Multi-dimensional Player Skill Progression Modelling for Procedural Content Generation

Francisco Bicho
INESC-ID & Instituto Superior Técnico
Lisbon, Portugal
francisco.nina@tecnico.ulisboa.pt

Carlos Martinho
INESC-ID & Instituto Superior Técnico
Lisbon, Portugal
carlos.martinho@tecnico.ulisboa.pt

ABSTRACT

Procedural Content Generation (PCG), i.e. how game content can be created algorithmically, is an increasingly important area and currently one of the most active topics within the software games industry and game research. One of the crucial aspects of PCG is the capacity to maintain the player engaged and in flow. In this work, we explore how *player skill progression* could be used by PCG to create more appropriate challenges for each player and propose a model for content adaptation that takes this concept as its core feature. Our approach introduces the player as an active element in the adaptation process and assumes both the player and the game should have an equal and active role in this process. Our adaptation explores how modelling the evolution of *multiple dimensions* of a same challenge while the game is played helps creating a better game experience for the player. To evaluate our approach, we present a novel validation process embedded in the game itself, with the purpose of providing a more direct and seamless way to analyse player preference. The results of the evaluation of our approach in the context of an endless running side-scrolling platformer game revealed that players have consistent and specific preferences regarding how difficulty should evolve over the course of a game, which should be taken into account when designing an engaging game progression.

CCS CONCEPTS

• **Applied computing → Computer games**; • **Human-centered computing → User models**;

KEYWORDS

Procedural Content Generation, Player Adaptation, Player Modelling, Progression Modelling, Player Skill

ACM Reference Format:
Francisco Bicho and Carlos Martinho. 2018. Multi-dimensional Player Skill Progression Modelling for Procedural Content Generation. In *Proceedings of Foundations of Digital Games (FDG'18).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

FDG'18, August 7-10 2018, Malmo, Sweden

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Procedural Content Generation (PCG) [11, 16] is a growing trend in the software games industry and games research. PCG is the programmatic generation of game content using a random or pseudo-random process that results in an unpredictable range of possible game play spaces[1]. PCG can be used to generate engaging and refreshing game worlds providing less predictable gameplay, as well as ensuring more replayability.

Level Design [17] teaches us how game levels should ensure that a player learns the required skills before making the player face greater challenges to allow for smooth game progression. While the current use of algorithms for automatically generating content has many advantages, the content itself is often random and not well suited to the player needs. In particular, it does not consider player skill progression, translating into either a boring gameplay, due to its simplicity, or a frustrating experience, due to an overwhelming difficulty (e.g. the difficulty spikes that break rogue-like experiences). The main issue with this approach is forcing the player to adapt entirely to the game, when it should be both the player and the game to adapt one to another [5].

We argue that, to improve player experience, difficulty should not be discretised into categories and used as a global setting affecting the whole game. Such an approach does not take into consideration that the player could be better at overcoming certain challenges with certain mechanics than others. The adaptation process should be continuously adjusted while the game is being played, providing challenges aligned with the evolution of the distinct skills of the player. It is important to note that players can take advantage of this, exploiting the game. Therefore, the game should not adapt blindly to the player, but instead feature mechanisms encouraging players to evolve. In other words, the game should consistently increase the overall challenge level but in a manner that will encourage the player to push forward.

Two essential dimensions of play in game design are player skill and challenge. As described by Csikszentmihalyi [2], to maintain a person in flow, the play activity must reach a balance between the challenges provided by the game and the skill of the player. If the player's perceived skill is much higher than the perceived difficulty of a challenge, the game becomes boring. On the other hand, if the player's perceived skill is much lower than the perceived difficulty of the challenge, the activity turns into a frustrating experience, provoking anxiety. It is important to understand that a same "overall" skill could mean a different proficiency level at different tasks for different players, and that these different proficiencies will also evolve differently over the course of the game. Treating difficulty as one, overall, dimension can quickly frustrate players. A player

[1] http://pcg.wikidot.com/ (online as of 11/Mar/2018)

could spend most of her time trying to overcome a specific challenge that requires a good grasp of specific mechanics, even though she is very proficient at all the remaining challenges and mechanics offered by the game. As such, we think that each challenge and the mechanics used to overcome it should have their own difficulty taken into account when thinking about game adaptation, rather than having a difficulty that covers the whole set of challenges and all manners of overcoming said challenges.

Experience-Driven Procedural Content Generation [19] explores various approaches for creating models of player experience based on data obtained from the players themselves, which are extremely relevant for assessing the content of online PCG [10] [3] [6]. To our knowledge, however, fine-grained player skill progression modelling is absent of such approaches. We believe player skill progression can play a major role in the process of adapting game content to create an improved experienced tailored to the player. In this work, we model the progression of the player's skills both in terms of how such progression affects overcoming the challenges offered by the game as well as how the player prefers to overcome such challenges using the available mechanics.

This paper is organised as follows: Section 2 presents a brief overview of the related work. Then, section 3 describes the game developed for our case study, section 4 details our skill progression model, and section 5 explains our approach to online player preference validation. Finally, section 6 presents our experimental procedure and reports the results of our study while section 7 summarizes and discusses our main findings.

2 RELATED WORK

This section presents a short overview of the research in procedural content generation, player and experience modelling, as well as progression modelling in games, that had an influence on our work.

2.1 Flow

The term *flow*, coined by psychologist M. Csikszentmihalyi [2], has been widely referenced across a variety of fields, and the study of games is no exception. In positive psychology, flow is a mental state of heightened focus and immersion in activities such as art, play and work, a sense that one's skills are adequate to cope with the challenges at hand. Concentration is so intense that there is no attention left to think about anything irrelevant, self-consciousness disappears momentarily and the sense of time becomes distorted. The two most important dimensions of flow are *challenge* and *skill*.

In our work, the concept of flow is specially important because it supports the idea of creating adaptive game content taken both of these dimensions into consideration. By measuring the player's skill, we can present challenges that closely match this player's skill, and maintain the player in the flow channel.

2.2 Procedural Content Generation

Togelius et al. [15, 16] present a detailed overview of recent research on PCG and discuss its multiple dimensions, such as dungeon generation [14], grammars and L-systems with applications to vegetation [13] and search-based approaches [12]. They provide a taxonomy that structures and highlights the differences and similarities between the approaches.

