CHALMERS
UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG

How Procedural Content Generation affects player experience in 2D platformer games

Bachelor's thesis in Computer science and engineering

Arvid Johansson
Irmeli Johansson
Zidane Nguyen
Jakob Ögnelod

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2026

Bachelor's thesis 2026

How Procedural Content Generation affects player experience in 2D platformer games

Arvid Johansson
Irmeli Johansson
Zidane Nguyen
Jakob Ögnelod

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

How Procedural Content Generation affects player experience in 2D platformer games

Arvid Johansson Irmeli Johansson Zidane Nguyen Jakob Ögnelod

Supervisor: Aris Alissandrakis, Department of Computer Science and Engineering
Co-examiner: Palle Dahlstedt, Department of Computer Science and Engineering
Examiner: Patrik Jansson, Department of Computer Science and Engineering

Bachelor's Thesis 2026
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A depiction of the letters 'PCG' over a screenshot from the game, created by Arvid Johansson..

Typeset in LaTeX
Gothenburg, Sweden 2026

How Procedural Content Generation affects player experience in 2D platformer games
Arvid Johansson, Irmeli Johansson, Zidane Nguyen, Jakob Ögnelod

Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Procedural content generation (PCG) has become an established technique in game development, offering increased content variety with reduced manual design effort. However, its impact on player experience remains an active area of study, particularly in genres that demand precise spatial design such as 2D platformers. This thesis investigates how PCG level design affects perceived flow, how constraint complexity shapes player experience, and whether prior platformer experience moderates players' perception of procedurally generated content.

A 2D side-view platformer, `Gone Fishing`, was developed in Unity, featuring a constraint-based PCG system inspired by the room-template approach used in Spelunky. Levels are assembled at runtime from a library of 82 handcrafted room templates distributed across 15 opening categories, connected via a random walk path generation algorithm. Two configurations were evaluated, a 3-fish (small) and a 6-fish (large) condition, differing in generated level size and complexity. A user study was conducted with 20 participants of varying platformer experience, who each played both configurations. Player experience was measured using the Game Experience Questionnaire (GEQ) alongside completion time data and observational notes.

Results indicate that the PCG system consistently produced structurally valid and broadly enjoyable levels. Immersion and flow subscales averaged close to 4 out of 5 across all experience groups, and negative affect remained low throughout. The fish counter parameter successfully produced a perceptible difficulty gradient, but uncontrolled enemy spawn probability introduced within-configuration variability that affected perceived fairness. Prior platformer experience significantly moderated challenge and tension scores, with novice players struggling primarily at the mechanical level before layout complexity became a factor. The findings suggest that even a lightweight constraint-based PCG system can produce engaging platformer content, but that consistent player-centered quality requires constraints that account for individual skill, not only structural validity.

Keywords: procedural content generation, PCG, 2D platformer, level generation, player experience, flow, immersion, constraint-based generation, room templates, game design, replayability, Game Experience Questionnaire

How Procedural Content Generation affects player experience in 2D platformer games
Arvid Johansson, Irmeli Johansson, Zidane Nguyen, Jakob Ögnelod

Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Sammandrag

Procedurell innehållsgenerering (PCG) har blivit en väl etablerad teknik inom spelutveckling och erbjuder ökad innehållsvariation med mindre manuell design. Inverkan på spelupplevelse är dock fortfarande ett aktivt forskningsområde, särskilt inom genrer som ställer höga krav på design av miljö och rum, såsom 2D-plattformsspel. Denna uppsats undersöker hur PCG-baserade nivåer och design av dessa nivåer, påverkar spelarens upplevelse av flöde. Även hur regleringar och komplexitet av använd PCG-model påverkar spelarupplevelsen, samt hur tidigare erfarenhet av plattformsspel har significans spelarnas uppfattning av procedurellt genererat innehåll.

Ett 2D-plattformsspel vid namn `Gone Fishing` har utvecklats i Unity med ett begränsningsbaserat PCG-system inspirerat av rum-uppbyggnaden och PCG-systemet använt i Spelunky. Nivåer sätts ihop vid körning från ett bibliotek med 82 handgjorda rumsmallar fördelade över 15 öppningskategorier, sammankopplade via en slumpmässig ban-algoritm. Två konfigurationer utvärderades, ett lätt villkor med 3 fiskar och ett svårt villkor med 6 fiskar. Båda skilde sig åt i genererad nivåstorlek och komplexitet. En användarstudie genomfördes med 20 deltagare med varierande erfarenhet av plattformsspel. Studien gick ut på att en deltagare spelade båda konfigurationerna där spelarupplevelsen sedan mättes med hjälp av The Game Experience Questionnaire (GEQ) tillsammans med övriga observationsanteckningar.

Resultaten visar att PCG-systemet konsekvent producerade strukturellt giltiga och överlag uppskattade nivåer. Mätningen av skalorna tidskrävande och flöde uppnådde i genomsnitt 4 av 5 poäng i samtliga erfarenhetsgrupper, och negativ affekt förblev låg genomgående. Fiskantalet som parameter i PCG-systemet lyckades framgångsrikt skapa en märkbar grad av svårigheter, men okontrollerad startpunkt av fiender vid körning introducerade variation inom konfigurationerna som påverkade upplevd rättvisa. Tidigare erfarenhet av plattformsspel påverkade poängen för utmaning och spänning påtagligt, där nybörjare framför allt kämpade på det mekaniska planet innan nivåkomplexiteten blev en avgörande faktor. Resultaten tyder på att även ett lättviktigt begränsningsbaserat PCG-system kan producera engagerande platformsinnehåll, men för att uppnå ett spel av konsekvent spelarcentrerad kvalitet, kräver begränsningar som tar hänsyn till individuell skicklighet, inte enbart strukturell giltighet.

Nyckelord: procedurell innehållsgenerering, PCG, 2D-plattformsspel, nivågenerering, spelarupplevelse, flow, uppslukande, begränsningsbaserad generering, rumsmallar, speldesign, omspelningsbarhet, Game Experience Questionnaire

Acknowledgements

We would like to express our sincere gratitude to our supervisor, Aris Alissandrakis, for his guidance, feedback, and support throughout the course of this project. His expertise and constructive input were invaluable in shaping both the research direction and the final outcome of this thesis.

We also wish to thank all participants who volunteered their time to take part in the user study. Their participation and willingness to provide feedback were essential to the empirical foundation of this work.

Also, a huge thank you to Hugo Johansson for the help in making the music in the game.

Arvid Johansson, Irmeli Johansson, Zidane Nguyen, Jakob Ögnelod, Gothenburg, June 2026

Contents

List of Figures

List of Tables

1
Introduction

This chapter introduces procedural content generation (PCG) and its application in game development, presents notable examples of PCG in games, and outlines the purpose, research questions, and scope of this study.

1.1 Background

Procedural content generation (PCG) for games is a method that utilizes algorithms to create game content based on parameters provided by the developer. It is intended to serve as a tool to expedite game development and establish a quality standard for games [1]. PCG can be applied in several ways in developing video games, such as creating highly replayable experiences, such as infinite worlds or randomized levels, as well as non-playable characters [2]. Replayability stems from procedural generation producing different content across sessions, extending a game's lifespan without requiring additional manual design effort [3]. This approach allows for a high volume of content and variety. Still, it can result in the game feeling "soulless" or introduce unpredictable variables that negatively affect gameplay balance and player experience [4]. As a result, it is relevant to investigate whether and how players notice differences in procedural content generation, and how these differences affect player engagement and experience. Through research on this problem, we can gain new insights into player perception and engagement, which is valuable both from an academic perspective and for improving video game development.

From an academic perspective, PCG raises important questions about how algorithmically generated content is perceived by players and how it influences engagement, enjoyment, and the overall player experience [1], [4], [5]. Previous studies on PCG show that while it is an effective solution to the cost and complexity of game development, PCG can also influence player experience in unwanted ways, introducing challenges related to quality, control, and uncertainty [6]. Studying these effects contributes to a better understanding of human perception of algorithmically generated content and its effects on players. The findings and insights can, in turn, improve the design and implementation of procedural content generation systems in games.

Central to evaluating these effects is the concept of player experience, which encompasses factors such as immersion, flow, challenge, and enjoyment [7]. Flow, as defined by Csikszentmihalyi, refers to a state of complete engagement where the

challenge of a task is perceived as balanced with the player's skill level [8]. Understanding how PCG influences these dimensions of player experience is therefore essential for evaluating the effectiveness of procedural systems as design tools.

There are a variety of methods for generating game content in PCG, each with its own strengths, weaknesses, and intended purposes. From a technical perspective, it is important to consider factors such as controllability, reproducibility, performance, and playability [1]. Additionally, PCG addresses the rising cost and complexity of modern games by automating content creation. Analyzing these factors highlights the technical trade-offs in PCG, showing how algorithms can be designed to balance control, efficiency, and playability while generating diverse and engaging content. Despite the growing body of research on PCG [9], [10], [11], research on how procedural level design in 2D platformers affects player experience remains comparatively limited, motivating the investigation carried out in this study [4], [5].

1.2 Notable Examples of PCG in Games

Perhaps the most widely recognized example of PCG in games is Minecraft (Mojang, 2009), where algorithms generate effectively infinite three-dimensional terrain, demonstrating how PCG can produce large-scale variety without manual design effort [12], [13].

More directly relevant to this study is Spelunky, developed by Mossmouth in 2008, which takes a different approach by applying PCG specifically to 2D level design. Rather than generating terrain from scratch, Spelunky assembles handcrafted room templates into randomized layouts each run, ensuring that levels remain both varied and structurally sound [14]. This hybrid approach combines predefined templates with algorithmic assembly, giving designers meaningful control over the quality of individual rooms while still producing unpredictable level layouts. This approach is particularly relevant to the present study, as the PCG system developed here adopts the same architectural foundation, described further in Section 3.4.

More recent titles demonstrate that the template-assembly approach pioneered by Spelunky continues to influence contemporary game design. Dead Cells (Motion Twin, 2018) extends the concept by combining procedurally assembled rooms with hand-authored encounter setups, allowing designers to maintain tight control over combat pacing while still generating unpredictable level layouts [15]. Similarly, Hades (Supergiant Games, 2020) uses procedural room selection within a curated graph of possible connections, demonstrating how PCG can be embedded within a strongly authored narrative framework without sacrificing structural variety [16]. These examples illustrate a broader trend toward hybrid PCG systems that blend algorithmic generation with deliberate human authorship, a principle that also underpins the approach taken in the present study.

1.3 Purpose

The purpose of this thesis is to focus on examining player experience in a 2D side-view platformer game containing procedurally generated content. Procedural content generation has been widely adopted in game development to increase content variety and reduce manual design effort. Still, its impact on player experience remains a key area of study [1], [17]. This study investigates how PCG-level design influences perceived flow, how the complexity of procedural constraints shapes the player experience, and whether prior knowledge and experience with platformer games affect players' perceptions of procedurally generated content.

To investigate this, a 2D platformer game was developed and used as the basis for a user study involving 20 participants. Two configurations of the PCG system were evaluated, differing in constraint complexity, and player experience was measured through questionnaires and observation.

The evaluation focuses primarily on player experience metrics, complemented by an analysis of the PCG system's constraint complexity. User testing is conducted to examine how the layout of procedurally generated platform levels affects player navigation and, in turn, the experience of traversing the game. Similar approaches to evaluating PCG through player experience have been shown to provide meaningful insight into the strengths and limitations of procedural systems [4], [18]. The findings aim to contribute knowledge that supports game developers in making informed design decisions when working with procedurally generated platformer games.

1.4 Problem/Task

Platformer games are a well-established genre in which players control a character who navigates environments by running, jumping, and avoiding obstacles across platforms positioned at varying heights. The genre places strong emphasis on precise jump timing, obstacle spacing, and spatial layout, making level design a central factor in shaping player experience [2]. In handcrafted platformer levels, these elements are deliberately planned by designers to balance challenge, fairness, and player satisfaction.

Procedural content generation offers an alternative approach by automatically generating level content using predefined rules and algorithms, thereby increasing variation and replayability while reducing manual design effort [1], [5]. However, prior research highlights that procedurally generated levels may struggle to consistently achieve the same levels of fairness, readability, and user satisfaction as handmade content, particularly in genres that demand precise spatial design, such as platformers [4], [6], [18]. This poses a central design challenge: how to generate platformer levels procedurally without adversely affecting the player experience.

