CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Designer Intentionality vs. Procedural Generation

Comparing Player Experience and Behavior in Maze Level Design

Bachelor's thesis in Computer Science and Engineering

Marcus Nilsson

Jeffrey Odeira

Anders Gustav Volkan Paul

Yahya Mohamed

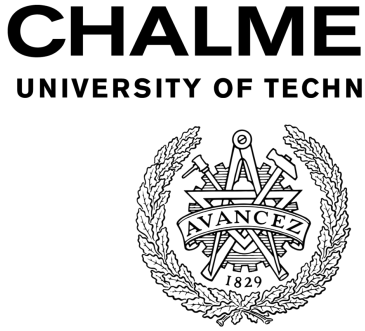
Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2026

Bachelor's thesis 2026

Designer Intentionality vs. Procedural Generation

Comparing Player Experience and Behavior in Maze Level Design

Marcus Nilsson
Jeffrey Odeira
Anders Gustav Volkan Paul
Yahya Mohamed



UNIVERSITY OF GOTHENBURG

Department of Computer Science and Engineering
DATX11 DIT561-VT26-35B
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

Designer Intentionality vs. Procedural Generation
Comparing Player Experience and
Behavior in Maze Level Design

Marcus Nilsson

Jeffrey Odeira

Anders Gustav Volkan Paul

Yahya Mohamed

Supervisor: Aris Alissandrakis, Interaction Design and Software Engineering, Data and Information Technology

Co-examiner: Palle Dahlstedt, Interaction Design and Software Engineering, Data and Information Technology

Examiner: Patrik Jansson, Department of Computer Science and Engineering

Bachelor's Thesis 2026
Department of Computer Science and Engineering
Name of research group: DATX11 DIT561-VT26-35B
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: In-game screenshot from Light the Way, showing the player's limited visibility range, ground torches emitting an orange glow, and a faint blue light in the distance hinting at the level exit.

Typeset in LaTeX
Gothenburg, Sweden 2026

Abstract

This thesis investigates how procedural content generation affects player experience in a 2D maze-based game. A playable prototype, Light the Way, was developed using the Godot engine, in which the player navigates dark hedge mazes using limited torchlight, with the ability to manipulate the level layout by burning through walls, at the cost of reducing their visibility. Maze layouts, item placements, and start and exit positions are all produced algorithmically and integrated with gameplay mechanics designed to make level structure central to the player experience.

The study compares procedurally generated and handcrafted levels to investigate how level structure affects player experience. To ensure the comparison is fair, the handcrafted levels were built on similar procedurally generated layouts and used the same visual style, tile system, mechanics, and gameplay rules. This allowed the study to focus only on differences in maze structure and design intent.

The findings indicate that both level types were playable and supported the core game loop. Handcrafted levels took longer to complete, required more player movement, and resulted in greater interaction with the wall destruction mechanic—outcomes that aligned with the deliberate design intent behind each handcrafted maze, where surgical modifications were made specifically to increase navigational complexity and encourage resource decision-making. Questionnaire responses showed that handcrafted levels were perceived as more structured, cohesive, and carefully constructed, while procedurally generated levels were associated with unpredictability or simpler navigation. At the same time, procedurally generated levels were still considered clear and playable.

The study concludes that procedural content generation can be an effective tool in game production, particularly for providing playable level diversity and reducing manual design work. However, the findings also indicate that procedural generation requires careful design constraints and parameter tuning to produce the same sense of pacing, structure, and intentionality as handcrafted levels.

Keywords: procedural content generation, maze generation, handcrafted level design, player experience, player behaviour, user study, Hunt-and-Kill algorithm, designer bias mitigation, designer intentionality, within-subject design.

Sammandrag

Denna avhandling undersöker hur proceduriell innehållsgenerering påverkar spelarupplevelsen i ett 2D-baserat labyrintspel. En spelbar prototyp, Light the Way, utvecklades med hjälp av Godot-motorn, där spelaren navigerar genom mörka häcklabyrinter med begränsat fackelljus, med möjligheten att manipulera nivålayouten genom att bränna sig igenom väggar, på bekostnad av minskad synlighet. Labyrintlayouter, placering av föremål samt start- och utgångspositioner produceras alla algoritmiskt, integrerade med spelmekaniker som är utformade för att göra nivåstruktur central för spelarupplevelsen.

Studien jämför procedurielllt genererade och handgjorda nivåer för att undersöka hur nivåstruktur påverkar spelarupplevelsen. För att säkerställa att jämförelsen är rättvis byggdes de handgjorda nivåerna på liknande procedurielllt genererade layouter och använde samma visuella stil, tilesystem, mekaniker och spelregler. Detta gjorde det möjligt för studien att endast fokusera på skillnader i labyrintstruktur och designavsikt.

Resultaten visar att båda nivåtyperna var spelbara och stödde den centrala spelloopen. Handgjorda nivåer tog längre tid att klara, krävde mer spelarrörelse och ledde till större interaktion med väggförstörelsemekaniken—resultat som stämde överens med den medvetna designavsikten bakom varje handgjord labyrint, där noggranna modifieringar gjordes specifikt för att öka navigationskomplexiteten och uppmuntra beslut kring resurshantering. Enkätsvaren visade att handgjorda nivåer uppfattades som mer strukturerade, sammanhängande och noggrant konstruerade, medan procedurielllt genererade nivåer associerades med oförutsägbarhet eller enklare navigation. Samtidigt ansågs de procedurielllt genererade nivåerna fortfarande vara tydliga och spelbara.

Studien drar slutsatsen att proceduriell innehållsgenerering kan vara ett effektivt verktyg inom spelproduktion, särskilt för att skapa spelbar nivåvariation och minska manuellt designarbete. Resultaten visar dock också att proceduriell generering kräver noggranna designbegränsningar och parameterjustering för att skapa samma känsla av tempo, struktur och avsiktlighet som handgjorda nivåer.

Nyckelord: proceduriell innehållsgenerering, labyrintgenerering, handgjord nivådesign, spelarupplevelse, spelarbeteende, användarstudie, Hunt-and-Kill-algoritm, mitigering av designerbias, designavsikt, inompersonsdesign.

Acknowledgements

We would like to express our sincere gratitude to our supervisor, Aris Alissandrakis, for his invaluable guidance, constructive feedback, and unwavering support throughout this project. His insights and encouragement were instrumental during both the development of the prototype and the writing of this thesis.

We would also like to thank our examiner, Palle Dahlstedt, for his academic guidance and thoughtful input.

A special acknowledgement goes to Michael Games, whose publicly available Godot tutorial series and asset pack provided an invaluable starting point for the prototype development, and to the developers and community members of the Godot Gaea plugin, who were candid about the limitations of their own tool and suggested that a custom solution would be better suited for our needs — prioritizing the success of our project over the promotion of their own.

We are especially grateful to everyone who participated in the playtesting of the prototype, both during internal testing and the formal user study. Their time, engagement, and feedback were essential to this research and made the study possible.

Finally, we would like to thank our families and friends for their patience, encouragement, and support throughout the thesis process.