According to this taxonomy, our work can be categorized as a content generator that is online (analyses player's behaviour and alters the game state on the fly), necessary (the generated content needs to be correct or else the player cannot progress) and adaptive (the content is generated taking player behaviour into account).

2.3 Player Modelling

Yannakakis et al. [18] provide a structured overview of player modelling and present a discussion of the key components of a player's model. Player modelling is defined as the study of computational means for the modelling of player cognitive, behavioural, and affective states which are based on data (or theories) derived from the interaction of a human player with a game. They detail the taxonomy of approaches for constructing a player model, the available types of data for the model's inputs and propose a categorisation of the model's outputs. According to this work, our approach could be classified as a hybrid approach in which we will essentially be using gameplay data to make the game adapt better to the player.

2.4 Experience Modelling

Shaker et al. [9] propose the use of Active Learning for player experience modelling. Active learning refers to a set of methods where the learning algorithm is allowed to select the data to learn from. The selection is based on the learning gain and therefore the main advantage of these methods is that they significantly reduce the amount of data needed for training. A dataset from hundreds of players of Infinite Mario Bros [8] was used to learn models of player experience through an active learning approach.

While the goal of this work is to create models of the player's experience (challenge, frustration), and our work focuses on modeling the player's skill progression, both address the issue of online content generation where the data is collected as needed, with the goal of providing a better gameplay experience to the player.

2.5 Progression Modelling

Aligned with Koster's theory of fun [4], Cook [1] highlights the need to begin establishing a more systematic approach to game design analogous to what transformed alchemy into modern chemistry. Cook describes a way of creating a testable model of game mechanics using the game actions available to the player as entities that can be grouped into skill chains. These skill chains can visually represent how the players learn the utility of each skill in the chain. Cook's work was one of the first works to attempt modelling the player's skill and progression. In our work, we will adapt the content to the player based on her skill progression.

Pereira [7] propose a model for progression modelling in video games using features like mechanics and challenges, alongside with the rate of success of a player in each one of the components, to increase the level of replayability in single-player platform games which consequently increases fun and engagement. Although we share similar features, e.g. the way player's skill is measured, we largely modified the approach to allow for online preference elicitation.

3 CASE STUDY: "GO GO HEXAHEDRON!"

In this section, we describe the components of "Go, Go Hexahedron!", the game developed for our case study[2]. "Go, Go Hexahedron!" is an endless running side-scrolling platform game with a minimalist design, featuring a white cube as the player character. The white cube, controlled with a gamepad or through the keyboard, is continuously "running" towards the right of the screen and its objective is to obtain the most points possible while avoiding every obstacle in the way. Points are mainly obtained by staying alive (the longer, the better) but catching different rewards along the path will also provide rewards. The game was designed to be challenging, has an accelerated rhythm and gives the player the ability to choose between challenges according to personal preferences.

3.1 Interface

The main screen of the game's interface follows a set of standards of the genre to make it easier to understand. The game and its interface components are depicted in Figure 1.

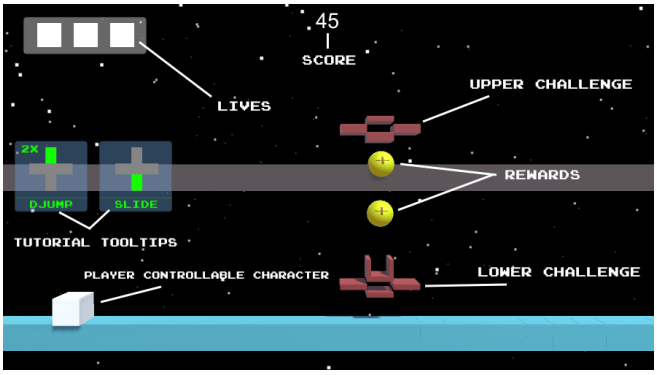


Figure 1: User interface of "Go, Go Hexahedron!"

On the bottom left corner, we have our player controlled character, the white cube. The player starts with 3 lives, represented as white squares on the top left corner. The score is displayed on top and increases by simply staying alive and catching rewards. To stay alive, the player must successfully chain the available mechanics to avoid the incoming challenges that enter through the right side of the screen. Rewards are represented by spheres with a plus sign and can be obtained by running into them while overcoming an obstacle. There are always two challenges (except during the first part of the tutorial), one on the top lane and one on the bottom lane. Each challenge can be overcome using different mechanics.

3.2 Mechanics

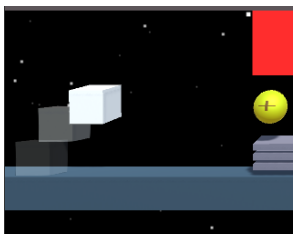
A small set of actions are available to the player in "Go, Go Hexahedron!' to overcome the presented challenges. The four base mechanics, shown in Figure 2, are:

single jump: activated by pressing the up key/button, it executes a low-height, long-distance, long-duration jump.

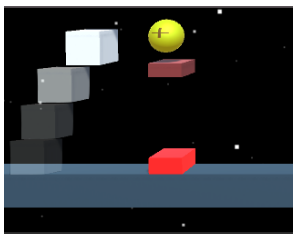
double jump: activated by pressing the up key/button while in the air, it executes a high-height, short-distance, short-duration jump.

dash: activated by pressing the right arrow/button, it is a swift move that can traverse grey obstacles.

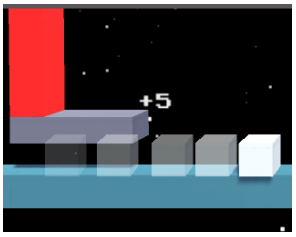
slide: activated by pressing the down arrow/button, it squishes the player cube allowing it to move under certain challenges. The hit-box is adjusted accordingly.



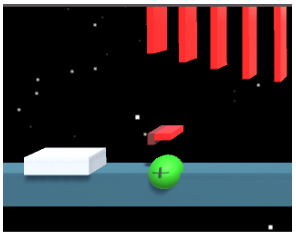
(a) single jump



(b) double jump



(c) dash



(d) slide

Figure 2: Base game mechanics

The base mechanics can be freely and fluidly chained together by the player, e.g. after a "single jump", it is possible to "dash" in mid-air, then "double jump".

