shoot them and choose the item best suited for the current situation, because the player is too busy dodging the attacks.
The answers of the survey show, that the perceived effects of the features are often much smaller than the monitored data might suggest. This is not true for the music speed feature, though, as it concurs with the data, which means that it also has no relevance in the shaping of the difficulty for the participant. The same applies for the bluntness of sound effects. Only two participants rate it to be relevant, which is about 15%. As this feature was very successful to manipulate the game when looking at the sampled data, this perception is interesting, although the subtlety of these results could be due to the participant not understanding the effect of this feature, as it only occurs when running low on health.
The feature for the amount of background layers shows a small positive rating, getting a 3.5 of 10 concerning the perceived effect on the difficulty. This means that the adjustment of the feature was noticeable, but not critical. The amount of object layers has a similar result, with a slightly more positive evaluation, but only a rating of 3.2 of 10. The global hit feedback shows a small negative score, while having a rating of 4.6 of 10, which means that the effects of this feature are clearly perceptible for the player. The terrain height, while expected to have one of the biggest impacts on the difficulty, surprisingly has one of the lowest ratings in this survey, 2.9 of 10, and only a very small positive effect.
A mid-size positive evaluation was given to the level eraser speed, rating its impact on the difficulty with 3.3 of 10. This shows that the heightened tension created by the level eraser is a welcome change for most of the players. The results for the projectile speed reveal a neither positive nor negative impression on the player, although a rating of 2.4 of 10 indicates a small impact on the difficulty. For more information, see appendix 4.
Part *B-2* focuses on the combination of two features to investigate, if their interaction changes their impact on the difficulty in comparison to their influence as single features. As with part *B-1*, only the positive and negative peak values are examined to determine the features with the biggest impact (see table 4.5).
By combining the speed of the music and the amount of background layers, two features with very different single impacts act together to shape the difficulty. The results are disillusioning, as it only affected the *projectile damage taken rate* in a positive way, which makes this combination not interactive enough to have a big influence.
The opposite is true for the combination of the terrain height with the amount of object layers. While both features did not leave much of an impact when examined alone, putting them together proves to be a very interactive combination. The only monitoring value it decreased is the *player health*, on the other hand many values were increased, like the *enemy hit rate*, the *jump rate*, the *double jump rate*, the *trap damage rate*, the *trap activation rate*, the *item health taken rate*, the *fall death rate*, and the *time needed*. Many of these monitored values indicate a strong growth of the difficulty, e.g. the decreased *enemy health* or the increased *trap damage rate* and *trap activation rate*. In addition, the higher *jump rate* and *double jump rate* are most probably due to more height changes.
The last pairing is the global hit feedback and the projectile speed. As with the first combination, the results show very little interaction. A peak value of the changes is only found when looking at the

projectile damage taken rate, which decreases after changing from low to high feature values.
In a second step, a more detailed look is taken on the difference between the summed single feature values and the results of the combined pairing to get a clearer evaluation of the situation. The higher the difference, the greater is the gap between the impact of the two single features in comparison to the combination. This does not take the actual values of the data into account, only the magnitude of the changes (see table 4.7).
Surprisingly, while only showing a small relevance when looking at the peak values, the combination of the music speed and the amount of background layers proves to have the highest number of peak differences, including 15 categories like the *shot rate*, the *enemy hit rate*, the *jump rate*, the *enemy destroyed rate*, the *projectile damage rate*, and the *time needed*. The smallest differences can be found in the *double jump rate* and the *fall death rate*. As these values include some of the major mechanics of the game like shooting and jumping and they are all in a positive direction, the amount of activity and interaction of the player with the game increases greatly with this combination. Additionally, for most of the categories, the difference is negative for the summed single feature values and positive for the combined feature values, showing a complete inversion of the effect. This indicates a higher tension of the player, as he immerses himself to a higher degree into the game. The differences between the two observations is due to the fact, that the previous examination included only peak values. A closer look at the data reveals that for some categories the music speed is close behind these values.
The pairing comprised of the terrain height and the amount of object layers shows a similar strength in this examination. It has the highest changes in the *double jump rate*, *the trap damage taken rate*, the *item health taken rate*, the *item time taken rate*, the *fall death rate*, and the *shot death rate*. In most of the cases, the magnitude of the change results from very low difference values for the summed single feature values and much higher difference values for the combined feature values, resulting in a much larger impact on the game in combined form. There is also a number of seemingly unaffected elements showing only marginal differences in the changes between low and high feature values. These are the *jump rate*, the *projectile damage taken rate*, the *enemy destroyed rate*, the *traps activation rate*, and the *item power taken rate*. It is interesting to see that the single *jump rate* is only a little higher in the combined version while the *double jump rate* shows a significant raise. It seems as if the increased use of trees and bushes makes the player jump more carefully through the levels, making use of the additional jump more often.
The last pairing examining the global hit feedback and the projectile speed disappoints in this analysis. When looking at the biggest differences between the two cases, none of the monitoring values shows a peak value. This is a strong indicator that these two features do not affect each other in a way that is interesting for future experiments. This assumption is also supported by the amount of monitoring values that show only very little variation between the two versions, including the *shot rate*, the *enemy hit rate*, the *trap damage rate*, and the *time needed*.
The perception of the interactions between the three pairings is also one part of the survey filled out by the participants. The results show, that the combination of the music speed and the amount of background layers was rated to have the smallest impact with 4.2 of 10 points. A little higher in the ranking the third feature pair can be found, comprised of the global hit feedback and the projectile speed, with 4.5 points of 10, only surpassed by the terrain height paired with the amount of object layers, which was rated with 4.7 of 10 (see 4.6). All the results are close to each other, but the small differences still show a variance in the player's perception. Only one of the participants gave the same rating to every field, meaning that the impact of the gameplay, the graphics, and the sound is generally differentially observed. The standard deviation indicates a high degree of fluctuation, too, being around 3 for every combination. In conclusion, it is difficult to derive information from these results, as there is hardly any direction to ascertain, whereas it is still remarkable that the combination of two features can lead to such a versatile outcome.
To gather further insight into the perception of the difficulty, every participant was asked to write down his impressions about the interaction between the tested feature pairings. The following section presents the most interesting statements.
Concerning the combination of the music speed and the amount of background layers, one participant notes, that it is indeed perceptible when the behavior of both features does not match. Another par-

	Music Speed & Background Layer	Terrain Height & Object Layer	Global Feedback & Projectile Speed
Player 1	0	0	0
Player 2	2	5	7
Player 3	5	4	5
Player 4	3	4	0
Player 5	6	8	9
Player 6	2	7	2
Player 7	8	6	0
Player 8	8	10	7
Player 9	10	2	9
Player 10	0	3	10
Player 11	5	4	2
Player 12	0	8	3
Player 13	5	0	5
Average	**4.15**	**4.69**	**4.54**
Std. Deviation	3.18	2.95	3.50

Table 4.6: Evaluation of the feature combinations using a 0 to 10 scale.

ticipant points out, that the combination of dramatic music and many background layers creates an increased perception of speed, which leads to more tension and a heightened difficulty. This theory is supported by another player who says, that the combination produces an uneasy feeling. While these statements do show that the combination has an effect, 50% of the participants believe that it does not influence the overall difficulty in any meaningful way.
The second pairing, consisting of the terrain height and the amount of object layers, successfully visualizes the consequences of getting stuck by letting the trees and bushes fall down when the terrain they are standing on crumbles, as one participant remarks. According to another player, the poor sight due to the amount of trees raises the tension when falling after a jump. This is also confirmed by another answer saying that the jumps become much more unpredictable. Additionally, bigger areas have to be observed to keep the complete overview. This is mostly due to the patchy object layers, which lead to problems judging the underlying terrain, as one player comments. Whereas this combination gets a lot of feedback, 25% of the participants claim there is no interaction at all.
The pairing of the global hit feedback and the projectile speed also provokes several statements about its interaction. One participant explains, that the more direct feedback of getting hit increases the perception of it, which then raises the tension of the game. In addition, the worsening of the viewing conditions after getting hit makes it harder for the player to dodge incoming projectiles. This also causes the player to concentrate more on the health bar at the top left corner of the screen, as another participant mentions. In this case, 33% of the participants state that they did not perceive any interaction between the two features.
In order to evaluate the game and find weak spots to improve upon, the survey ended with the question, whether the participant felt bored or frustrated in the course of the experiment and which element caused these signs of fatigue. The results show that the matter most players complain about is the monotony of the experiment, 50% would have liked more variety in the levels. 33% of the participants did not enjoy the background music, also noting the monotony of the tracks made the test seem even longer. In addition, 33% complained about the big number of levels, which made the experiment time-consuming. 25% of the participants were unsatisfied with the jumping mechanics, stating that some of their failures were due to incorrect input identification. Apart from that, there were also participants that did not perceive the experiment to be boring or frustrating. 42% commented that they actually enjoyed the experiment and were curious about the next one.

Feature	Shots	Enemy Hits	Jumps	Projectile Damage	Fall Deaths
Music Speed + Background Layer	−13.54	−6.54	−6.31	−1.54	−0.31
Music Speed & Background Layer	7.23	4.69	2.08	1.38	0.15
Terrain Height + Object Layer	2.08	−0.46	4.46	1.23	−0.23
Terrain Height & Object Layer	5.77	4.92	7.08	1.08	0.85
Global Feedback + Projectile Speed	6.38	3.92	3.85	0.69	0.62
Global Feedback & Projectile Speed	3.77	2.54	−1.15	−1.38	−0.31

Table 4.7: Magnitude of the changes between the summed single feature values (+) and the combined pairing values (&).

4.2.4 Experiment C: Dynamic Adjustment

In the third experiment, the DDA is employed for the first time in its full range as a completely automated and integrated adjustment mechanic. All the results from the previous two experiments helped shaping the settings responsible for the balance of the game and which monitoring value would be connected to which feature value. Again, the experiment is separated into two parts.
The first part *C-1* takes a closer look at the three fields gameplay, graphics, and sound again to determine if the changes made by the DDA make a difference for the player and how they affect the overall experience. There are three levels for every field, each starting with the same basic settings and dynamically adjusting the game from there.
The second part *C-2* is the endless mode. That means, that the game is played as it is without any external regulations. The player can continue the game as long as he likes and the DDA keeps adjusting the whole feature set to the needs of the player. There are three example cases examined in this part, each from a player with a different set of skills: one novice player, one moderate player, and one advanced player. For every player, the data sampled for the first 20 levels is compared to one another to get an understanding of the functionality of the DDA.
The survey shown to the player after each segment of *C-1* once again informs about the field that has been targeted for the changes and asks how these dynamic changes were perceived and how its impact on the difficulty was evaluated. After quitting the game, several statements are shown to the player which require to choose the degree of approval using a Likert scale. A 0 equals that the statement does not apply, a 5 stands for an abstention and a 10 translates to a complete approval. The statements are as follows:

- You were able to comprehend the progression of the difficulty.
- You felt supported by the game.
- You felt limited by the game.
- There was one or more levels that demanded too much of you.
- Some changes of the difficulty were too abrupt.