The primary task of this project is therefore to examine how procedurally gener-

ated level design in a 2D platformer affects player experience in terms of perceived flow, constraint complexity, and the influence of prior platformer experience. This is addressed by generating playable platformer levels using a PCG approach, collecting user data, and analyzing player responses. By grounded analysis in established research on PCG and player experience [1], [4], [6], [17], the project aims to identify key design trade-offs and determine how procedural methods can be refined to support player-centered design in platformer games.

The three research questions were selected to address complementary dimensions of this challenge: RQ1 targets the experimental outcome most directly linked to engagement, RQ2 examines the technical lever available to the designer, and RQ3 accounts for the individual differences that any fixed generative system must contend with. Together, they provide a structured basis for evaluating the PCG system both technically and from the player's perspective. Based on this, the following research questions have been formulated to guide the study:

RQ1 How does PCG level design affect the perceived flow in a 2D platformer game?
RQ2 How does the complexity of the constraints for the content created by procedural generation affect player experience?
RQ3 Is the player's experience with the game affected by previous knowledge and experience with platformer games?

1.5 Scope

This study focuses on applying PCG to level layout generation in a 2D platformer. The aim is to evaluate how different levels of constraint complexity affect player experience. The PCG system is evaluated in two configurations: a small and a large. The two configurations differ in the number of fish collectibles required to complete the level, which directly influences the size and complexity of the generated layout.

While many factors may influence how players perceive procedurally generated content, this study focuses primarily on prior gaming experience. This scope was chosen due to time constraints and the limitations of recruiting a larger and more diverse participant group.

Throughout this report, the terms small/large and 3-fish/6-fish are used interchangeably to refer to the two configurations of the PCG systems.

2

Method

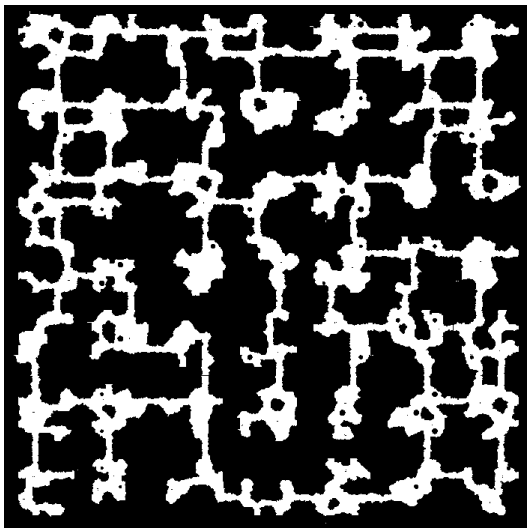
This chapter describes the research methodology used to investigate how procedural content generation affects player experience in a 2D platformer. It covers the research approach, participant recruitment, data collection instruments, and the methods for analyzing the collected data.

2.1 Research Approach

The project followed a design-centered research methodology that combined game development, procedural content generation, and user-study-based evaluation. The methodology was structured to support the investigation of player experience in procedurally generated level designs within a 2D platformer context. Such mixed approaches are commonly used in research on procedural content generation, as they allow both technical properties of generation systems and player-facing outcomes to be evaluated [1].

The way that was chosen to develop the procedural content generation according to 2D could be compared to Diablo's dungeon generation, where it incorporates pre-authored set pieces, much like the room templates that we use in our implementation of PCG [19]. This approach of having set pieces is close but not the same, as Diablo's levels are algorithmically generated on a per-tile basis while inserting handcrafted set pieces at specific positions, unlike the room template assembly used in Spelunky.

In Figure 2.1, there is a clear resemblance in how these games generate their layout, but also very different. Spelunky creates a path and then diverges from the path, and creates the rooms and Diablo that only checks if there is a path from the start to the end.



(a) A Diablo genereated chunk layout [20]

(b) A Spelunky generated room layout [21]

Figure 2.1: Example of different implementation of a chunk/room implementation.

2.2 Game Development

The game `Gone Fishing` was developed in 2D using Unity, which was chosen for its widespread use in the game development industry and its strong support for 2D game mechanics and procedural systems [2], [6]. The game contains procedurally generated levels built from a set of handcrafted room templates, which are assembled at runtime into a complete map layout using a constraint-based generation algorithm. Each room template defines the tile layout, hazard placement, and potential enemy spawn positions, while the generation algorithm determines how rooms are connected and arranged to form a playable level.

The development followed an iterative and agile workflow, where a minimum viable product (MVP) was implemented early to establish core mechanics such as movement, jumping, and collision handling [22], [23]. Level design and procedural generation were refined through continuous playtesting by the developers [24]. This iterative approach aligns with recommendations from prior PCG research, which emphasize repeated testing to ensure playability and control unintended spikes in difficulty [5]. Throughout development, particular attention was given to edge cases in the generation system, such as ensuring the player always spawns in a valid position, that all levels contain the required collectibles, and that a navigable path through the level always exists.

2.3 Participants

To evaluate player experience, a user study was conducted with participants ($n = 20$) of varying experience with platformer games. Prior research highlights the importance of accounting for player skill level when evaluating procedurally generated content, as perceptions of difficulty and fairness can vary significantly between novice and experienced players [18].

The participants were recruited through convenience sampling within our personal networks, targeting a diverse group of individuals with varying backgrounds, ages, and gaming experience. Each participant was tasked to play both the small and large configurations. To reduce order effects, the starting configuration was alternated between participants. Half of the participants began by playing with a small configuration, and the other half began playing with the large one. Playtesting sessions were conducted both in person and remotely, with each session lasting approximately 30 minutes. Remote participants were asked to share their screen and were observed and guided throughout the session in the same manner as in-person participants. This was done to ensure a consistent experience across both formats. Participants were encouraged to think aloud during play, verbalizing their thoughts and reactions as they navigated the generated levels. This provided qualitative insight into player decision-making and immediate responses to the level design that would not be captured by questionnaire data alone [25].

Before the playtesting session, participants completed a pre-study questionnaire collecting demographic information and self-reported gaming experience, including how frequently they played games and their familiarity with the platformer genre. This background data was used to contextualize individual differences in gameplay performance and experience.

2.4 Measuring Constraint Complexity

Constraint complexity in this study refers to the degree to which the PCG system restricts or guides level generation through explicit rules and validation requirements. A system with few constraints produces highly variable output but risks generating unplayable or unfair layouts, while a heavily constrained system produces more predictable but potentially repetitive content [1], [5]. In this study, constraint complexity was evaluated both technically and perceptually. From a technical perspective, the constraints of the generation system were documented and analyzed, including path validation, minimum room counts, enemy spawn thresholds, and collectible placement rules. The primary parameter differentiating the two configurations is the fish count, which controls the size and complexity of the generated level. From a player perspective, participants were asked in the post-study questionnaire (Tables A.2 - A.6) whether the levels felt consistent, fair, and readable, which provided an indirect measure of how well the constraints translate into perceived design quality [4], [18].

2.5 Measuring Player Experience and Flow

After completing the play sessions, participants completed a post-game survey using the Game Experience Questionnaire (GEQ), a widely adopted and extensively validated instrument for measuring player experience in game studies [26]. The GEQ provides a structured and reproducible means of measuring dimensions such as immersion, flow, and challenge, making it well-suited for comparing experiences across the small and large configurations.

Flow, as defined by Csikszentmihalyi [8], refers to a state of complete immersion and engagement where challenge and skill are perceived as balanced. In this study, flow is operationalized using the flow-related subscales of the GEQ, which include items on immersion, challenge balance, and sense of control. Participants rated these items on a Likert scale after each play session, enabling a comparison of perceived flow across the small and large configurations. Gameplay metrics such as time spent per section and number of failed attempts were used as secondary indicators, as prolonged struggle or near-instant completion may both indicate a disruption of the flow state [4].

2.6 Data Analysis

The collected data was analyzed to compare player experience across procedurally generated levels and to assess procedural content quality under algorithmic constraints. Gameplay metrics and survey responses were examined to identify trends related to difficulty, fairness, and playability. Descriptive statistics were used to summarize quantitative responses, and patterns across player experience groups were identified to evaluate how well procedurally generated content supports players with differing skill levels.

Qualitative responses were thematically analyzed, grouping recurring observations and opinions to identify common points of friction or enjoyment across both configurations. These themes were used to contextualize and enrich the interpretation of the quantitative results, providing a more complete picture of how each design approach affects the player experience.

Comparisons across player experience groups further evaluated whether procedurally generated content scales appropriately in perceived difficulty for both novice and experienced players. This analysis follows established practices in PCG evaluation, where player-centered metrics are used to assess the effectiveness of procedural systems as design tools [1].

2.7 Limitations

Several limitations should be acknowledged when interpreting the results of this study. First, the time available to conduct the user study was limited, as the study

had to be completed within a fixed project period. This limited the number of playtesting iterations that could be conducted, leaving little opportunity to make design adjustments based on the collected feedback and then test again with new participants. Ideally, a study of this kind would benefit from multiple rounds of playtesting, with findings from each round informing refinements to both the game and the study design. With only one iteration possible, some issues with the game design or the procedural generation system may not have been identified and addressed before the final evaluation. Secondly, the study relies on convenience sampling within a university environment, meaning the participant group is unlikely to be representative of the broader gaming population. The sample may skew towards younger, more technically experienced individuals, which could influence perceptions of difficulty and fairness in ways that do not reflect those of a general audience. Third, the randomness inherent in procedural content generation means that no two participants necessarily played an identical procedurally generated level, which introduces variability that makes direct comparison between participants more difficult. While the validation system ensures a minimum standard of playability, the qualitative feel of generated levels may vary considerably between runs.

2.8 Societal and Ethical Aspects

While procedural content generation is not concerning or harmful in societal aspects, it can, from an ethical perspective, unintentionally produce content that is unfair or overly difficult, which can negatively affect player experience and exclude certain groups of players. In a platformer game with poorly designed jumps or obstacles that exceed a player's reasonable capabilities, the results of a procedurally generated game could raise ethical concerns about fairness, accessibility, and player trust. These are relevant when evaluating the user experience, as players generally expect games to provide consistent, solvable challenges.

The project involves user studies, which raise concerns about informed consent, voluntary participation, and the responsible handling of collected data. All participants were informed of the study's purpose, and the collected data were anonymized and used solely for research purposes, in accordance with established research ethics guidelines [27].

From a societal perspective, the findings of this study could inform how developers use PCG in games targeted at diverse audiences, helping to avoid designs that unintentionally exclude or frustrate certain groups of players.

3

Game Development

This chapter describes the development of the game created for this study, including the design decisions that shaped its mechanics, visual identity, and procedural generation system. The game's narrative follows a penguin on a continuous journey home to Antarctica. Having escaped human captivity, the penguin must find fish to survive as it makes its way back to its flock. The player's goal of collecting fish scattered across each level is thus grounded in this broader narrative of survival and return, giving the gameplay loop a purposeful context beyond mere point-scoring. The game serves as the primary instrument for evaluating player experience, and the decisions made during its development are therefore directly relevant to the study's validity.

3.1 Game Design Decisions

The game `Gone Fishing` was developed in Unity, chosen for its robust support for 2D tilemaps and procedural systems, as well as its widespread use in both industry and academic game development research [6].

After initial research and a field review of PCG, we decided to focus the project on constraint-based level generation in a 2D platformer to examine its effects on player experience. During our research, we found that the field of PCG within the 2D platformer world was relatively limited. We decided to use a constraint-based approach to isolate which parameters and elements of the game to focus on and reduce variance across playtesting sessions. Originally, the plan was to implement a power-up system with different abilities. Still, after careful consideration, it was concluded that the system would involve unnecessary complexity and shift the focus away from the PCG system. For the same reason, a tutorial environment was implemented to ensure that unfamiliarity with the controls would not confound participants' responses to the PCG system during the user study.

3.2 Core Game Loop

The game is a 2D side-view platformer in which the player navigates procedurally generated levels to collect all fish scattered throughout them. The core loop consists of the player spawning in a designated start room, exploring the level, collecting fish, avoiding hazards, and reaching the exit once all collectibles have been gathered. The player has access to basic directional movement and a dash, which was included to

provide a tool for both traversal and enemy avoidance without introducing additional complexity. Two types of enemies are present in the levels alongside damaging blocks, which together serve as the primary obstacles the player must navigate around. The decision to keep the mechanics minimal was deliberate, as the project focuses on the PCG system rather than the mechanics themselves.

Upon collecting all fish, the player is presented with a completion screen offering two options: returning to the main menu or proceeding to the next level, which generates an entirely new level of equal difficulty using the same fish count configuration. There is no dedicated failure screen; if the player dies, they are respawned and continue attempting the same level that was generated. This decision was deliberate, as introducing a lose condition with level regeneration on death would have introduced additional variables into the user study by exposing players to different generated layouts mid-session.

