Procedurally Generating a Digital Math Game's Levels: Does it Impact Players' In-Game Behavior?

Luiz L. Rodrigues[a,1,*], Jacques Brancher[a]

[a]*Department of Computer Science, Londrina State University, Rodovia Celso Garcia Cid - Pr 445 Km 380, Londrina - PR, Brazil*

Abstract

Developing games can be improved with Procedural Content Generation (PCG), the automatic creation of game contents, such as levels, music, and vegetation. However, few researches have addressed the impacts of PCG on players, especially in terms of games with educational ends. Hence, this understanding is a research area that demands further studies. To expand on this gap, this article presents a study concerning whether PCG for level creation impacts players' in-game behavior, based on interactions with a digital math game. This game features two versions that contain a unique difference: whilst one features human-designed levels, these are procedurally generated on the other version. To compare them, PCG's impact on players' in-game behavior was measured through the total played time, number of played levels, and number of retained players. In this context, the findings demonstrated a significant difference in the number of retained players, which was higher for the PCG version in comparison to the other. In contrast, the other two metrics were insignificantly different between versions. Therefore, game designers and developers can exploit these findings to employ PCG in games, taking advantage of its impacts on development and players, knowing how it is expected to affect players' in-game behavior.

Keywords: Procedural Content Generation, Player Engagement, Player Retention, A/B test, Digital Math Game, Educational Game

1. Introduction

A reliable approach to improve games is to use Procedural Content Generation (PCG) [1], that can be defined as the algorithmic generation of any content

*Corresponding author

Declaration of Interest: none

Email addresses: luiz_rodrigues17@hotmail.com (Luiz L. Rodrigues), jacques@uel.br (Jacques Brancher)

[1]Present address: Institute of Mathematics and Computer Science, University of São Paulo, Av. Trab. São Carlense, 400, São Carlos - SP, Brazil

[2]. However, it is mostly applied on games, such as levels [3, 4], music [5, 6], vegetation [7], buildings [8, 9], race tracks [10], history [11], and puzzles [12, 13], for instance. A complete review of examples can be found in Hendrikx et al. [14], where the authors provide a survey on game contents that can be procedurally generated, along to methods for generating them. In sum, by enabling the automatic creation of a wide range of contents, PCG is an important technique for developing games.

However, PCG's contributions to games go beyond improving the development process, being valuable to players as well. Developing a game is a complex task which commonly requires human effort, time, money, and a multidisciplinary team working together [15, 14]. Creating games' parts through algorithms can improve this process and remedy these problems [1]. Furthermore, players expect technologies to provide them with positive experiences; otherwise, they are unlikely to accept or interact with those [16]. Considering this context, a game with a limited number of contents might fail to promote those experiences. Consequently, players might lose interest and left the game. Algorithms for automatic generation of content also contributes to this vein, being capable of providing players with a pseudo-infinite number of game contents. Thereby, using PCG can contribute to games, not only on their development process, but also to the experience they provide to players.

Nevertheless, identifying the impact of PCG techniques according to players is yet an emerging field of research [17]. To this end, Korn et al. [18] argue that evaluating a single game, with versus without PCG, is the most feasible approach. Additionally, the authors confirm that this is a scantly explored perspective which requires more researches. Based on this context, this article tackles this gap, presenting an empirical study concerning the impacts of Procedural Level Generation (PLG) on players' behavior. Moreover, although PCG's main application is in games [14], there are little researches in terms of educational games that feature algorithms for automatic content generation [19]. Therefore, this study adopted a Digital Math Game (DMG) as its testbed, which is a specific type of educational game.

Hereby, this article's main contribution is to demonstrate the PLG's impact on the behavior of a DMG's players. Consequently, this research's scientific impact is to provide evidence of whether using an algorithm for PLG leads players to behave differently from players who interacted with human-designed levels. Our proposition is that players from both situations will behave indifferently, which is relevant because our experiment is based on a simple and straightforward PLG method that does not produce personalized outputs. Thus, developers and designers can rely on this paper's findings when creating their games. They can take advantage of PCG's benefits for development-time, knowing that it will not affect players' behavior, despite using a simple algorithm.

Lastly, in order to outline this research article, the remaining of this paper is following presented. Section 2 provides a background on PCG and Section 3 reviews related works on the impacts of procedurally generated contents. Sections 4 and 5 describe this study's testbed game and experiment, respectively. Section 6 shows the experiment's results, Section 7 discusses its findings, and,

lastly, Section 8 draws this article's final considerations.

2. Procedural Content Generation

This section presents a background on PCG in three perspectives: contents, algorithms classifications, and procedurally generated outputs evaluation. Mainly, the aim is to show what a specific type of content is consisted of, different forms of using and automatically generating contents, and ways to evaluate procedurally created outputs. At last, a summary of these is presented towards the motivation behind selecting one option or another.

Hendrikx et al. [14] surveyed several papers reporting the use of PCG algorithms to classify game contents that might be procedurally generated, as well as to provide what methods were used to create them. Six game content classes were defined, which are bits, space, systems, scenarios, design, and derived contents. This paper investigates the procedural generation of levels, which fits in the fourth class. Levels are present in basically all games, they are one of the most studied contents on PCG's context, and nearly all games can benefit from them, according to the authors. Additionally, they claim that, following the presented order, contents from the latest classes can be generated from contents of the previous ones. Accordingly, it helps to understand what is a level (game scenario). It is a content, composed of a game space, which is fulfilled with game elements. For instance, an adventure game's level might be an empty map (space) containing several game elements (bits), such as buildings, walls, enemies, and a character. In summary, there are six classes of contents that might be procedurally generated, wherein most of these are composed of simpler ones.

Regardless of what type of content will be generated, there are different characteristics the generation algorithm might feature. Togelius et al. [20] presented a taxonomy of search-based generators, distinguishing them through opposed characteristics. Of high relevance to this article, there are the differences between online and offline systems, as well as constructive and generate-and-test algorithms. A system that uses PCG to create contents while the application is running is considered an online system; otherwise, it is considered an offline system. Online generation allows providing endless gameplays, or even endless game levels, for example. However, it often demands predictable running time and quality. In contrast, offline generation is often used to aid developers in creativity. One example is the Mixed-Initiative approach, wherein the designer is helped by the generation system [21]. In this case, the demands of the online generation are mostly absent since there is a human in the loop, which is able to evaluate the content's quality. Also, running time is less critical, although it is still important in some situations.

Moreover, the difference between constructive and generate-and-test algorithms is that the former creates its outputs in a single step, whereas the later generates and tests multiples results until satisfying a predefined criterion. Constructive algorithms usually run faster but have to feature generation rules that guarantee the feasibility of its generated contents. For instance, it must have

a rule that connects a race track's begin and end if players' goal is to perform two or more laps. Differently, generate-and-test methods are usually computationally more expensive but can achieve more complex results by optimizing its outcomes toward passing the desired test function (e.g. content's quality or difficulty). Hence, its design guarantees contents' feasibility naturally. Thereby, according to the presented possibilities, selecting how the system's use of PCG operates mainly depends on the goal of using it, while the algorithm's type plays a role on the outputs' quality and computational performance.

Once the algorithm is completely set up, evaluating it is important for both academic and commercial ends. Basically, there are three options that might be adopted to analyze the use of PCG, depending on the evaluation's goals. One of them is to assess the algorithm's capacities, which probably is the most common approach (e.g. [22, 23, 3]). This procedure is often performed through the analysis of contents' expressivity [24]. A large number of outputs are generated and, according to the evaluator's selected metrics (e.g. variety), they are visually analyzed. While the former procedure is interesting, it is insufficient to replace user studies [25], which is the second option. Studying players' interaction with the algorithmically generated contents captures another perspective. It allows to identify their opinions through self-reports or to capture their reactions while playing them [26]. The first option is known as Top-down approach, whilst the second is known as Bottom-up [27]. However, neither of them can measure the impact of using PCG on players by itself. To this end, the third option is the most recommended, which expands from the Bottom-up approach: performing an A/B test [18]. This is a scantly used perspective [17], wherein one game, which contains two versions that only differ by using PCG or not, is evaluated according to players (e.g. opinions or behavior). Therefore, whereas analyzing generators through their expressivity is the most used approach, the impact of using procedurally generated content lacks further researches.