In addition to the base mechanics, the player can also use the "**switch**" key/button. Challenges enter through the right side of the screen and are presented to the player in pairs, one challenge on the top lane and another on the bottom lane. The player will always have to overcome the challenge on the bottom lane. While the challenges are still a certain distance away from the player, they can be switched using the switch key/button. This allows the player to choose which challenge to overcome. Every time the player activates the switch command, the two challenges will swap places.

3.3 Challenges

The game offers six different challenges. Each challenge was designed so it could be overcome in different ways and using different mechanics.

As the game becomes more difficult, each challenge will evolve across two dimensions. Each evolution will require more precision from the player. Figure 3 depicts an example of the evolution space of a challenge (the cross challenge). The base challenge is depicted in the bottom left corner image. The yellow axis represents the evolution of the challenge requiring the player to improve at jumping,

[2]The submission is accompanied with a gameplay video of "Go, Go Hexahedron!"

while the blue axis represents the evolution of the challenge requiring the player improve at sliding. Our approach will be to select the best dimension to evolve the challenge based on the player's skill progression.

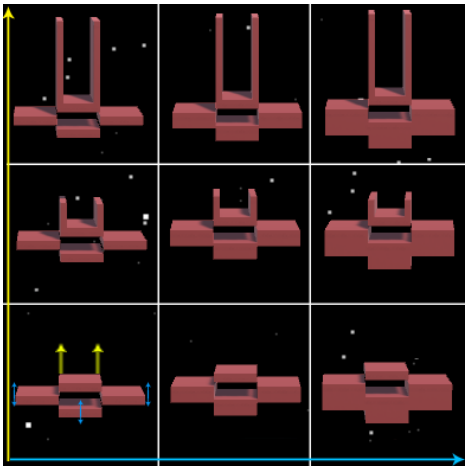


Figure 3: "cross" challenge dimensions difficulty and their respective evolution

3.4 Rewards

Players increase their score by overcoming challenges and staying alive. Additionally, they can collect rewards to increase their score. Rewards are represented by coloured spheres with a plus sign in the game. Rewards were added, on the one hand, to add some diversity to the game but, more importantly, to motivate players to push themselves while the game progressively gets more difficult.

3.5 Tutorial Mode

The tutorial mode was created to initiate players to the game. As such, some things were changed to allow for a smoother introduction. One of the main differences is that every time the player is presented with a challenge, the game enters a "slow motion" mode thus giving the player more time to analyse the presented challenges and how to react accordingly to them. This slow motion mode is held until the player decides to either switch between the top and bottom lane challenges using the switch command, or to confirm the bottom lane challenge by pressing the confirmation button. While in this state, a tool tip shows the players which command they should press to express their choice.

The tutorial introduces players to all challenges in their easiest form, but also gathers some initial player data: the proficiency at overcoming the different challenge types and the starting skill with all the available mechanics. This initialises our player model. The game always requires the player to go through the tutorial at least once.

4 SKILL PROGRESSION MODEL

Our progression model's approach consists in considering difficulty in terms of distinct components of the game, instead of making difficulty influence the game as a whole. In other words, increasing the game's difficulty does not correspond to increasing the difficulty of every aspect of the game. Each player can progress differently in each skill within the game and therefore evolve differently. The ideal is to increase difficulty progressively but in a manner that will maintain the player challenged, not frustrated. For the game to adapt to the player progression, our model needs to capture how well the player is doing. Thus, the model starts by measuring the player skill. Instead of measuring the overall player skill, we started by defining skill in terms of mechanics and associated challenges. Distinguishing skill in terms of the mechanics chosen to overcome a certain challenge allows our model to have finer information regarding skill. For every attempt, the model saves if the player succeeded or failed at the challenge, by how much, and the mechanics used to overcome the challenge. Our model supports both a binary approach, i.e. accounting only for success or failure, as well as a continuous assessment of success, i.e. accounting for how close the player was to the obstacle while overcoming it. While a "true master" could potentially overcome an obstacle while moving very close to it, we observed exactly the opposite when players had their first contact with the game. The easier the challenge was perceived, the further away from the obstacle they moved."

4.1 Overview

Before delving into the details of the model, we provide a general overview. Our model's fundamental objective is to increase the difficulty where the player skill is best. Each time the game presents a new challenge to the player, our model evolves this challenge. In particular, it increases the difficulty in one of the dimensions of the challenge, the dimension the player is more skilled at overcoming. To inform the selection of the dimension to evolve, our model maintains the player skill value for all pairs (mechanics and corresponding challenge) used by the player during the game.

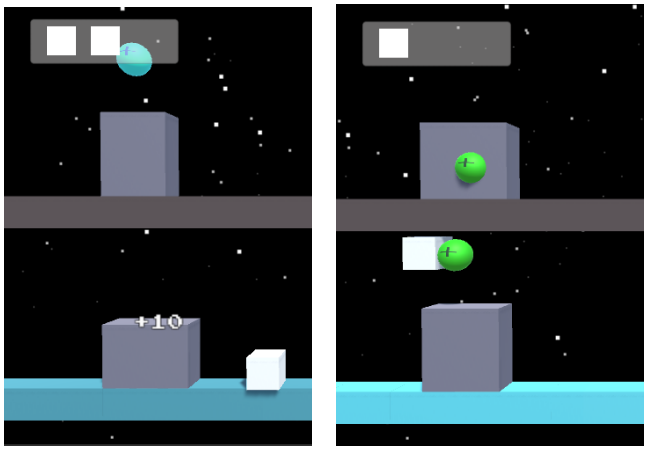
Each challenge can be overcome using certain combinations of mechanical actions (combos) and these combos are associated to the different dimensions of the challenge, e.g. for the dash-box challenge (see Figure 4), the 'dash' combo is associated with an increase in the length of the challenge, while the 'double jump' combo is associated with an increase in the height of the challenge. Since each time a player overcomes or fails a specific challenge the model register this information, it can now search the data using the challenge name and each one of the combos. For every match, the model calculates the skill associated with overcoming the challenge with this combo based on the most recent attempts, then compares these calculations to find the dimension with the best skill. After this step, the model knows exactly which pair has the best skill and also which dimension it is associated to. Therefore, the model is now able to increase the challenge's difficulty in this particular dimension.

The next subsections details the representation of challenges, combos, pairs, how the player skill is calculated, how challenges are evolved, and the reason for including the reward system.