Results

The results for the first part of the third experiment are meant as an indicator about the competence of the DDA, as the short test runs prevent more detailed statements. They rather give a clue about the processes of the dynamic adjustments. This does not answer the question, whether the feature-monitoring links are chosen reasonably, though, as they have been selected solely based on intuition and expert knowledge. Apart from the verification of the technical side, the answers given in the survey are an important sign for the functionality of the DDA.
The survey answers verify what the previous experiments already indicated. The question about the impact of the specific fields of Dream Runner shows, that gameplay features clearly dominate the experience with a rating of 5.5 of 10 points. The graphics and sound leave a much smaller impression on the player with 2.2 and 1.4 of 10 points. 72% of the participants say that the gameplay field has a strong impact on the difficulty, making the game much more intense, while only 18% perceive it as small. 45% claim that the graphics have no or only a small effect, responding that the viewing conditions are the biggest factor in this. The benefits of graphical adjustments are a change in the perception of the speed and an enhanced control over the focus of the player. Concerning the sound, only 8% of the participants say that it has no impact at all. 42% have the opinion that the effect is small and 17% even consider it to be high. The most common response is that the background music emphasizes the tension, which is also responsible for a bigger error rate. Also the sound achieves to manipulate the atmosphere of the game in a very subtle way.
The opinion of the participants about the possibilities for the gameplay, the graphics, and the sound to influence the difficulty perception in general shows a very similar result. The gameplay again has the highest rating with 6.5 of 10 points, followed by the graphics with 3.9 of 10 and the sound with 3.2 of 10 (see appendix 8 and 9 for more information).
The results of the Likert scale can be seen in figure 4.1. They indicate that the difficulty can be perceived well enough and that the DDA incorporates the changes in a subtle way, although the game could support the player more and give a stronger direction for the difficulty.

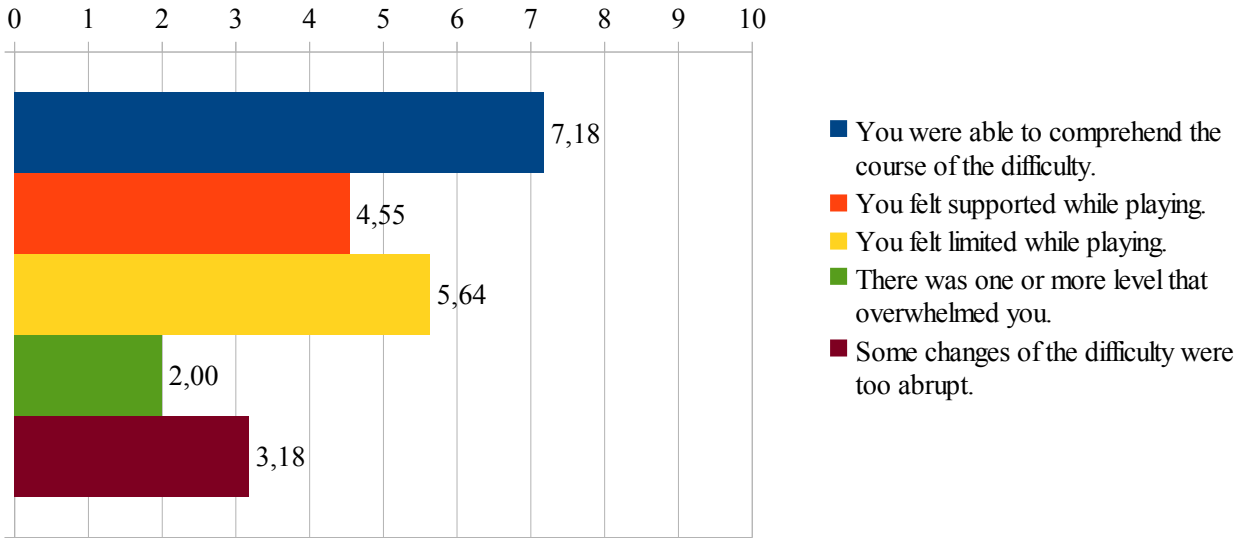

Figure 4.1: The results from the Likert scale, showing the player impressions of the last experiment.

Before conducting the second part of the experiment, the endless mode, two problems had to be solved. The first one was that *C-2* was optional and every participant had to play nine levels before reaching it, which resulted in only a small number of participants playing the endless mode at all. To solve this problem, an additional experiment consisting only of the endless mode had to be conducted. The second

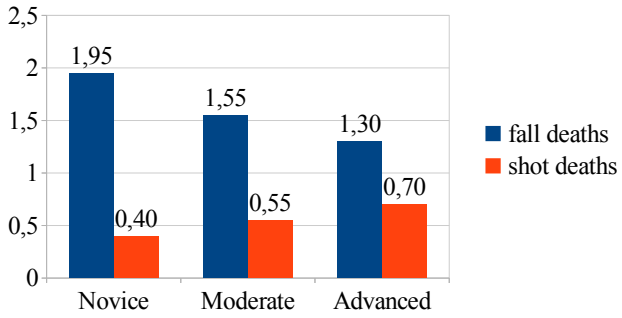

Figure 4.2: Illustrating the rates for the various deaths in the third experiment.

problem was that there were few volunteers in general, which made the amount of data too small to find out about the average estimation of the game. Instead, a different approach was chosen, which was to concentrate on samples and the course of the values. To understand the results of the experiment, it is important to keep in mind that the feature values are affected by two mechanisms, the DDA itself and the damping function, which lowers the feature values each time the players fails to complete the level. The monitoring values of the novice, the moderate, and the advanced player over the course of 20 levels give an insight on the player behavior and the influence of the difficulty on the different aspects of the game. Therefore, the results of the most important areas are compared to one another.

There are two ways of losing in Dream Runner, falling into a gap and losing all of the player energy. These two monitoring values show an interesting difference between the three participants, as the *fall death rate* sinks more and more, while the *shot death rate* keeps increasing (see fig. 4.2). This is due to the fact, that the *fall death rate* also includes being caught up by the level eraser and careless mistakes, which are more likely to happen to a novice player. In addition, an advanced player is prone to fail less often at the beginning of the game, when the feature values are still not accustomed to a style of playing. The increase of the *shot death rate* is the result of the gain of the damage values of projectiles and traps.

The monitoring values of the shooting shows a vast increase between the novice and the moderate player and only small changes between the moderate and the advanced player (see fig. 4.3). As there are much more enemies, projectiles, items, and traps on higher difficulties, it is only logical that the *shot rate* rises. The small increase on better player performances is due to the experience becoming too overwhelming to react to every source of danger. Instead, the player prioritizes and focuses on the most urgent dangers first. The accuracy shows a very similar result for the same reasons (see fig. 4.4).

The jumping, divided into normal jumping from the ground and double jumping in mid-air, also shows an increase concerning the normal jump, while having more unstable values for the double jump (see fig. 4.5). In higher difficulties, the jumping mechanism does not only serve as a tool to overcome gaps and higher ground, but also as a means to dodge traps and projectiles. The double jump is a good way

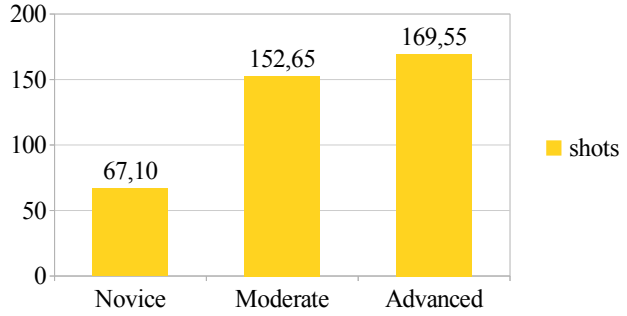

Figure 4.3: The mean values for the shot rate.

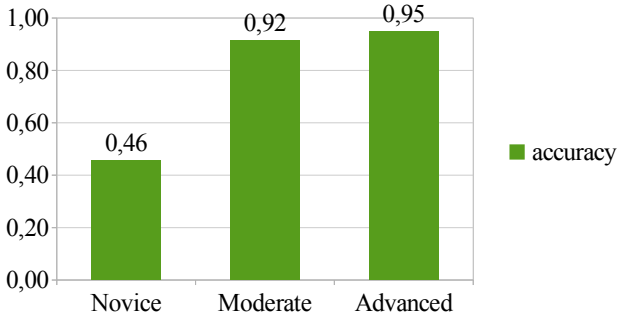

Figure 4.4: The evaluation for the accuracy.

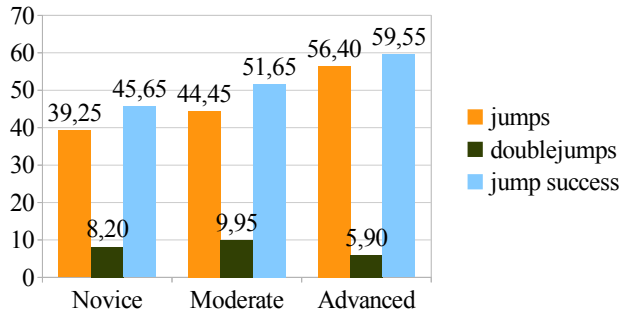

Figure 4.5: The comparison between the two jump types and the overall jump success.

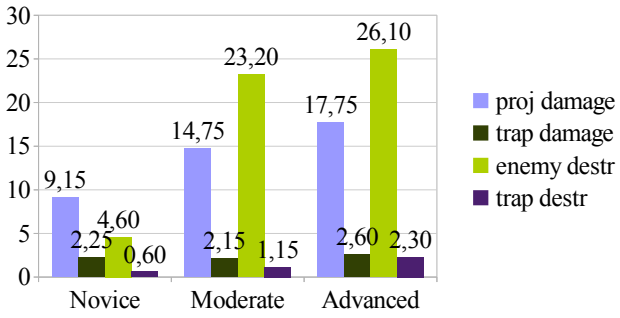

Figure 4.6: The damage dealt by projectiles and traps in comparison to the amount of destroyed enemies and traps.

to jump over larger gaps and gives the player much more freedom while navigating through the level, but it also can be difficult at times to keep track of the terrain, especially when the height changes become bigger. Therefore, it is logical to make more use of the normal jump on higher difficulties than the double jump.

When looking at the two different ways of getting damage, either by getting hit by a projectile or by stepping on a trap, the results show a continuous increase of the *projectile damage rate*, while the *trap damage rate* rises only very little (see fig. 4.6). The reason for this is the fact, that most of the players, even if inexperienced, are able to dodge many traps, as it is not difficult to simply jump over them. The small raise is due to the larger gaps on higher difficulties, which leave less terrain on which the traps are located. The higher differences of the *projectile damage rate* are attributable to a bigger number of projectiles shot by the enemies.

Both enemies and traps can be shot by the player in order to eliminate their threat. The results show much bigger values for the moderate and advanced player than for the novice when looking at the *enemy destroyed rate*, while the *trap destroyed rate* rises only on a very small level (see fig. 4.6). The very high amount of destroyed enemies for the moderate and advanced player matches the development of the *shot rate* (see fig. 4.3). Additionally on low difficulty settings more enemies can be ignored by the player without the level becoming impossible to beat. The small increase of the amount of destroyed traps can be explained by the fact that traps have much more energy on a higher difficulty, which makes it much harder to destroy them before they either pass by or explode. In conclusion, it is more appealing for the player to dodge the traps rather than trying to destroy them. For the complete monitoring values, see appendix 10, 14, and 18.

In order to get a deeper insight into the effect of the monitoring values and their interaction with the features, one gameplay feature area and the most interesting graphics and sound features are examined. The flying enemy created for Dream Runner is a gameplay element that is not only responsible for the shooting of projectiles, but also has a number of features controlling its behavior and outer appearance (see fig. 4.7). The energy feature has a very small value for the novice player, meaning that enemies could be destroyed by a few hits. The moderate and advanced player had to deal with enemies with much more energy. Similar to this, the average values for the enemy speed and size take on small values for the novice player and much higher values for the more experienced players. The enemy rate shows medium-sized values for the novice player and even higher values for the moderate and advanced player, reaching almost the maximum. The shooting frequency shows the most static increase of the enemy features. In conclusion, the enemy rate is rather high for every player type, but this is balanced out on easier difficulty settings by the other feature values attaining lower values. For additional information, see appendix 11, 15, and 19.