Before entering the procedurally generated levels, the controls were verbally explained to each participant, who was then given time to explore a simple, isolated environment (see Figure 3.1) to familiarize themselves with the movement, jump, and dash mechanics.

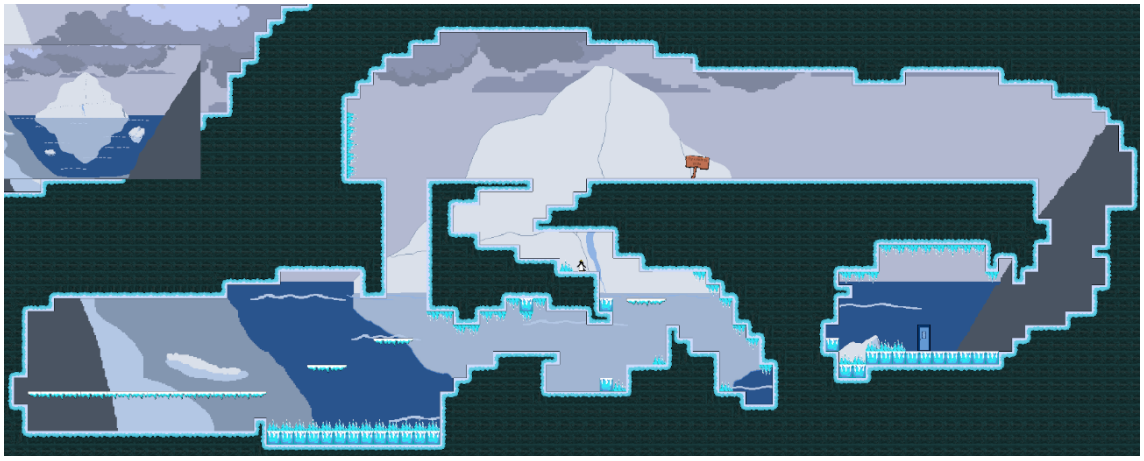


Figure 3.1: The handcrafted tutorial level.

3.3 Visual and Audio Design

All visual assets in the game, including character sprites, enemy sprites, tiles, and collectibles, were created by hand rather than sourced from existing asset libraries. This decision was made to ensure visual consistency across all game elements and to avoid introducing bias from asset quality into the player study [18]. Using pre-made assets from different sources risks visual inconsistency that could distract players or influence their experience in ways unrelated to the PCG system being evaluated [4]. By creating all sprites manually, the visual presentation remains uniform and neutral, keeping player attention focused on navigating the procedurally generated levels rather than on the aesthetics of individual assets.

The visual style was kept deliberately simple and readable, prioritizing the clarity of the game elements over the visual complexity. Clear visual distinction between traversable space, solid blocks, hazards, and enemies was considered essential, as players need to parse the layout of procedurally generated rooms quickly, which they have not seen before [2]. This aligns with the broader design philosophy of the project, where all decisions were made in service of evaluating the PCG system rather than creating a polished commercial product. Original music was composed for both the main menu and in-game environments to support player immersion, as audio has been identified as a contributing factor to overall game experience [26].

3.4 Procedural Content Generation System

The procedural content generation system is based on the generative approach used in the game Spelunky [14], using an existing implementation [21] as a starting point. The core concept of defining room templates as pixel-based layouts, along with the overall code structure consisting of a configuration file, room, level, and level generator classes, was adopted from this implementation and then modified and expanded to fit this project's needs. Spelunky was chosen as a reference due to its well-established PCG approach. The primary changes include the path generation algorithm, the number of room templates, and the specific templates used, suggesting that the original implementation served mainly as an architectural foundation rather than a direct solution.

The system generates levels by first determining a path through each level using a random-walk algorithm [28]. Unlike the original Spelunky implementation, where the path is generated downwards, our system generates paths in all directions. The generation begins by creating a main path with a fish placed at the end of it. Additional branch paths are then generated, each starting from a random point along the main path and ending with a fish. The number of fish determines the number of branches. This branching structure was chosen over a single linear path to ensure that the fish are distributed across the routes, requiring players to navigate in multiple directions rather than following a single linear path through the level. To ensure generated levels meet minimum playability requirements, the system employs a retry-based validation loop. After each generation attempt, the level is checked to confirm that a valid entrance room exists and that the correct number of fish rooms has been assigned. If either condition fails, the level is discarded and regenerated, up to a maximum of ten attempts. This approach prioritizes correctness over speed, and in practice, levels were consistently produced within the first few attempts.

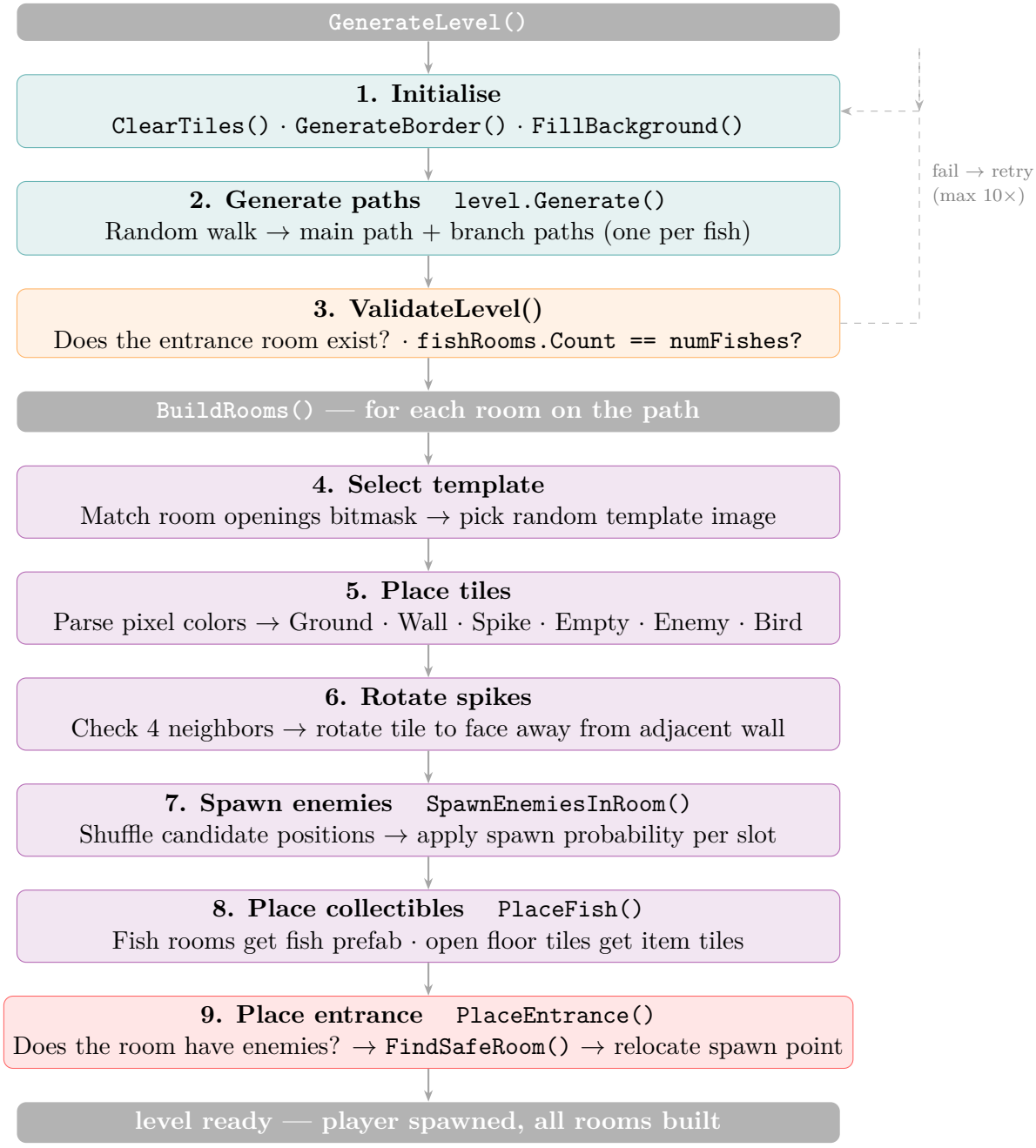


Figure 3.2: Overview of the procedural level generation pipeline, from path generation through room building and spawn safety resolution. The dashed line indicates the retry loop: if validation fails, the system discards the level and regenerates from scratch, up to ten attempts.

3.5 Room Templates Generation

Each room in the PCG is generated using predefined room templates. Each room template is of equal size, 16 x 10 tiles, where each pixel in the template image corresponds to one tile in the game world. The color of each pixel determines what is placed at that position. The mapping between pixel colors and tile types

is visually illustrated in Figure 3.3. The tile types in the room templates follow this configuration: solid blocks (black), spikes (red), flying enemies (blue), ground enemies (yellow), and empty space (white).

Spike tiles are additionally rotated at runtime by inspecting each tile's four cardinal neighbors; a spike is oriented to face away from any adjacent solid wall, ensuring that the hazard's direction is always consistent with the room geometry regardless of which template is selected.

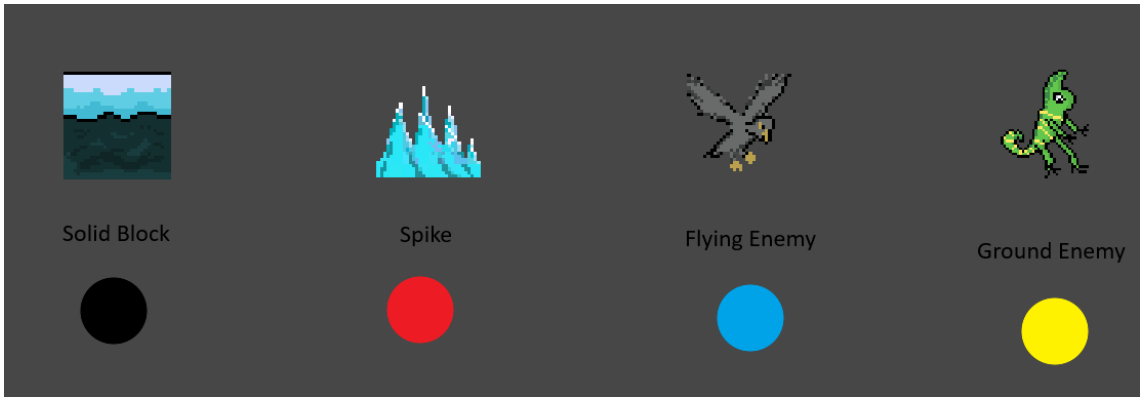
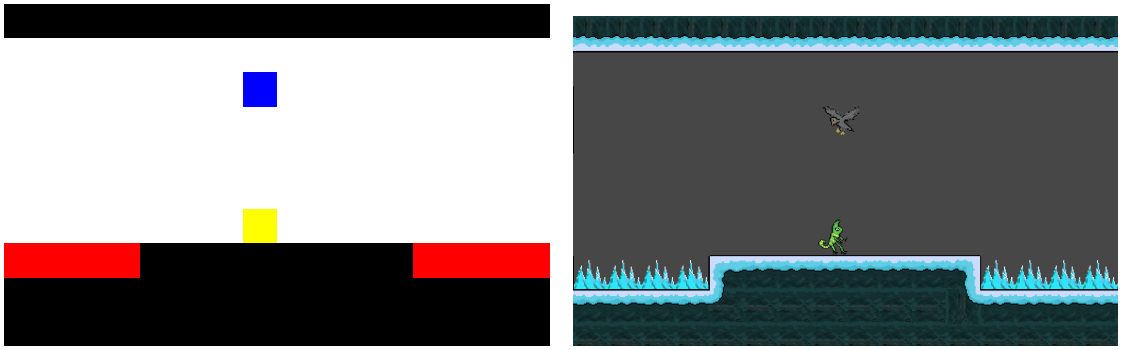


Figure 3.3: Visual representation of the tile types

To illustrate this process, Figure 3.4a shows an example of a room template represented as a 16-by-10-pixel image. During level generation, the template is converted into a playable room as shown in 3.4b.



(a) Room template represented as a 16 × 10 pixel image.

(b) Generated in-game room based on the template.

Figure 3.4: Example of a room template and its generated room.