In sum, game contents might be distinguished through six classes, where contents of one are used to build up contents of another. In order to automatically generate these, algorithms might be used in running-time to provide endless experiences or on development-time to aid designers and developers. Generated outputs might be evaluated from players' opinions, behavior, or other reactions, as well as through comparisons with contents created by specialists; whereas generators might be analyzed according to the expressive range of their contents. Hence, selecting the generator's characteristics and evaluation method depends on what is intended, whilst creating a specific game content relies on contents of simpler classes.

3. Related Works

This section surveys previous studies of PCG's impact on players. To this end, evaluating the same game with and without PCG (A/B test) is the most feasible approach [18]. Therefore, this section presents researches that performed this test and, in addition, used player-related data (e.g. opinion or behavior) as evaluation metrics. To the best of the authors' knowledge, Butler

et al. [28] was the first work to meet the aforementioned criterion. Thereafter, two other similar researches that meet those criteria were found [18, 17]. However, these latter studies evaluated PCG's impact based on players' opinions, whereas the former analyzed players' in-game behavior. Following, this section further reviews them.

Butler et al. [28] used the DMG *Refraction* as the testbed for their proposed algorithm for game progression, comparing it to the originally human-designed progression. Their data collection procedure relied on the testbed game being available online. Players played on one game version or another (i.e. originally designed or procedurally designed) through the browser. Then, based on in-game data, the authors evaluated the impact of their proposed approach on players' behavior. The behavioral metrics used were (i) the number of played seconds and (ii) the number of played levels. Comparing metrics of both versions, they found an insignificant difference on the first and a small significant difference on the second. Although the significant difference in the number of played levels, the automatically designed version was played 92% as much as the human-designed. Thereby, showcasing the feasibility of their approach and that it had a small impact on one behavioral metric and no impact on the other.

Connor et al. [17] used an abstract game as the testbed to evaluate a PLG algorithm's impact, which was compared to human-generated levels. The authors performed a controlled experiment to collect players' feedback. Players were assigned to one version or another (i.e. procedurally generated or human-generated) and, after playing it, responded to a game immersion questionnaire composed of 30 items. According to players' self-reported immersion, the authors compared whether the PCG version impacted their experiences. The experiment's findings showed a significant difference, where the human-generated version led to a higher total immersion. However, with further analyses on each item separately, the authors found significant differences in only five out of the 30 items. Therefore, providing empirical evidence that PLG impacted players' self-reported total immersion, which emerged from significant differences on less than 17% of the questionnaire's items.

Korn et al. [18] used the tactical strategy game *Conquest of the Seven Seas* to evaluate the procedural generation of reefs. These were compared to reefs created by artists according to players' feedback. Feedback was collected on a controlled experiment, wherein participants played both versions on a randomized order. PCG's impact was then analyzed based on players' opinions of reefs visual aspects and preference for one version or another. The results showed that the procedurally generated reefs were considered significantly more realistic, aesthetically pleasing, and preferred in comparison to the artist created. Also, it was found that players with 45 years or more were the ones who appreciated the automatically generated contents the most. Hence, this research presented empirical evidence that PCG can promote experiences that overcome the experiences provided by artist-designed reefs and, additionally, demonstrated that older players prefer procedurally generated reefs more than younger players.

In order to sum up the key aspects of the reviewed researches, along with

aspects of the experiment presented in this paper, Table 1 was designed. For all of these studies, it demonstrates the year of publication, generated content, sample's age range and size (N), evaluation metric, and analysis design. A two-sample design refers to the cases wherein players interacted with one game version or another; while one-sample refers to the case wherein players interacted with both versions.

Table 1: Comparison of researches on PCG's impact.

Aspect	Butler et al.	Connor et al.	Korn et al.	This
Year	2015	2017	2017	?
Content	Progression	Levels	Reefs	Levels
Age-sample	?	18-35	18-≈45	7-72
N-sample	2377	16	41	724
Metric	in-game behavior	game immersion	aesthetics; preference	in-game behavior
Design	two-sample	two-sample	one-sample	two-sample

? = data not available.

As can be seen in Table 1, the latest studies analyzed PCG's impact according to players' self-reported feedback through questionnaires. Also, these analyses were based on controlled experiments with modest samples' size ($\leq$ 41). Additionally, the table shows that Connor et al. [17] is the only published research that actually evaluated the generation of levels itself. Although Butler et al. [28] also used procedurally generated levels, the authors' main focus was on their game progression system's impact. Another fact is that PCG's impact was evaluated by adult players only, considering that the participants of these samples were all over 18 years. Notice that, despite *Refraction* was hosted on a platform of games for children, Butler et al. [28] did not provide the sample's age. Consequently, there is no evidence of their samples' age neither that they share similar characteristics. Analyzing samples with similar characteristics is important on two-sample designed studies, which was adopted on two out of three researches, because failing to meet this criterion might bias the analysis. Once these exist, there are two differences in the analysis: (i) participants' characteristics and (ii) whether the game version uses PCG. Therefore, this might represent a threat to analysis validity.

Based on the aforementioned context, this article expands the literature on four main aspects. Firstly, recent researches have focused on analyses based on players' opinions. This paper tackled a problem alike to theirs, however, it adopted players' in-game behavior as evaluation metrics as these are seldom recently. Nevertheless, we acknowledge that evaluating measures of both players' opinions and behaviors, in a complementary way, would provide more thoughtful contributions and that not providing this combination is a limitation of this article's scope. Secondly, the only study using this metric type was mainly concerned with a game progression design system. Differently, this paper's experiment was mainly concerned with a level generation system. Thirdly,

analyses of PCG's impact did not involve children and teenage players, which was addressed in this research by having a sample composed of players with ages from seven to over 70 years. Lastly, no concerns were provided regarding whether there exist significant differences in samples' characteristics in previous studies. This article presents empirical analyses that addressed this lack.

4. Testbed Game

To perform this article's experiment, it was used as testbed the DMG *SpaceMath*[2]. This game requires players to solve basic math operations to advance from one level to another. Each level is represented by a universe where players control an astronaut. This astronaut must explore levels, using a teleport device to find out what is hidden below level's elements. Most level's elements are boxes, which might hide aliens (i.e. enemies), numbers, or nothing. While aliens are intended to difficult level's exploration, numbers are the elements that must be collected to solve the math puzzle of each level. These puzzles are the arithmetic operations (e.g. summations and subtraction) that are the key to progress in the game. However, simply collecting the numbers is not enough. The order these numbers are collected must form the exact puzzle's solution. For instance, collecting the number one and then number two if the right answer is 12. As can be noted, collected numbers are concatenated, forming the astronaut's answer to the level's puzzle. Additionally, boxes might have different colors until players achieving a 10-wins-streak to help them in collecting numbers. These colors guide players toward the right order of numbers selection, which can improve their in-game performance by aiding on puzzles' resolution. Finally, once the avatar is led to the portal, which appears once that any number is collected, another parallel universe (i.e. game level) is presented and the game continues if the puzzle's answer is correct, or restart if it is wrong. Figure 1 presents two screenshots of the game, showing the same level at two different stages.