Marcus Nilsson, Jeffrey Odeira, Anders Gustav Volkan Paul, Yahya Mohamed, Gothenburg, Tuesday 2nd June, 2026

Contents

Glossary

Below is the list of words that have been used throughout this thesis listed in alphabetical order:

Between-Subject Design:	An experimental design in which different groups of participants are each assigned to a separate test condition, as opposed to within-subject design where the same participants experience all conditions.
Blob TileSet:	A game development tile pattern system derived from Wang tile theory, in which a mathematically minimal set of tile variants (47 textured and one blank, for a total of 48 in this project) is sufficient to connect seamlessly with any possible neighbouring tile arrangement. See also: Wang Tiles.
Dead End:	A corridor segment in the maze that connects to the rest of the maze on only one side, requiring the player to backtrack upon reaching it.
Designer Bias:	The unintended influence a human designer's preferences, skill, or aesthetic instincts may have on handcrafted content, potentially confounding comparisons with procedurally generated content.
Determinism:	The property of a system whereby the same inputs always produce the same outputs. The procedural generation pipeline in this project is fully deterministic given the same seed, algorithm, and parameters.
Fisher-Yates Shuffle:	A standard algorithm for producing an unbiased random permutation of a list. In this project, a seeded variant is used to ensure deterministic shuffling of candidate positions.
Hard-locking:	A game state in which the player is permanently unable to complete the game — in this prototype, occurring when all torch charges are depleted and the remaining path to the exit requires wall destruction. As opposed to soft-locking, where progress is hindered but not permanently blocked.

Hunt-and-Kill:	A maze generation algorithm that alternates between a random walk phase (carving passages until reaching a dead end) and a systematic scan phase (finding an unvisited cell adjacent to the existing maze). Produces structurally deep, winding dead ends with low apparent repetition.
Island:	A closed loop of path tiles that has no connection to the rest of the maze structure except through wall destruction. Players entering an island must burn their way out.
Manhattan Distance:	A distance metric calculated as the sum of the absolute differences in X and Y coordinates between two points on a grid. Used in this project to enforce a minimum separation between start and exit positions.
Maze Baker:	A custom Godot editor plugin developed for this project that runs the procedural generation pipeline at design time, rendering a maze directly into an editable scene file.
Poisson Disk Sampling:	A distribution method that places items such that no two items are closer than a specified minimum distance, producing naturalistic spacing without visible clustering or artificial regularity.
Prim's Algorithm:	A maze generation algorithm adapted from minimum spanning tree construction. Maintains a frontier of candidate walls and selects one at random to carve through. Produces more uniformly distributed mazes with frequent short branches.
Pseudo-random Numbers:	Numbers that appear random but are fully determined by an initial seed value. Given the same seed, the same sequence of numbers is always produced, making them reproducible and suitable for deterministic procedural generation.
Recursive Backtracking:	A depth-first maze generation algorithm that carves passages by randomly visiting unvisited neighbours and backtracking when none remain. Produces long, winding corridors with relatively few branches.

Seed:	An initial integer value that fully determines the sequence of random decisions made during procedural generation. Given the same seed, algorithm, and configuration, the output is always identical.
Soft-locking:	A game state in which the player is unable to make meaningful progress but the game has not ended. The player is not permanently stuck, but no useful actions are available without backtracking or changing strategy. As opposed to hard-locking, where the player is permanently unable to complete the game.
Solution Path:	The uninterrupted navigable route through a maze from the start position to the exit, traversable without wall destruction. Wall destruction may create additional solution paths during gameplay.
Telemetry:	Automated in-game data collection. In this project, telemetry records per-level metrics including completion time, movement distance, torches collected, and walls destroyed.
TileMap:	A grid-based system used in 2D game engines to compose game levels from individual tiles. In Godot, a TileMap layer stores which tile variant is placed at each grid coordinate.
TileSet:	A collection of tile textures and their associated metadata, used by a TileMap to render the visual appearance of a level.
Torch Charge:	A resource unit representing the player's current visibility level and capacity to destroy walls. Players begin each level with one charge, expandable to a maximum of three by collecting ground torches.
Wall Destruction:	A game mechanic allowing the player to burn through a single hedge wall at the cost of one torch charge, creating a shortcut through the maze.
Wang Tiles:	A mathematical tiling concept introduced by Hao Wang, in which square tiles with colored edges are arranged so that adjacent edges always match. The Blob TileSet used in this project is a game development application of this theory. See also: Blob TileSet.

Within-Subject Design:	An experimental design in which the same participants are exposed to all test conditions, allowing direct comparison between conditions for each individual, as opposed to between-subject design where different participant groups are assigned to separate conditions.

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

2D	Two-Dimensional
3D	Three-Dimensional
AI	Artificial Intelligence
FOV	Field of View
GDD	Game Design Document
HUD	Heads-Up Display (the in-game UI showing torch charge count)
JSON	JavaScript Object Notation (file format used for telemetry storage)
MDA	Mechanics, Dynamics, and Aesthetics (framework)
PCG	Procedural Content Generation
RQ	Research Question
SD	Standard Deviation
UI	User Interface

List of Figures

1. Introduction

Playing video games is among the most common hobbies for teenagers and young adults, with many individuals spending close to three hours per day gaming [1]. Over the past decades, the video game industry has grown tremendously into a global, mainstream, multi-billion-dollar industry [2]. As the gaming industry has grown, the quality of modern games has also increased and as such, productions of modern games require higher quality content with increased detail, better graphics and an expectation of more engaging gameplay. This may lead to video game developers needing to work in increasing amounts and intensity to reach deadlines. Recent studies show that game developers needing to work extensive periods of overtime is highly common, which has been shown to increase stress, fatigue and burnouts [3].

To help the developers, tools to improve production efficiency have become an important area of research within the industry. Artificial Intelligence (AI) tools are being used to help with the coding and bug-fixing processes and improvements in game engines. Procedural Content Generation (PCG) is one of these tools. PCG is a method of creating content with algorithms combined with computer-generated randomness, as opposed to manually handcrafted content. PCG relies on a set of parameters, rules, and inputs to generate data. These inputs can be pseudo-random numbers, which are seemingly randomly generated but usually determined by an initial smaller value known as a seed, where the aforementioned randomness is reproducible given the same seed. PCG, with help of seeds, can be used on many different aspects of game development, from map generation to interactive object creation.

The use of PCG is not a recent development; it has been utilized by the industry for many years, including in the development of some of the most popular video games, such as *Minecraft* [4] with an impossibly large world to be created by hand, the *Borderlands* series [5] with more than a billion unique gun designs, and the *Remnant* series, where procedural generation is used to create replayable worlds with changing maps, enemy encounters, quests, events, and loot [6].

Despite the advantages that PCG provides, the content it generates can differ fundamentally from handcrafted content in the way the user perceives it. Manually made content benefits from deliberate design choices, where at each step of the creation of handmade content, developers can control structure, pacing and the relationship between different elements and how they come together more compared to using PCG tools where the output may feel less intentional potentially leading to repetition and less coherence in user experience.

1.1 Background

PCG has been studied widely in game development research, often in response to the growing cost and effort required to produce game content and with a focus on how PCG can reduce manual workload within limited development times, similar to the thesis at hand.

1.1.1 Related Work

Within the field of procedural generation, several studies have investigated how player experience is influenced. Among these, a previous bachelor's thesis conducted within this study program has been an inspiration for this work [7]. In that thesis, the authors note the need for further research, linked to the lack of clear evidence on how procedural generation affects player experience. While PCG offers clear advantages in efficiency and scalability, its impact on the quality of gameplay remains uncertain. The previous thesis's authors argue that developers often rely on assumptions when choosing between procedural and handcrafted design. While that thesis provides a useful foundation, this thesis does not follow the same approach. Instead, it adopts a more focused research design. This thesis limits its scope to a simpler and more controlled question to isolate a single design variable. This allows for a clearer interpretation of how level generation affects players' perception.