4.2 Challenge Representation

The model was designed to work with challenges that could be surpassed through the use of different mechanical combos. The

player has, at least, two different paths to overcome every challenge. As an example, consider the "dash-box" challenge displayed in Figure 4. In this case, the player can choose between 'dashing' through the grey matter or jumping over it using 'double jump'.



(a) Overcoming the challenge using 'dash' **(b) Overcoming the challenge using 'double jump'**

Figure 4: Two ways to overcome the "dash-box" challenge

Each time a challenge spawns, its difficulty will increase on one of the available dimensions. In "Go, Go Hexahedron!", the increase in difficulty is represented by modifying the size of certain core parts of the challenges. For instance, in the "dash-box" case, the model will increase its difficulty in two dimensions that correspond to two possible ways of overcoming it: for the 'double' jump path, the difficulty increases by increasing the height of the challenge, for the 'dashing' path, by increasing the width of the challenge.

Although, in this case, difficulty increases by changing the size of the challenge, our model is not limited to size, i.e, the requirement is only to have a minimum and maximum value associated with the lower and higher difficulty, and a form to interpolate between the two. As such, the values could represent how fast a slicing blade is, the number of objects in a challenge, etc. In our testbed game, there are 6 challenges with two dimensions for evolving difficulty. There are, however, many more multiple mechanical ways for the player to overcome each challenge.

In "Go, Go Hexahedron!", the value used for interpolation between the two extremes of a certain dimension of a challenge is D, calculated by the following equation:

$$D = log_5(spawn + 1) \tag{1}$$

where *spawn* represents the number of times this particular challenge was spawned in the current game. As such, for the first spawn of a particular challenge, the difficulty for a certain increased dimension would be $log_5(1 + 1)$ resulting in a 43% difficulty. The second spawn would be $log_5(2 + 1)$ resulting a 68% difficulty and so on. After 4 spawns (considering both the challenge and the dimension chosen were the same for the 4 spawns), the dimension will be at its maximum difficulty (100%).

Although our initial approach was to increase the difficulty linearly, the feedback we received from the preliminary evaluations showed that players wanted a faster progression of the challenge difficulty. The original evolution of the challenges was too subtle to be understood by all players. To allow players to more clearly understand the changes taking place in the game, we opted to use a logarithmic curve to increase the challenge difficulty.

4.3 Combos

In "Go, Go Hexahedron!", it is possible to combine multiple mechanics when overcoming any type of challenge. A combo is a combination of mechanical actions. Each mechanical action is represented by a constant state.

The combo detection mechanism works as follows: every time a player uses a mechanical action, the associated state is inserted into the combo. Every combo always starts with the idle state. The idle state refers to 'doing nothing', i.e., the player did not execute any action. The model uses this state to identify when the player did not have enough time to react to a challenge. Thus, the smallest possible combo is made of 1 state, the idle state, while the largest combo is composed of 5 states (the idle state plus the 4 mechanical actions). Our current implementation does not take into account the order of the mechanical actions in the sequence or the number of times a specific action was used in a combo.

The duration in which the mechanical combo is kept in memory is approximately 0.6 seconds (tuned through playtesting), i.e., if a player does not use any other mechanical actions during this time, the combo is considered over and resets to the idle state.

Every challenge is limited by two planes: the left plane (represented as a green line in the debug orthographic view illustrated in Figure 5), and the right plane (represented as a red line in the same view). The left plane represents the beginning of the challenge, while the right plane denotes its end. A challenge is considered to be in progress when the player's cube is between the left plane and the right plane. Success or failure information on the use of the combo in the context of this specific challenge is saved when the player either (1) crosses the left plane first and then the right plane without 'colliding' with the challenge (success), or (2) 'collided' with the challenge after having crossed the left plane, meaning that it failed the challenge. Colliding will mean different thing based on the type of the challenge (e.g. you can traverse a grey obstacle with 'dash' but not a red one).

4.4 Player Skill

To compute player skill, we recorded, for each attempt at a challenge, a success rating, measuring how well the player performed. Our conceptual definition of success rating is the following: 0.0 means the player totally failed the challenge; 1.0 means that the player totally succeeded at the challenge; values in between represent how easy or hard the challenge was to overcome (e.g. based on how close to the obstacle the player landed).

These success ratings, that represent how good (or bad) the player is at surpassing a certain challenge with a set of mechanics, are stored under a key $< challenge, mechanics >$ in a hashtable. The multi-dimensional key is constituted by the array of mechanical actions used (combo) and the identifier of the challenge. The value associated with this key is the list of objects containing the success rating of the most recent attempts at overcoming the challenge using

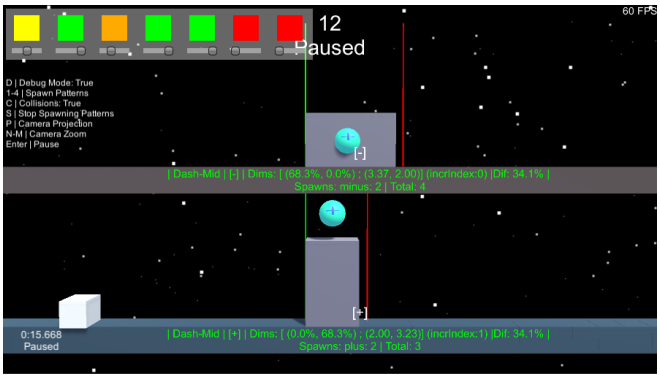


Figure 5: Debug view: While not available to the players, a debug view was developed to monitor our model dynamics in real-time and validate that all the implemented functionality behaved as expected.

this set of mechanics. In other words, every time a player attempts to overcome a challenge, a new pair $< challenge, mechanics >$ is created (or an existing one updated if the key already exists) and the associated list updated with the most recent success rating.

The list associated with each pair $< challenge, mechanics >$ represents a sliding "window" recording the success ratings of the most recent attempts. Every time the player attempts to overcome a challenge using a specific combination of mechanics the new success rating is inserted in the respective sliding "window". In our case study, the sliding "windows" registered the 10 more recent attempts using a FIFO (First-In, First-Out) policy.