More specifically, the main pedagogical objective of the testbed game is related to mathematical arithmetic operations, namely: summation, subtraction, multiplication, and division. The goal is to encourage players to practice their arithmetic skills as much as possible while playing the game. To achieve this goal, the arithmetic operations are inserted into the game world as puzzles that must be solved in order to win each level and, consequently, advance to the next one, as well as to accumulate more points. Hence, players need to put efforts in correctly solving the puzzles, otherwise, they will not win any and, therefore, yield poor performances. For instance, in the level presented in Figure 1, the arithmetic problem players must solve is 20 divided by five, which the answer is four. As this puzzle's solution has a single number (4), the player just has to find the number, collect it (pass under the number with the avatar), and then it would be possible to win the level. However, if the player was not aware that the answer has a single number, s/he would, possibly, unnecessarily search for a

[2]spacemath.rpbtecnologia.com.br

(a) **(b)**

Figure 1: SpaceMath's screeshots. Figure *a* shows a level as generated and Figure *b* showcases the same level after the player used the teleport device to teleport two boxes (top-center and bottom-right) to another parallel universe, finding an alien and the problem's solution, respectively.

second number. Similarly, in a level that the arithmetic puzzle is, e.g. 3 times 4, if the player wrongly processes the operation, s/he could find the number one, think it is the right answer (confounding multiplication with subtraction, e.g. by lack of attention), and lose the level when trying to advance. In the same way, the player could collect both the numbers two and one, forming the answer 21, and also lose the level, whereas the player should collect first the number one, then the number 2 in order to formulate the right answer (12). Thus, despite the game aids in the puzzle's resolution process, the pedagogical goal is accomplished by making players mentally solve arithmetic operations to prevent from losing.

Moreover, given the game's context, players are always led to other parallel universes that, similarly, require them to solve new puzzles. Thereby, as long as the game is being played, its players are repeatedly practicing their math skills. This repetition is a pedagogical aspect that is valuable to players because it helps them to practice the mental resolution of arithmetic problems [29, 30], which is similar to homework but in a ludic way. For example, when a new subject is introduced (e.g. multiplication), it is common to ask the students to solve a set of multiplication problems in order to fix the learning. The pedagogical approach of *SpaceMath* is similar to that: it makes players solve many arithmetic problems in terms of the four basic math operations but, differently from a common homework, it does that in a ludic fashion. Therefore, it is expected to encourage players to practice their math skills in an entertaining way. Furthermore, as the math puzzles are automatically generated, the game provides a pseudo-infinite number of those for players to practice. Thus, besides allowing them to practice math in a ludic fashion, it provides a substantially large number of problems to do so. Nevertheless, describing the puzzles' automatic generation is not the scope of this paper.

Additionally, we highlight that every time some level is finished, regardless

of the player winning or losing, a new parallel universe is presented. Thereby, *SpaceMath* promotes endless gameplay, which means players will never face a *last level*. To yield this feature, PCG is necessary, once it can automatically create a pseudo-infinite number of those (as well as for the math puzzles). Otherwise, developers and designers would have to create thousands of levels, making this feature unfeasible. Therefore, the importance of exploiting PCG algorithms for games with this characteristic. Nevertheless, there exists a trade-off on this approach. On one side, humans can create a limited number of contents using their specialist knowledge and creativity. On the other side, computers rely on predefined algorithms that might be limited in outputs quality but promote several contents with less manpower. Hence, using PCG is important for games to deliver endless gameplay, whilst the trade-off between development improvement and content's quality must be considered. Hereby, the importance of comparing contents created from both ways to understand how the experience they provide differs according to players.

4.1. Static Version

This version contains 20 levels that were manually designed by a game developer. He is graduated in Technology in Digital Games and had three years of experience in the field at the time of designing them. During the gameplay, the order in which these levels are presented for players increases the quantity of elements in them based on the sequence of wins players have at that point. The aim is to raise their difficulty accordingly. Thus, the first levels will contain few elements and the last ones (the twentieth is the last one) will contain several. In this context, levels' difficulty progression respects the following specifications. At first, levels have few boxes with no aliens hidden beneath any of them. Then, after each win, the quantity of boxes is increased and harmful elements, which are boxes hiding aliens, progressively begin to exist. Thereafter, levels will present boxes that are actually necessary only if the puzzle's answer was bigger than 99, seeking to lure players towards wasting their devices' charges and difficult their path.

Furthermore, another characteristic of this version is that when players lose after reaching a three-wins-streak, or in any other streak less than 20, they will have to replay the three levels they just played again. This characteristic is present in most games with a limited number of levels (or any other content) available. However, when players reach a 20-wins-streak, they are redirected to the *dynamic* version to guarantee the endless gameplay feature of *SpaceMath*. Also, after achieving this point, the player will always play in the *dynamic* version from the next login. Summarizing, this version provides predefined levels, designed by a human developer, which presents a progressive increase of difficulty as players achieve higher sequences of wins. Due to the limited number of contents, players might play repeated levels and, in order to maintain the testbed game's endless gameplay, players that reach a 20-wins-streak are transferred to the *dynamic* version. Despite that, for data analysis, those that started playing on the *static* were always considered as Control group's players,

regardless of being transferred to the *dynamic* version or not in order to maintain the experiment's consistency.

4.2. Dynamic Version

In this version, players interact with levels automatically created. A constructive method of PCG was adopted to this end, which was inspired on previous similar implementations [31, 32]. This method operates in a *generic* fashion, which means that levels are generated independently of players' types, characteristics, and personalities. Also, it runs *online*, enabling the endless gameplay characteristic. Its implementation's description might be presented as follows. Firstly, it creates a tilemap (grid-like) representation for the level under generation. Secondly, it positions the level's avatar (astronaut) at its fixed position (left-bottom). Thirdly, numbered elements are placed at selected positions of the tilemap. These elements might be single boxes and/or rectangular clusters of boxes (blocks). Fourthly, harmful elements are positioned as well, which are those that might spawn one or two aliens. Fifthly, the procedure places empty elements. Thus, constructing levels in a straight-forward procedure, through a specific sequence of steps that run independently of players. Figure 2 demonstrates the step-by-step of this process for the game's screenshot presented in Figure 1.

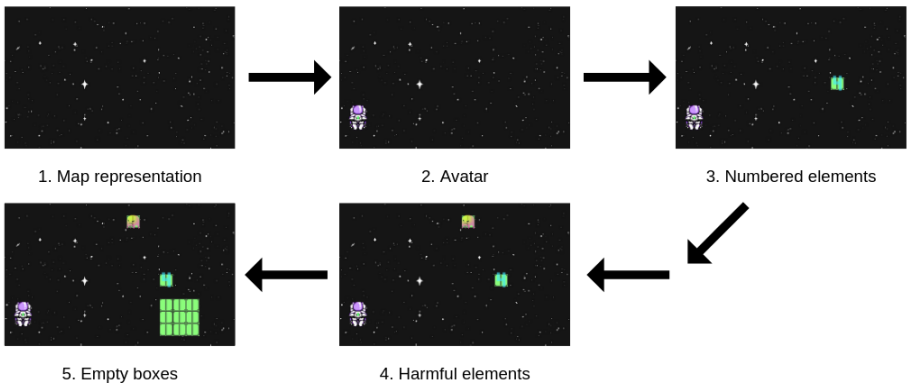

Figure 2: Step-by-step level generation.

Because Figure 2 shows the level's final design, the tilemap representation is invisible. At the second step, the level is incremented, compared to step one, by featuring the avatar. At step three, the numbered element, a single box hiding the number for in this level's case, is inserted. Subsequently, the level's single harmful element (single box), which is hiding an alien, is added in step four. Lastly, step five increments the level by adding a block of empty boxes. Note that the portal used to teleport the astronaut from one level to another was not mentioned. The reason is that this generation process runs before the gameplay's beginning, whereas the portal is created during it. Therefore, its positioning is executed as a separated step. Following, we provide further

insights into how the generation of levels happens, expanding on the overview already mentioned. To illustrate this method, Figure 3 showcases a high-level flow diagram of the process. It presents the five steps presented on the step-by-step generation of Figure 2 and, in addition, it shows the sub-steps performed in each of those. For all steps that create some content but step number two, where all elements will be at is selected similarly, which is: choosing tilemap's available positions at random. This approach aims at increasing the algorithm's outputs variation, inserting a degree of randomness on every generated level. However, it is necessary to design the algorithm in a way that no output fails to meet the developers' intends or testbed game's characteristics. Thereby, this method features a set of predefined rules that guarantee the feasibility of each level, in terms of both the math puzzle's resolution and difficulty progress. These rules are following described, relating each rule to the algorithm's step in which it is executed.