Another example is research on experience-driven procedural content generation, which examines how generated content can be linked to player experience rather than following a rigid set of rules [8]. The research focuses on how procedural systems can adapt content based on player behaviour and feedback. The work describes a range of studies where player testing is used to evaluate generated content. In these studies, players interact with different versions of content, and data is collected from gameplay metrics as well as self-reported data through questionnaires.

The research also discusses more advanced methods for measuring player experience, such as physiological data collection. This can include facial expression tracking or other forms of biometric data. While these methods may provide detailed insights, they also require wider research and require more complex testing setups. For this reason, the present study does not include such methods and instead focuses on gameplay data and questionnaires only.

These works highlight the need for controlled studies that isolate specific design variables, which motivates the approach used in this thesis.

1.2 Purpose and Problem Statement

The purpose of this study is to examine how procedural content generation influences player experience in game design. Procedural methods are widely used in the industry, yet there is limited empirical evidence on how players perceive procedurally generated content compared to handcrafted design. This study addresses that gap by comparing how players experience and evaluate different types of level design.

The study uses a controlled user test. Participants play both handcrafted and procedurally generated levels without being informed of how the levels were created. All other game elements remain constant. This design isolates the effect of level structure. The levels are small and focused on directing attention to navigation and layout.

Data is collected through gameplay metrics and questionnaire responses. Gameplay data captures how players move through the levels, while questionnaire responses capture how players interpret and evaluate their experience.

The study is guided by the following research questions:

- RQ1: Do players perceive differences between handcrafted and procedurally generated levels?
- RQ2: How do players perceive the structure and intentionality of handcrafted levels compared to procedurally generated levels?
- RQ3: How does level type influence player behavior and interaction during gameplay?

These questions address both recognition and perception. The aim is to determine whether the differences in level structure are noticeable and how they influence the player's experience.

1.2.1 Hypothesis

This thesis presents the following hypothesis: *players will consider handcrafted levels to be more unified and deliberate than procedurally generated levels.*

This hypothesis is based on the difference between direct and indirect design control. Handcrafted levels allow designers to shape the layout in relation to a planned player experience. Procedurally generated levels are instead created through rules and parameters, which means that the final layout is shaped by the generation system. Since this study focuses on navigation and level structure, it is expected the handcrafted levels to be more likely to appear cohesive and intentional to players.

In the hypothesis, the deliberateness relates to the level's logical connection, readability, and consistency in structure. Intentionality relates to whether players believe the level was purposefully created, such as through meaningful paths, controlled pacing, and strategic placement of gameplay items like torches, obstacles, and exits.

At the same time, procedurally generated levels are expected to offer more variety between playthroughs. However, because these levels are produced using computational principles rather than direct manual placement, they may appear to be less planned or structured. This distinction is especially important in the prototype, as navigation, limited sight, torch usage, and the quest for the exit are all integral parts of the player experience.

To test the hypothesis, players' questionnaire responses will be compared to gaming observations. The analysis will focus on whether players describe handcrafted levels as easier to grasp, more purposefully designed, or more coherent, versus procedurally generated levels as more varied, unpredictable, or confusing.

This hypothesis should be understood within the context of *Light the Way* and the specific procedural generation pipeline used in this project. It is not a general claim that handcrafted levels are always more coherent or better designed than procedurally generated levels. Rather, the comparison focuses on how the selected generator, handcrafted level modifications, and core mechanics of limited visibility, torch collection, wall destruction, and maze navigation shaped player perception in this prototype.

1.3 Game Design

This study's game was designed as a controlled test environment to distinguish between handcrafted and procedurally generated levels. The prototype focuses on games where level structure is crucial because the thesis's goal is to investigate how level generation affects player experience. The design places a higher priority on player movement, navigation, and visual organization than it does on narrative advancement or other types of content.

This design approach meets the criteria for the study question since it makes each level's layout an essential component of the game's experience. In games where movement through space is central, the arrangement of corridors, branches, dead ends, and progression paths can directly influence how players experience orientation, exploration, and progression [9]. Therefore, rather than using other types of game content like story, dialogue, or character behavior, the prototype offers a suitable environment for researching procedural generative content production through level design. Figure 1.1 shows an example of a maze level in the game.

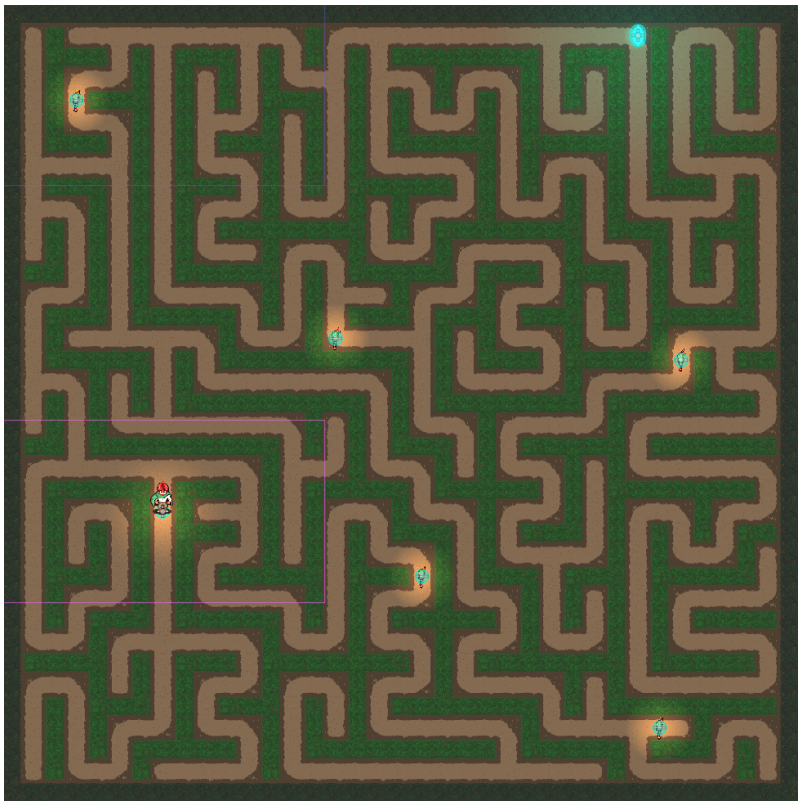


Figure 1.1: Maze game design. The player (middle left) needs to navigate the maze and reach the exit (blue light, top right). In-game, the player cannot see the entire map at once, and their vision is dynamically reduced, so they need to find some of the five torches to be able to see further and/or create shortcuts. This maze was generated using the Hunt-and-Kill algorithm, as described in 4.3 and 4.3.1.

Furthermore, the prototype was made to maintain a reasonably straightforward overall game structure. The study may focus more precisely on how variations in level layout may impact player perception by minimizing the impact of irrelevant design factors. This makes the game useful as a controlled environment for exploring whether handcrafted and procedurally generated levels are perceived differently by players. In Chapter 4, the prototype, its mechanics, and its implementation are described in further depth.