The graphics features aim to influence the difficulty mostly by adding more graphical elements or altering the viewing conditions for the player (see fig. 4.8). The amount of background layers shows a steady but small increase between the different player types. As it is dependent on the *accuracy* and the *fall death rate* of the player and these values show changes in opposite directions, the small differences are

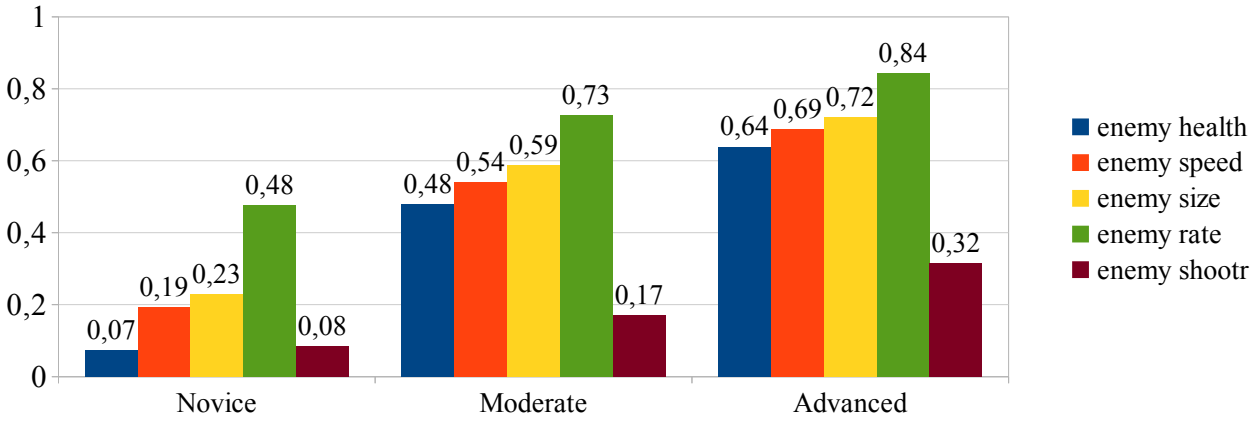

Figure 4.7: The different features of the flying enemy.

comprehensible. The amount of foreground layers on the other hand seems much more unstable. It is influenced by the *projectile damage rate* and the *fall death rate*. The differences between the amount of background layers and the amount of foreground layers, although partly dependent on the same monitoring value, lie in the stronger variations of the *projectile damage rate* and the fact, that the background layers are not influenced by the damping function, as this would have been too disrupting for the player.
The global hit feedback, determined by the *projectile damage rate* and the *trap damage rate*, surprisingly has the lowest average for the advanced player. This is explicable by looking at the two monitoring values, which both are strongly regulated by the difficulty setting, meaning that the feature reacts stronger when the player takes a lot of damage. The local hit feedback on the other hand shows a large growth between the three player types, the average being under 0.1 for the novice player and over 0.6 for the advanced player. Its adjustment takes both the *enemy destroyed rate* and the *trap destroyed rate* into account and therefore serves as a counterpart to the global hit feedback. The dramatic changes of the average feature values are due to the increasing relevance to destroy enemies and traps, which has a direct connection to the mentioned monitoring values.
The amount of object layers, displayed as trees and bushes on the terrain, is influenced by the *accuracy* and the *enemy destroyed rate*. It also exhibits an enormous jump when comparing the novice and the moderate player, which is also a result of the growing relevance to destroy enemies in order to survive.

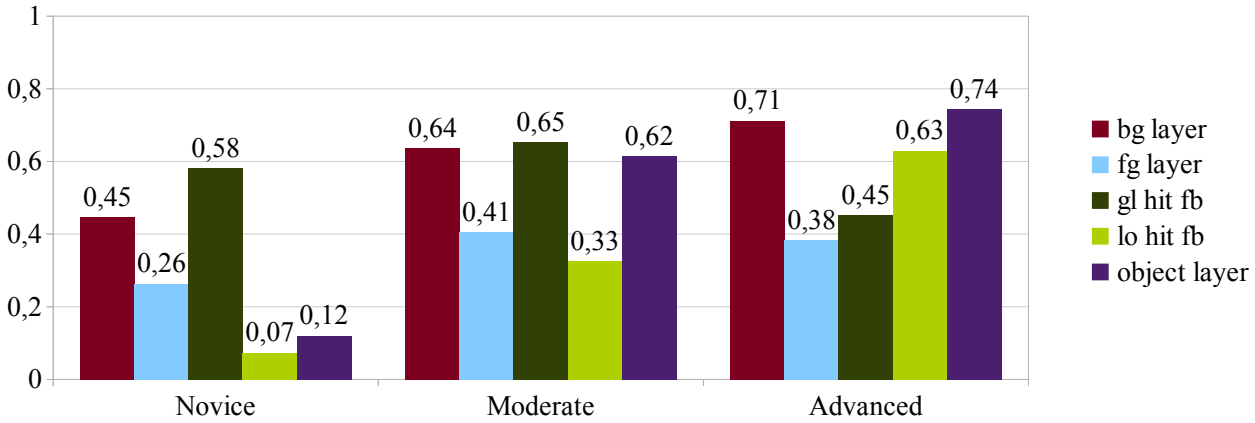

Figure 4.8: The comparison of the foreground and background layer, the global and local hit feedback, and the object layer.

The much higher *accuracy* – compared to the aforementioned *trap destroyed rate* – contributes further to this development. More detailed information can be seen in appendix 12, 16, and 20.
The sound features range from simply adjusting the volume of the sound effects and music to changing the whole instrumentation of the music track (see fig. 4.9). The adjustment of the volume of sound effects is affected by the *jump success rate* and the *accuracy*. It increases steadily from novice to advanced player, which is mostly due to the same level of growth of the *jump success rate*. The bluntness of the sound effects is dependent on the *enemy destroyed rate* and the *player health*, as it only becomes important when the player is low on health. Again the *enemy destroyed rate* is the substantial monitoring value, because the *player health* delivers only alternating results.
The background music volume is increased steadily between the three players. It is defined by the *fall death rate* and the *accuracy*. Although the *fall death rate* would suggest a decreasing music volume, the dramatic gain of the *accuracy* cancels this effect. The intensity of the background music is influenced by the same monitoring values. The difference of the results is due to the varying damping values if the player fails to complete the level. The speed of the music, affected by the *fall death rate* and the *jump success rate*, has a small increase from the novice player to the advanced player, which is a result of the successful jumps also showing a growing monitoring value. The last sound feature is the hit feedback of the background music, which determines how long the music fades when the player is hit. It is influenced by the *projectile damage rate* and the *enemy destroyed rate* and has a strong gain between the novice and the moderate player. This can be explained by looking at the volatile increase of the *enemy destroyed rate*, which is also supported by the *projectile damage rate*. For more information, see appendix 13, 17, and 21.

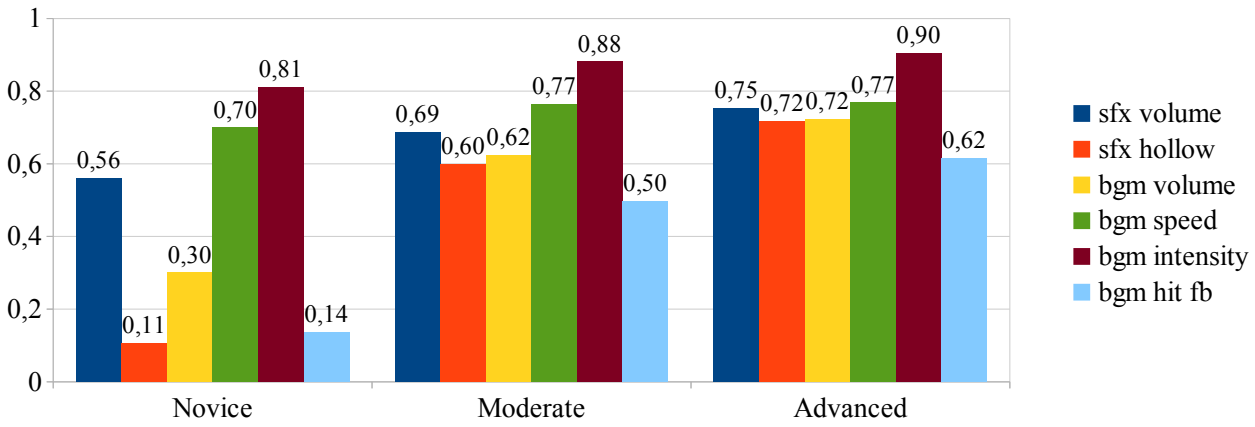

Figure 4.9: The comparison of all sound features in Dream Runner.

4.3 Discussion

The results of the experiments give room for a number of interpretations, which are presented and discussed in this section.
What can be noted first of all is that the DDA accomplishes to deliver results that are *stable*, *measurable*, and *versatile*. *Stable*, because every input value leads to a valid output value due to the use of thresholds. In addition, the same input always returns the same output, as there is no ambiguity involved in the process. All of the data of the system is *measurable*, because every feature and monitoring value is represented as a number, which makes the comparison of input and output values possible. The system is *versatile*, because the difficulty in the game ranges from *very easy* to *very hard*, which aims to make the game playable for every type of player and give the DDA enough room to adjust the settings of the game. Additionally, features can be edited and exchanged between play sessions, which makes the

adaptation of the experience very easy.
There are three questions that are the most relevant for the evaluation of the practical part:

1. Does the DDA succeed to adjust the difficulty according to the needs of the player?
2. Does it increase the fun for the player through a non-disruptive, smooth game experience?
3. How big is the impact of graphics and sound adjustments for the difficulty of the game?

Whether the DDA is capable of determining critical situations for the player and react accordingly is of the utmost importance, as the player would not be able to enter the state of flow otherwise and thereby lose his motivation very soon. The results of the experiment show the features have a very varied influence on the difficulty, behaving differently dependent on their application in the game, the state of the player, or interactions with other features, which often shift the effect of the feature into another direction, as shown in the second experiment. In conclusion, the difficulty is shaped by a multitude of influences. Additionally, the progression of the feature values shows a clear distinction between the novice, the moderate, and the advanced player (see figure 1 in the appendix). The first figure shows the feature values of the novice over the course of 20 levels. The values span the whole range, reaching from low feature values to high ones. Most feature values can be found in the middle or at the top of the scale for the moderate player, as the second figure illustrates. The majority of the feature values of the advanced player are placed at higher regions, with only a few features positioned at middle-sized values.
The second question aims to investigate if the DDA manages to interact with the game without revealing the adjustments to the player. Although not directly linked to the work of the DDA, the jump mechanism has been the most frequent complain of the participants, often destroying the immersion of the player and thereby also preventing the state of flow. Another issue has been the monotony, which was expressed by 50% of the participants as a major problem of the game. Although this was mainly due to the high number of levels in the second experiment, it might most likely carry over to the endless mode, whereas the presence of an appropriate challenge will keep the player motivated for a longer period of time. A good sign for the non-disruptive procedure of the game is the fact, that 42% of the participants did not find any frustrating or boring elements and enjoyed the overall experience. In addition, in every experiment the player was asked after each level how much fun he had while playing it. The average values of the complete tests show a steady growth over the course of the experiments, beginning with 5.1 points of 10 in the first one, 5.7 of 10 in the second experiment, and 6.5 of 10 in the last one (see appendix 3). This is due to the fact, that the experiments continuously concentrated on a more detailed level, with the first experiment examining only the changes of all gameplay, graphics, or sound features, the second experiment focusing on single features and combinations of features, and the third one adjusting each feature dynamically in a much more subtle way. This serves as a proof that the DDA does accomplish to increase the fun of the player and for almost half of the participants, this was achieved without interrupting the experience.
For answering the third question, the relevance of adjustments of the graphics and sounds has to be determined. It is safe to say, that the gameplay field wields the biggest influence. This can be seen in every conducted experiment, both in the evaluation of the monitoring values and in the survey results. It has the most obvious impact on the difficulty, as it influences the way the player interacts with the game.
In the first and third experiment, the participants were asked about their assessment of the importance of graphical and sound adjustments for the configuration of the difficulty settings. The results vary between the two tests, as the graphics are on a par with gameplay changes in the first experiment (5.9 of 10), while the relevance of the sound is evaluated to be much lower (3.1 of 10). In the last experiment, the order is much clearer, rating the gameplay with 5.5 of 10, the graphics with 2.2 of 10 and the sound with 1.4 of 10. As the first experiment only looked at the extreme values of every field, the results of the last experiment are much more significant, because it utilized the DDA to adjust the features smoothly. This trend is underpinned further when looking at the second part of the survey of the third experiment. It reveals that 72% of the participants rate the gameplay to have a strong impact, while only 18% believe it has a small influence. Concerning the graphics, 45% think that its impact is