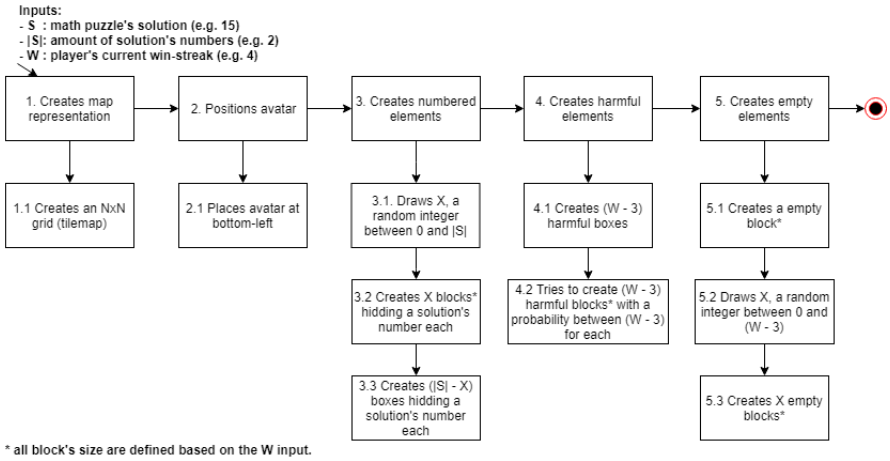

Figure 3: Flow diagram of the level generation method.

The feasibility of a testbed's level is achieved through both the existence of the numbers required to solve the math puzzle and the progressive increase in the quantity of elements inside it according to players' win-streak. From one perspective, if a puzzle's answer is 15, for example, the level must feature two numbered boxes hiding both the numbers one and five. To do so, the level generation is aware of the puzzle's answer and creates exactly one game element for each one of its numbers (steps 3.1). That is possible because the math puzzle will feature a level, as well as its solution, are available before the level's generation process start. Hence, the solution can be passed as an input to the level generation algorithm (see step 1), which uses it to determine (i) the exact amount of numbered elements that must be created (steps 3.2 and 3.3) and (ii) which number (or set of numbers) will be hidden below each numbered element. Hereby, while generating a level, the existence of its math puzzle's solution is guaranteed, avoiding the necessity of verifying whether the solution exists after

the procedure's execution. From the other perspective, it is necessary that the number of elements inside levels increases progressively as players achieve higher win-sequences. To this end, the method is also aware of players' win-sequences (input on step 1), which allows it to determine how many elements will be inserted in its outputs (throughout steps 4 and 5). Additionally, there are specific questions of the generation process that aid the generated results for both aforementioned perspectives.

From math puzzles' perspective, there is another important fact to improve levels' challenging and variety. To avoid numbers being always in single boxes or in blocks of boxes, randomness is employed (step 3.1). It is used to determine whether all numbers will be beneath single boxes, blocks of boxes, or in both (e.g. one number in each type or two numbers in one and one number in another). Consequently, preventing players from looking for numbers on only one element type or another, which improves the generated outputs' challenging level. Moreover, in blocks' cases, randomness is also used to determine their sizes as well as in which block's box the number will be hidden. A block with size two has two rows and two columns (total of four boxes), whilst a block with size three has three rows and three columns (total of nine boxes; see Figure 1). However, they can end up being smaller than the expected, if some box's position is occupied by a previously generated element or if its position falls outside levels boundary. Even though, blocks offer options of where the number can be placed. Thereby, using randomness enhances levels' variety by making where the numbers will be at less predictable, both in terms of which element type is hiding them and in which block's position the number is at, besides inserting variation on blocks' size.

From levels' difficulty perspective, there are other specificities that play an important role in the contents' variety as well. The quantity of not numbered elements respects players' performance, however, randomness also affects this aspect (steps 4 and 5). It is used as a way to enhance levels' variation, affecting the quantity of elements available on levels, as well as affecting both blocks' size and harmfulness. Despite the quantity of harmful elements increases as players achieve higher win- streak, this increase is not fixed (step 4.2). Due to the randomness use, it is possible that 10 of those elements are available at the fifth level in one time, and that 12 of them are available when the player reaches that fifth level once again. Similarly, the size of blocks will respect wins-streak, although they will vary as well, given the use of randomness. In addition, blocks' harmfulness (i.e. the number of aliens hidden within a block) also benefits from it. Hence, the use of randomness yields blocks with varied harmfulness, in spite of always increasing according to the sequence of wins players have.

Furthermore, there are specific generation restrictions that were designed in order to aid players. One of them is that the avatar's neighbors positions are excluded from the tilemap. The goal of this restriction is to prevent the astronaut from beginning the level stuck in the corner, giving some space for the player to perform an initial exploration. Another restriction is that harmful elements only begin to appear from the four-wins-streak (step 4), aiming to facilitate players advancing the first levels and preparing themselves to the subsequent

ones. Also, it makes them practicing the math operations even if they are beginner players and have difficulty to win levels with harmful elements. Lastly, we highlight that, if there are harmful boxes in the level, each one will spawn a single alien, on its own position, once it is teleported out of the level. However, as players' wins-streak increases, there is a chance for each one of those boxes to spawn a second alien, on a random position, increasing the levels' challenging and variety.

In sum, this version provides the game with several computer-generated levels which are created following a specific sequence of steps and highly relies on randomness to improve its outputs' variation. The generation method design manages to guarantee that levels will always contain the math puzzle's solution (step 3), while also increasing the outputs' difficulty, according to players' performance, through placing more boxes that hide aliens (step 4) as well as empty ones (step 5). At the same time, it uses random selections to avoid levels always having the same amount of elements, with the same size each, seeking to make them less predictable and more innovative. Therefore, being able to create a large number of varied outputs that respects the game's and designer's intents.

5. Experiment

Identifying PLG's impact on players' behavior was the goal of this experiment. It was performed according to an A/B test, which is recommended in the literature to evaluate PCG algorithms' impact [18]. This test requires a game with two versions differing in a single aspect. In this case, the single difference between versions must be whether they use PLG or not. Hence, the studied game versions were named as *dynamic*, which features the PLG algorithm, and *static*, which features human-generated levels. In addition, this experiment used an adaptation of the two-sample design. This design itself implies that a player plays in a single version only, leading to two samples. Thereby, the samples consisting of players from the *dynamic* and the *static* version were named Intervention and Control groups, respectively.

However, guaranteeing that players will interact with only one version is hampered due to the limited number of human-designed levels plus the game's feature of providing endless gameplay. Players from the Control group might achieve a 20-wins-streak and, to maintain the game's endless characteristic, in this case, they begin to play on procedurally generated levels after this point. Hereby, while participants from the Intervention group only played on the *dynamic* version, participants from the Control group that achieved a 20-wins-streak ended up playing on both versions, despite playing at least 20 levels on the *static* version. Thus, the rationale of arguing that an adapted two-sample design was used. Regardless, this approach enabled the comparison of players' behavior from each group to identify PLG's impact. As result, this experiment's hypothesis was: **The behavior of players that started playing on the *static* version is indifferent from the behavior of the ones that played only on the *dynamic* version.**

Therefore, differences in players' behavior suggest that PLG impacted their behavior since it is expected to be the only distinction between groups. We highlight that, although PCG can be used to create personalized contents, aiming at driving or improving players' experience [26], the algorithm analyzed in this experiment is generic, besides simple and straight-forward. Consequently, it is interesting for a game featuring it to lead players to behave in the same way they behave on specialist-generated contents, besides being feasible for *online* use due to its fast running-time. Lastly, to accomplish this experiment, all statistical tests and analyses were performed on R [33].