To summarize, the game was created as a controlled testing environment, with level structure playing a key part in influencing player experiences. By reducing the influence of irrelevant design aspects and stressing exploration, restricted knowledge, and spatial decision-making, the prototype provides an appropriate foundation for investigating how handcrafted and procedurally generated level design may be interpreted differently by players.

1.4 Ethical Considerations

This study involved human participants as test users and included data collection. Ethical considerations are important to ensure that the study is conducted in a responsible and transparent manner.

Participation in the study was voluntary. Before taking part, participants were informed about the procedure and their role in the study. No form of compensation or incentive was provided for participation. This was done to reduce the risk of participants focusing on completing the task as quickly as possible rather than engaging with the game and questionnaire in a consistent and attentive manner. The aim was to support more reliable observations from players.

Since the study aims to get information without swaying opinions with prior knowledge about what the study aims to answer, some information, mainly the question of "whether players can identify whether a game level was handcrafted or procedurally generated", some details about the specific research focus are not disclosed before testing to avoid influencing participant responses. The participants were only informed about the full purpose of the study later.

The study did not collect personal or sensitive information. The data consisted of gameplay metrics and questionnaire responses. All data was anonymized and only accessible to researchers.

The game used in the study is designed to be simple and safe to play. However, some players may experience frustration due to difficulty, so participants will be allowed to stop partaking in the study if they choose to.

1.4.1 Potential Societal Impact

The increasing usage of PCG in game development carries potential societal implications, particularly in relation to labor practices in the industry. As a tool that is used for automating parts of the work done by game developers, PCG has the potential to reduce the workload and excessive overtime of workers within the industry, but the implementation of automation technologies does not necessarily lead to improved working conditions.

It is possible that more implementations of efficiency methods may lead to an expectation of higher output from the workers, increasing the expected amount of output rather than lowering the workload. Furthermore an increase in procedural generation in more parts of game development may lead to lower employment rates within certain roles of production.

1.5 Scope

PCG can be used in many different parts of game development, from aesthetics to parts that have a much bigger impact on mechanics and gameplay, such as map generation. Because of this wide range of applications it is not possible to study the full scope of PCG on user experience within a single Bachelor thesis.

Even with a focused research question, the topic can lead to many related questions; these may concern different parts of game development that can be automatized so that the effect on user behaviour is minimal while the efficiency gain is maximized.

To keep the study focused, this thesis limits its scope to a specific use of procedural generation. The work examines how PCG is used to generate maze layouts and place interactive elements within a 2D top-down game.

The number of participants is limited due to time and resource constraints. This affects the statistical strength of the results, and findings should be interpreted as indicative rather than conclusive.

2. Theory

This chapter presents the theoretical concepts that are relevant to the study. It outlines procedural content generation as a game design approach, discusses how level structure can influence player experience, and explains why the comparison between handcrafted and procedurally generated levels is relevant in the context of this project.

2.1 Procedural Content Generation in Game Design

The algorithmic development of game content is known as procedural content generation (PCG). Designers create norms, limitations, and processes that enable a system to produce content automatically rather than building each component by hand. PCG has been utilized to an extensive variety of game content, such as environments, systems, objects, and levels. It has frequently been proposed as a solution to the growing expense and scope of game content creation. At the same time, PCG is a design strategy that can add variation, re-playability, and new types of player experiences, in addition to being a technical solution for efficiency [10], [11].

PCG's combination of automation and designer control is critical. While procedural systems can reduce manual burden, designers must still affect the generated results via rules, restrictions, and parameters. Previous research has demonstrated that procedural generation depends not just on automation, but also on designer control. Hendrikx et al. argue that designers should be able to influence procedurally generated content by adjusting parameters [10], while van der Linden et al. describe control as an essential feature of generative methods because it determines what the algorithm can and cannot design [12].

In this thesis, the relevant form of PCG is level generation. The study focuses on one specific implementation of procedural generation, the development of maze layout in a 2D game prototype, rather than the concept as a whole. This allows for the study of PCG in a controlled environment, where one essential aspect of the player experience, level structure, can be examined across different design techniques.

2.2 Level Structure and Player Experience

Level design is more than just the technical organization of paths and obstacles; it is also a critical component of how players comprehend and interact with a game. Jenkins claims that game designers do more than just telling stories; they create worlds and sculpt locations, implying that player experience can evolve through movement, navigation, and interaction with the environment rather than narrative alone [9]. This emphasizes exploration and orientation, as the level structure becomes one of the primary means of producing meaning and experience.

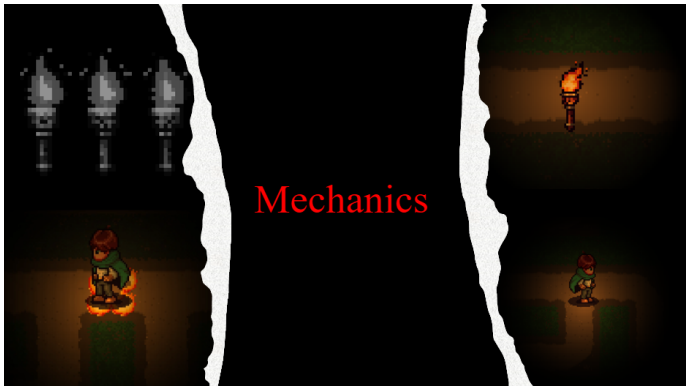
This perspective is also connected to the MDA framework, which describes games in terms of their Mechanics, Dynamics, and Aesthetics. In this approach, "mechanics" refers to the game's formal components, "dynamics" to how those components behave during gameplay, and "aesthetics" to the emotional responses elicited in the player [13]. The application of the MDA framework used in this study is shown in Figure 2.1.

MDA is significant for this project because it investigates not only how the maze is formed but also how the generated structure influences play. The mechanisms of *Light the Way* include player movement, maze structure, torch system, limited sight, destructible hedge walls, stone walls, and exit. These are the fundamental rules and elements that determine what the player can and cannot accomplish.

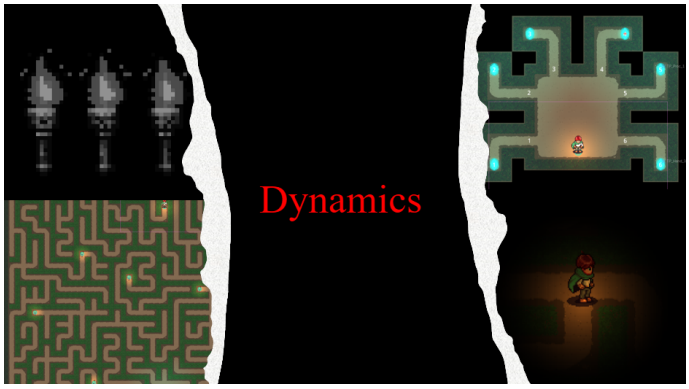
When these mechanics are applied in game development, they produce unique dynamics. Because the player cannot see the entire maze, they must explore step by step, recall paths, and make judgments based on incomplete information. The torch mechanic also influences how the player goes through the level, as light is associated with visibility and interaction with the environment. If the player chooses to destroy a hedge wall, this can result in a shortcut, but it also alters how the player uses the available light. In this way, the level structure and light system interact to influence navigation and decision-making.