4.5 Skill-based Evolution

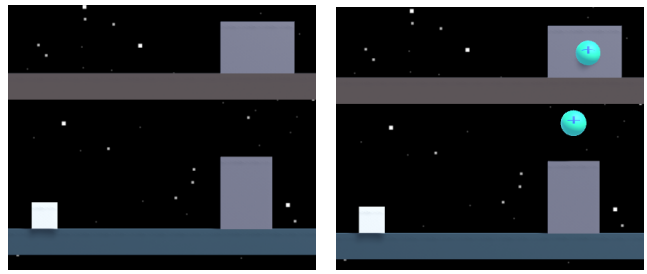
Each challenge is defined by the number of dimensions (in our case study, the challenges had 2 dimensions for evolution), the minimum and maximum values of each dimension, as well as all the combos associated with each dimension. These combos and the challenge name are used as a key to search the hash table. If there is a match, the sliding "window" is returned. Then, the model computes the skill in this dimension. The player's skill measurement in each pair is calculated through a weighted arithmetic mean, the skill S for a given pair being computed by the following expression:

$$S = \frac{\sum_{i=1}^{A} w_i * A_i}{\sum_{j=1}^{A} w_j} \tag{2}$$

where A_i represents each attempt, rated by a numerical value between 0 (total failure) and 1 (total success): $[\,0,\ 0.5\,[\ \cup\ ]\,0.5,\ 1\,]$. Values between 0 and 0.5 represent negative states such as, for example, totally unsuccessful or near-miss attempts. Values in between 0.5 and 1 represent successful attempts like a clutch success or a fully successfully attempt. Each attempt in this "window" of most recent attempts contributes with a different weight w_i to the skill assessment of the gameplay element. In our case study, the chosen weight distribution was linear (specifically: $w_1 = 1$, $w_2 = 2$, .., $w_{10} = 10$) to emphasise that the more recent attempts are more relevant when evaluating player skill than previous attempts.

4.6 Reward System

Rewards were added to pique the player's curiosity through gameplay diversity in an otherwise very simple and repetitive game while encouraging the player to get better as the game progressively becomes more difficult.



(a) If a player is more skilled at 'double jump' than 'dash', he/she may choose to swap the challenges and engage the one originally on top using the 'double jump' mechanics, when the challenge was tuned to promote the 'dash' mechanics.

(b) By using rewards, we disclose more clearly what is happening under the hood and promote player engagement with the adaptation systems, while still giving the player the total freedom to choose how to best overcome the challenge.

Figure 6: Two paths to overcome a dash-box challenge

Rewards were also introduced to mitigate an issue that appeared during our preliminary testing. Figure 6 gives an example of this issue and illustrates how the introduction of rewards helped. The scenario on the left (without rewards) depicts what would typically happen when a player had to choose which challenge to overcome based on the two proposed challenges previously evolved to create meaningful choice while providing with an overall equally challenging obstacle (e.g. in this case, the volume of both obstacles would be roughly the same). As players would get better at using certain mechanics over others, they would use these mechanics consistently but would not push themselves to get better at them, typically choosing the path demanding less effort. In the example of Figure 6, and assuming the player had developed a better skill at 'double jump' than 'dash', the typical course of action would be to switch between the two challenges then use 'double jump' to overcome the challenge originally presented on top, when that particular challenge was evolved to encourage the player to get better at 'dash' mechanics.

While our model is robust to such an effect, this issue does not encourage the player to improve on the skills she had developed and maintain her engagement with the game. As we believe both the game and the player should participate actively in the adaptation process, we introduced rewards to mitigate this effect and better disclose the state of the adaptation process. To motivate the player to get better at the game, rewards were introduced to highlight the path related to the mechanics being evolved by our adaptation process. While not preventing the player from choosing any other alternative way of overcoming an obstacle, it would make the adaptation more visible through the creation of a (very slightly) higher risk, higher reward path. In the example of Figure 6, rewards were placed to disclose the 'path of the dash' and the 'path of the jump'

that influenced the evolution of both challenges. When a reward is present, it is present in both the top and bottom challenges, to ensure equally rewarding choices in terms of the impact on the player score (it is after all, a high-score single player game). An important note is that the values for the rewards provided are quite low when compared to the reward of surviving for a longer period of time in the game.

Again, this is only an issue of communicating more effectively what is happening under the hood to the player to encourage engagement with the game systems: the player is always free to choose how to combine the four available mechanics to overcome each obstacle. And if the player chooses a different path, our model will seamlessly adapt to it.

5 EVALUATION

This section describes the evaluation process used to validate our model. We start by presenting our experimental approach, briefly describe the evaluations that took place during development, then detail the main procedure followed while evaluating the final version of our skill progression model.

5.1 Experimental Approach

The validation process used to test our model consists in giving the player the option to choose between two variations of a challenge each time the player has to overcome this challenge in the game. These two variations, the plus mode of the challenge and the minus mode of the challenge, represent two different evolutions of the same original challenge. The plus mode is the result of evolving the challenge according to the dimension of play the player is the most skilled at that time, while the minus mode is the exact opposite. Each time the minus mode challenge evolves, it does so in the dimension of play the player is the least skilled.

Once evolved, one of the modes is spawned on the top lane while the other is spawned on the bottom lane. The player is then given the choice to choose between the two versions of the challenge, without being explicitly told which variation is the plus mode and which variation is the minus mode. Additionally, the player has no prior knowledge of the existence of this process. The player is able to choose the top lane challenge through the switch mechanics or simply wait for the bottom lane challenge to reach her. We ensure that, in a window of six challenges, the plus mode of the challenge will be placed thrice on the top lane and thrice on the bottom lane, although how this happens in the window is totally random.

With the purpose of verifying our model, we will analyse if players preferred mostly the plus mode (+) of the challenge over the minus mode (−) of the challenge, and if there was in fact a preference for switching to the plus mode challenge over switching to the minus mode challenges, supporting our hypothesis that players express their preference for certain dimensions in detriment of others and that games should take this into account when modelling difficulty progression.

5.2 Preliminary Evaluation

The preliminary evaluation's main objective was to track and solve any usability issues that our game could have. Preliminary playtests were performed in the early stages of the model and game development. The preliminary tests were performed in small groups (4 to 6 people) with multiple iterations throughout 2 to 3 months. The preliminary evaluations did not follow a formal procedure. The consequent changes emerged from informal discussions and interactions with the game.

5.3 Final evaluation

30 participants took part in the final evaluation of our game based on player skill progression. Each participant was asked to perform two tasks: answer a short questionnaire then play our game. The full test was performed autonomously by all participant using a PC or Mac computer with an Internet connection. The web application[3] provided all the information required for any player to participate in the evaluation.