5.1. Participants

Players often have distinct characteristics, personalities, and types, and, thus, behave differently according to them [34, 35, 36]. Thereby, due to the two-sample design, another distinction on the A/B test might emerge: players from different samples having distinct characteristics. For instance, the Intervention group might be composed of players older than the players from the Control group. Consequently, this age difference might bias the experiment's results. In this case, it is possible to emerge a difference in players' behavior due to groups' age difference rather than due to the different game versions they played. To mitigate this threat, it is expected for both groups to share similar characteristics. Therefore, we compared demographic data from players of both groups aiming to provide evidence that the aforementioned threat was remedied in this experiment. This analysis' results are shown in Table 2.

Table 2: Samples' characteristics.

Characteristic	**Intervention**	**Control**	**Measure**	**Result**
Genre	F=137; M=234	F=113; M=249	χ^2	2.4117
Age	13 (11-17)	13 (11-16)	U test	68433
Gamer	N=180;Y=191	N=188;Y=174	χ^2	0.72413
Playing time	5 (2-14)	4 (1.25-14.75)	U test	67300

* = $p < 0.05$; M = male; F = female; Y = yes; N = not

Table 2 demonstrates both the characteristics of this experiment's samples and the results of comparing their data. Represented as median(1^{st} quartile-3^{rd} quartile) within the table, numeric data refers to participants' current age and to how many hours they weekly play (playing time), considering any game and any device. For these, the Shapiro-Wilk normality test provided significant evidence they do not follow a normal distribution ($p < 0.05$). Consequently, we compared groups' median through the Mann-Whitney U test (U test hereafter). Differently, the table represents categorical data (i.e. genre and gamer) according to their classes' distribution. This data type was compared through the Chi-Squared homogeneity test (χ^2). For numeric data, the null hypotheses were that groups' medians do not differ; whereas the null hypotheses for categorical data were that groups' distributions are the same. As can be seen on the table, no significant difference ($p < 0.05$) was found on all tests. Hence,

demonstrating that the threat of comparing the behavior of samples with different characteristics is not present in terms of players' demographics and gaming habits.

5.2. Behavior Measures

To evaluate participants beyond those characteristics, this experiment used three metrics to measure players' in-game behavior: number of played levels, time played in seconds, and number of retained players.

Total time played has been used on previous studies as a metric of players' engagement [37, 38, 39]. In this research, we analyzed it based on the total amount of time that the game was played in seconds. The number of played levels was used in the study of Butler et al. [28] as an additional metric to assess players' in-game behavior. This was adopted in this paper's experiment, in the same way, considering how many levels participants played at all. Additionally, this experiment analyzed players in terms of how many of them were retained in each version. Although retention does not have a definition that is universally accepted [40, 41], a number of attempts exist, as discussed in Viljanen et al. [42]. Nacke and Drachen [43] defines it as the game's ability to keep users playing the game. Weber et al. [44] claim it is a measure of player experience that refers to players' complete gameplay history in terms of how long one continues to play the game. These authors measured retention as both the number of football games played (similar to out levels played metric) and as the number of sessions a played initiated [44, 45]. Harrison and Roberts [46] argue that, despite the many definitions, it often refers to the percentage of players that still play the game after some period of time. In addition, some authors also define distinct types of retention. Harrison and Roberts [46] also defines session-level retention, the percentage of players that complete a single game session. Another case is the differentiation between short- and long-term retention, which refers to continue playing the same session and quitting the game or leaving it for an extended period, respectively [47]. Also, there are the aggregate and individual retention, which refer to measurements that consider the entire player base and those applied to a player in specific, respectively [44].

Considering the aforementioned context, the retention measure adopted in this paper was inspired by a combination of definitions and types present in the literature. In the scope of this article, this metric measures the number of participants that played at least a specific number of levels, considering as threshold the median of total played levels based on both Intervention and Control groups' players together. Hence, a retained participant played more than half of all participants. Thereby, this is a type of aggregate retention because it considers the entire player base [44], it is session independent [46], and might reflect both short- and long- term retention [47]. Additionally, it is based on the number of played levels [45], capturing players' commitment towards the game [48] in terms of those who continued playing after achieving the average number of levels [43]. We highlight that this is relevant in the context of this study because the more levels users play, the more they practice math. Note that we selected this average-based threshold because the *SpaceMath* does not have

an *end*, therefore, it is impossible to evaluate players' retention in terms of those who finished playing (c.f. [43, 46]). Therefore, in summary, this experiment analyzed players considering two numeric metrics and one categorical measure, concerning their behavior from the perspectives of engagement and retainment.

5.3. Procedure

Participants were reached through two interventions. One was to disclose the research via e-mail. The other was to contact colleagues teachers. On the latter case, four colleagues agreed to perform the research on institutions they teach. In both cases, participants were informed they would be participating in a research before the procedure started. Hence, all subjects agreed to participate in the experiment, regardless of being reached via e-mail or in their institutions through their teachers.

The procedure itself consisted of three steps: (i) description, (ii) registering, and (iii) playing. The first step's goal was to introduce the research and the testbed game to participants. For those reached via e-mail, its corpus had the description of this procedure. Differently, colleague teachers who ran the experiment with their students on their institutions' classes performed the procedure's first step. On the second step, the goal was to capture players demographics in order to enable comparing characteristics of players from different groups. This is important because having groups with different characteristics inserts a threat to the study's validity. Hence, registering into the testbed game before playing it was a requirement for all players, regardless of how they were reached. Lastly, players' behavior was captured on the third step, when they finally got to play the game. Their behavior was constantly captured in terms of in-game metrics (e.g. number of played levels). Also, they were able to play as much as they wanted because the game is available for free access online. Therefore, even participants not reached via e-mail could play it after the institutional activity. In sum, this procedure captured participants' in-game behavior after they were introduced to both the research and the testbed game and provided their demographic data.

6. Data Analysis and Results

Firstly, numeric behavioral data (i.e. number of played levels and time played) were analyzed through the Shapiro-Wilk test. These tests considered an alpha level of 0.05 to reject the null hypothesis. Hence, p-values less than or equal to 0.05 suggest that data are unlikely to follow a normal distribution, similar to the analysis presented in Section 5.1. Table 3 shows all tests' results for both Intervention and Control groups in terms of numeric behavioral data normality.

Table 3 demonstrates that all p-values are less than the alpha value. Therefore, all normality tests provided sufficient evidence to reject the hypothesis that some of these metrics follows a normal distribution. Thus, numeric measures of players' behavior were compared based on the non-parametric U test. This test

Table 3: Normality tests of numeric behavioral metrics.

	Intervention		Control	
Measure	**W**	**P-value**	**W**	**P-value**
Number of played levels	0.676	< 0.001	0.764	< 0.001
Time played in seconds	0.664	< 0.001	0.770	< 0.001

W = Shapiro-Wilk statistic

compares two samples' medians. They significantly differ if the test provides sufficient evidence to reject its null hypothesis. This is, a p-value less than or equal to the selected alpha level, which is 0.05 for the following analyses as well. Therefore, U tests resulting in p-values less than or equal to 0.05 indicate a significant difference in the behavior of players from different groups according to numeric measures. Consequently, suggesting that PLG impacted participants' behavior in terms of levels and time played. Table 4 presents statistics of these metrics for both groups.

Table 4: Statistics of players' numeric measures of in-game behavior.