The aesthetics of this prototype are connected to the experiences produced from these settings. The ideal experience is a mix of curiosity, uncertainty, difficulty, and discovery. A level with obvious paths, regulated pacing, and significant obstacle placement can give the player a more intentional experience. A level with unclear routes, repetitive patterns, or unexpected dead ends may appear more random. This is why MDA is useful for comparing handcrafted and mechanically generated levels. The same mechanisms might produce distinct player experiences depending on how the level is designed.

Figure 2.1 shows how the MDA framework is applied in the prototype. The mechanics are the concrete game systems, such as torches, visibility, walls, and maze layout. These mechanics create dynamics such as exploration, route choice, memory, and resource management. These dynamics then influence the aesthetics of the game, including uncertainty, tension, and the player's perception of intentional design.



(a) Mechanics



(b) Dynamics

(c) Aesthetics

Figure 2.1: Application of the MDA framework to *Light the Way*. The mechanics include torches, limited visibility, maze layout, and wall interaction. These mechanics create dynamics such as exploration, route choice, memory, and resource management. These dynamics shape the aesthetics of the game by creating uncertainty, challenge, discovery, and a sense of intentional or random level design.

2.3 Handcrafted and Procedurally Generated Levels

A critical theoretical distinction in this work is that between handcrafted and procedurally generated level design. In a handcrafted design, the level structure is directly created by the designer, who has control over path connections, pacing, advancement, and the placement of key gameplay elements. This allows for a high degree of intentionality in the final arrangement, as each level component can be modified in reference to a larger design aim.

In procedural generation, the designer creates a system that generates the level using rules and parameters. This indicates that control is more indirect. The final layout is no longer created tile by tile but rather emerges from the logic of the creation engine. As a result, the perceived quality of procedurally generated material is determined not just by the created output, but also by how well the system has been designed and optimized. Poorly adjusted generation may result in levels that appear repetitive, vague, or less deliberate, whereas well-designed procedural algorithms may produce structures that players perceive as coherent and meaningful [10].

This distinction is significant for the current investigation. The comparison of handcrafted and procedurally produced levels is not intended to establish one method as universally superior, but rather to study how players interpret the resulting level structures in a given game environment. In a prototype where navigation, exploration, and spatial decision-making are key to gameplay, variances in layout may be most noticeable in how players describe coherence, orientation, and overall experience.

3. Methods

3.1 Method overview

The project combines game development and comparative evaluation. The first phase of the project involved creating a playable prototype in which both procedurally generated and handcrafted levels can be used inside the same game. The second portion focuses on how these two forms of level design affect the game and the player experience.

3.2 Development method

The game was developed iteratively. This means that the game was developed in stages, with major features initially deployed in a basic version and then refined through repeated internal testing and modifications. This approach is ideal for game development projects that require simultaneous testing of playability, technical functionality, and design considerations.

The development began by presenting the core concept of the game: a dark maze game where the player must find their way to the next exit despite steadily decreasing visibility. Then, fundamental features, including player movement, level transitions, lighting systems, environmental structure, and level control, were included. After establishing the core gaming experience, the procedural generation of the maze was integrated and enhanced.

The iterative process allowed us to regularly analyze if the technical implementation supported the project's fundamental goal, which was to establish a practical basis for comparing procedurally generated and hand-crafted levels within a similar game experience.

3.3 User Study Setup

Both on-site and remote (online) testing were utilized, where it was possible to observe the participants, ensure they followed the instructions, and collect additional feedback afterwards. To keep the procedure consistent, both groups of participants followed the same protocol. They played the same game version, experienced both level types, and answered the same questionnaire (see Appendix C). This helped reduce the difference between the two testing formats and made the collected responses easier to compare.

The study applies a within-subject design, where each participant experiences both conditions that are being tested. In this type of design, the same participants are exposed to all test conditions, allowing a direct comparison between them, as opposed to a between-subject design where different groups of participants are assigned to

separate conditions. This form of design is commonly used when comparing closely related variations of the same system, as it allows differences in user experience to be observed more directly [14].

Exposing participants to multiple conditions introduces the risk that experience with one level type may influence perception of the next. To reduce this effect, participants completed both handcrafted and procedurally generated mazes in a randomized order. The game featured a central level hub from which all six mazes were accessible via individual portals, with the play order determined by rolling a six-sided die — one face corresponding to each maze. This ensured that level order varied between participants, reducing the effect of play order and game familiarity on the results.

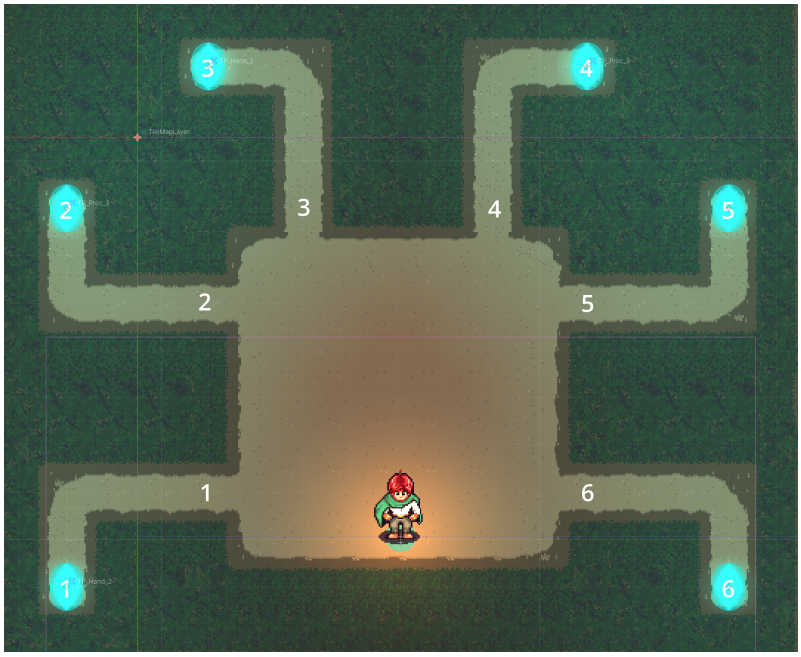


Figure 3.1: Level hub which enabled randomized order for each playtester.

3.3.1 Designer bias mitigation

A key consideration in designing a fair comparison was mitigating the bias introduced by human level designers. A designer's experience, aesthetic preferences, and creative instincts could influence the handcrafted levels in ways unrelated to maze structure — producing levels that players preferred or disliked for reasons entirely separate from the research question. Beyond deliberate design choices, subtler factors such as texture alignment, spatial pattern repetition, or the absence of repetition could act as unintended cues that reveal to the player which type of level they are navigating.

To address this, the handcrafted levels were not designed entirely from scratch. Instead, each handcrafted maze was derived from its procedural counterpart using the same generation seed as a structural baseline. Rather than applying modifications freely, the handcrafting process was governed by a deliberate separation between two categories of design constraints:

- **Local geometric constraints (constant)** — variables kept identical across both level types to eliminate surface-level visual and spatial bias. These include tile textures, lighting behavior, path widths, minimum dead-end lengths, the odd-coordinate grid structure inherent to the generation algorithm, and minimum separation distances between torches, between torches and the start and exit positions, and between the start and exit positions themselves — as described in Sections 4.3.3 and 4.3.4.
- **Global topological constraints (variable)** — higher-level structural configurations intentionally modified in the handcrafted levels to evaluate specific navigation behaviors. These include the introduction of isolated path structures (islands) and the deliberate exploitation of known player sightlines — placing torches at positions the designer knew would be visible from specific locations, and shaping the connecting paths to create intentional resource trade-offs that the procedural algorithm, operating without spatial awareness of the player's perspective, cannot produce. These design decisions are described in detail in Section 4.4.