Room templates are divided into 15 categories based on their openings, meaning which sides of the room connect to neighboring rooms. This is achieved using a bitmask system, where each bit in a binary number represents a Boolean condition. In this case, four bits correspond to the four possible openings of a room, left, right, up, and down, where the value of 1 indicates an opening and 0 indicates a solid block. Each binary combination is converted into an integer value that represents a

unique room configuration. Excluding the configuration with no neighboring rooms, this results in 15 categories, shown in Table 3.1. Each category contains multiple room templates, from which the level generator randomly selects during generation. This approach ensures that every room remains traversable.

Integer	**Binary representation**	**Openings**
1	0001	Left
2	0010	Right
3	0011	Left + Right
4	0100	Up
5	0101	Left + Up
6	0110	Right + Up
7	0111	Left + Right + Up
8	1000	Down
9	1001	Left + Down
10	1010	Right + Down
11	1011	Left + Right + Down
12	1100	Up + Down
13	1101	Left + Up + Down
14	1110	Right + Up + Down
15	1111	All directions

Table 3.1: The 15 bitmask categories and their corresponding openings

The 15 categories contain a combined total of 82 room templates, distributed unevenly across categories to reflect the varying frequency with which each opening configuration appears during generation. Categories with more directional combinations, such as categories 13 and 14 (Left + Up + Down and Right + Up + Down), contain the most templates, with 8 each. The full distribution across categories is shown in Table 3.2. Within each category, the generator selects a template at random, meaning that players are unlikely to encounter identical room layouts even across multiple runs. Also, there is a chance of new layouts appearing that haven't appeared in previous generations.

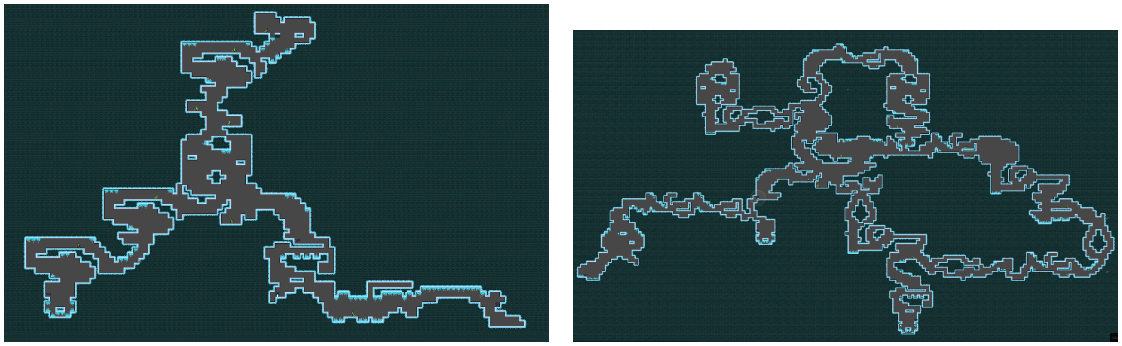
Category	Number of Templates
1	6
2	6
3	8
4	4
5	5
6	5
7	3
8	5
9	6
10	4
11	4
12	7
13	7
14	7
15	5
Total	**82**

Table 3.2: Number of room templates per opening category.

3.5.1 Constraint System

The constraint system is intentionally kept minimal, with only the fish count serving as the parameter controlling level generation. The map size is directly correlated with the number of fish required to complete the level. The generated path length increases according to the fish count, with one additional path segment added per fish. Two configurations were used for the user study. One small configuration with 3-fish and one large configuration with 6-fish, producing levels of different sizes and complexity. The number of fish was determined through internal playtesting during development, whereas when the difficulty changes the difference in length in branches and in the amount of paths created is increased for a bigger variance in level generation. Multiple levels were tested and timed to ensure that both configurations produced levels that could be completed in a reasonable timeframe while still offering a noticeable difference in size and complexity.

Each fish added to the configuration extends the generated path by one additional segment of randomly selected rooms, meaning the 6-fish configuration produces a level roughly twice the size of the 3-fish configuration in terms of total rooms traversed, while also increasing the probability of encountering more enemies due to the greater number of rooms generated.



(a) Small configuration, 3-fish **(b)** Large configuration, 6-fish

Figure 3.5: Example of generated maps for the small and large configurations.

Although the enemy and spike positions are defined within the room templates themselves, generating a unique template for every possible enemy configuration would result in an unmanageable number of templates. Instead, enemy density is determined at runtime through a configurable spawn probability applied to each candidate position defined in the template. This means that the same template can produce varying levels of difficulty between runs. Additionally, the system includes a runtime safety check ensuring the player never spawns in a room containing enemies, relocating the spawn point to a safe room elsewhere on the path if necessary. These runtime behaviors represent the procedural layer applied on top of the handcrafted template content.

4
Results

This chapter presents the results of the user study conducted with all gathered participants ($n = 20$), all of whom played both the 3-fish (Small) and 6-fish (Large) configurations of the procedurally generated levels. Results are organized into four subsections outlined in the thesis structure: Game Experience Questionnaire responses, level-generation observations, path-generation characteristics, and perceived differences in difficulty between configurations.

4.1 The Game Experience Questionnaire (GEQ)

The Game Experience Questionnaire (GEQ) was completed by every participant following their play sessions, score [26], and using the Likert evaluation system for having a basis in how we should evaluate the data that we discuss [4].

4.1.1 Overall Experience

Preliminary results suggested that participants generally responded positively to the overall game experience, though scores on the challenge subscales varied notably between the two configurations and across player experience groups. The flow subscale, which indicates the balance between perceived skill and challenge, showed the most variability, particularly among participants who self-reported little prior experience with 2D platformer games. The Tables 4.1 reads as Pos. = Positive affect, Comp. = Competence, Tens. = Tension, Imm. = Immersion, Neg. = Negative affect, Chall. = Challenge. Note that Tension and Negative affect are negatively valenced: higher scores indicate more tension and frustration, respectively, and 4.2 reads as follows: experience group (Beginner, Casual, Experienced), and configuration (Small, Large). Pos. = Positive affect, Comp. = Competence, Tens. = Tension, Imm. = Immersion, Neg. = Negative affect, Chall. = Challenge. Note that Tension and Negative affect are negatively valenced: higher scores indicate more tension and frustration, respectively.

Config	Pos.	Comp.	Flow	Tens.	Imm.	Neg.	Chall.
Small (3-fish)	3.88	3.42	3.98	2.88	3.92	1.88	2.80
Large (6-fish)	3.88	3.52	4.22	3.25	4.00	1.90	3.12

Table 4.1: Mean GEQ subscale scores for the small and large configurations across all participants ($n = 20$).

Group	Config	Pos.	Comp.	Flow	Tens.	Imm.	Neg.	Chall.
B ($n = 7$)	S	3.86	3.36	3.86	3.43	4.29	1.57	3.57
	L	4.29	3.50	4.50	2.71	4.21	1.71	3.07
C ($n = 8$)	S	4.06	3.75	4.19	2.56	3.75	2.12	2.56
	L	3.38	3.31	3.94	3.44	3.69	2.00	3.19
E ($n = 5$)	S	3.60	3.00	3.80	2.60	3.70	1.90	2.10
	L	4.10	3.90	4.30	3.70	4.20	2.00	3.10

Table 4.2: Mean GEQ subscale scores by experience group.

Full per-participant GEQ subscale scores split by configuration are provided in Tables A.3 and A.4 in the appendix.

4.1.2 Effect of Prior Platformer Experience

Participants were grouped according to their self-reported familiarity with 2D platformer games, collected via the pre-study questionnaire. Those with little or no prior experience with the genre reported lower scores on the flow and enjoyment subscales, and higher scores on the challenge subscales, compared with participants with moderate or extensive platformer experience. This pattern was consistent across both difficulty configurations, though it was more pronounced in the 6-fish condition.

A few beginner participants commented during observation that they found it difficult to navigate the platformer mechanics themselves, independent of the procedurally generated layout. Traversing platforms, timing jumps, and avoiding hazards such as spikes and enemies proved challenging at a mechanical level before the structural qualities of the generated levels were a factor. This suggests that for this participant group, the baseline difficulty of the game mechanics may have influenced GEQ responses as much as the PCG system itself.

4.2 Level Generation

The level generation system performed consistently across all sessions. In all observed runs, levels were successfully generated within the maximum of ten allowed attempts, and in practice, the majority of levels were produced on the first or second attempt. No participant encountered a generation failure during their session.

The bitmask-based room template system ensured that all generated rooms were structurally compatible with their neighbors, meaning no invalid connections, such as a solid wall abutting an open passage, were observed. All participants spawned in a valid start room free of enemies, consistent with the runtime safety check described in Section 3.4.

The 82 available room templates distributed across 15 opening categories provided

sufficient variety that participants were unlikely to traverse identical room sequences within a single session. Across the full set of playtesting sessions, no two participants reported playing what appeared to be the same level layout, although direct comparison of generated layouts was not systematically recorded.

4.3 Path Generation

The random walk algorithm used to generate the main and branch paths produced levels with varied spatial layouts across sessions. Because the systems generate paths in all four cardinal directions, unlike Spelunky's downward-only, the resulting levels exhibited a wider range of spatial configurations, including horizontal, vertical, and diagonal traversal patterns.

The main path, which always terminates with a fish collectible, was consistently navigable in all generated levels. Branch paths, generated from random points along the main path and each ending with a fish, contributed additional exploration space proportional to the fish count configuration. In the 3-fish configuration, this resulted in a compact level with limited branching, while the 6-fish configuration produced considerably larger levels with multiple branch paths extending in different directions from the main route.

During observation, it was noted that some participants, particularly those less experienced with the platformer genre, had difficulty determining which direction to navigate after entering a new room. The multi-directional path generation, while increasing structural variety, occasionally produced layouts in which the intended route was not immediately apparent, particularly in rooms with multiple open sides.

4.4 Difference of Difficulty

The two configurations, 3-fish (Small) and 6-fish (Large), produced a perceivable difference in difficulty among participants. The 6-fish configurations generated levels approximately twice the size of the 3-fish configuration in terms of total rooms, and exposed players to a higher number of enemies due to the greater number of rooms traversed.

Most participants reported that the 6-fish configuration felt notably more demanding. This was reflected both in observed play behavior: participants spent more time per session, made more failed traversal attempts, and more frequently retraced their steps; and in post-session questionnaire responses, where challenge scores were consistently higher for the 6-fish condition across all experience groups.

Among beginner participants, the difficulty gap between configurations was particularly pronounced. Several novice players struggled significantly with the 6-fish configuration, with enemies and spike placements described as overwhelming relative to their navigation ability. In contrast, more experienced participants generally

managed both configurations, with some reporting that neither felt particularly challenging from a mechanical standpoint.

The runtime enemy spawn probability system, which applies a configurable spawn chance to candidate positions defined in each room template, contributed to within-configuration variability. Two participants playing the same fish-count configuration could encounter meaningfully different enemy densities, which introduced some inconsistency in perceived difficulty that is independent of the structural layout of the generated path. This variability was more noticeable in the 6-fish configuration due to the larger number of rooms (and therefore more candidate spawn positions) involved.

4.5 Qualitative Observations

In addition to the quantitative GEQ data, observational notes and spontaneous verbal comments were recorded during play sessions. Several recurring themes emerged across participants. Several players, primarily in the casual and experienced groups, commented that certain rooms felt unfair, citing narrow corridors in which enemies left insufficient space to pass without taking damage. This was more frequently reported in the 6-fish configuration, where the greater number of rooms increased the probability of encountering a narrow room with a high enemy spawn density.

Conversely, a consistent theme of accomplishment was noted across all experience groups when players successfully navigated a difficult room or completed a level they had initially struggled with. Several participants made unprompted remarks to this effect, suggesting that the hazard density, while sometimes perceived as unfair, also contributed to a sense of reward upon success. This aligns with the GEQ challenge scores, which remained moderate-to-high across groups without producing correspondingly high negative affect scores. Looking at Table 4.3 and Table 4.4, we can see the difference in the negative and the positive item scores from the GEQ. Table 4.3 shows that the negative scores are showing a certain difficulty from the players in some areas in the question of "I felt frustrated" and "I felt irritated" that showed the highest averages, but also seeing that the players also scored low on the scale in "I felt bored" so while the players felt irritated and frustrated during the playtest they were still invested in the game and didnt feel like the game was boring.

Question	**1st Playthrough**	**2nd Playthrough**
I felt frustrated	3.30	3.10
I felt stressed	2.65	2.30
I felt bored	1.35	1.95
I felt irritated	2.85	3.00

Table 4.3: Mean scores per negative GEQ item across all participants.