	Intervention		Control	
Measure	**Mdn**	**IQR**	**Mdn**	**IQR**
Number of played levels	39	19-75.5	33.5	17-68
Time played in seconds	998	515-1914	1000.5	505.5-1792.2

As can be seen in Table 4, Intervention and Control groups played a median of 39 and 33.5 levels, respectively. It also demonstrates that most participants from the Intervention group played between 19 and 75.5 levels, while most participants from the Control group played between 17 and 68 levels. Thus, showing that the Intervention group played more levels than the Control group. However, the difference between groups in terms of this behavioral metric is statistically insignificant according to the U test ($U = 62480$, $p = 0.1031$). Moreover, Table 4 shows that Intervention and Control groups played a median of 16.63 and 16.68 minutes, respectively. In addition, it demonstrates that most participants from the Intervention group played between 8.58 and 31.90 minutes, while most participants from the Control group played between 8.43 and 29.87 minutes. Thereby, demonstrating that the Control group played slightly more than the Intervention group in terms of played seconds. Yet, this difference is statistically insignificant as well, according to the U test ($U = 65617$, $p = 0.5926$). Therefore, we can conclude that there is no evidence to support the claim that the behavior of players from distinct groups differed in terms of numeric behavioral measures.

Furthermore, players' behavior was also analyzed in terms of how many of them were retained. This metric's threshold was the median value of played levels from both samples together, i.e. 36 levels. Therefore, participants that played more than this threshold were considered retained. Otherwise, they were considered non-retained. This measure was compared based on the distribution

of players (i.e. retained or non-retained) on each version through the Chi-Squared homogeneity test. It analyzed the homogeneity of groups based on the null hypothesis that their distributions are the same. Hence, rejecting the null hypothesis provide significant evidence they differed. This test used the same alpha level than previous tests. Thereby, a p-value less than or equal to 0.05 suggests that retained players' distribution differ between the two samples. Table 5 displays the number of retained and non-retained players from each group.

Table 5: Distribution of players in terms of retainment

Group	Retained	Non-retained
Intervention	193 (0.52)	178 (0.48)
Control	145 (0.40)	217 (0.60)

The number of players retained and non-retained for both Intervention and Control groups are shown in Table 5. Also, it showcases the proportion they represent within each sample. As can be seen, over than half of the players from the Intervention group were retained, whilst 60% of players from the Control group were non-retained. Hence, showing that players from the Intervention group were more retained than players from the Control group. Additionally, through the homogeneity test, we found statistical evidence that the difference in groups' retainment distribution is highly significant (χ^2=10.082, df = 1, p = 0.001497). Thereby, we can conclude that the distribution of participants who only played on the procedurally generated levels is significantly different from those who began playing on the human-generated levels according to the retainment metric.

7. Discussion

Overall, on one hand, the experiment's findings showcased that players' behavior did not significantly differ in terms of engagement-related metrics (i.e. number of played levels and seconds). Therefore, demonstrating that PLG had no impact on these behavioral metrics. On the other hand, the number of retained players from the Intervention group was significantly different from the number of retained players from the Control group. In this case, presenting empirical evidence that the game version using PLG retained players better than the version using human-designed levels. Hereby, while the number of played levels and seconds suffered no significant impact by the use of a PCG algorithm, a highly significant impact became evident when assessing players' retainment. Thus, rejecting the experiment's hypothesis and yielding empirical evidence that groups behaved differently in one of the analyzed metrics.

These findings corroborate with the literature by showing that further researches are needed to establish the impacts of PCG on players. Butler et al. [28] found that their approach had a small impact on the number of played levels, which can be considered negative since the game's original version was

played more. Connor et al. [17] also achieved results that might be considered negative. Players' immersion was small for the ones that played on procedurally generated levels in comparison to players from the human-designed levels, in their research. However, Korn et al. [18] demonstrated that players considered the human-created reefs less realistic and preferred in comparison to the procedurally created. Hence, yielding positive results for the algorithmically generated contents. We argue that this paper's results fit in a middle-term, wherein a positive impact was found in one metric, whilst no impact was found on the other two. Thereby, the literature presents some contradictions in terms of what are the impacts of PCG on players. Therefore, demonstrating the need for more researches to ground this understanding.

Nonetheless, both game development and player experience communities can benefit from this article's findings. Based on them, researchers can gain insights into how players' interactions with human-generated levels differ from interactions with procedurally generated levels, according to their in-game behavior in a DMG. Game developers and designers can incorporate PLG algorithms into their projects to improve the development process being aware that, even though they use a simplistic PCG method, it is unlikely that it will have a negative effect on players' in-game behavior. Additionally, based on this research's findings, it is likely that players will be more retained by the game than if they had to interact with a predefined amount of human-created content. In this latter case, unavoidably, they would end up having to play the same levels again. In contrast, using PLG allows the game to constantly provide players with new and unseen contents. Consequently, we argue that presenting not-repeated contents was one factor for the Intervention group's players being more retained than Control group's players, despite the simplicity of the generation algorithm. Therefore, based on this paper's findings, our argument is that people can take advantage of PLG's contributions for development time, without negatively influencing players, at the same time its benefits will probably be reflected on players' experience as well.

In spite of that, a key limitation of this research is one of the testbed game's main feature. Comparing the behavior of players that interacted with a single game version is difficult given the endless gameplay feature. Once there is a limited set of human-designed levels, it might be necessary to transfer players from the *static* version to the *dynamic* version in some cases. This change was necessary when players achieved a 20-wins-streak. Otherwise, they would whether have to stop playing or going back to the first level despite the win, and both cases would violate the game's rules. Hence, it was unavoidable the presence of participants that started playing on human-created levels and then played on algorithmically generated ones. Nevertheless, to maintain the experiment's measures consistent, data (levels and time played) from players that started playing on the *static* version always counted to the Control group for the sake of data analysis. That is, regardless of being transferred from one version to another, players belonging to the Control group never changed to the Intervention group. Therefore, maintaining all metrics consistent.

However, that limitation represents a threat to the experiment's construct

validity [49]. For instance, it could be mitigated if the analysis were focused on players' opinions. They could be captured after playing a predefined number of levels that is equal to the number of human-created levels. Thereby, guaranteeing that no players would have to be changed from *static* to *dynamic* version before providing their opinions. Nonetheless, this approach is not available in order to assess players' behavior. Therefore, in order to tackle the gap of evaluating in-game behavior left from recent studies, this paper's experiment analyzed whether the behavior of participants who started playing on human-designed content and then, possibly, also played on procedurally generated levels, differed from the behavior of participants that only played on automatically generated levels. Hereby, we argue that the possible design threats are small as we managed to explore a research question that considers this limitation.

Another possible limitation of this study is the *static* game version promoting repeated levels. While players might have to play the same level again if they lose before reaching a 20-wins-streak in the aforementioned version, a new level is generated each time a player wins or looses in the *dynamic* version. This fact might be seen as a limitation because whereas players from one version are forced to replay the same content, players from the other are unlikely to do so, although it is possible that the same level is generated twice (or more). We argue that this not a limitation, though. It is common for games that do not use some PCG algorithm to operate in this way: Once you lost and have to restart, the same content is available. Consequently, it is valuable for research purposes that the control intervention (i.e. playing the *static* version) provides this feature. Hence, the research setup is approximated to realism. Thereby, mitigating threats the experiment's external validity [49]. Thus, increasing the chances that the experiment's results will be generalized to industrial practice. Additionally, we highlight that playing the same level more than once did not affect the number of played levels metric neither the retained players metric. Although, e.g. losing at the second level twice means levels one and two were played twice, the log system stores that this player played four levels (i.e. first and second, then, first and second again). Therefore, computing the number of played levels rather than the number of distinct levels played.