negligible. The sound features are evaluated by 50% of the participants as having no or only a small impact, although 17% are of the opinion that it exerts a strong influence.
The focus on features from all three fields in the second experiment shows, that not only gameplay elements accomplish to change the way the difficulty is perceived. The bluntness of sound effects, a rather subtle effect that is strongly attached to the performance of the player, has many peak values when looking at the differences between low feature values and high feature values. The amount of background layers exhibits a similar impact, producing high differences in key elements like the *jump rate* or the *time needed*. Another interesting examination is, that the sampled values and the perception of the player are often very different. Whereas the first experiment shows a much higher evaluation of graphics features by the player than the monitored data might suggest, it is the other way around in the second experiment. This time, the results from the survey are generally much smaller than the monitoring values. This is affirmed by the fact, that no participant gave the same rating to all three fields, meaning that there was definitely a perceived difference between them.
To further investigate the meaning of the player's perception, the statements about the major effects of the features are examined. There are three main influences stated by the participants, the first being the alteration of the perception of the speed, which has been detected to be evoked by the music speed in combination with the amount of background layers. The more movement is on the screen, e.g. by the use of many parallax layers or many objects like enemies, and the faster the music plays, the higher is the perceived speed of the game. The second factor is the perception of tension. It is mostly affected by the viewing conditions in the level, e.g. by the use of fog with different transparency values or strong visual hit feedback, but also by a large amount of objects that ask for the player's attention, making the game much more intense. The third influence is the perception of consequences. Demonstrating to the player what happens if he does not successfully play through the level is a powerful tool to encourage a better performance. A feature in Dream Runner that accomplishes to create this effect is the level eraser. As it destroys every block and trap that it touches and causes the trees to fall down and vanish, the player can immediately see why it is important to prevent getting stuck in the terrain, even if it gives time to aim and plan the next moves.

Chapter 5

Conclusion

5.1 Summary

The center of this thesis is the conception and implementation of a *feature-based Dynamic Difficulty Adjustment* system (DDA). Its key feature is the automatic manipulation of the balancing of the accompanying game project *Dream Runner* in order to offer a difficulty setting that reacts to the needs of the player. The goal of the DDA is to prevent the experience from being frustrating, by providing a challenge that stimulates the skills of the player to become continuously better, while also reacting on critical states, like being overwhelmed.
The application of the DDA is constructed around the use of *features*. Features serve as a minimum unit for every customizable element of the game. Each feature contains a numerical value in the interval of $[0, 1]$. This value represents the difficulty for this specific feature, meaning that higher values correspond to a higher difficulty. A feature value of 0 translates to *very easy*, which implies that this particular feature effects the overall difficulty only on a minimal level. 1 means *very hard* and provides an enormous challenge, i.e. with all feature values set to 1, it is very unlikely that any player finishes the level in a reasonable amount of time. As every element that is important for the shaping of the difficulty is manipulated by changing the numerical value of its feature, the DDA can easily be configured and adapted.
The conducted research shows, that both scientific works and commercial video games that concentrate on dynamic difficulty adjustments include mostly gameplay elements into the evaluation process, while neglecting the impact of graphics and sound on the difficulty. The DDA in this thesis is built to be able to work with any type of feature, as it does not differentiate between gameplay elements and graphical or sound parameters, as long as the layout of the feature is correct. Thus the execution of special experiments can answer the question, whether these two fields help to shape the difficulty and thereby increase the fun of the game. The experiments each consist of a test play gathering data from the game that is saved into a single file. In order to extract as much information from the experiments as possible, additional data is obtained through the use of surveys that the players fill in after every play session. This allows for the comparison of the objective data created by the DDA and the subjective player impressions.
The results of the experiments verify that the DDA is stable, measurable, and versatile. The courses of the feature values for a novice, a moderate, and an advanced player show, that the DDA reacts according to the style of playing. Although there are elements that disrupt the flow effect like the jumping mechanics, almost half of the participants enjoyed the overall game and did not find any elements that bored or frustrated them while playing. Thus, the DDA works subtly and does rarely interfere with the experience of the player.
Both the sampled data and the the answers from the survey clarify, that the gameplay features have by far the biggest impact on the difficulty. They wield a strong influence on the tension of the player, as they control the amount of enemies and obstacles the player has to overcome. In the second place are

the features adapting the graphics. Their main effect is to manipulate the viewing conditions, by either putting more graphical elements between the player and the actual gameplay or by adding motion to the scene. Both can act as a way to increase the tension or to draw the attention of the player to a specific element. The sound features get third place, affecting the perception of speed by changing the speed of the background music. Additionally, by adjusting the atmosphere and intensity of the track, the tension felt by the player can be enhanced.
In conclusion, the feature-based DDA created for this thesis accomplishes to intensify the experience of the game for different player types, without impairing the playability. Through the inclusion of graphics and sounds into the adaptation process, new possibilities for shaping the difficulty can be utilized to encourage the player and increase his motivation and enjoyment. Despite the fact, that these two fields only have a minor impact on the overall difficulty, they can be used as an addition to gameplay changes to create a more balanced and multifaceted experience.

5.2 Evaluation

As a means to evaluate the accomplishments reached in this thesis, four goals have been defined. These goals all contain specific information about the demands put on the feature-based DDA. In order to be able to analyze the success of the system, every experiment is examined regarding its relevance for meeting the particular objective.
The data gathered by the DDA system concerning the difficulty concurs by 80% with the surveys filled in by the players.
One of the main purposes of the DDA is to support the player by creating a challenge based on his skills. Therefore, it is crucial that the definition of the difficulty is the same for the player and the DDA. To verify this, the preparation experiment tests, if the borders of the difficulty are matching the expectations of the player. The experiment contained three levels with three different difficulty presets.

1. Easy: all the feature values are set to 0.2
2. Medium: all values are set to 0.5
3. Hard: all values are set to 0.8

The results of the survey show, that the difficulty of the easy level was rated with 4.2 of 10, the medium level with 6.8 of 10, and the hard level with 9.6 of 10 points (see table 4.2). After converting these values to the same scale, the difference between the actual data and the perceived values is 22% for the easy level, 18% for the medium level, and 16% for the hard level. As only the first value exceeds the 20% variation between the gathered data and the impression of the players by 2%, the overall goal is achieved.
The game project designed for the thesis manages to keep the player motivated for at least 70% of the time.
In order to get information about the perception of the game, every experiment included a prompt after every level asking how much the player enjoyed this particular level. The results for the preparation experiment reveal an average enjoyment of 63% over three levels, although the third level was not designed to be enjoyable due to its high difficulty settings. The first official experiment shows a slightly smaller outcome with an average amount of enjoyment of 51% over the course of ten levels. Higher values were measured in the second experiment, with an average of 57%. This value is still too low, which is due to the fact, that the experiment was, with a total of 23 levels, too long and that the examination of the different features was not interconnected. The first part of the last experiment performs better and reaches an average enjoyment of 65% over the course of nine levels, but as it employs the functionality of the DDA only for one single field at a time, the results are not significant enough. The second part of the experiment consists of the endless mode, giving the player the opportunity to play as long as they like. The results are much higher, rating the overall enjoyment with 78%. As every player decided independently how long the game would be, this outcome is the most important for the evaluation of

the immersion. Therefore, this goal is also achieved.
The surveillance and modification of the game through the DDA system takes place in a non-disruptive manner by 80% and does not interfere with the gaming experience.
Providing an experience that is not impaired by the actions of the DDA is very important to prevent the player from feeling cheated. The second experiment aimed to inquire if the participants feel frustrated or bored during the game session and which elements of the game evoke these feelings. The biggest complaint is the monotony, which is rather attributable to the structure of the experiment than the feature values. The same is true for the amount of levels, that 33% of the participants perceive as too high. Directly connected to the effect of the DDA is the music and its properties. Again, 33% claim it adds to the monotony and does not entertain for the whole time of the experiment. The jumping mechanics, which are not affected directly by the DDA, are rated by 25% of the participants as disruptive. In conclusion, only one field that is influenced by the DDA, the music, has been mentioned to impair the gaming experience. 42% of the participants even stated that the game does not contain any frustrating or boring elements.
One part of the survey of the third experiment asked the players a number of questions concerning the overall playability and the perception of the DDA. The results show, that the participants rate the statement, that they were able to comprehend the progression of the difficulty, with 72%. The perception of the support and restrictions of the game is at 46% and 56%, respectively, showing that the adjustment accomplishes to improve the experience, although there are elements that do not work well enough. Furthermore, the statement, that there was one or more levels which was too hard to complete, is rated with 20%, proving that the participants did not feel overwhelmed by the difficulty. The last question is, whether there were changes of the difficulty that were too abrupt. The rating shows that the amount of approval is at 32%, which as a result means, that the players perceive the adjustments of the DDA as non-disruptive by 68%.
As neither the 42% of the participants not finding any impairing elements, nor the 68% approval that the changes were not too abrupt reach the intended 80%, this goal is not achieved. One way to proceed with this problem would be to use smaller adjustments with a smoother transition in future experiments.
The modification of graphics and sound effects has a measurable impact on the difficulty and 40% of the players name it a game-defining element.
As the impact of graphics and sound on the difficulty of a game has not been explored yet, the focus of this work is to find a way of incorporating all elements into one evaluation structure. This is achieved by using features that do not differentiate between gameplay, graphics, or sound, which facilitates to compare the results of different areas of the game to one another. In the first experiment, the participants state that only gameplay elements play an important role in the definition of the difficulty. The second experiment proves otherwise, as the monitoring data shows that the sound feature *sound bluntness* and the graphics feature *amount of background layers* do have a measurable impact. The combination of the music speed with the background layers also has a big influence on the monitored values. The answers of the survey confirm this, as the impact of the aforementioned combination is rated with 50%. In addition, the combination of the terrain height and the amount of object layers is assessed with 75% and the global hit feedback with the projectile speed with 66%. In the third experiment, an overall ranking shows, that the impact of graphics on the difficulty is rated with 55%, while the impact of the sound is at 50% (see appendix 9). All these values exceed the requirement of 40%, which leads to the conclusion that his goal is achieved.