The scores being shown in Table 4.4, the highest scoring questions were "I forgot

about the time", "The game was visually good looking" and "I was busy with the game". The Likert scale of points that we are using in this suggests that the game, while using the procedurally generated content, still made the playtesters immersed in the game forget about other variables like the time limit of the playtest(Max 20min). This shows that using PCG in the game design and the game didn't take away the addictiveness of the game [4]. This overall response of the averages being around the score of 4 in our scale(1-5) shows great promise for the practice and that the model that was being used could be replicated in other projects, but with more refining to work on the areas that either scored too high if it was a negative item or too low on a positive item.

Question	**1st Playthrough**	**2nd Playthrough**
I thought it was fun	3.95	3.95
I felt skillful	3.05	3.40
I forgot about the time	4.15	4.05
The game felt impressive	3.90	3.55
The game felt tiresome	2.05	2.20
I was busy with the game	4.05	4.15
It felt like i succeeded	3.95	3.50
The game was visually good looking	4.20	4.20
I thought it was hard	3.30	3.60
I felt completed	3.95	3.65

Table 4.4: Mean scores per positive GEQ item across all participants.

The GEQ mean scores being shown in Tables 4.5 and 4.6 show the averages in the different dimension items from the questionnaire. Table 4.5 shows the mean scores in the small configuration, and shows which starting configuration the playtesters started with. The playtesters who started with the large configuration showed a mean score of 3.73 while playing the small configuration, and the playtesters starting with the small configuration showed a mean score of 4.06 while playing the small configuration in the Positive item. Higher scores for Tension and Negative indicate more tension and frustration, respectively, in Tables 4.5 and 4.6.

Subscale	**Small**	**Large**
Positive	4.06	3.73
Flow	3.17	3.64
Competence	3.50	4.36
Tension	3.16	2.64
Immersion	4.44	3.50
Negative	1.67	2.05
Challenge	3.27	2.41

Table 4.5: Mean GEQ subscale scores for the small configuration.

Subscale	Small	Large
Positive	3.89	3.86
Flow	3.22	3.77
Competence	3.78	4.59
Tension	3.56	3.00
Immersion	4.33	3.72
Negative	2.11	1.73
Challenge	3.61	2.72

Table 4.6: Mean GEQ subscale scores for the large configuration.

The results from the GEQ indicate that participants generally responded positively to the procedurally generated levels. As shown in Figure 4.2 follow the configuration of (B=Beginner, C=Casual, E=Experienced) and (S=Small, L=Large). Each box shows median, IQR, and whiskers to min/max. Tension and Negative affect are negatively valenced – higher scores indicate more tension and frustration, respectively. The dimensions of Immersion and Flow received the highest average scores across all experience groups and both difficulty configurations, consistently approaching 4 out of 5. This suggests that the PCG system successfully produced levels that engaged players and held their attention, regardless of prior platformer experience. The Negative dimension, comprising items related to frustration and irritation, remained low across all groups and both conditions, indicating that the procedurally generated content did not produce a broadly negative emotional response.

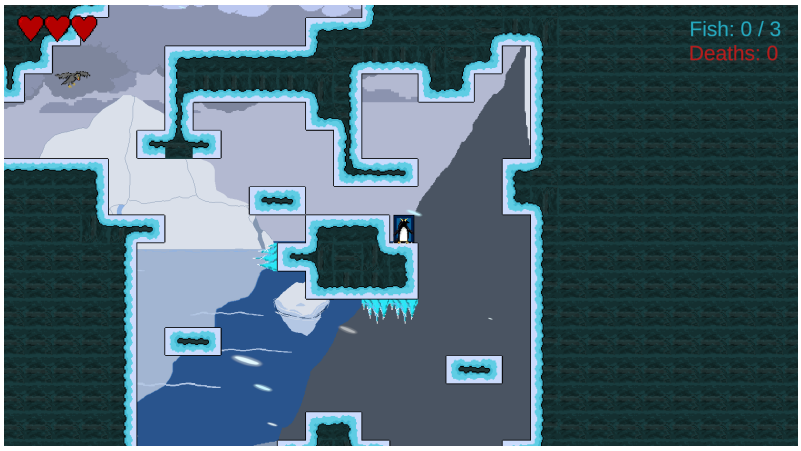


Figure 4.1: PCG generated spawn location in a 3-fish configuration.

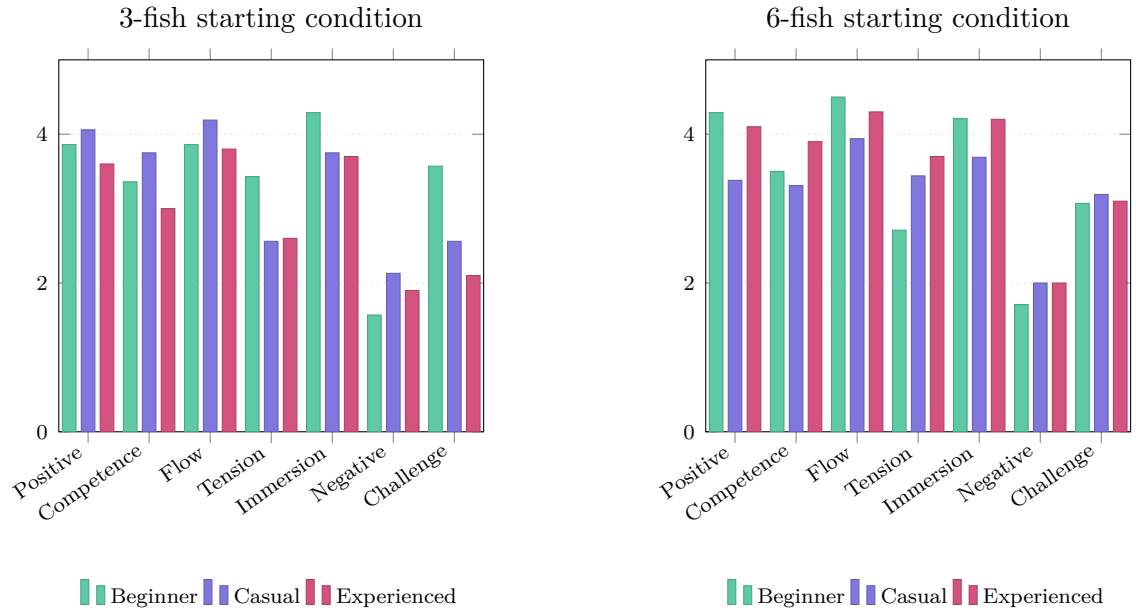


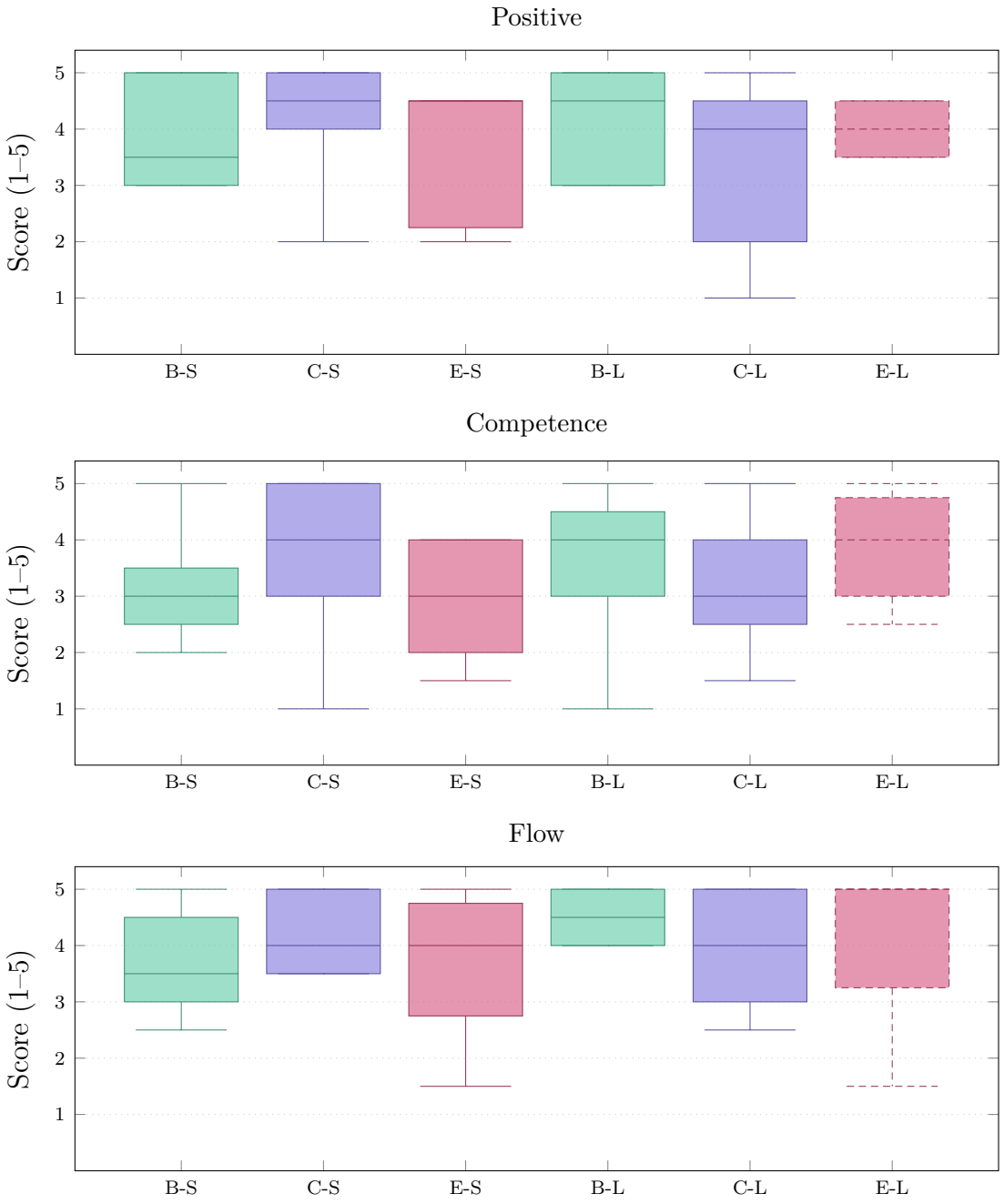
Figure 4.2: GEQ dimension scores by experience level.

The most notable variation between groups appears in the Challenge and Tension dimensions. In the 3-fish condition, beginner participants reported substantially higher challenge scores than experienced participants, reflecting the expected effect of prior platformer experience on perceived difficulty. This pattern persisted in the 6-fish condition, where the difficulty gap between experience groups became more pronounced. Interestingly, experienced participants reported higher tension scores than both beginner and casual participants in the 6-fish condition, a result that is discussed further in Section 5.1.

Comparing the two configurations, the 6-fish condition produced a noticeable shift in scores across several dimensions, most clearly in challenge and tension. Casual participants remained the most stable group between conditions, with relatively small score differences across dimensions, while Beginner and Experience participants each showed stronger responses to the increased difficulty. This indicates that the fish count parameter, despite being the sole constraint governing level size and complexity, was effective in producing a perceptible and meaningful difference in difficulty between the two configurations.

The boxplots in Figure 4.3 reveal within-group variance that is not captured by the group averages alone. The Flow dimensions show the widest spread across all groups and conditions, confirming that individual players within the same experience group had meaningfully different flow experiences. This variability is consistent with the nature of the procedural system. Since enemy density is determined at runtime through a configurable spawn probability rather than fixed by the constraint system, two participants playing the same fish-count configuration may have encountered substantially different levels of hazard. The Negative dimension, by contrast, shows tight distributions centered on low values across all groups, indicating that the low average frustration scores are reliable and not the result of a skewed dis-

tribution. The Positive dimensions display a moderate spread, particularly among Casual participants, suggesting that enjoyment within this group varied more than the group average would imply.