The questionnaire contained exclusively demographics related questions (e.g. age, gender), to help us characterise our sample. The playtest procedure for every participant was to: play the tutorial at least once and then play the normal game, although the players could play the tutorial and the game multiple times if they so desired. Information about the play session, such as how many time and for how long the player played our game were logged on our server. The logged data was exported to a CSV file and imported into IBM SPSS for analysis. The logged data includes the sequence of challenges, the position of the different modes of each challenge, which mode the player attempted to overcome and if a switch command was executed before the attempt.

6 RESULTS

This section presents the analysis of the data collected in the final evaluation. We start by describing our population sample then provide a detailed analysis of the player logs as a way to evaluate it and to conclude whether or not our progression model is valid and viable. The statistical analysis was processed using the IBM SPSS software version 24.

6.1 Sample

The final evaluation was carried out with the participation of 30 persons with ages between 16 and 30 years old (M = 22.97, SD = 3.77, 7 female). 43.3% of the participants reserved time in their schedule to play games (dedicated players [5]); 40% played games only when the opportunity presents itself (casual players [5]), and; 16.7% did not play games. Most participants were familiar with endless runner games (60%), 23.3% were not familiar with endless runner games although familiar with games in general. No statistically significant effect was found when comparing the different demographic groups.

6.2 Data Collected

Players played at least one time the tutorial before playing the normal game. For the tutorial, the time users played ranged between 2.25 and 16.53 minutes (M = 5.50, SD = 3.66). For the normal game, the time played ranged between 2.05 and 20.15 minutes (M = 6.83,

[3]Web application implemented in React and Node.js using Nginx as a web server.

SD = 4.86). From the logged data, we calculated the following ratios: Plus_Ratio_Total (equation 3), Minus_Ratio_Total (equation 4), Plus_Ratio_Total (equation 5) and Plus_Ratio_Total (equation 6). The result of the Shapiro-Wilk test showed that most of the calculated ratios do not follow a normal distribution, as such, we used non-parametric tests to analyse our data.

6.3 Overall Player Preference

In order to know which mode (Plus or Minus) the players preferred, we conducted a Wilcoxon signed-rank test between two ratios: Plus_Ratio_Total and Minus_Ratio_Total. These ratios were calculated using the following formulas:

$$Plus\_Ratio\_Total = \frac{Total\_Plus\_N}{Total\_Plus\_N + Total\_Minus\_N} \quad (3)$$

$$Minus\_Ratio\_Total = \frac{Total\_Minus\_N}{Total\_Plus\_N + Total\_Minus\_N} \quad (4)$$

where Total_Plus_N refers to the total number of times a player attempted the plus mode challenges and Total_Minus_N refers to the total number of times a player chose to do the minus mode challenge. Theses values include the result of switching, e.g., if a player switched a minus challenge for a plus challenge, Total_Plus_N value is increased by one.

Results

The Plus_Ratio_Total presents a median of 0.59 (0.46 to 0.88, IQR = 0.13) whereas the median for the Minus_Ratio_Total is 0.41 (0.13 to 0.54, IQR = 0.13). The Wilcoxon signed-rank test showed that there is a statistically significant change when comparing the player preference between Plus_Ratio_Total and Minus_Ratio_Total ($Z = -4.742$, $p < 0.01$), with a large effect size (r=0.61).

This test suggest that players, in fact, have a statistically significant preference for plus mode challenges throughout the play sessions.

6.4 Switch Usage

The idea of this test is to analyse if players opted to switch more to the plus mode challenge (when a minus mode challenge was presented on the bottom lane) than to the minus mode challenge (when a plus mode challenge was presented on the bottom lane). A Wilcoxon signed-rank test was conducted between two ratios: Switch2plus_Ratio and Switch2minus_Ratio. These ratios were calculated using the following formulas:

$$Switch2plus\_Ratio = \frac{Switch2plus\_Total\_N}{Minus\_Total\_N + Switch2plus\_Total\_N} \quad (5)$$

$$Switch2minus\_Ratio = \frac{Switch2minus\_Total\_N}{Plus\_Total\_N + Switch2minus\_Total\_N} \quad (6)$$

Switch2plus_Total_N refers to the total number of times the player switched from a minus mode challenge to a plus mode challenge in the game. The Minus_Total_N variable refers to the total number of times a player did not switch when a minus was presented (and therefore chose the minus mode without switching). The sum in the denominator of the first equation refers to the total number of times a minus mode challenge was presented on the bottom lane. For the second equation, the denominator refers to the total number of times a plus mode challenge was presented on the bottom lane.

Results

The Switch2plus_Ratio presents a median of 0.37 (0.11 to 0.80, IQR = 0.30) whereas the median for the Switch2minus_Ratio is 0.07 (0.00 to 0.31, IQR = 0.17). A Wilcoxon signed-rank test shows that there is a statistically significant change when comparing the player preference in switching to the plus mode or to the minus mode, i.e., Switch2plus_Ratio and Switch2minus_Ratio ($Z = -4.762$, $p < 0.01$), with a large effect size (r=.61).

The interpretation of this analysis is that when players opted to use the switch, they switched significantly more to the plus mode challenge than to the minus mode challenges.

6.5 Player Preference over Time

For this test, we divided the play sessions into three equal segments (the initial segment, the middle segment and the final segment) based on the number of attempted challenges in each segment. The purpose of dividing the play sessions into segments is to understand whether there are differences in preference between these segments. In particular, we wanted to understand the evolution of these preferences throughout the game, i.e. if the preference were uniform throughout the game or if there were segments in which player preferences were more strongly expressed. We also wanted to understand if the increase in difficulty would motivate a change in the player preference. A Wilcoxon test was conducted for each Plus and Minus ratios corresponding to each segment. These ratios were calculated similarly to equations 3 and 4.

Results

For the initial segment (Plus_Ratio_1 and Minus_Ratio_1), the medians were respectively 0.60 (0.25 to 1.00, IQR = 0.17) (plus ratio) and 0.40 (0.00 to 0.75, IQR = 0.17) (minus ratio). The Wilcoxon signed-rank test revealed a statistically significant change when comparing the player preference between Plus_Ratio_1 and Minus_Ratio_1 ($Z = -2.842$, $p = 0.004$), with a medium effect size ($r = 0.36$).