Moreover, there are other threats that might have affected this study's validity. Mainly, they are threats that concern the experiment involving humans. From one perspective, there are the manually generated levels, which might be of poor quality and end up biasing the results. This threat was mitigated by selecting an experienced developer which is a specialist in the field. Another concern is that data from Table 4 suggest that players from the Control group spend more time per level (30 seconds) than those of the Intervention group (26 seconds), indicating that the former group faced more difficult than the latter. On the one hand, one might argue that this performance difference appeared because players from one group are more skilled than those of the other. However, data from Table 2 indicates participants do not differ in terms of both playing time and being gamers. In spite of that, some players might be more skilled in the specific genre of the testbed game and, therefore, this might be the reason for the performance difference. On the other hand, the

cause might be the procedurally generated levels which, possibly, were easier to play and, consequently, led players to yield better performances. The latter case would evidence that PCG led to higher retention, while the former would indicate that players themselves were the cause. Nevertheless, further research is required to confirm or reject those hypotheses. From another perspective, participants might have behaved within the game differently than they would if they were not aware they were participating in a research. The approach to mitigate this threat was not to inform what was under analysis (i.e. PCG versus human-generation) and neither that in-game behavior was the evaluation metric. Despite that, they were aware their in-game data would be collected and no personal information would be disclosed. Additionally, it was sought to mitigate threats to conclusion validity through the use of robust, well known statistical tests and a substantially large sample, based on previously related researches. Therefore, although the existence of some threats to the validity of this paper's experiment, we argue they are insufficient to invalidate its findings.

8. Final Considerations

Few researches have tackled the impact of PCG use on games, especially on the ones with educational ends, although it is a reliable approach to improve them. This article addressed this gap, presenting the analysis of PLG's impact on players' behavior based on their interactions with a DMG. Players' behavior in two versions of this game were compared, wherein the only difference was one version featuring human-generated levels and the other providing procedurally generated levels. Hence, to identify whether playing only on algorithmically generated levels impacts players, compared to starting playing on human-designed levels, three behavioral measures were assessed through empirical analyses: number of played levels, time played in seconds, and number of players that played more than half of all experiment's participants (retained players).

The experiment's results showed insignificant differences in two out of the three measures. The number of retained players was the single metric that presented a significant difference, wherein the ones who only interacted with the automatically generated levels were more retained than the remaining players. Thereby, demonstrating that PCG impacted players behavior in terms of the number of retained players, but had no impact on time played and number of played levels. These findings mainly contribute to designers and developers, providing them with empirical evidence that using PLG algorithms to create levels of similar games is likely to have no negative impact on players' behavior. Instead, it can lead them to be more retained than when they start to play on human-created levels and have to face repeated contents. Nonetheless, this analysis' perspective is recent and, given the contradictory results of similar researches, further studies are required in order to ground the understanding of PCG's impact on players. Furthermore, consider only in-game metrics is a scope limitation that, if mitigated (e.g. using those in combination to play-

ers' opinions), would strengthen this study's contributions by allowing a more reliable interpretation of our findings.

Consequently, as the main directions of further researches, we recommend performing similar experiments on different game's genre, using different algorithms as well as sets of evaluation metrics. Adopting other games as testbed can demonstrate whether PCG impacts players differently on them. Similarly, advanced algorithms of content generation might showcase more advantages over human-designed contents, considering that the simple constructive algorithm used in this paper's experiment already demonstrated to be of value. Evaluating these impacts from different perspectives is interesting as well. While PCG impacted a behavioral metric but did not on others, it might be found that it impacts players differently in terms of their opinions or physiological reactions.

Acknowledgments

This research was supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] B. Bontchev, Modern trends in the automatic generation of content for video games, Serdica Journal of Computing 10 (2017) 133–166.

[2] J. Togelius, E. Kastbjerg, D. Schedl, G. N. Yannakakis, What is procedural content generation?: Mario on the borderline, in: Proceedings of the 2Nd International Workshop on Procedural Content Generation in Games, PCGames '11, 2011, pp. 3:1–3:6. URL: http://doi.acm.org/10.1145/2000919.2000922. doi:10.1145/2000919.2000922.

[3] R. v. d. Linden, R. Lopes, R. Bidarra, Designing procedurally generated levels, in: IDPv2 2013 - Workshop on Artificial Intelligence in the Game Design Process, AAAI, AAAI Press, AAAI Press, Palo Alto, CA, 2013, pp. 41–47. ISBN 978-1-57735-635-6.

[4] B. Horn, S. Dahlskog, N. Shaker, G. Smith, J. Togelius, A comparative evaluation of procedural level generators in the mario ai framework, in: Foundations of Digital Games 2014, 2014, pp. 1–8. URL: http://www.fdg2014.org/.

[5] M. Scirea, Y.-G. Cheong, M. J. Nelson, B.-C. Bae, Evaluating musical foreshadowing of videogame narrative experiences, in: Proceedings of the 9th Audio Mostly: A Conference on Interaction With Sound, AM '14, ACM, New York, NY, USA, 2014, pp. 8:1–8:7. URL: http://doi.acm.org/10.1145/2636879.2636889. doi:10.1145/2636879.2636889.

[6] M. Scirea, P. Eklund, J. Togelius, S. Risi, Evolving in-game mood-expressive music with metacompose, in: Proceedings of Audio Mostly (AM 2018), 2018, pp. 1–8.

[7] I. Interactive Data Visualization, Speedtree, 2017. URL: http://www.speedtree.com/, access: February 19, 2019.

[8] F. C. M. Rodrigues, J. B. C. Neto, C. A. Vidal, Split grammar evolution for procedural modeling of virtual buildings, in: SVR 2015, 2015, pp. 75–83. doi:10.1109/SVR.2015.18.

[9] I. Silveira, D. Camozzato, F. Marson, L. Dihl, S. R. Musse, Real-time procedural generation of personalized facade and interior appearances based on semantics, in: 2015 14th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), 2015, pp. 89–98. doi:10.1109/SBGames.2015.32.

[10] L. Cardamone, D. Loiacono, P. L. Lanzi, Interactive evolution for the procedural generation of tracks in a high-end racing game, in: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, GECCO '11, ACM, New York, NY, USA, 2011, pp. 395–402. URL: http://doi.acm.org/10.1145/2001576.2001631. doi:10.1145/2001576.2001631.

[11] Y. G. Cheong, R. M. Young, Suspenser: A story generation system for suspense, IEEE Transactions on Computational Intelligence and AI in Games 7 (2015) 39–52.

[12] A. M. Smith, E. Andersen, M. Mateas, Z. Popović, A case study of expressively constrainable level design automation tools for a puzzle game, in: Proceedings of the International Conference on the Foundations of Digital Games, FDG '12, ACM, New York, NY, USA, 2012, pp. 156–163. URL: http://doi.acm.org/10.1145/2282338.2282370. doi:10.1145/2282338.2282370.

[13] J. Valls-Vargas, J. Zhu, S. Ontañón, Graph grammar-based controllable generation of puzzles for a learning game about parallel programming, in: Proceedings of the 12th International Conference on the Foundations of Digital Games, FDG '17, ACM, New York, NY, USA, 2017, pp. 7:1–7:10. URL: http://doi.acm.org/10.1145/3102071.3102079. doi:10.1145/3102071.3102079.

[14] M. Hendrikx, S. Meijer, J. Van Der Velden, A. Iosup, Procedural content generation for games: A survey, ACM Trans. Multimedia Comput. Commun. Appl. 9 (2013) 1:1–1:22.

[15] E. A. Boyle, T. Hainey, T. M. Connolly, G. Gray, J. Earp, M. Ott, T. Lim, M. Ninaus, C. Ribeiro, J. Pereira, An update to the systematic literature review of empirical evidence of the impacts and outcomes of computer games and serious games, Computers & Education 94 (2016) 178 – 192.

[16] C. Bauckhage, K. Kersting, R. Sifa, C. Thurau, A. Drachen, A. Canossa, How players lose interest in playing a game: An empirical study based on

distributions of total playing times, in: 2012 IEEE Conference on Computational Intelligence and Games (CIG), 2012, pp. 139–146. doi:10.1109/CIG.2012.6374148.