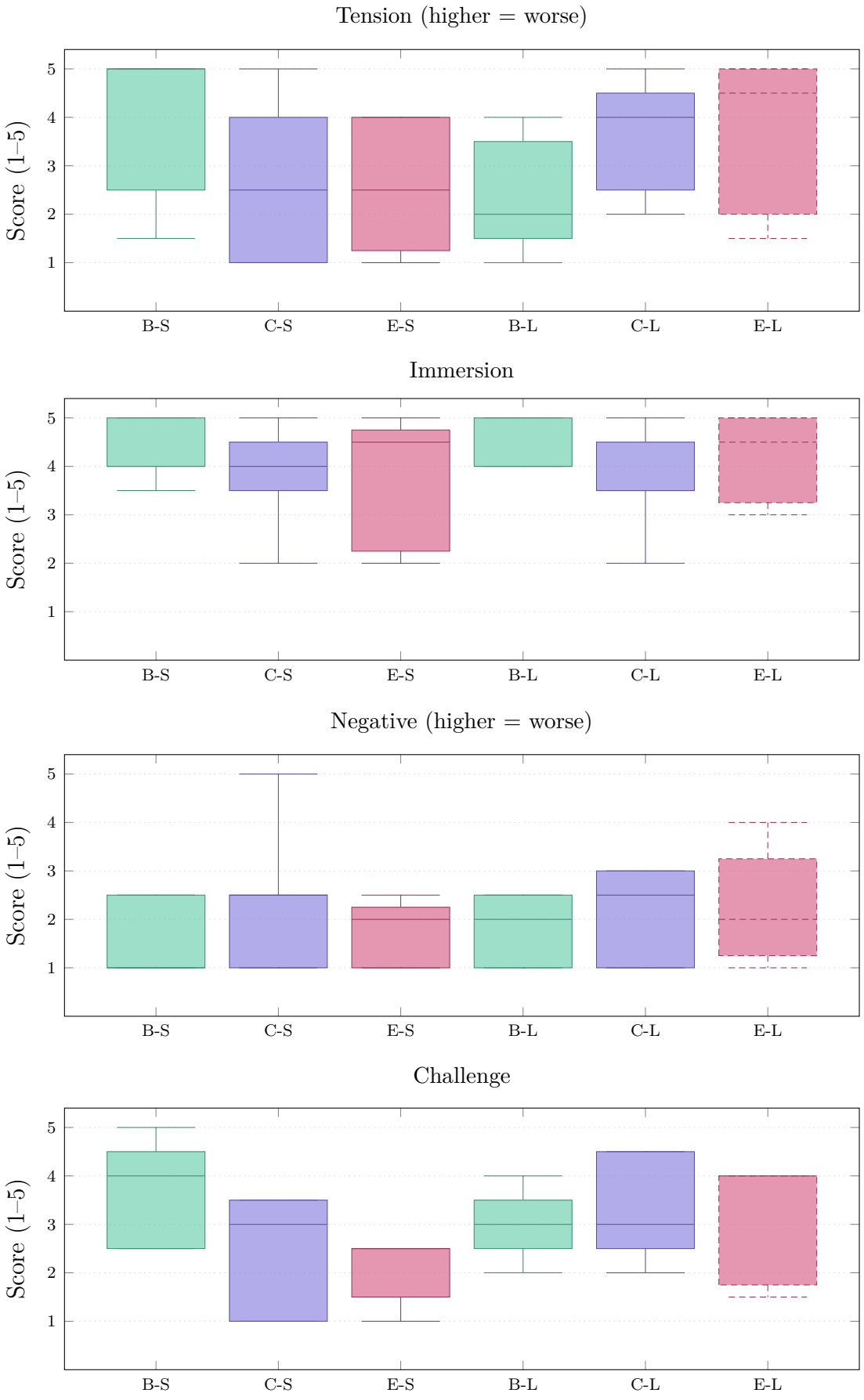


Figure 4.3: Boxplots of all GEQ subscales by experience level.

5

Discussion

This chapter discusses the findings of the study in relation to the three research questions, evaluates the methodology, and assesses the validity and generalizability of the results. The chapter concludes with directions for future work that addresses the limitations identified throughout the study.

5.1 Results Evaluation

The results of this study offer descriptive insight, interpreted cautiously given the sample size, into how procedural content generation influences player experience in a 2D platformer context, addressing each of the three research questions formulated in Chapter 1.

Overall, the results indicate that the PCG system performed well across both configurations. Immersion and flow scores consistently approached 4 out of 5 across all experience groups, and negative affect remained low throughout, suggesting that the procedurally generated levels were broadly engaging and did not produce a strongly negative emotional response. However, the results also reveal that structural validity alone is insufficient for consistent player-centered quality. The most significant source of variability across all three research questions was the uncontrolled enemy spawn probability, which introduced difficulty variance that the constraint system did not govern.

RQ1 - Flow in PCG-generated levels. The GEQ flow subscale showed the highest variability across participants, which aligns with Csikszentmihalyi's model of flow as a state that is disrupted when challenge and skill are mismatched [8]. The procedural generation system, while consistently producing structurally valid and playable levels, could not account for individual differences in player skill when determining enemy density or spatial layout. This is an inherent limitation of constraint-minimal PCG: with only fish count as the governing parameter, the system cannot dynamically calibrate challenge to the individual. As a result, the same generated level could place a novice player in a state of anxiety, while an experienced player would find it insufficiently challenging, both conditions representing a disruption of flow [8]. This pattern was most visible in the 6-fish configuration and is consistent with prior findings that PCG systems without adaptive difficulty mechanisms risk producing uneven flow experiences [4], [18].

The wide interquartile range observed in the Flow dimension (Figure 4.3) is consistent with Sweetser and Wyeth's GameFlow model, which identifies clear goals and immediate feedback as prerequisites for sustained flow, both of which are harder to guarantee in procedurally assembled environments [29]. The multi-directional path generation also contributed to flow disruption in some cases, where the room templates didn't link up because the path generation would sometimes traverse diagonally to check if there was a viable path, which created flow disruption. Unlike Spelunky's downward-only traversal, which communicates directional intent implicitly, the four-directional random walk occasionally produced layouts in which the intended route was unreadable, particularly in rooms with multiple open sides. This navigational ambiguity caused some participants (especially beginners) to backtrack or pause, breaking the exploratory momentum that supports flow [2]. This is consistent with Shaker et al. [9], who note that player disorientation is a common failure mode in PCG systems that lack explicit guidance structures to direct players through generated space.

RQ2 - Constraint complexity and perceived quality. The constraint system used in this study was deliberately minimal, with fish count as the sole parameter. From a technical standpoint, this simplicity produced consistent, playable output: levels were generated successfully on the first or second attempt in all observed sessions, all rooms were structurally compatible via the bitmask system, and no participant encountered any generation errors. This confirms that even a low-complexity constraint system can guarantee baseline playability, supporting findings from prior PCG research that validation-based approaches are effective at filtering unplayable outputs [1], [5].

However, the runtime enemy spawn probability introduced variability that the constraint system did not govern. Two participants playing the same fish-count configuration could encounter meaningfully different enemy densities, which caused perceived difficulty to diverge in ways unrelated to structural layout. This within-configuration inconsistency complicates the interpretation of difficulty-related GEQ scores, as participants' responses may reflect enemy density rather than level structure. For future constraint design, explicitly bounding enemy spawn probability per configuration, or coupling it to the fish count, would reduce uncontrolled variance and improve the validity of comparisons across sessions.

Qualitative feedback supports the view that constraint complexity affects fairness perception. Several participants commented that certain rooms felt "unfair" due to enemies appearing in spaces too narrow to dodge through. This is consistent with the ethical concern raised in Section 2.8: procedurally generated obstacle placement that exceeds a player's reasonable capability undermines trust in the system [4]. Since enemy positions are baked into room templates and spawn probability is applied at runtime, the current system offers no mechanism to prevent a worst-case combination of narrow room geometry and high enemy density from co-occurring. A constraint linking spawn probability to room traversal width would address this.

From a design ethics standpoint, Barlet and Spohn argue that procedurally generated obstacle placement that cannot be tuned to player capability risks systematic exclusion of players with lower baseline motor skill [30]. This connects directly to the ethical concern raised in Section 2.8 and suggests that future constraint design should treat spawn probability as a first class parameter rather than a runtime afterthought.

RQ3 - Effect of prior platformer experience; The Effect of prior platformer experience is reflected in Table 4.6, where results are divided by experience group alongside aggregate scores for all participants. Overall, participants responded positively to the game. The categories with the highest overall scores were immersion, flow, and positive, all of which hovered around a score of 4. The greatest disparity in the categories was in the challenge category for the small level and the tension category for the large level. Beginner participants reported a mean challenge score of 3.6 in the small configuration, compared to 2.1 for experienced participants. While in the large configuration, the experienced playtesters felt around a score of 3.7 tension, and the beginners felt around a score of 2.7 tension.

These results follow the expected pattern, where prior experience with the genre reduces perceived difficulty. Experienced playtesters were able to adapt to the game mechanics and level layouts more quickly, while beginner playtesters faced a steeper initial learning curve

One interesting point in the scores is that the experienced playtesters felt more tension than the beginner and casual playtesters during the harder level. This aligns with the concept of genre literacy described by [31], where experienced playtesters carry internalized expectations about genre conventions that make deviations from established design patterns, such as PCG-introduced layout unpredictability, more noticeable and potentially more frustrating [31]. The challenges presented to participants were generally well received, and the controls were perceived as intuitive across all experience levels. Participants noted that the hazards of the levels were hard and unforgiving, but this also contributed to a greater sense of accomplishment when successfully navigating a difficult room or completing a level. This was observed across all experience groups, and even though the game itself is challenging and features different layouts and compositions of room templates between runs, the overall experience still felt accessible across different skill levels, which indicates a success for the use of PCG in this context.

Notably, several beginner participants struggled with core platformer mechanics, such as jumping, timing, and hazard avoidance, before the structural qualities of the generated levels became a factor. This echoes Malone's distinction between challenge arising from environment and challenge arising from mechanics themselves, which can compound in ways that make it difficult to isolate which source of difficulty is driving a negative experience [32]. Future studies should consider separating mechanics familiarity from PCG exposure more explicitly, for instance, by extending the tutorial or restricting beginner participants to a single configuration in their first session.

5.2 Evaluation of Methodology

The mixed-methods approach (combining GEQ survey data, observational notes, and completion timing) proved adequate for generating preliminary insights but revealed several design limitations. The use of convenience sampling within a university environment produced a sample skewed toward younger, technically literate individuals, limiting generalizability to broader gaming populations. The between-subjects variability introduced by procedural generation (no two participants necessarily played identical layouts) also makes direct comparisons more difficult to make than a fixed-level design would allow.

The alternating order of small/large configurations was a sound design choice to control for learning effects. However, the 30-minute session length, while practical, may have been insufficient for beginner participants to overcome the initial learning curve and reach a stable gameplay state. Future iterations should consider either extending session time or restricting the beginner group to the small configuration only in the first session. However, the choice of not using seeds or a different way of storing generated layouts should have been considered for replayability and recreational value so that the playtesters could test the same level layout and get a more concrete value in the scores. This also creates a library for how the generation looked like during the development, and could make for a more decisive way of showing progress and evolution in the PCG development, and what exactly worked.

5.3 Validity and Generalization

The findings should be interpreted cautiously, given several methodological constraints that limit the scope and generalizability of the conclusions.

5.3.1 Internal Validity

A central threat to internal validity is the uncontrolled variability introduced by the procedural generation system itself. Because enemy density is resolved at runtime through a configurable spawn probability rather than being fixed by the constraint system, two participants assigned to the same fish-count configuration may have encountered meaningfully different gameplay conditions. This makes it difficult to attribute GEQ score differences solely to structural layout or fish count, as enemy density represents a confounding variable that was not systematically recorded or controlled across sessions. Future studies should log enemy spawn count per session to allow post-hoc statistical control of this variable.

A further threat to internal validity arises from the within-subject, counterbalanced design. Although alternating the starting configuration was intended to control for order effects, the 30-minute session length may have been insufficient for beginner participants to reach a stable gameplay state before switching configurations. Participants who began with the 6-fish(large) configuration may have experienced frustration that carried over to their perception of the 3-fish configuration, and vice

versa. This asymmetric transfer effect is a recognized limitation of within-subject designs in game studies [33] and should be addressed in future iterations by either extending session duration or adopting a between-subject design for participants with low baseline experience.

The Game Experience Questionnaire (GEQ) is a widely validated instrument [26], but its use here is constrained by sample size. With $n = 20$ participants, the study is underpowered for inferential statistical testing. Descriptive trends are reported throughout, but effect sizes and significance levels cannot be meaningfully computed. Consequently, the findings are best understood as indicative rather than conclusive, suitable for informing the design of a larger follow-up study rather than serving as definitive evidence about how PCG affects player experience in 2D platformers.

5.3.2 External Validity

The study relies on convenience sampling within a university environment. The resulting sample is likely skewed toward younger, technically literate individuals who may approach unfamiliar game mechanics with greater comfort and lower frustration tolerance thresholds than the general gaming population. This limits the generalizability of findings to broader player demographics, including older adults, casual mobile gamers, or individuals with motor impairments who may respond differently to procedurally generated spatial challenges.