For the middle segment (Plus_Ratio_2 and Minus_Ratio_2), the medians were respectively 0.62 (0.00 to 1.00, IQR = 0.24) (plus ratio) and 0.38 (0.00 to 1.00, IQR = 0.24) (minus ratio). The Wilcoxon signed-rank test revealed a statistically significant change when comparing the player preference between Plus_Ratio_2 and Minus_Ratio_2 ($Z = -3.123$, $p = 0.002$), with a medium effect size ($r = 0.40$).

For the final segment (Plus_Ratio_3 and Minus_Ratio_3), the medians were respectively 0.62 (0.33 to 1.00, IQR = 0.25) (plus ratio) and 0.38 (0.00 to 0.67, IQR = 0.25) (minus ratio). The Wilcoxon signed-rank test revealed a statistically significant change when comparing the player preference between Plus_Ratio_3 and Minus_Ratio_3 ($Z = -3.776$, $p < 0.01$), with a relatively large effect size ($r = 0.49$).

This tests suggest that players prefer the plus mode over the minus mode from the beginning and continue to do so, over time. The difference between plus mode challenges and minus mode challenges increases over time, when the plus mode becomes even more adapted to the player's skill. The analysis suggests that players

continue to use the preferred mechanic to overcome a challenge even when the difficulty is close to the maximum value and players still have the option of using another mechanic to overcome the challenge in another dimension.

6.6 Switch Usage over Time

This test analyses how much players switched over time while playing the game. A Wilcoxon signed-rank test for Switch2plus_Ratio and Switch2minus_Ratio was conducted for each segment, similarly to section 6.5. The ratios were calculated similarly to ones presented in section 6.4, the only difference being we limit each dataset to one of the three segments (initial, middle, and final).

Results

For the initial segment, the medians were respectively 0.33 (0.00 to 1.00, IQR = 0.36) (Switch2plus_Ratio_1) and 0.00 (0.00 to 1.00, IQR = 0.25) (Switch2minus_Ratio_1). The Wilcoxon signed-rank test revealed a statistically significant change when comparing the player preference in switching to the plus mode over the minus mode for the initial segment (Z = -2.892, p = 0.004), with a medium effect size (r = 0.37).

For the middle segment, the medians were respectively 0.20 (0.00 to 1.00, IQR = 0.50) (Switch2plus_Ratio_2) and 0.00 (0.00 to 0.60, IQR = 0.00) (Switch2minus_Ratio_2). The Wilcoxon signed-rank test revealed a statistically significant change when comparing the player preference in switching to the plus mode over the minus mode for the middle segment (Z = -3.346, p = 0.001), with a medium effect size (r = 0.43). In this segment of the play session, players displayed a tendency for not using the switch as much as the initial segment. Although there was an increase comparing the Switch2plus and Switch2minus ratios with the initial segment.

For the final segment, the medians were respectively 0.45 (0.00 to 1.00, IQR = 0.52) (Switch2plus_Ratio_3) and 0.00 (0.00 to 0.50, IQR = 0.13) (Switch2minus_Ratio_3). The Wilcoxon signed-rank test revealed a statistically significant change when comparing the player preference in switching to the plus mode over the minus mode for the final segment (Z = -4.092, $p < 0.01$), with a large effect size (r = 0.53).

These tests reveal that players opted to switch into the plus mode significantly more than minus mode over time in a consistent manner.

6.7 Comparison between Segments

For this comparison, we conducted a Friedman test to detect differences in preference across multiple test attempts over time. We examined the three different segments for each individual ratio: Plus, Minus, Switch2plus and Switch2minus.

Plus Mode Preference

A Friedman test showed that there was no statistically significant difference between each segment (initial, middle and final) regarding the preference between plus and minus modes, $\chi^2(2) = 1.264$, p = 0.531. Although the values of the mean ranks increase gradually over the segments (Plus_Ratio_1's Mean Rank = 1.85, Plus_Ratio_2's Mean Rank = 2.03, Plus_Ratio_3's Mean Rank = 2.12), the test did not find that this increase was statistically significant.

Minus Mode Preference

A Friedman test showed that there was no statistically significant difference between each segment regarding the preference between plus and minus modes, $\chi^2(2) = 1.264$, p = 0.531. Although the values of the mean ranks decrease gradually over the segments (Minus_Ratio_1's Mean Rank = 2.15, Minus_Ratio_2's Mean Rank = 1.97, Minus_Ratio_3's Mean Rank = 1.88), the test did not find that this decrease was statistically significant. This was expected since minus ratios are the complement to 1 of the plus ratios.

Switching to Plus Mode Preference

A Friedman test showed that there was in fact a statistically significant difference between each segment in preference of switching to plus mode challenges, $\chi^2(2) = 8.072$, p = 0.018.

To find between which segments this difference actually occurs, we conducted post hoc analysis with Wilcoxon signed-rank tests between each segment with a Bonferroni correction applied, resulting in a significance level set at $p < 0.017$. The results showed that there were no statistically significant differences between the initial and middle segment (Z = -0.216, p = 0.829) or initial and final segment (Z = -1.1721, p = 0.085). However, there was a statistically significant increase in preference for plus mode challenges from the middle segment to the final segment (Z = -3.044, p = 0.002).

This analysis suggests that in the initial segment there is an exploratory phase where players experiment different mechanics. In the middle and final segments the analysis conveys that players still continues to use the preferred mechanic to overcome a challenge even when the difficulty is reaching its peak and there is an option of using another mechanic to overcome the challenge in another dimension.

Switching to Minus Mode Preference

A Friedman test showed that there was a statistically significant difference between each segment in preference of switching into minus mode challenges, $\chi^2(2) = 6.633$, p = 0.036.

To find between which segments this difference actually occurs we conducted post hoc analysis with Wilcoxon signed-rank tests between each segment with a Bonferroni correction applied, resulting in a significance level set at $p < 0.017$. The results showed that there were no statistically significant differences in preference of switching into minus mode challenges between the initial and middle segments (Z = -2.262 p = 0.024) or initial and final segments (Z = -1.068 p = 0.286) or even between the middle and final segments (Z = -1.201 p = 0.230).