As described in Section 4.2, both level types are rendered using the same Blob TileSet and Terrain system, meaning there is no visual distinction at the texture level between a procedurally generated tile and a handcrafted one. The local geometric constraints ensured that minimum path lengths and dead-end behaviour remained identical between level types—a property explained in detail in Section 4.3.3. Any difference a player perceives between the two level types, therefore, reflects deliberate changes to the global topology introduced during the handcrafting process, rather than surface-level visual variation or designer skill.

3.4 Data Collection and Analysis

The study gathered two sorts of data: questionnaire responses and in-game telemetry. This combination allowed for an elucidation of participants' subjective experiences and actual behavior while playing the levels. Previous research on procedural content generation and player experience compared procedural versus handcrafted content based on player ratings, immersion, attentiveness, and perceived quality [11], [15], [16].

Telemetry data was collected throughout each play session. This includes level details, completion time, movement distance, torches collected, and walls destroyed. The tutorial level was eliminated from the comparison, as it was meant for introducing controls and mechanics rather than assessing level structure.

After completing each of the levels and also at the end, participants were asked about their experience, including clarity, direction, apparent structure, difficulty, and satisfaction (see Figures C.1 and C.2). These responses were recorded using a five-point rating scale. At the end of the playtest, participants also answered broader questions about the overall experience, whether they noticed differences between levels, which levels felt more structured or intentionally designed, and which levels felt more random. These particular questions allow us to compare the handcrafted and programmatically created levels.

The analysis evaluates handcrafted versus procedurally generated levels, utilizing telemetry and questionnaire results. Telemetry data was collected to determine the average completion time, movement distance, torch usage, and wall destruction for each level type. These values help demonstrate how the level structure influenced player behavior. Longer completion durations or increased moving distance may indicate a level requiring more exploration or ambiguity.

In addition to questionnaire data, observed gameplay activity was employed to supplement the analysis. This includes how participants navigated the maze, whether they appeared to be lost, how they used the torches, and how they dealt with dead ends or unknown paths. These observations are interpreted as supporting qualitative evidence rather than exact statistical measurements.

The questionnaire responses and gameplay observations are used to determine how variances in level structure influenced the player experience. The analysis thus ties the established level-generating methods to the core research topic, which concerns how players perceive handcrafted versus procedurally generated levels.

3.5 Participant profile

To better comprehend the participants' backgrounds, the questionnaire asked about their gaming frequency and familiarity with developing and designing games. These questions were used to clarify the player replies, as previous experience with games or level design may have influenced how participants interpret level structure, challenge, and intention.

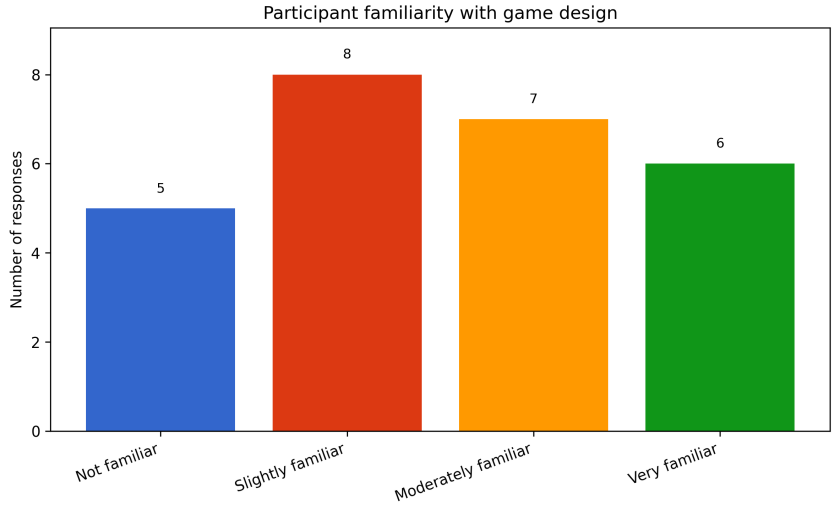


Figure 3.2: Mean completion time for each maze level.

In total, 26 people answered these questions. Figure 3.2 indicates that participants claimed that they were unfamiliar with game design, whereas eight were slightly familiar, seven were highly familiar, and six were quite familiar. This suggests that the majority of participants had some knowledge of game design, but the group was not made up entirely of professional designers.

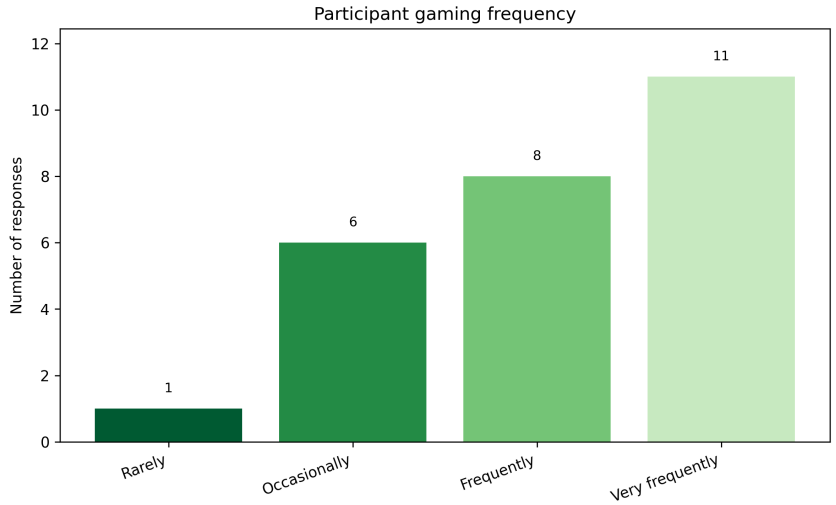


Figure 3.3: Mean completion time for each maze level.

Figure 3.3 indicates that the participants were typically comfortable with playing games. Only one participant reported playing games infrequently, while six played occasionally, eight regularly, and eleven quite frequently. This shows that the majority of participants had enough gaming experience to understand the prototype's basic concepts and structure.

These findings are significant when understanding subsequent discoveries. Participants with more gaming or game design expertise may be more aware of level layout, pacing, navigation, and whether a level appears to have been constructed with intention. At the same time, the group included participants with lower familiarity, indicating how the levels were interpreted by players from various backgrounds.

3.6 Delimitations

The project is limited to a single sort of game: a 2D top-down maze game in which exploration, orientation, limited visibility, and route choice are important aspects of the gameplay. As a result, the findings are most applicable to games in which level layout significantly influences navigation and player decision-making. The findings should not be construed as a blanket statement about all game genres or applications of procedural content generation.

The study also focuses specifically on level structures. Other aspects of game design, such as narrative, character design, combat systems, soundtrack, visual style, and long-term advancement, are not addressed. These elements may also have an impact on the player experience, but they are beyond the scope of this project.

Another delimitation is that the comparison is dependent on the specific implementation of *Light the Way*. The procedurally generated levels were created utilizing the maze-generating methods, settings, and placement rules chosen for this prototype. A different generator, a set of constraints, or game design might yield different results.