5.3 Limitations

In order to be able to create a fully functional DDA and a complex game project over the course of this thesis, several tradeoffs had to be made. But also technical limitations due to the choice of the development tools demanded attention in the conception and design of the program.
One drawback is, that some features only have discontinuous changes. This does not mean that they are treated differently than the continuous features, as their states are simply mapped onto the feature

interval, but the evaluation of the effect becomes much more imprecise. This is true for features like the amount of background layers or the music. As one background layer is an indivisible unit, the feature only adds a new layer once a certain threshold is exceeded. The same applies for the music tracks. The transition from one instrumentation setup to the next cannot be executed smoothly. A more direct approach would be needed for this kind of interaction in order to provide a smoother and non-disruptive difficulty adjustment.
Due to the experiments being conducted over the internet, it is not possible to guarantee that every player used the same computer setup. Variables like the settings of the monitor or the audio devices probably had very different configurations accustomed to the personal preferences of the user. In order to minimize this effect, an introductory text points out, that the graphics and sound are an important part of the experiment, and asks the player to play the game with all settings turned on.
To keep the complexity of the DDA manageable, features react only on the current state of the monitoring values at the end of each level. It does not take any knowledge from previous levels or play sessions into account. This limits the actions of the DDA, as it is not possible to predict a future state of the game. Additionally, in order to increase the performance of the game, the DDA applies its adjustments only once for each level. Although this prevents eventual latency issues, the DDA cannot react directly on critical states of the player. This problem is mitigated by the implementation of a damping function that responds to frequent failures by reducing each feature value by a preset amount. The function is only a workaround to correct adjustments that were too big and to prevent the player from losing his motivation, though.
Another concept that proved to be too complex to integrate is the inclusion of hybrid features into the evaluation process. In this thesis, every feature is embedded in one of the three classes gameplay, graphics, or sound. But there are features that actually belong to two classes, the gameplay class and the graphics class, as the effect of the feature influences both the interaction with this particular element and its graphical representation. One example for this is the size of the enemies. By changing the feature value, the dimensions of the enemy image are modified. In Dream Runner, this feature belongs solely to the gameplay class, as it has an impact on the chances to hit the enemy, but at the same time, the visual effect of a bigger shape is neglected. This is due to the decision to keep the concept of features as simplistic as possible, as they serve as the minimal unit for the DDA. Including a whole new type of features would have raised many new questions about the human perception and the weight of the particular influences. These studies are postponed for future experiments.

5.4 Future Work

The algorithms and techniques used for the realization of the dynamic difficulty adjustment in this thesis only serve as the groundwork to collect the data necessary for the evaluation. There are numerous ways to optimize and modify the current setup and the most interesting possibilities are discussed in the following section.
As one focus of this work was to study the impact of graphical and sound elements on the overall difficulty, additional mechanics could be incorporated into the game. To get a better understanding of the influence of sound effects and the background music, the adjustment could take place on a deeper level, working directly on the raw data of sounds and music. This would enable the use of effects like echoes, reverberations, etc. For enhanced possibilities concerning the graphical adjustments, a shader technology could be used to add visual effects during run-time. The advantage would be a less disruptive modification through smoother effects.
An area of the DDA that has been neglected due to the small time frame is the continuous level generation. A fully developed level generator that does not rely so heavily on random functions could be applied in order to create more diverse levels, while also offering more control over the structure of the particular parts. This could be used to avoid critical problems, like two gaps merging into one very large gap, but also to incorporate the idea of features better into the level design. In the current game, there are only a few features controlling the height changes of the terrain, the width of gaps or the

length of the level. With an enhanced level generator, much more detailed and precise features could be introduced, like the minimum and maximum space between gaps or the course of the terrain height throughout the level.
The structure of features is currently very simplistic, as every feature holds exactly one value that determines its difficulty. As a result, the evaluation of the monitoring values and the manipulation of the feature values takes place in a one-dimensional manner. This setup could be expanded by further parameters. For example, the addition of weights to every connection between a monitoring value and its respective feature value would lead to more control over its impact on the difficulty.
The implementation of player profiles would be another possibility to enhance the effect of the DDA and make its modifications more accurate. It could analyze the style of the player and thereby provide a more personal experience. This could be achieved by adding a memory to the DDA, which stores all the previous results into a profile that can then be used for every play session, as it is continuously extended and refined. A step further would be to implement a DDA that learns from all user data to define own player classes.
Finally the DDA could be tested in different environments. By way of example, the DDA could be ported to other video game genres, like strategy games, where the artificial intelligence of the enemies and the accessibility of the terrain could be altered, or racing games, where parts of the race track or the weather conditions could be modified. This would facilitate to test the customizability and versatility of the DDA and enhance its scope of application.

Bibliography

Gustavo Andrade, Geber Ramalho, Hugo Santana, and Vincent Corruble. Extending reinforcement learning to provide dynamic game balancing. *In Computer Games*, 1:7–12, 2005. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.9922&rep=rep1&type=pdf#page=11.

Gustavo Andrade, Geber Ramalho, Alex Sandro Gomes, and Vincent Corruble. Dynamic game balancing: An evaluation of user satisfaction. In John E. Laird and Jonathan Schaeffer, editors, *AIIDE*, pages 3–8. The AAAI Press, 2006. ISBN 978-1-57735-235-8. URL http://www.aaai.org/Papers/AIIDE/2006/AIIDE06-005.pdf.

Christine Bailey and Michael Katchabaw. An experimental testbed to enable auto-dynamic difficulty in modern video games. *In Proceedings of the 2005 GameOn North America Conference*, 1:1–5, 2005. URL http://www.cp.eng.chula.ac.th/~vishnu/gameResearch/AI_november_2005/AN%%20EXPERIMENTAL%20TESTBED%20TO%20ENABLE%20AUTO-DYNAMIC%20DIFFICULTY%20IN%20MODERN%20VIDEO%20GAMES.pdf.

Capcom. Capcom homepage. URL http://www.capcom.com.

Capcom. Resident Evil 5. [DVD-Rom], 2009.

Darryl Charles, Michael Mcneill, Moira Mcalister, Michaela Black, Adrian Moore, Karl Stringer, Julian Kücklich, and Aphra Kerr. Player-Centred Game Design: Player Modelling and Adaptive Digital Games. In *Digital Games Research Association 2005 Conference: Changing Views - Worlds in Play*, 2005. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.97.735&rep=rep1&type=pdf.

Jenova Chen. Flow in games (and everything else). *Commun. ACM*, 50(4):31–34, April 2007. ISSN 0001-0782. doi: 10.1145/1232743.1232769. URL http://doi.acm.org/10.1145/1232743.1232769.

Mihaly Csikszentmihalyi. *Flow: The Psychology of Optimal Experience*. Harper Perennial, 1991. ISBN 0060920432. URL http://books.google.de/books?id=epmhVuaaoK0C&dq=%22Flow:+The+Psychology+of+Optimal+Experience%22&source=bl&ots=Hx6i8q2pZG&sig=bDAfDvxZo1LixpEcYeGyf_nx7PU&hl=de&sa=X&ei=dKooUOiqCsjEswb444HwCA&ved=0CDQQ6AEwAA.

Robby Goetschalckx, Olana Missura, Jesse Hoey, and Thomas Gärtner. Games with dynamic difficulty adjustment using pomdps. *Artificial Intelligence*, 1:1–6, 2010. URL http://www-kd.iai.uni-bonn.de/icml2010mlg/papers/GoetschalckxEtAl.pdf.

Robin Hunicke. The case for dynamic difficulty adjustment in games. *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, 265:429–433, 2005. URL http://portal.acm.org/citation.cfm?id=1178573.

Robin Hunicke and Vernell Chapman. Ai for dynamic difficulty adjustment in games. *Assessment*, 1: 91–96, 2004. URL https://www.aaai.org/Papers/Workshops/2004/WS-04-04/WS04-04-019.pdf.

Martin Jennings-Teats, Gillian Smith, and Noah Wardrip-Fruin. Polymorph: dynamic difficulty adjustment through level generation. In *Proceedings of the 2010 Workshop on Procedural Content*

Generation in Games, PCGames '10, pages 11:1–11:4, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0023-0. doi: http://doi.acm.org/10.1145/1814256.1814267. URL http://doi.acm.org/10.1145/1814256.1814267.

Rudolf Kruse, Kai Michels, Frank Klawonn, and Andreas Nürnberger. *Fuzzy-Regelung: Grundlagen, Entwurf, Analyse*. Springer-Verlag, Berlin, 2002. ISBN 3-540-43548-4. URL http://www.springer.com/978-3-540-43548-8.

Ricardo Lopes and Rafael Bidarra. Adaptivity challenges in games and simulations: A survey. *IEEE Trans. Comput. Intellig. and AI in Games*, 3(2):85–99, 2011. URL http://dx.doi.org/10.1109/TCIAIG.2011.2152841.

Martin Mladenov. Offline learning for online difficulty prediction. *Proceedings of the ICML Workshop on Machine Learning and Games*, 1:1–6, 2010. URL http://www-kd.iai.uni-bonn.de/icml2010mlg/papers/MladenovMissura.pdf.

Nintendo Co., Ltd. Nintendo homepage. URL http://www.nintendo.de.

Nintendo Co., Ltd. Mario Kart: Double Dash!! [Cartridge], 2003.

A. Joy James Prabhu. Improving dynamic difficulty adjustment to enhance player experience in games. In *ICT*, pages 303–306, 2010. URL http://dx.doi.org/10.1007/978-3-642-15766-0_44.

Pieter Spronck, Ida Sprinkhuizen-Kuyper, and Eric Postma. Difficulty scaling of game ai. *International Journal of Intelligent Games and Simulation*, 3(1):33–37, 2004. URL http://www.dcc.ru.nl/~idak/publications/papers/SpronckGAMEON2004.pdf.

Pieter Spronck, Marc Ponsen, Ida Sprinkhuizen-Kuyper, and Eric Postma. Adaptive game ai with dynamic scripting. *Mach. Learn.*, 63:217–248, June 2006. ISSN 0885-6125. doi: 10.1007/s10994-006-6205-6. URL http://dl.acm.org/citation.cfm?id=1137598.1137603.

Valve Corporation. Valve homepage. URL http://www.valvesoftware.com.

Valve Corporation. Half Life. [CD-Rom], 1998.

Valve Corporation. Left 4 Dead. [DVD-Rom], 2008.

Chang Yun, Philip Trevino, William Holtkamp, and Zhigang Deng. Pads: enhancing gaming experience using profile-based adaptive difficulty system. In *Proceedings of the 5th ACM SIGGRAPH Symposium on Video Games*, Sandbox '10, pages 31–36, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0097-1. doi: http://doi.acm.org/10.1145/1836135.1836140. URL http://doi.acm.org/10.1145/1836135.1836140.

Appendix

Feature - Monitore Value Links

Segment Length	**Eraser Speed**	**Terrain Width**	**Terrain Height**
Shot Death *Fall Death*	*Fall Death*	*Jump Success* *Fall Death*	*Jump Time* *Fall Death*
Gap Width	**Item Health**	**Item Power**	**Item Time**
Jump Success *Fall Death*	*Player Health*	*Enemy Destroyed*	*Projectile Damage Rate* *Trap Damage*
Item Rate	**Enemy Health**	**Enemy Speed**	**Enemy Size**
Item Pickup	*Accuracy* *Enemy Destroyed*	*Accuracy*	*Accuracy*
Enemy Rate	**Enemy Shoot Rate**	**Projectile Health**	**Projectile Damage**
Shot Death *Accuracy*	*Shot Death*	*Player Health* *Shot Death*	*Player Health* *Projectile Damage Rate*
Projectile Speed	**Projectile Size**	**Projectile Rate**	**Trap Health**
Shot Death	*Accuracy* *Player Health*	*Shot Death* *Enemy Destroyed*	*Player Health* *Shot Death*
Trap Damage	**Trap Size**	**Trap Rate**	**Background Layer**
Player Health *Trap Damage Rate*	*Trap Damage Rate*	*Trap Destroyed* *Player Health*	*Accuracy* *Fall Death*
Foreground Layer	**Globel Feedback**	**Local Feedback**	**Object Layer**
Projectile Damage Rate *Fall Death*	*Projectile Damage Rate* *Trap Damage Rate*	*Enemy Destroyed* *Trap Destroyed*	*Accuracy* *Enemy Destroyed*
Dark Squares FX	**Slow Motion FX**	**Sound Volume**	**Sound Bluntness**
Jump Success *Player Health*	*Jump Success* *Fall Death*	*Jump Success* *Accuracy*	*Enemy Destroyed* *Player Health*
Music Volume	**Music Speed**	**Music Intensity**	**Music Feedback**
Fall Death *Accuracy*	*Fall Death* *Jump Success*	*Fall Death* *Accuracy*	*Projectile Damage Rate* *Enemy Destroyed*

Table 1: Enumeration of every **feature** and the *monitoring values* having an influence on it.