[17] A. M. Connor, T. J. Greig, J. Kruse, Evaluating the impact of procedurally generated content on game immersion, The Computer Games Journal 6 (2017) 209–225.

[18] O. Korn, M. Blatz, A. Rees, J. Schaal, V. Schwind, D. Görlich, Procedural content generation for game props? a study on the effects on user experience, Comput. Entertain. 15 (2017) 1:1–1:15.

[19] Y. Dong, T. Barnes, Evaluation of a template-based puzzle generator for an educational programming game, in: Proceedings of the 12th International Conference on the Foundations of Digital Games, FDG '17, ACM, New York, NY, USA, 2017, pp. 40:1–40:4. URL: http://doi.acm.org/10.1145/3102071.3106347. doi:10.1145/3102071.3106347.

[20] J. Togelius, G. Yannakakis, K. Stanley, C. Browne, Search-based procedural content generation: A taxonomy and survey, Computational Intelligence and AI in Games, IEEE Transactions on 3 (2011) 172–186.

[21] D. Karavolos, A. Bouwer, R. Bidarra, Mixed-initiative design of game levels: integrating mission and space into level generation, in: FDG 2015, 2015, p. 8. URL: http://graphics.tudelft.nl/Publications-new/2015/KBB15.

[22] A. B. Moghadam, M. K. Rafsanjani, A genetic approach in procedural content generation for platformer games level creation, in: 2017 2nd Conference on Swarm Intelligence and Evolutionary Computation (CSIEC), 2017, pp. 141–146. doi:10.1109/CSIEC.2017.7940160.

[23] S. Dahlskog, J. Togelius, M. J. Nelson, Linear levels through n-grams, in: Proceedings of the 18th International Academic MindTrek Conference: Media Business, Management, Content & Services, AcademicMindTrek '14, ACM, New York, NY, USA, 2014, pp. 200–206. URL: http://doi.acm.org/10.1145/2676467.2676506. doi:10.1145/2676467.2676506.

[24] G. Smith, J. Whitehead, Analyzing the expressive range of a level generator, in: Proceedings of the 2010 Workshop on Procedural Content Generation in Games, PCGames '10, ACM, New York, NY, USA, 2010, pp. 4:1–4:7. URL: http://doi.acm.org/10.1145/1814256.1814260. doi:10.1145/1814256.1814260.

[25] J. R. H. Mariño, W. M. P. Reis, L. H. S. Lelis, An empirical evaluation of evaluation metrics of procedurally generated mario levels, in: Proceedings of the Eleventh AAAI, 2015, pp. 44–50.

[26] G. N. Yannakakis, J. Togelius, Experience-driven procedural content generation, IEEE Transactions on Affective Computing 2 (2011) 147–161.

[27] N. Shaker, G. Smith, G. N. Yannakakis, Evaluating content generators, in: N. Shaker, J. Togelius, M. J. Nelson (Eds.), Procedural Content Generation in Games: A Textbook and an Overview of Current Research, Springer, 2016, pp. 215–224.

[28] E. Butler, E. Andersen, A. M. Smith, S. Gulwani, Z. Popović, Automatic game progression design through analysis of solution features, in: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI '15, ACM, New York, NY, USA, 2015, pp. 2407–2416. URL: http://doi.acm.org/10.1145/2702123.2702330. doi:10.1145/2702123.2702330.

[29] F. Ke, A case study of computer gaming for math: Engaged learning from gameplay?, Computers & Education 51 (2008) 1609 – 1620.

[30] I. Ahmad, A. Jaafar, Computer games: Implementation into teaching and learning, Procedia - Social and Behavioral Sciences 59 (2012) 515 – 519. Universiti Kebangsaan Malaysia Teaching and Learning Congress 2011, Volume I, December 17 – 20 2011, Pulau Pinang MALAYSIA.

[31] L. Rodrigues, R. P. Bonidia, J. D. Brancher, A math educational computer game using procedural content generation, in: SBIE 2017, 2017, pp. 756–765. URL: http://www.brie.org/pub/index.php/sbie/article/view/7604/5400. doi:http://dx.doi.org/10.5753/cbie.sbie.2017.756.

[32] A. Khalifa, D. P. Liebana, S. M. Lucas, J. Togelius, General video game level generation, in: 2016 GECCO, 2016, pp. 253–259. doi:10.1145/2908812.2908920.

[33] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2018. URL: https://www.R-project.org/.

[34] L. A. L. Rodrigues, J. D. Brancher, Improving players' profiles clustering from game data through feature extraction, in: 2018 17th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), 2018, pp. 177–186. doi:10.1109/SBGAMES.2018.00029.

[35] Z. Halim, M. Atif, A. Rashid, C. A. Edwin, Profiling players using real-world datasets: Clustering the data and correlating the results with the big-five personality traits, IEEE Transactions on Affective Computing (2017) 1–1.

[36] L. E. Nacke, C. Bateman, R. L. Mandryk, Brainhex: A neurobiological gamer typology survey, Entertainment Computing 5 (2014) 55 – 62.

[37] E. Andersen, Y.-E. Liu, R. Snider, R. Szeto, Z. Popović, Placing a value on aesthetics in online casual games, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '11, ACM, New

York, NY, USA, 2011, pp. 1275–1278. URL: http://doi.acm.org/10.1145/1978942.1979131. doi:10.1145/1978942.1979131.

[38] E. Andersen, E. O'Rourke, Y.-E. Liu, R. Snider, J. Lowdermilk, D. Truong, S. Cooper, Z. Popovic, The impact of tutorials on games of varying complexity, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '12, ACM, New York, NY, USA, 2012, pp. 59–68. URL: http://doi.acm.org/10.1145/2207676.2207687. doi:10.1145/2207676.2207687.

[39] D. Lomas, K. Patel, J. L. Forlizzi, K. R. Koedinger, Optimizing challenge in an educational game using large-scale design experiments, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '13, ACM, New York, NY, USA, 2013, pp. 89–98. URL: http://doi.acm.org/10.1145/2470654.2470668. doi:10.1145/2470654.2470668.

[40] X. Fu, X. Chen, Y.-T. Shi, I. Bose, S. Cai, User segmentation for retention management in online social games, Decision Support Systems 101 (2017) 51–68.

[41] B. Jolley, R. Mizerski, D. Olaru, How habit and satisfaction affects player retention for online gambling, Journal of Business Research 59 (2006) 770–777.

[42] M. Viljanen, A. Airola, A.-M. Majanoja, J. Heikkonen, T. Pahikkala, Measuring player retention and monetization using the mean cumulative function, arXiv preprint arXiv:1709.06737 (2017).

[43] L. Nacke, A. Drachen, Towards a framework of player experience research, in: Proceedings of the second international workshop on evaluating player experience in games at FDG, volume 11, 2011.

[44] B. G. Weber, M. Mateas, A. Jhala, Using data mining to model player experience, in: FDG Workshop on Evaluating Player Experience in Games, ACM Press, 2011.

[45] B. G. Weber, M. John, M. Mateas, A. Jhala, Modeling player retention in madden nfl 11, in: Twenty-Third IAAI Conference, 2011.

[46] B. E. Harrison, D. Roberts, Analytics-driven dynamic game adaption for player retention in a 2-dimensional adventure game, in: Tenth Artificial Intelligence and Interactive Digital Entertainment Conference, 2014.

[47] K. B. Shores, Y. He, K. L. Swanenburg, R. Kraut, J. Riedl, The identification of deviance and its impact on retention in a multiplayer game, in: Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing, ACM, 2014, pp. 1356–1365.

[48] T. Debeauvais, B. Nardi, D. J. Schiano, N. Ducheneaut, N. Yee, If you build it they might stay: retention mechanisms in world of warcraft, in: Proceedings of the 6th International Conference on Foundations of Digital Games, ACM, 2011, pp. 180–187.

[49] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, A. Wessln, Experimentation in Software Engineering, Springer Publishing Company, Incorporated, 2012.