Finally, the research was limited by the number of participants and the time of the playtesting sessions. While the collected data can suggest tendencies in how players perceived the levels, the results should be seen as suggestive rather than statistically applicable. The collected data can reveal patterns in how players perceived the levels; the findings should be interpreted as suggestive rather than statistically applicable.

4. Implementation

The prototype was developed using the Godot game engine, an open-source and cross-platform engine supporting both 2D and 3D game creation [17]. Godot's scene- and node-based architecture, combined with its TileMap system for 2D level building, made it particularly well-suited for this project — both for procedural maze generation and handcrafted level editing, as described in Section 4.4. The initial player movement system, state machine, camera setup, and base character and path sprites were derived from a publicly available Godot tutorial series [18] by Michael Games, and its associated asset pack [19], used and modified under their permitted terms — the character sprite was subsequently modified to include a torch and the path TileSet modified to fit the game's aesthetic. The torch item sprite [20] served as both a standalone pickup asset and the basis for the player's torch-holding animation. The wall destruction animation and sound effects for wall burning and torch collection were sourced from freely available assets [21], [22]. All remaining assets and code were produced by the development team.

At its core, the prototype is a maze-navigation game in which the player must find the exit of a dark hedge maze. What distinguishes it from other maze games is the ability to manipulate the maze layout itself — hedge walls can be burned through to create shortcuts, but at the direct cost of visibility. This trade-off is governed by a resource management system built around a torch with charges ranging from zero to three. Each charge represents a level of illumination — at maximum charges, the player can see a substantial area around them, while at zero charges, visibility is reduced to near-blindness, creating immediate pressure to manage resources carefully.

Wall destruction is therefore the central mechanic of the game. Spending a charge to bypass a dead end or reach a visible path sooner comes at the direct cost of reduced future visibility. The player must weigh the short-term benefit of a shortcut against the long-term consequence of navigating with diminished sight—a decision that becomes increasingly consequential as charges are depleted.

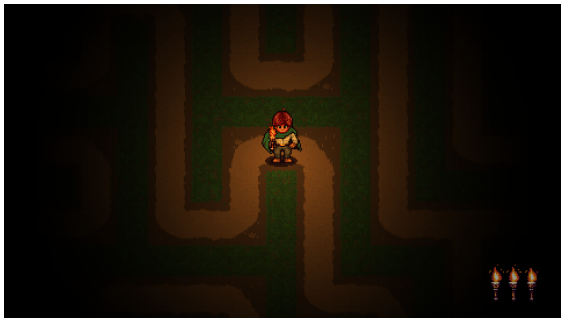
Additional torches are scattered on the ground throughout each maze, always visible even in near-darkness, and refill one torch charge for the player when collected. These serve as navigation beacons as much as resources—a visible torch glow in the distance gives the player both a goal to move towards and the promise of restored visibility. Together, these three elements — charge conservation, wall destruction, and torch collection — form a three-way resource tension that sits at the heart of the gameplay experience.

Central to the prototype's research purpose is the distinction between two types of maze levels. Half (three out of six) of the mazes are procedurally generated — their layouts, torch placements, and start and exit positions are determined algorithmically at the design stage using a seeded random number generator. The other half are handcrafted, meaning that their layouts were deliberately designed by

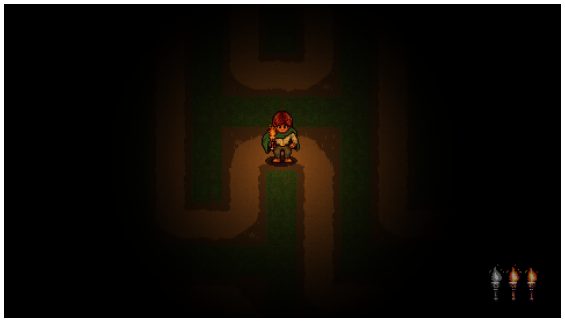
the development team. However, rather than being designed entirely from scratch, the handcrafted levels use the procedurally generated mazes as a structural baseline — each handcrafted maze shares its seed with a procedural counterpart and was then modified through deliberate, surgical changes to the layout. This approach and the reasoning behind it are discussed in more detail in Section 4.4. Both level types are visually identical in terms of tile aesthetics and game mechanics — the only difference between them is the structure of the maze itself, the placements of torches, and the start and exit positions. This was a deliberate design decision to ensure that any difference in player experience between the two level types could be attributed to maze structure alone, rather than to visual or mechanical variation, as well as the experience of the development team as level designers, as described in Section 3.3.1.

4.1 Player interaction systems

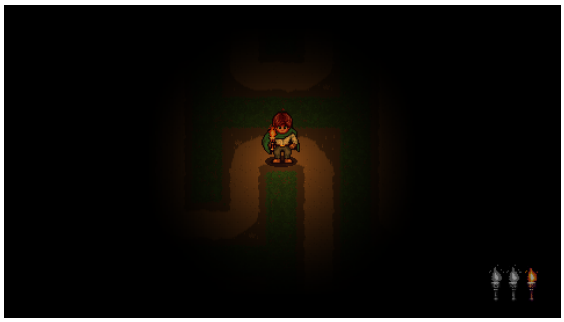
Lighting and level manipulation are at the heart of the game mechanics, and they are made up of a handful of systems that all function together. The maze environment is rendered in total darkness, and the only light sources are the torches scattered throughout the maze and the light emitted by the player themselves. The player illuminates their surroundings in a circular shape, the visibility radius of which is determined by the number of torch charges the player currently holds, as shown in Figure 4.1. This limits how much of the maze the player can see at any given time — a larger visibility radius makes it easier to spot dead ends, identify paths, and navigate the maze from a top-down perspective, while a smaller radius forces the player to make decisions with incomplete information.



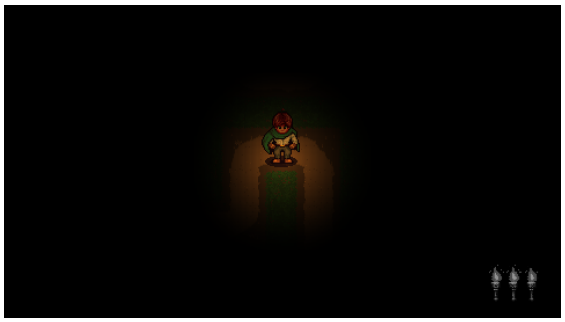
(a) Three charges



(b) Two charges



(c) One charge



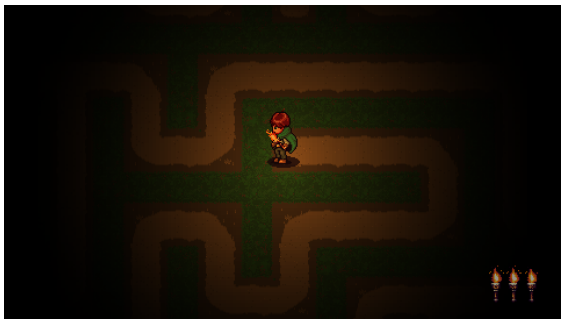
(d) Depleted

Figure 4.1: Number of torch charges, represented by the UI element in the bottom right corner.