Experiment A Data

	Shots	Enemy Hits	Item Hits	Overheating	Projectile Damage	Jumps
Level 1	36.00	16.92	3.63	0.00	1.42	18.46
Level 2	240.58	139.00	10.71	0.58	16.25	77.71
Level 3	202.92	130.75	9.00	1.75	22.75	51.04
Level 4	109.79	63.33	4.79	0.25	7.21	31.71
Level 5	31.38	18.04	2.38	0.00	1.25	19.08
Level 6	33.54	17.38	2.96	0.00	1.13	18.00
Level 7	31.67	16.04	2.75	0.00	1.25	17.25
Level 8	34.96	17.13	3.96	0.00	1.29	18.21
Level 9	31.67	17.33	3.46	0.00	0.83	17.71
Level 10	36.38	17.46	4.63	0.00	1.13	18.83

	Item Pickup	Fall Deaths	Shot Deaths	Level Skip	Player Health
Level 1	3.04	0.38	0.00	1.00	96.17
Level 2	8.76	7.21	0.00	0.83	63.82
Level 3	5.85	2.63	0.92	0.88	50.53
Level 4	3.74	1.00	0.04	1.00	71.40
Level 5	3.11	0.25	0.00	1.00	96.03
Level 6	2.97	0.13	0.00	1.00	95.61
Level 7	3.00	0.17	0.00	1.00	96.37
Level 8	3.04	0.25	0.00	1.00	95.36
Level 9	2.93	0.08	0.00	1.00	97.27
Level 10	3.15	0.17	0.00	1.00	96.11

Table 2: Monitoring results of the first experiment.

	Experiment A	Experiment B	Experiment C
Level 1	5.00	6.69	7.54
Level 2	4.22	6.23	6.15
Level 3	5.11	5.77	6.08
Level 4	6.22	6.15	6.15
Level 5	4.56	5.92	6.54
Level 6	4.89	5.69	6.38
Level 7	5.00	5.77	6.54
Level 8	5.00	5.31	6.62
Level 9	5.56	5.38	6.54
Level 10	5.44	5.38	
Level 11		5.54	
Level 12		5.85	
Level 13		5.31	
Level 14		5.46	
Level 15		5.77	
Level 16		5.38	
Level 17		5.85	

Table 3: The mean values for the enjoyment rating for every experiment.

Experiment B Data

	Music Speed		Sound Bluntness		Background Layer		Object Layer	
Player 1	5	0	5	0	5	0	5	0
Player 2	5	0	5	0	7	1	7	2
Player 3	0	0	5	0	5	0	7	6
Player 4	8	2	5	0	5	0	10	3
Player 5	5	0	5	0	6	6	5	1
Player 6	5	0	5	0	6	4	7	6
Player 7	6	0	5	0	8	0	8	0
Player 8	5	0	7	5	9	9	7	8
Player 9	6	7	7	2	0	8	10	5
Player 10	5	0	5	0	5	0	5	0
Player 11	5	1	5	0	6	5	7	4
Player 12	6	1	5	0	7	2	6	2
Player 13	5	0	5	0	10	10	5	5
Average	**5.08**	**0.85**	**5.31**	**0.54**	**6.08**	**3.46**	**6.85**	**3.23**
	Global Feedback		Terrain Height		Eraser Speed		Projectile Speed	
Player 1	6	5	5	0	5	2	5	0
Player 2	8	9	3	7	9	8	5	7
Player 3	8	6	5	0	8	4	5	3
Player 4	5	2	5	0	5	0	5	2
Player 5	3	5	4	7	7	5	5	2
Player 6	6	0	5	1	5	0	5	1
Player 7	0	5	7	2	5	0	5	1
Player 8	0	10	5	5	5	0	4	7
Player 9	3	6	8	10	5	7	5	0
Player 10	8	7	7	4	7	5	6	5
Player 11	3	5	5	1	6	4	5	3
Player 12	5	0	5	1	8	3	5	0
Player 13	5	0	5	0	10	5	5	0
Average	**4.62**	**4.62**	**5.31**	**2.92**	**6.54**	**3.31**	**5.00**	**2.38**

Table 4: Evaluation of the studied features. In the left column, the perception of it is stated with 0 for a negative impression, 5 stands for an abstention, and 10 for a positive impression. In the right column, the impact of the feature on the difficulty is rated with 0 to 10 points.

Feature	Projectile Hits	Enemy Destroyed	Doublejumps	Jump Success	Overheat
Music Speed[1)]	−1.15	−1.08	0.38	−2.62	0.08
Sound Hollow	2.62	0.23	−1.15	0.62	0.15
Background Layer[2)]	−1.00	−1.85	−0.15	−3.69	0.08
Object Layer[3)]	−0.62	0.38	−0.23	2.15	0.15
Global Feedback[4)]	1.23	1.38	1.00	3.08	0.08
Terrain Height[5)]	−0.23	−0.38	0.31	1.92	0.00
Eraser Speed	−2.69	−1.38	−1.46	−2.69	−0.15
Projectile Speed[6)]	−0.23	1.31	1.85	0.69	−0.15
1) & 2)	−0.77	1.00	1.54	2.54	−0.08
3) & 5)	0.00	1.31	3.85	6.92	0.00
4) & 6)	0.15	1.15	0.31	−0.15	0.00
Feature	Trap Damage	Traps Activated	Trap Hits	Trap Destroyed	Item Hits
Music Speed[1)]	−0.23	−1.38	0.31	0.31	−0.08
Sound Hollow	−0.08	0.08	0.62	0.62	1.15
Background Layer[2)]	−0.46	−0.46	−0.46	−0.46	−1.31
Object Layer[3)]	0.15	1.15	0.15	0.15	1.69
Global Feedback[4)]	0.23	0.46	0.38	0.38	0.31
Terrain Height[5)]	−0.08	0.46	−0.69	−0.69	0.85
Eraser Speed	0.31	0.23	−0.38	−0.38	−0.46
Projectile Speed[6)]	0.54	1.46	−0.08	−0.08	−1.46
1) & 2)	−0.23	0.23	−0.08	−0.08	0.62
3) & 5)	0.77	3.23	−0.54	−0.54	0.92
4) & 6)	0.54	0.15	0.23	0.23	0.00
Feature	Item Health	Item Power	Item Time	Time Needed	Health
Music Speed[1)]	−0.38	−0.38	0.00	−176.69	5.60
Sound Hollow	0.15	−0.08	0.31	−123.00	30.64
Background Layer[2)]	−0.92	−0.85	0.54	−354.15	−6.14
Object Layer[3)]	−0.08	0.08	0.31	54.31	−6.27
Global Feedback[4)]	−0.15	0.46	0.38	397.38	−3.25
Terrain Height[5)]	−0.08	0.15	0.85	113.08	−4.07
Eraser Speed	−0.92	0.92	−0.92	−211.69	−4.01
Projectile Speed[6)]	0.15	0.92	−0.08	64.08	1.32
1) & 2)	−0.08	−0.38	0.69	268.46	−2.68
3) & 5)	1.38	0.69	0.00	786.38	−13.03
4) & 6)	−0.77	0.69	0.15	−47.15	1.81

Table 5: Changes between low feature values and high feature values for each segment.

Feature	Projectile Hits	Item Hits	Enemy Destroyed	Doublejumps	Jump Success
Music Speed + Background Layer	−2.15	−1.38	−2.92	0.23	−6.31
Music Speed & Background Layer	−0.77	0.62	1.00	1.54	2.54
Terrain Height + Object Layer	−0.85	2.54	0.00	0.08	4.08
Terrain Height & Object Layer	0.00	0.92	1.31	3.85	6.92
Global Feedback + Projectile Speed	1.00	−1.15	2.69	2.85	3.77
Global Feedback & Projectile Speed	0.15	0.00	1.15	0.31	−0.15
Feature	Overheat	Trap Hits	Trap Activated	Trap Destroyed	Trap Damage
Music Speed + Background Layer	0.15	−0.15	−1.85	−0.15	−0.69
Music Speed & Background Layer	−0.08	−0.08	0.23	−0.08	−0.23
Terrain Height + Object Layer	0.15	−0.54	1.62	−0.54	0.08
Terrain Height & Object Layer	0.00	−0.54	3.23	−0.54	0.77
Global Feedback + Projectile Speed	−0.08	0.31	1.92	0.31	0.77
Global Feedback & Projectile Speed	0.00	0.23	0.15	0.23	0.54
Feature	Item Health	Item Power	Item Time	Time Needed	Health
Music Speed + Background Layer	−1.31	−1.23	0.54	−530.85	−0.53
Music Speed & Background Layer	−0.08	−0.38	0.69	268.46	−2.68
Terrain Height + Object Layer	−0.15	0.23	1.15	167.38	−10.34
Terrain Height & Object Layer	1.38	0.69	0.00	786.38	−13.03
Global Feedback + Projectile Speed	0.00	1.38	0.31	461.46	−1.94
Global Feedback & Projectile Speed	−0.77	0.69	0.15	−47.15	1.81

Table 6: Magnitude of the changes between the summed single feature values (+) and the combined pairing values (&).

Experiment C Data

Evaluation Data

	1)	2)	3)	4)	5)
Player 1	5	5	5	0	0
Player 2	10	10	8	8	0
Player 3	10	5	0	0	5
Player 4	5	5	5	0	7
Player 5	3	3	6	2	8
Player 6	6	7	6	1	1
Player 7	7	5	5	0	0
Player 8	9	7	3	0	0
Player 9	7	3	7	3	5
Player 10	10	0	10	6	4
Player 11	7	0	7	2	5
Average	**7.18**	**4.55**	**5.64**	**2.00**	**3.18**

1) You were able to comprehend the course of the difficulty.
2) You felt supported while playing.
3) You felt limited while playing.
4) There was one or more level that overwhelmed you.
5) Some changes of the difficulty were too abrupt.

Table 7: Results of the Likert Scale.

	Gameplay	Impact	Graphics	Impact	Sound	Impact
Player 1	5	0	6	0	5	0
Player 2	8	8	5	0	5	0
Player 3	7	6	5	3	5	0
Player 4	10	9	5	0	5	7
Player 5	7	5	6	2	4	1
Player 6	7	5	5	1	5	0
Player 7	7	5	6	5	5	0
Player 8	7	8	3	5	5	5
Player 9	7	3	5	0	7	3
Player 10	1	1	5	0	4	2
Player 11	7	7	5	2	5	0
Player 12	6	8	5	0	3	5
Player 13	6	6	5	0	6	6
Player 14	6	5	5	0	4	0
Average	**6.54**	**5.46**	**5.08**	**1.38**	**4.92**	**2.23**
Std. Deviation	1.88	2.56	0.70	1.79	0.91	2.52

Table 8: Survey results showing the impression and impact of the gameplay, graphics, and sound features. An impression of 0 means negative, 5 means neutral and 10 means positive. The impact is rated with 0 to 10 points.