6.8 Discussion

Our results suggest players prefer significantly and consistently plus mode challenges over minus mode challenges, over time. The analysis suggests the initial segment is an exploratory phase where players experiment with different mechanics. While there is a significant preference for plus mode challenges over minus mode challenges in the initial segments, it is not as strong as in later segments. Players express a stronger preference for plus mode over minus mode in later segments, revealing an increasing preference for plus mode challenges over time and, consequently, a decreasing preference for minus mode challenges. We also verified that players

continue to use their preferred mechanics to overcome challenges, even when the difficulty reaches its peak and they have the option to use other mechanics to overcome the challenge. This supports our hypothesis that players express preference for certain dimensions in detriment of others in the context of a game with a progressive increasing difficulty, and, as such, difficulty should not be modelled as a global setting in games where player adaptation is key.

7 CONCLUSIONS

In this paper, we explored how player skill progression could be used in PCG to create games better adapted to each player, and proposed a model for online content adaptation using player skill progression as a core feature.

To adapt the game to the player's skill progression, the model needs to know how well the player is doing over time at using the different game mechanics to overcome the different challenges proposed by the game. To that end, we propose that player performance could be measured based on the recent performance at using a certain combo of mechanics to overcome a certain challenge, i.e. associating a success rating with all pairs ($< challenge, mechanics >$) used by the player. By measuring the player performance in each one of these dimensions, it is possible to infer which dimension the player is currently better at when confronting a certain challenge and use this information to progressively increase the difficulty in a manner that will encourage the player to improve at the game and promote engagement with its gameplay systems.

We presented "Go, Go Hexahedron!", an endless running side-scrolling platform game with a minimalist design specially developed to serve as a case study for our skill progression model. To evaluate our model, we included a validation process within the game itself that presented different variations of a same base challenge from which the player could choose from. In our case study, this choice was expressed by allowing the player to switch between two different challenges, one aligned with the skills the players is most proficient at and the other more adapted to the skills the player has been the least proficient with, and select which one to attempt. While part of the gameplay itself, this process allowed us to better understand how player preference evolved throughout the game.

Finally, we detailed our experiment with 30 participants and reported the results obtained from the analysis of the data logged during the play sessions. The results suggest there is a significant pattern in player choice, with players consistently preferring overcoming the challenges through the use of the skills they are more proficient at, even when the difficulty is reaching its peak and they have the option of using another set of mechanics to overcome such challenges. These results support our hypothesis that, when adapting content to the player in the context of a game with increasing difficulty (which includes most of the current generation of games), difficulty should not be modelled as a global setting. Knowing the specific preferences of each player based on the progression of their skill at overcoming the various dimensions of play offered by the game could provide invaluable insight for the generation of content adequately adapted to each player.

ACKNOWLEDGMENTS

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013.

REFERENCES

[1] Daniel Cook. 2007. The Chemistry of Game Design. Gamasutra: https://www.gamasutra.com/view/feature/1524/. (July 2007). [Online as of 30/Dec/2017].
[2] Mihaly Csikszentmihalyi and Isabella Selega Csikszentmihalyi. 1988. *Optimal Experience: Psychological Studies of Flow in Consciousness.* Cambridge University Press.
[3] Martin Jennings-Teats, Gillian Smith, and Noah Wardrip-Fruin. 2010. Polymorph: Dynamic difficulty adjustment through level generation. In *Foundations of Digital Games Proceedings of the 2010 Workshop on Procedural Content Generation in Games (FDG'10).* ACM.
[4] Ralf Koster and Will Wright. 2004. *Theory of Fun for Game Design.* Paraglyph Press.
[5] Carlos Martinho, Pedro Santos, and Rui Prada. 2014. *Game Design and Development.* FCA.
[6] Ravi Parekh. 2017. *Staying in the flow using procedural content generation and dynamic difficulty adjustment.* Master's thesis. Worcester Polytechnic Institute.
[7] Pedro Pereira. 2016. *Modelling Progression in Video Games.* Master's thesis. Instituto Superior Técnico, Universidade de Lisboa, Portugal.
[8] Markus Persson. 2008. Infinite Mario Bros. (Super Mario Bros. open-source clone). (2008).
[9] Noor Shaker, Mohamed Abou-Zleikha, and Mohammad Shaker. 2015. Active Learning for Player Modeling. In *Proceedings of the 10th International Conference on the Foundations of Digital Games (FDG 2015).* ACM.
[10] Noor Shaker, Georgios N. Yannakakis, and Julian Togelius. 2011. Experience-Driven Procedural Content Generation. In *Proceedings of Artificial Intelligence and Interactive Digital Entertainment (AIIDE'10).* AAAI, AAAI Press.
[11] Julian Togelius, Emil Kastbjerg, David Schedl, and Georgios N. Yannakakis. 2011. What is procedural content generation?: Mario on the borderline. In *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games.* ACM.
[12] Julian Togelius and Noor Shaker. 2016. The search-based approach. In *Procedural Content Generation in Games: A Textbook and an Overview of Current Research.* Springer, Chapter 2, 17–30.
[13] Julian Togelius, Noor Shaker, and Joris Dormans. 2016. Grammars and L-systems with applications to vegetation and levels. In *Procedural Content Generation in Games: A Textbook and an Overview of Current Research.* Springer, Chapter 5, 73–98.
[14] Julian Togelius, Noor Shaker, Antonios Liapis, Ricardo Lopes, and Rafael Bidarra. 2016. Constructive generation methods for dungeons and levels. In *Procedural Content Generation in Games: A Textbook and an Overview of Current Research.* Springer, Chapter 3, 31–55.
[15] Julian Togelius, Noor Shaker, and Mark J. Nelson. 2016. Introduction. In *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*, Noor Shaker, Julian Togelius, and Mark J. Nelson (Eds.). Springer, Chapter 1, 1–15.
[16] Julian Togelius, Georgios N. Yannakakis, Kenneth O. Stanley, and Cameron Browne. 2011. Search-based procedural content generation: A taxonomy and survey.. In *IEEE Transactions on Computational Intelligence and AI in Games 3.* ACM, 172–186.
[17] Christopher Totten. 2014. *An Architectural Approach to Level Design.* A K Peters/CRC Press.
[18] Georgios N. Yannakakis, Pieter Spronck, Daniele Loiacono, and Elisabeth Andre. 2013. Player Modeling. In *Dagstuhl Seminar on Artificial and Computational Intelligence in Games.* Schloss Dagstuh.
[19] Georgios N. Yannakakis and Julian Togelius. 2011. Experience-Driven Procedural Content Generation. In *IEEE Transactions on Affective Computing.* IEEE, IEEE Press.