Generalizability is also bounded by genre and platform specificity. The PCG system evaluated here is tailored to a 2D side-view platformer with maze-like room assembly, and the findings may not transfer to other platformer sub-genres (e.g., auto-runners, precision platformers) or to procedural systems used in top-down, three-dimensional, or narrative game contexts. Rodrigues et al. [18] note that comparisons between procedural and handcrafted content tend to be highly genre-specific, and that results obtained in one design context are rarely portable to another without replication.

Despite these limitations, the study provides sufficient evidence to support three provisional conclusions: (1) a constraint-minimal PCG system based on room templates and bitmask-driven assembly can produce structurally valid and broadly enjoyable platformer levels without requiring complex generative machinery; (2) fish count as a sole structural constraint is effective at producing a perceptible difficulty gradient, but is insufficient for controlling perceived fairness due to unbound runtime parameters such as enemy spawn probability; and (3) prior platformer experience significantly moderates how player interpret both difficulty and procedural quality, consistent with findings from comparable PCG evaluation studies [4], [18]. These conclusions should be treated as preliminary, pending replication with a larger, more diverse sample and a tighter experimental design.

5.4 Future Work

Several directions for future development and research emerge from the limitations and findings of this study, spanning improvements to the PCG system, experimental design, and the generalizability of the evaluation framework.

5.4.1 Adaptive Difficulty

The most significant limitation of the current system is its inability to calibrate challenge to individual player skill. The most promising near-term extensions are the introduction of adaptive difficulty mechanisms that adjust generation parameters in response to real-time performance metrics. Concretely, enemy spawn probability and path segment length could be coupled to observable indicators such as per-room death count, time-to-completion, and the number of backtracking events detected by the path traversal system. This would allow the PCG system to function as a dynamic difficulty adjustment (DDA) mechanism [34], tightening the challenge-skill balance that Csikszentmihalyi identifies as the precondition for sustained flow [8]. Prior work on experience-driven PCG by Yannakakis and Togelius [3] demonstrates that incorporating player modeling into the generation loop can significantly improve flow consistency across diverse player populations, and this approach warrants direct application to the room-template system developed here.

5.4.2 Expanding the Room Template Library

Figure 5.1: Example of room template with one fish.

The current library of 82 templates distributed across 15 opening categories provides adequate variety for short play sessions, but the risk of recognizable layout repetition increases as session length grows. Expanding the template pool, particularly for complex multi-opening categories (categories 13, 14, and 15 - Left + Up + Down, Right + Up + Down, and All Directions), would reduce the probability of

players encountering familiar room sequences across runs. Additionally, introducing thematic or biome-based template variants, for example, ice caverns, flooded passages, or ruined architecture, could increase perceived variety without altering the underlying generation algorithm, a strategy employed successfully in commercial roguelites such as Dead Cells [15] and Hades [16].

5.4.3 Constraining Enemy Spawn Probability

As identified in Section 5.1, the unconstrained runtime enemy spawn probability is the primary source of within-configuration variability in perceived difficulty. A straightforward improvement would be to define per-configuration spawn probability bounds, for example, capping the spawn rate at 0.3 for the 3-fish configuration and 0.5 for the 6-fish configuration, and to link spawn probability inversely to room traversal width so that narrow rooms cannot receive the maximum enemy density. This would address the fairness concern raised by Barlet and Spohn [30] while preserving the run-to-run variation that motivates the use of procedural generation in the first place. Logging actual spawn counts per session would also make it possible to include enemy density as a covariate in future statistical analyses of GEQ scores.

5.4.4 Study Design and Evaluation

Future iterations of the user study should address several methodological shortcomings identified in Section 5.2. Increasing the sample size to at least 40-60 participants would provide sufficient statistical power for mixed-effects regression analyses that can simultaneously model the effects of fish count, experience level, and enemy density on GEQ subscale scores. Recruiting participants through gaming communities rather than exclusively within a university environment would also improve demographic diversity and external validity.

A between-subjects design, in which each participant plays only one configuration, would eliminate carry-over effects between difficulty conditions. Alternatively, a three-session design: tutorial, small configuration, and a large configuration, each on separate days, would allow learning effects to stabilize before exposure to the harder configuration, reducing the noise introduced by skill acquisition during the study itself.

Finally, qualitative methods such as think-aloud protocols or retrospective verbal reports could complement the GEQ by capturing moment-to-moment player responses to specific generated layouts. This would make it possible to identify which room configurations or spatial patterns are most disruptive to flow, providing actionable design feedback that aggregate questionnaire scores cannot offer [35].

5.5 Usage of AI

Generative AI was used in the development of the game, as well as in the writing of this report. For development, Claude and ChatGPT were used to assist in implementing enemies and room templates, including generating, debugging, and refining code related to these systems. For the report, Claude was used to suggest table structures and overall document layout, Grammarly was used to check spelling and grammar, and Overleaf Copilot assisted with writing fluency, all authored content was reviewed and verified by the authors.

6
Conclusion

This study investigated how procedural content generation affects player experience in `Gone Fishing`, addressing three research questions concerning flow, constraint complexity, and the role of prior gaming experience. Regarding RQ1, PCG level design produced moderate but highly variable flow across participants. The primary disruptions were attributable to navigational ambiguity in multi-directional layouts, and the system's inability to dynamically calibrate challenge to individual skill level [9], [29].

Regarding RQ2, the minimal constraint system proved sufficient for structural playability, all levels were generated successfully, and no participant encountered a generation error. However, unconstrained runtime parameters, particularly enemy spawn probability, introduced difficulty variance that the constraint system did not govern, undermining perceived fairness in ways that more tightly coupled constraints could address [30].

Regarding RQ3, prior platformer experience significantly shaped how players perceived procedurally generated content. Novice players were challenged primarily at the mechanical level before layout complexity became a factor, while experienced players exhibited heightened tension in the harder configuration, likely reflecting stricter genre expectations [31], [32]. Combined, the findings suggest that even a lightweight constraint-based PCG system can produce engaging and broadly accessible content.

However, achieving consistent player-centered quality requires constraints that account for individual skill and gameplay context, not only structural validity. Future work should explore adaptive difficulty mechanisms that couple enemy density and layout complexity to real-time player performance metrics, which would address the most significant limitation identified in this study.

Ultimately, this study demonstrates that procedural content generation need not be complex to be effective. A lightweight, template-based system governed by a single parameter was sufficient to produce levels that engaged players across a range of experience levels, maintained low frustration scores, and delivered a perceptible and meaningful difficulty gradient. The findings suggest that the principal challenge in PCG-driven platformer design is not structural generation, which modern constraint-based approaches handle reliably, but rather the finer calibration of runtime parameters to individual player skill. Addressing this gap, through adaptive

mechanisms that respond to real-time player behavior, represents the most promising path toward procedural systems that are not only technically sound but genuinely player-centered.

Bibliography

[1] Y. Zhang, G. Zhang, and H. Xinyuan, "A survey of procedural content generation for games," in *2022 International Conference on Culture-Oriented Science and Technology (CoST)*, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9898527

[2] M. Nitsche, C. Ashmore, W. Hankinson, R. Fitzpatrick, J. Kelly, and K. Margenau, "Designing procedural game spaces: A case study," in *FuturePlay 2006 Proceedings, London, ON October 10-12, 2006 (digital proceedings)*, 2006. [Online]. Available: https://homes.lmc.gatech.edu/~nitsche/download/Nitsche_DesigningProcedural_06.pdf

[3] G. N. Yannakakis and J. Togelius, "Experience-driven procedural content generation," *IEEE Transactions on Affective Computing*, vol. 2, no. 3, pp. 147–161, 2011. [Online]. Available: https://dl.acm.org/doi/abs/10.1109/T-AFFC.2011.6

[4] N. Andreasson, M. Granath, G. Johnsen, D. Memedov, B. Suhail, and E. Young, "Procedurally generated content in games and its effect on player experience," 2025. [Online]. Available: https://odr.chalmers.se/items/63d4bc8d-e1dd-47e2-9172-390081245be0

[5] J. Blomberg, R. Jemth, A. Lennar, R. Lilius-Lundmark, M. P. Johnsson, and T. Svensson, "An exploration of procedural content generation for top-down level design," 2018. [Online]. Available: https://publications.lib.chalmers.se/records/fulltext/256132/256132.pdf

[6] J. T. Tarigan, E. Y. P. Sukku, and S. M. Hardi, "Hybrid approach for procedural level generation for 2d platformer game," in *2024 Beyond Technology Summit on Informatics International Conference (BTS-I2C)*, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10941760

[7] L. Michailidis, E. Balaguer-Ballester, and X. He, "Flow and immersion in video games: The aftermath of a conceptual challenge," *Frontiers in Psychology*, vol. 9, 2018. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC6134042/

[8] M. Csikszentmihalyi, *Flow: The Psychology of Optimal Experience*. New York: Harper & Row, 1990.

[9] N. Shaker, J. Togelius, and M. J. Nelson, *Procedural Content Generation in Games*. Springer, 2016. [Online]. Available: https://doi.org/10.1007/978-3-319-42716-4

[10] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural content generation for games: A survey," *ACM Transactions on Multimedia*

Computing, Communications, and Applications, vol. 9, no. 1, 2013. [Online]. Available: https://dl.acm.org/doi/10.1145/2422956.2422957

[11] M. U. Farooq *et al.*, "Procedural content generation in games: A survey with insights on emerging LLM integration," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024. [Online]. Available: https://arxiv.org/pdf/2410.15644

[12] B. Patterson and M. Ward, "Adaptive procedural generation in minecraft," CIS 5603 Final Project, Temple University. [Online]. Available: https://cis.temple.edu/~wangp/5603-AI/Project/2022S/pattersonblaker/Ward_Patterson_Final_Report.pdf, 2022.

[13] Mojang Studios, "About minecraft," [Online]. Available: https://www.minecraft.net/en-us/about-minecraft.

[14] Mossmouth, "Spelunky," [Online]. Available: https://www.spelunkyworld.com/.

[15] Motion Twin, "Dead cells," Video game, Bordeaux, 2018. [Online]. Available: https://dead-cells.com

[16] Supergiant Games, "Hades," Video game, San Francisco, 2020. [Online]. Available: https://www.supergiantgames.com/games/hades

[17] M. Dahren, "The usage of PCG techniques within different game genres," 2021. [Online]. Available: https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1604550

[18] L. Rodrigues, R. Bonidia, and J. Brancher, "Procedural versus human level generation: Two sides of the same coin?" *International Journal of Human-Computer Studies*, 2020. [Online]. Available: https://doi.org/10.1016/j.ijhcs.2020.102465

[19] BorisTheBrave, "Dungeon generation in diablo 1," [Online]. Available: https://www.boristhebrave.com/2019/07/14/dungeon-generation-in-diablo-1/, 2019, accessed: May 2026.

[20] P. Kankiewicz, "Random generation done right," https://paulkankiewicz.com/2023/08/18/random-generation-done-right/, Aug. 2023, accessed: May 2026.

[21] pol-lozano, "Spelunky_pcg," [Online]. Available: https://github.com/pol-lozano/Spelunky_PCG.

[22] T. McKenzie, M. Morales-Trujillo, and S. Hoermann, "Software engineering practices and methods in the game development industry," in *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play (CHI PLAY EA '19)*. Barcelona, Spain: ACM, 2019, pp. 181–193. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3341215.3354647

[23] T. McKenzie, M. Morales-Trujillo, S. Lukosch, and S. Hoermann, "Is agile not agile enough? A study on how agile is applied and misapplied in the video game development industry," in *2021 IEEE/ACM Joint 15th International Conference on Software and System Processes (ICSSP) and 16th ACM/IEEE International Conference on Global Software Engineering (ICGSE)*. Madrid, Spain: IEEE, 2021, pp. 94–105. [Online]. Available: https://ieeexplore.ieee.org/document/9461025

[24] A. Kultima, "Developers' perspectives on iteration in game development," in *Proceedings of the 19th International Academic Mindtrek Conference.*

Tampere, Finland: ACM, 2015, pp. 26–32. [Online]. Available: https://dl.acm.org/doi/10.1145/2818187.2818298

[25] J. Nielsen, "Thinking aloud: The #1 usability tool," [Online]. Available: https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/, 2012.

[26] W. IJsselsteijn, Y. de Kort, and K. Poels, "The game experience questionnaire," Technische Universiteit Eindhoven, Eindhoven, Netherlands, Tech. Rep., 2013.

[27] Vetenskapsrådet, "God forskningssed," 2017. [Online]. Available: https://www.vr.se/analys/rapporter/vara-rapporter/2017-08-29-god-forskningssed.html

[28] G. F. Lawler and V. Limic, *Preface*. Cambridge University Press, Jun 2010, vol. 123, pp. ix–xii.