	Gameplay Impact	Graphics Impact	Sound Impact
Player 1	5	2	0
Player 2	10	1	10
Player 3	7	4	0
Player 4	10	2	6
Player 5	6	7	4
Player 6	5	2	0
Player 7	8	6	0
Player 8	9	5	1
Player 9	5	4	5
Player 10	0	1	5
Player 11	9	4	0
Player 12	5	6	4
Player 13	5	7	6
Player 14	5	0	1
Average	**6.46**	**3.92**	**3.15**
Std. Deviation	2.61	2.26	3.05

Table 9: Results of the evaluation of the overall impact of gameplay, graphics, and sound modifications with 0 to 10 points.

Data of the Novice, the Moderate, and the Advanced Player

	Shots	Accuracy	Jumps	Projectile Damage	Trap Dmg.	Enemy Destroyed	Trap Destr.	Fall Death	Shot Death
Level 1	27	1.00	22	6	2	3	1	0	0
Level 2	88	0.98	57	16	6	14	2	1	2
Level 3	112	0.53	42	9	3	10	1	4	0
Level 4	72	0.49	36	9	1	5	2	1	0
Level 5	44	0.59	37	8	4	3	0	3	0
Level 6	24	0.96	36	13	3	3	0	0	1
Level 7	38	0.11	36	11	2	0	0	6	0
Level 8	187	0.13	31	6	1	3	0	1	0
Level 9	76	0.13	18	5	1	0	0	0	0
Level 10	90	0.33	23	3	1	2	0	4	0
Level 11	3	1.00	43	12	1	0	0	1	0
Level 12	40	0.20	26	9	0	2	0	1	0
Level 13	81	0.41	22	4	0	7	1	0	0
Level 14	94	0.43	49	9	3	6	2	8	0
Level 15	78	0.44	43	4	3	4	0	2	0
Level 16	82	0.46	47	7	1	5	0	3	0
Level 17	17	1.00	21	4	0	4	1	0	0
Level 18	76	0.47	87	26	4	5	1	2	2
Level 19	84	0.71	87	18	8	12	0	2	3
Level 20	29	0.69	22	4	1	4	1	0	0
Average	**67.10**	**0.46**	**39.25**	**9.15**	**2.25**	**4.60**	**0.60**	**1.95**	**0.40**

Table 10: Endless Mode: *Novice Player* Monitoring Values

	Enemy Health	Enemy Speed	Enemy Size	Enemy Rate	Enemy Shots
Level 1	0.30	0.40	0.40	0.50	0.30
Level 2	0.30	0.45	0.46	0.60	0.26
Level 3	0.17	0.37	0.41	0.51	0.04
Level 4	0.12	0.36	0.40	0.55	0.07
Level 5	0.00	0.26	0.32	0.52	0.00
Level 6	0.00	0.30	0.36	0.59	0.10
Level 7	0.00	0.13	0.23	0.45	0.00
Level 8	0.00	0.00	0.10	0.44	0.03
Level 9	0.00	0.00	0.01	0.47	0.13
Level 10	0.00	0.00	0.00	0.30	0.00
Level 11	0.00	0.00	0.00	0.34	0.03
Level 12	0.00	0.05	0.06	0.44	0.06
Level 13	0.00	0.00	0.00	0.48	0.16
Level 14	0.00	0.00	0.00	0.22	0.00
Level 15	0.00	0.00	0.00	0.21	0.00
Level 16	0.00	0.00	0.00	0.18	0.00
Level 17	0.00	0.00	0.00	0.25	0.10
Level 18	0.01	0.01	0.03	0.31	0.00
Level 19	0.00	0.00	0.00	0.28	0.00
Level 20	0.05	0.05	0.05	0.33	0.00
Average	**0.07**	**0.19**	**0.23**	**0.48**	**0.08**

Table 11: Endless Mode: *Novice Player* Enemy Feature Values

	Background Layer	Foreground Layer	Global Feedback	Local Feedback	Object Layer
Level 1	0.10	0.00	0.30	0.40	0.30
Level 2	0.25	0.10	0.39	0.33	0.33
Level 3	0.38	0.12	0.37	0.15	0.28
Level 4	0.39	0.17	0.46	0.00	0.25
Level 5	0.46	0.23	0.49	0.01	0.16
Level 6	0.53	0.30	0.59	0.00	0.13
Level 7	0.67	0.27	0.53	0.00	0.00
Level 8	0.47	0.24	0.59	0.00	0.00
Level 9	0.49	0.35	0.72	0.00	0.00
Level 10	0.52	0.41	0.74	0.00	0.00
Level 11	0.47	0.46	0.88	0.00	0.00
Level 12	0.61	0.52	0.93	0.00	0.00
Level 13	0.63	0.61	1.00	0.00	0.00
Level 14	0.70	0.64	0.83	0.00	0.00
Level 15	0.62	0.58	0.88	0.00	0.00
Level 16	0.69	0.65	0.95	0.00	0.00
Level 17	0.73	0.72	1.00	0.00	0.00
Level 18	0.86	0.83	0.96	0.07	0.05
Level 19	0.93	0.77	0.95	0.00	0.00
Level 20	1.00	0.82	1.00	0.00	0.05
Average	**0.45**	**0.26**	**0.58**	**0.07**	**0.12**

Table 12: Endless Mode: *Novice Player* Graphics Feature Values

	SFX Volume	SFX Bluntness	Music Volume	Music Speed	Music Intensity	Music Feedback
Level 1	0.20	0.40	0.20	0.20	0.50	0.40
Level 2	0.33	0.40	0.32	0.34	0.65	0.41
Level 3	0.42	0.18	0.37	0.45	0.78	0.24
Level 4	0.48	0.11	0.36	0.51	0.79	0.22
Level 5	0.52	0.08	0.38	0.63	0.86	0.18
Level 6	0.59	0.05	0.45	0.70	0.93	0.15
Level 7	0.64	0.00	0.46	0.81	1.00	0.00
Level 8	0.66	0.00	0.24	0.83	0.80	0.00
Level 9	0.69	0.05	0.26	0.97	0.82	0.02
Level 10	0.66	0.00	0.19	0.97	0.85	0.00
Level 11	0.71	0.01	0.13	1.00	0.80	0.01
Level 12	0.85	0.00	0.24	0.99	0.94	0.00
Level 13	0.88	0.00	0.26	1.00	0.96	0.00
Level 14	0.84	0.00	0.16	0.94	1.00	0.00
Level 15	0.86	0.00	0.04	0.96	0.92	0.00
Level 16	0.92	0.00	0.06	0.98	0.99	0.00
Level 17	0.99	0.06	0.10	1.00	1.00	0.03
Level 18	0.98	0.08	0.20	0.99	1.00	0.07
Level 19	0.97	0.06	0.23	0.99	1.00	0.00
Level 20	1.00	0.12	0.30	1.00	1.00	0.00
Average	**0.56**	**0.11**	**0.30**	**0.70**	**0.81**	**0.14**

Table 13: Endless Mode: *Novice Player* Sound Feature Values

	Shots	Accuracy	Jumps	Projectile Damage	Trap Dmg.	Enemy Destroyed	Trap Destr.	Fall Death	Shot Death
Level 1	31	1.00	23	5	3	6	1	0	0
Level 2	71	0.56	15	6	4	6	1	0	0
Level 3	175	0.70	60	18	7	20	1	0	4
Level 4	193	0.95	53	14	5	31	1	2	1
Level 5	68	0.97	19	6	2	12	0	0	0
Level 6	88	0.76	22	5	1	11	1	1	0
Level 7	380	0.97	121	45	3	63	4	2	5
Level 8	142	0.94	36	9	2	26	1	2	0
Level 9	135	0.90	42	13	1	21	2	1	0
Level 10	107	0.90	33	10	2	12	1	1	0
Level 11	81	0.88	23	8	0	12	1	0	0
Level 12	393	0.93	117	39	4	62	3	8	0
Level 13	67	1.03	20	6	1	14	0	0	0
Level 14	83	0.98	23	8	0	11	0	0	0
Level 15	426	0.92	122	39	4	66	5	8	1
Level 16	150	0.94	32	14	1	25	0	1	0
Level 17	68	0.93	19	7	1	10	0	0	0
Level 18	209	0.97	62	22	2	31	1	4	0
Level 19	80	0.98	18	8	0	12	0	0	0
Level 20	106	0.91	29	13	0	13	0	1	0
Average	**152.65**	**0.92**	**44.45**	**14.75**	**2.15**	**23.20**	**1.15**	**1.55**	**0.55**

Table 14: Endless Mode: *Moderate Player* Monitoring Values

	Enemy Health	Enemy Speed	Enemy Size	Enemy Rate	Enemy Shots
Level 1	0.30	0.40	0.40	0.50	0.30
Level 2	0.41	0.50	0.50	0.64	0.40
Level 3	0.38	0.45	0.46	0.63	0.36
Level 4	0.37	0.45	0.47	0.59	0.15
Level 5	0.48	0.54	0.56	0.70	0.24
Level 6	0.55	0.58	0.61	0.78	0.27
Level 7	0.49	0.53	0.58	0.77	0.09
Level 8	0.47	0.54	0.60	0.69	0.00
Level 9	0.54	0.59	0.66	0.79	0.03
Level 10	0.60	0.64	0.71	0.87	0.06
Level 11	0.67	0.73	0.80	1.00	0.16
Level 12	0.50	0.57	0.68	0.76	0.00
Level 13	0.57	0.66	0.78	0.89	0.10
Level 14	0.71	0.75	0.87	1.00	0.20
Level 15	0.52	0.60	0.76	0.76	0.00
Level 16	0.56	0.66	0.82	0.84	0.02
Level 17	0.67	0.75	0.91	0.97	0.12
Level 18	0.62	0.70	0.88	0.87	0.00
Level 19	0.69	0.79	0.98	1.00	0.10
Level 20	0.74	0.84	0.96	0.95	0.13
Average	**0.48**	**0.54**	**0.59**	**0.73**	**0.17**

Table 15: Endless Mode: *Moderate Player* Enemy Feature Values

	Background Layer	Foreground Layer	Global Feedback	Local Feedback	Object Layer
Level 1	0.10	0.00	0.30	0.40	0.30
Level 2	0.24	0.13	0.43	0.45	0.41
Level 3	0.32	0.23	0.50	0.45	0.42
Level 4	0.41	0.26	0.52	0.32	0.44
Level 5	0.50	0.33	0.59	0.31	0.55
Level 6	0.63	0.44	0.69	0.32	0.64
Level 7	0.72	0.52	0.76	0.24	0.64
Level 8	0.82	0.46	0.68	0.24	0.67
Level 9	0.91	0.53	0.77	0.25	0.75
Level 10	1.00	0.58	0.82	0.35	0.83
Level 11	1.00	0.67	0.91	0.33	0.90
Level 12	1.00	0.70	0.88	0.28	0.84
Level 13	1.00	0.50	0.85	0.28	0.91
Level 14	1.00	0.62	0.97	0.31	1.00
Level 15	1.00	0.65	0.88	0.17	0.84
Level 16	1.00	0.44	0.78	0.22	0.89
Level 17	1.00	0.51	0.85	0.24	1.00
Level 18	1.00	0.58	0.89	0.16	0.91
Level 19	1.00	0.48	0.92	0.14	0.98
Level 20	1.00	0.57	0.98	0.10	0.97
Average	**0.64**	**0.41**	**0.65**	**0.33**	**0.62**