[29] P. Sweetser and P. Wyeth, "Gameflow: A model for evaluating player enjoyment in games," *Computers in Entertainment*, vol. 3, no. 3, pp. 3–es, 2005.

[30] M. C. Barlet and S. D. Spohn, *Includification: A Practical Guide to Game Accessibility*. The AbleGamers Foundation, 2012. [Online]. Available: https://www.includification.com

[31] T. H. Apperley, "Genre and game studies: Toward a critical approach to video game genres," *Simulation & Gaming*, vol. 37, no. 1, pp. 6–23, 2006.

[32] T. W. Malone, "What makes things fun to learn? Heuristics for designing instructional computer games," *SIGSOC Bulletin*, vol. 13, no. 2–3, pp. 26–36, 1981.

[33] A. Field and G. Hole, *How to Design and Report Experiments*. London: SAGE Publications, 2003.

[34] R. Hunicke and V. Chapman, "AI for dynamic difficulty adjustment in games," in *Challenges in Game AI Workshop, Proceedings of the AAAI Conference on Artificial Intelligence*, San Jose, CA, 2004.

[35] K. Isbister and N. Schaffer, *Game Usability: Advancing the Player Experience*. Burlington, MA: Morgan Kaufmann, 2008.

A

Data Collection

Playtest	Time: first test	Time: second test	Conditions
1	DNF	1m40s	6/3
2	11m13s	36s	6/3
3	4m23s	DNF	3/6
4	4m04s	1m50s	3/6
5	17m32s	3m05s	6/3
6	18m33s	2m17s	3/6
7	5m15s	2m30s	6/3
8	8m23s	5m30s	6/3
9	DNF	4m42s	3/6
10	DNF	4m12s	6/3
11	5m19s	19m1s	3/6
12	DNF	DNF	3/6
13	DNF	DNF	6/3
14	14m36s	3m46s	6/3
15	6m23s	4m08s	6/3
16	7m55s	9m14s	3/6
17	3m49s	8m31s	3/6
18	19m37s	DNF	3/6
19	02m05s	03m07s	6/3
20	DNF	DNF	6/3
Average	9m47s	5m34s	3 Fishes: 9, 6 Fishes: 11

Table A.1: The time it took players to finish each of the levels. Condition column indicates the order in which configurations were played.

P#	Exp.	Start	Pos.	Comp.	Flow	Tens.	Imm.	Neg.	Chall.
P03	B	S	4.25	2.50	3.75	2.13	4.50	1.50	3.00
P09	B	S	4.00	3.00	4.50	3.00	4.50	3.00	3.50
P11	B	S	3.50	3.50	3.75	3.13	3.50	2.00	4.00
P14	B	L	4.00	4.50	4.75	2.00	4.00	2.00	2.50
P13	B	L	4.25	3.25	4.25	1.75	4.25	1.50	3.50
P18	B	S	3.50	3.50	3.50	3.50	3.75	2.50	4.50
P19	B	L	5.00	5.00	5.00	3.00	5.00	1.00	4.00
P01	C	L	3.00	3.00	3.50	2.25	3.50	3.50	1.50
P02	C	L	3.50	4.00	4.50	1.50	2.50	1.00	2.50
P04	C	S	4.00	4.00	4.25	2.75	4.50	1.50	4.00
P10	C	L	5.00	5.00	5.00	1.25	4.50	1.00	2.00
P12	C	S	3.50	3.00	3.50	2.75	3.75	2.75	4.50
P15	C	L	4.50	3.00	5.00	2.25	3.50	1.50	4.00
P16	C	S	4.50	4.00	3.50	1.75	5.00	2.50	3.00
P20	C	L	2.00	1.00	3.50	5.00	1.50	5.00	4.50
P05	E	L	2.50	1.50	4.50	2.00	2.50	2.00	3.00
P06	E	S	4.50	2.50	1.75	2.50	5.00	3.50	3.50
P07	E	L	2.50	4.00	4.50	2.50	2.00	1.00	3.00
P08	E	L	4.50	5.00	5.00	2.00	5.00	1.50	3.50
P17	E	S	4.75	4.00	5.00	1.50	5.00	1.00	4.00

Table A.2: Per-participant GEQ subscale averages by configuration. Note that higher Tension and Negative scores indicate more tension and frustration. Continued in Table A.4.

P#	Exp.	Cfg	Pos.	Comp.	Flow	Tens.	Imm.	Neg.	Chall.
P01	C	S	4.50	5.00	4.00	1.00	3.50	2.50	1.00
P01	C	L	2.00	3.00	3.50	4.50	3.50	3.00	3.00
P02	C	S	3.50	4.00	4.50	3.50	2.50	1.00	2.00
P02	C	L	3.50	3.50	4.50	1.50	3.00	1.00	2.50
P03	B	S	5.00	2.50	3.50	1.50	5.00	1.00	2.50
P03	B	L	4.00	1.00	4.00	4.00	4.50	2.50	4.00
P04	C	S	4.00	4.00	4.00	1.00	4.50	1.50	3.50
P04	C	L	4.00	4.00	4.50	2.50	4.50	2.00	2.50
P05	E	S	2.00	1.50	4.00	1.00	2.50	2.00	1.00
P05	E	L	3.50	3.50	5.00	5.00	3.50	2.50	4.00
P06	E	S	4.50	2.50	1.50	1.50	5.00	2.50	2.00
P06	E	L	3.50	2.50	1.50	4.50	4.50	3.00	4.00
P07	E	S	2.50	3.00	4.00	4.00	2.00	2.00	2.50
P07	E	L	4.00	4.50	5.00	2.50	3.00	2.00	1.50
P08	E	S	4.50	4.00	4.50	2.50	4.50	2.00	2.50
P08	E	L	4.50	5.00	5.00	1.50	5.00	1.50	2.00
P09	B	S	3.50	2.00	4.50	5.00	4.50	1.00	4.00
P09	B	L	4.50	4.00	4.50	3.50	5.00	1.00	3.00
P10	C	S	5.00	5.00	5.00	1.00	4.50	1.00	1.00
P10	C	L	5.00	5.00	5.00	4.00	4.50	1.00	2.00

Table A.3: Per-participant GEQ subscale averages by configuration. Note that higher Tension and Negative scores indicate more tension and frustration. Continued in Table A.4.

P#	Exp.	Cfg	Pos.	Comp.	Flow	Tens.	Imm.	Neg.	Chall.
P11	B	S	3.00	3.00	2.50	4.00	3.50	2.00	4.00
P11	B	L	4.00	4.00	4.50	3.00	3.50	2.00	3.50
P12	C	S	4.00	4.00	4.00	4.00	4.00	2.50	3.50
P12	C	L	2.50	2.50	3.50	4.00	3.50	2.50	4.00
P13	B	S	4.00	3.00	3.50	2.50	4.00	1.50	3.00
P13	B	L	5.00	4.00	5.00	2.00	4.00	1.00	2.00
P14	B	S	3.00	4.50	5.00	1.50	4.00	2.00	2.00
P14	B	L	4.50	3.50	4.50	2.50	4.00	2.00	2.50
P15	C	S	5.00	4.00	5.00	2.50	4.00	2.50	3.00
P15	C	L	4.50	3.00	5.00	3.50	3.50	1.00	2.50
P16	C	S	4.50	3.00	3.50	2.50	5.00	1.00	3.00
P16	C	L	4.50	4.00	2.50	2.50	5.00	2.50	4.50
P17	E	S	4.50	4.00	5.00	4.00	4.50	1.00	2.50
P17	E	L	5.00	4.00	5.00	5.00	5.00	1.00	4.00
P18	B	S	3.50	3.50	3.00	5.00	4.00	2.50	4.50
P18	B	L	3.00	3.00	4.00	3.00	3.50	2.50	3.00
P19	B	S	5.00	5.00	5.00	4.50	5.00	1.00	5.00
P19	B	L	5.00	5.00	5.00	1.00	5.00	1.00	3.50
P20	C	S	2.00	1.00	3.50	5.00	2.00	5.00	3.50
P20	C	L	1.00	1.50	3.00	5.00	2.00	3.00	4.50

Table A.4: Per-participant GEQ subscale averages (P11-P20), continued from Table A.3.

Q#	Question	Subscale
Q1	I thought it was fun	Positive
Q2	I felt satisfied	Positive
Q3	I felt skillful	Competence
Q4	It felt like I succeeded	Competence
Q5	I forgot about the time	Immersion
Q6	I was busy with the game	Immersion
Q7	I felt frustrated	Tension
Q8	I felt irritated	Tension
Q9	The game felt impressive	Positive
Q10	The game was visually good looking	Positive
Q11	The game felt tiresome	Negative
Q12	I felt bored	Negative
Q13	I felt stressed	Challenge
Q14	I thought it was hard	Challenge

Table A.5: GEQ question key. Q1–Q14 correspond to the column headers in Table A.6 below.

Table A.6: Raw GEQ item scores per participant per configuration. Q1–Q14 correspond to the question key in Table A.5 above.

P#	Exp.	Cfg	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14
P01	C	S	4	5	5	5	3	5	1	1	2	5	2	3	1	1
P01	C	L	2	2	5	1	4	3	4	5	2	5	3	3	4	2
P02	C	S	4	3	3	5	5	4	3	4	3	2	1	1	1	3
P02	C	L	4	3	3	4	5	4	2	1	3	3	1	1	3	2
P03	B	S	5	5	2	3	4	3	2	1	5	5	1	1	2	3
P03	B	L	4	4	1	1	3	5	4	4	4	5	1	4	3	5
P04	C	S	4	4	3	5	4	4	1	1	4	5	2	1	3	4
P04	C	L	4	4	4	4	5	4	3	2	4	5	2	2	1	4
P05	E	S	3	1	1	2	5	3	1	1	1	4	3	1	1	1
P05	E	L	2	5	2	5	5	5	5	5	3	4	4	1	3	5
P06	E	S	5	4	2	3	2	1	1	2	5	5	1	4	1	3
P06	E	L	3	3	2	3	1	2	5	4	4	5	3	3	4	4
P07	E	S	3	2	3	3	4	4	4	4	2	2	3	1	1	4
P07	E	L	4	4	4	5	5	5	4	1	2	4	3	1	1	2
P08	E	S	5	4	5	3	5	4	3	2	4	5	3	1	1	4
P08	E	L	4	5	5	5	5	5	2	1	5	5	2	1	1	3
P09	B	S	4	3	1	3	4	5	5	5	4	5	1	1	4	4
P09	B	L	5	4	4	4	4	5	3	4	5	5	1	1	3	3
P10	C	S	5	5	5	5	5	5	1	1	5	4	1	1	1	1
P10	C	L	5	5	5	5	5	5	5	3	4	5	1	1	1	3
P11	B	S	4	2	2	4	3	2	5	3	3	4	2	2	4	4

Table A.6 – continued

P#	Exp.	Cfg	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14
P11	B	L	4	4	4	4	5	4	3	3	3	4	2	2	3	4
P12	C	S	4	4	4	4	4	4	4	4	4	4	3	2	3	4
P12	C	L	3	2	3	2	3	4	4	4	3	4	3	2	3	5
P13	B	S	4	4	3	3	4	3	2	3	4	4	2	1	2	4
P13	B	L	5	5	4	4	5	5	2	2	4	4	1	1	1	3
P14	B	S	3	3	5	4	5	5	1	2	4	4	2	2	2	2
P14	B	L	4	5	3	4	5	4	3	2	4	4	3	1	2	3
P15	C	S	5	5	4	4	5	5	3	2	4	4	3	2	2	4
P15	C	L	4	4	1	5	5	5	4	3	5	2	1	1	1	4
P16	C	S	4	5	3	3	5	2	2	3	5	5	1	1	5	1
P16	C	L	4	5	3	5	3	2	2	3	5	5	3	2	4	5
P17	E	S	5	4	3	5	5	5	4	4	5	4	1	1	2	3
P17	E	L	5	5	3	5	5	5	5	5	5	5	1	1	3	5
P18	B	S	4	3	2	5	2	4	5	5	4	4	4	1	4	5
P18	B	L	3	3	4	2	4	4	4	2	4	4	2	3	2	4
P19	B	S	5	5	5	5	5	5	5	4	5	5	1	1	5	5
P19	B	L	5	5	5	5	5	5	1	1	5	5	1	1	4	3
P20	C	S	3	1	1	1	2	5	5	5	1	3	5	5	3	4
P20	C	L	1	1	2	1	1	5	5	5	2	2	5	1	4	5