Table 16: Endless Mode: *Moderate Player* Graphics Feature Values

	SFX Volume	SFX Bluntness	Music Volume	Music Speed	Music Intensity	Music Feedback
Level 1	0.20	0.40	0.20	0.20	0.50	0.40
Level 2	0.34	0.47	0.34	0.35	0.64	0.50
Level 3	0.39	0.50	0.38	0.49	0.72	0.51
Level 4	0.46	0.51	0.44	0.63	0.81	0.42
Level 5	0.59	0.57	0.53	0.73	0.90	0.49
Level 6	0.71	0.60	0.64	0.87	1.00	0.57
Level 7	0.77	0.60	0.67	0.98	1.00	0.55
Level 8	0.87	0.61	0.72	0.99	1.00	0.44
Level 9	0.99	0.67	0.80	1.00	1.00	0.49
Level 10	0.99	0.76	0.89	0.99	1.00	0.53
Level 11	1.00	0.79	1.00	1.00	1.00	0.59
Level 12	0.92	0.71	0.88	0.96	1.00	0.48
Level 13	1.00	0.78	0.89	0.99	1.00	0.45
Level 14	1.00	0.92	1.00	1.00	1.00	0.58
Level 15	0.92	0.84	0.88	0.96	1.00	0.45
Level 16	0.99	0.83	0.87	0.99	1.00	0.38
Level 17	1.00	0.90	0.99	1.00	1.00	0.45
Level 18	0.96	0.89	0.93	0.98	1.00	0.43
Level 19	1.00	0.96	0.99	1.00	1.00	0.37
Level 20	0.98	0.97	0.98	0.99	1.00	0.42
Average	**0.69**	**0.60**	**0.62**	**0.77**	**0.88**	**0.50**

Table 17: Endless Mode: *Moderate Player* Sound Feature Values

	Shots	Accuracy	Jumps	Projectile Damage	Trap Dmg.	Enemy Destroyed	Trap Destr.	Fall Death	Shot Death
Level 1	61	0.97	23	2	0	12	5	0	0
Level 2	88	0.93	22	4	1	15	6	0	0
Level 3	88	0.99	22	6	1	15	4	0	0
Level 4	201	0.96	59	15	4	27	1	0	2
Level 5	156	0.88	59	17	5	23	4	3	0
Level 6	180	1.04	46	18	2	30	1	1	1
Level 7	105	0.90	35	13	0	14	3	1	0
Level 8	238	0.92	81	23	6	32	4	0	3
Level 9	308	0.94	102	32	7	47	1	3	2
Level 10	68	0.99	20	7	1	11	1	0	0
Level 11	366	0.96	120	34	4	60	5	5	1
Level 12	281	0.96	77	20	4	48	3	3	0
Level 13	64	0.98	24	10	0	7	1	0	0
Level 14	321	0.92	101	35	5	48	1	3	1
Level 15	50	0.88	23	8	1	7	1	0	0
Level 16	268	0.93	78	26	2	45	3	2	1
Level 17	112	1.04	50	17	1	18	1	1	0
Level 18	87	0.95	44	17	0	12	0	1	0
Level 19	285	0.96	117	44	7	39	0	3	3
Level 20	64	0.94	25	7	1	12	1	0	0
Average	**169.55**	**0.95**	**56.40**	**17.75**	**2.60**	**26.10**	**2.30**	**1.30**	**0.70**

Table 18: Endless Mode: *Advanced Player* Monitoring Values

	Enemy Health	Enemy Speed	Enemy Size	Enemy Rate	Enemy Shots
Level 1	0.30	0.40	0.40	0.50	0.30
Level 2	0.44	0.49	0.49	0.64	0.40
Level 3	0.58	0.59	0.59	0.78	0.50
Level 4	0.69	0.66	0.66	0.90	0.53
Level 5	0.65	0.64	0.66	0.87	0.38
Level 6	0.68	0.69	0.72	0.96	0.41
Level 7	0.73	0.74	0.78	0.96	0.42
Level 8	0.77	0.81	0.85	0.97	0.45
Level 9	0.71	0.78	0.84	0.88	0.16
Level 10	0.78	0.87	0.93	0.97	0.22
Level 11	0.67	0.78	0.85	0.82	0.00
Level 12	0.66	0.79	0.87	0.85	0.00
Level 13	0.78	0.88	0.96	0.98	0.10
Level 14	0.76	0.90	0.94	0.92	0.00
Level 15	0.85	0.99	1.00	1.00	0.09
Level 16	0.85	0.94	0.95	0.94	0.05
Level 17	0.93	0.97	0.98	0.97	0.07
Level 18	0.96	0.97	0.97	0.97	0.10
Level 19	0.87	0.89	0.91	0.89	0.00
Level 20	0.94	0.98	1.00	0.96	0.00
Average	**0.64**	**0.69**	**0.72**	**0.84**	**0.31**

Table 19: Endless Mode: *Advanced Player* Enemy Feature Values

	Background Layer	Foreground Layer	Global Feedback	Local Feedback	Object Layer
Level 1	0.10	0.00	0.30	0.40	0.30
Level 2	0.24	0.15	0.37	0.55	0.44
Level 3	0.38	0.29	0.41	0.70	0.58
Level 4	0.53	0.40	0.46	0.84	0.71
Level 5	0.67	0.43	0.48	0.75	0.73
Level 6	0.74	0.44	0.48	0.69	0.77
Level 7	0.87	0.47	0.51	0.65	0.85
Level 8	1.00	0.53	0.56	0.66	0.90
Level 9	1.00	0.52	0.47	0.62	0.89
Level 10	1.00	0.49	0.50	0.59	0.97
Level 11	1.00	0.54	0.51	0.57	0.88
Level 12	1.00	0.34	0.38	0.55	0.91
Level 13	1.00	0.33	0.39	0.61	1.00
Level 14	1.00	0.40	0.45	0.60	0.95
Level 15	1.00	0.37	0.47	0.57	1.00
Level 16	1.00	0.46	0.55	0.58	0.96
Level 17	1.00	0.42	0.50	0.62	0.98
Level 18	1.00	0.46	0.54	0.62	0.98
Level 19	1.00	0.48	0.55	0.48	0.93
Level 20	1.00	0.45	0.58	0.39	1.00
Average	**0.71**	**0.38**	**0.45**	**0.63**	**0.74**

Table 20: Endless Mode: *Advanced Player* Graphics Feature Values

	SFX Volume	SFX Bluntness	Music Volume	Music Speed	Music Intensity	Music Feedback
Level 1	0.20	0.40	0.20	0.20	0.50	0.40
Level 2	0.34	0.49	0.34	0.35	0.64	0.55
Level 3	0.48	0.64	0.48	0.50	0.78	0.69
Level 4	0.62	0.71	0.61	0.65	0.93	0.79
Level 5	0.72	0.71	0.70	0.78	1.00	0.75
Level 6	0.84	0.76	0.75	0.84	1.00	0.70
Level 7	0.97	0.77	0.86	0.97	1.00	0.70
Level 8	0.99	0.82	0.97	1.00	1.00	0.73
Level 9	0.96	0.78	0.94	0.98	1.00	0.60
Level 10	1.00	0.86	1.00	1.00	1.00	0.57
Level 11	0.94	0.81	0.91	0.97	1.00	0.51
Level 12	0.97	0.84	0.89	0.99	1.00	0.40
Level 13	1.00	0.98	0.96	1.00	1.00	0.43
Level 14	0.97	0.95	0.96	0.99	1.00	0.43
Level 15	1.00	1.00	1.00	1.00	1.00	0.40
Level 16	0.98	0.96	0.97	0.99	1.00	0.42
Level 17	0.99	0.98	0.99	1.00	1.00	0.40
Level 18	0.99	0.98	0.98	0.99	1.00	0.41
Level 19	0.96	0.93	0.95	0.98	1.00	0.30
Level 20	1.00	0.96	1.00	1.00	1.00	0.20
Average	**0.75**	**0.72**	**0.72**	**0.77**	**0.90**	**0.62**

Table 21: Endless Mode: *Advanced Player* Sound Feature Values

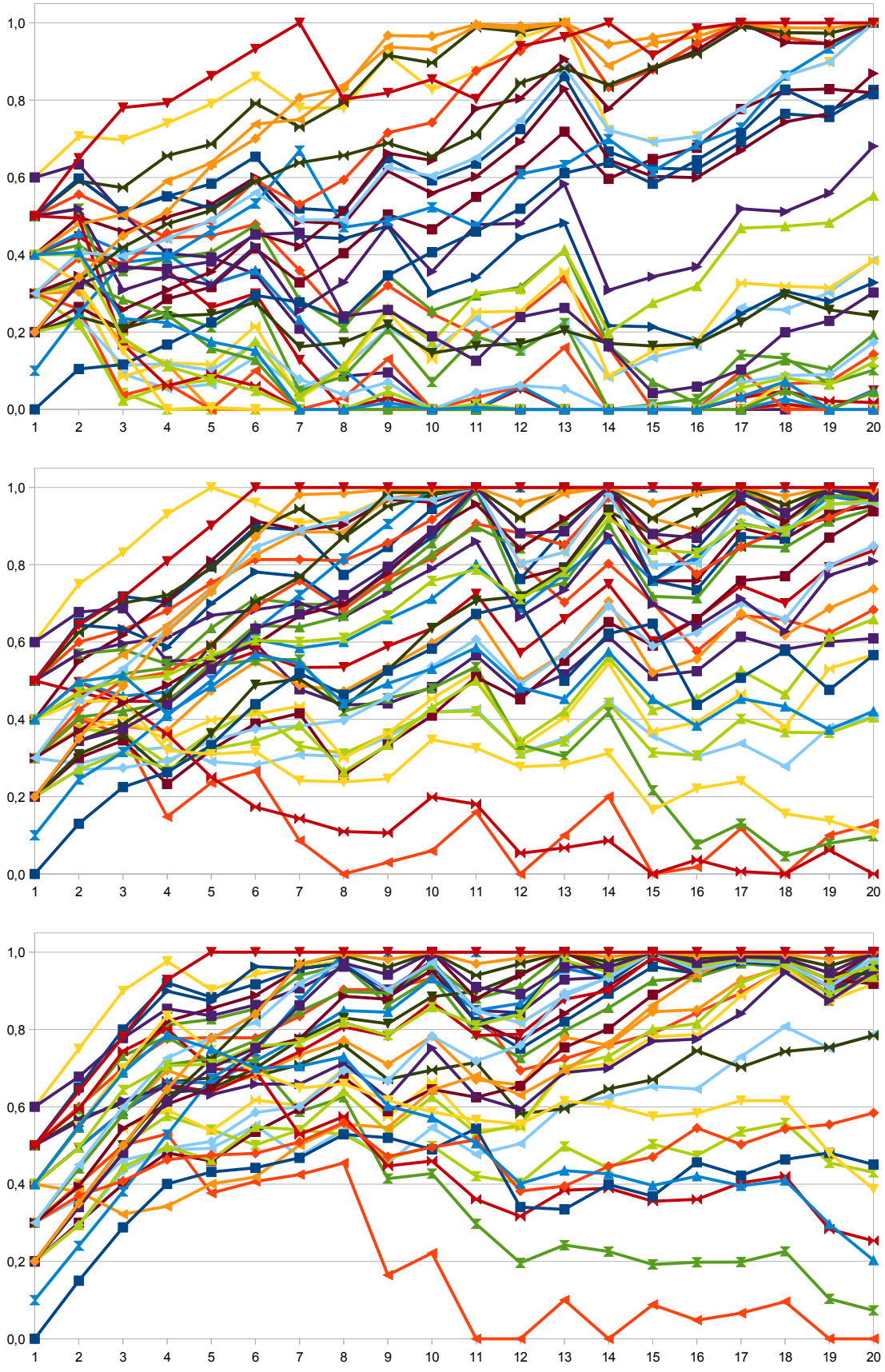

Figure 1: An illustration of the feature evolution for the novice (top), moderate (middle), and advanced player (bottom) over 20 levels.

Statement of Authorship

I, Constantin Graf, confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g. ideas, equations, figures, text, tables, programs) are properly acknowledged at the point of their use. A full list of the references employed has been included.

Magdeburg, August 13, 2012

Constantin Graf