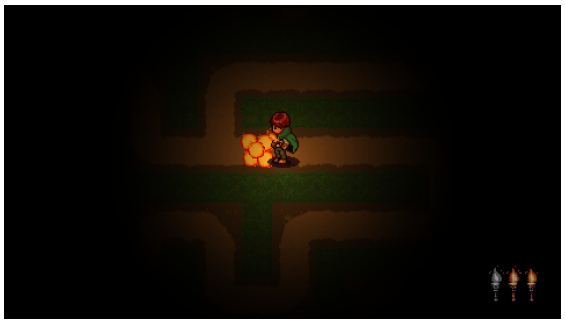
The player begins each level with a single torch charge, expandable to a maximum of three by collecting ground torches placed within the maze. These torches emit a small-radius orange glow, which the player can spot even at near-zero visibility, making them navigation beacons as much as resources.

Figure 4.2: Torches in the distance function as beacons for the player to move towards.

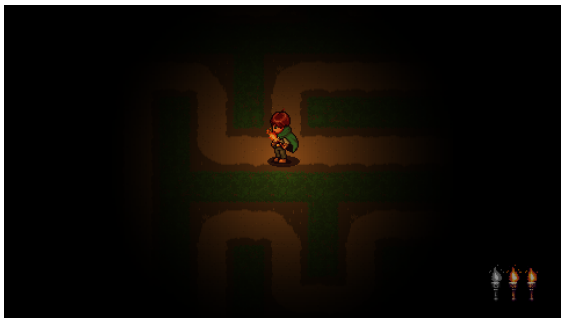
While maintaining charges preserves visibility, spending them offers a different kind of advantage: level manipulation. For the cost of one charge, the player can burn through a single hedge wall, carving a shortcut through the maze. This creates a direct trade-off: a shortcut gained now comes at the immediate cost of reduced visibility for the remainder of the level, unless the player can locate and collect another ground torch.



(1) The player encounters a dead end.



(2) They use one torch charge to burn through the hedge wall.



(3) Proceeds forward, but with reduced visibility.

Figure 4.3: Hedge burning, step by step.

The exit marker also functions as a light source, emitting a faint blue glow with a larger radius but lower intensity than the orange torches scattered on the ground. The contrasting color makes the exit visually distinct from torch beacons, giving the player an unambiguous signal when the exit is close in proximity.

Figure 4.4: A faint blue light is emitted by the exit, functioning as a visual hint.

These systems, with their advantages, and trade-offs have been deliberately designed to keep the player aware of their surroundings, and encourage conscious decisions based on the maze layout, rather than aimlessly wandering around until they stumble upon the exit. Differences in structure between handcrafted and procedurally generated levels — such as the frequency of dead ends, the availability of shortcuts, and the placement of torches relative to the solution path — should therefore manifest as measurable differences in how players engage with these mechanics, as reflected in the results discussed in Chapter 5.

4.2 Maze creation pipeline

Creating six comparable mazes (three procedurally generated and three handcrafted) required a shared pipeline that both workflows could operate on without introducing inconsistencies. Existing tools were evaluated: in particular, the Godot plugin Gaea, designed to support procedural generation within the engine. After consulting the plugin's community documentation and members of the official Gaea Discord [23], it was determined that Gaea lacked native support for maze generation and that integrating it with the custom generation logic required by this project would offer no meaningful advantage over a fully custom implementation. The decision was therefore made to develop the pipeline from scratch.

This custom pipeline is built around four core components:

- `MazeConfig`
- `ProceduralMazeGenerator`
- `MazeData`
- `TileMapPainter`

`MazeConfig` serves as the single source of truth for all generation parameters — maze dimensions, random seed, chosen algorithm, torch count, and the minimum Manhattan distance [24] between the start and exit positions, described more in Section 4.3.3. In the initial implementation, the random seed was configured directly on each level scene in the Godot editor, allowing different procedural levels to produce different mazes while sharing the same generation code. This approach was later superseded by the Maze Baker, as described in Section 4.2.2, which took over responsibility for seed configuration at design time.

The `ProceduralMazeGenerator` takes a `MazeConfig` as input and produces a `MazeData` object: a two-dimensional grid of tile types together with the generated start position, exit position, and torch placement coordinates, collectively representing the complete state of a generated maze. The `TileMapPainter` translates `MazeData` into the visual game world using Godot's TileMap system. The tile textures for the paths and hedge walls both follow the Blob TileSet specification [25] — a game development pattern system derived from Wang tile theory [26], in which a mathematically minimal set of tile variants is sufficient to connect seamlessly with any possible neighboring tile arrangement. This approach significantly reduces the number of unique tile variants an artist must produce. In this case, 47 variants are sufficient to cover all possible configurations, as defined by the Blob TileSet specification — in this implementation, one additional blank tile was included for unassigned cells, bringing the total to 48. The TileSet for the hedge walls were designed in Aseprite by the authors themselves, but the TileSet for the paths was instead a modified version of that found in an asset pack by Michael Games [19]. Combined with Godot's Terrain system, which uses these variants to automatically select the correct tile based on each tile's neighbors, neither a designer nor a procedural algorithm needs to manually specify which exact variant to place — only the logical tile type needs

to be defined. This also ensures there is no texture-level visual difference between a procedurally generated tile and a handcrafted one, since both use the same set of variants and the same selection logic — eliminating the risk of a designer accidentally placing an incorrect tile variant and introducing an unintended visual inconsistency between level types.

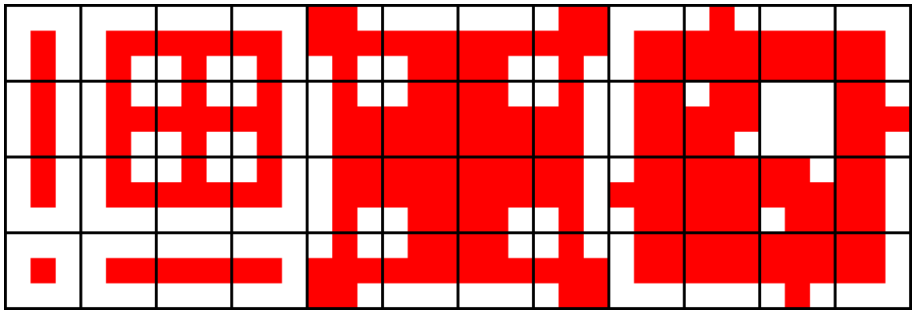
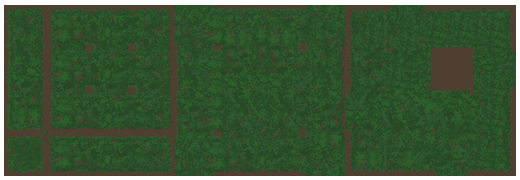


Figure 4.5: The 47 tile variants of the Blob TileSet pattern, each representing a unique combination of neighbouring tile connections [25], [27].



(a) Hedge Wall TileSet

(b) Path TileSet

Figure 4.6: The hedge walls and path TileSets used in the game, each comprising 48 tile variants following the Blob TileSet specification.

4.2.1 From runtime generation to the Maze Baker

In the initial implementation, maze generation and rendering were handled at runtime. Each time a procedural level was loaded, the game would generate a `MazeData` object from the configured seed and immediately paint it into the TileMapLayer. This worked well for the procedural levels, where no manual editing was required. However, as the project developed iteratively and a deliberate design decision was made by the game developers to derive all handcrafted levels from procedural counterparts (as described in Section 3.3.1), a fundamental problem emerged: there was no way for a designer to see or edit a maze that only existed at runtime.

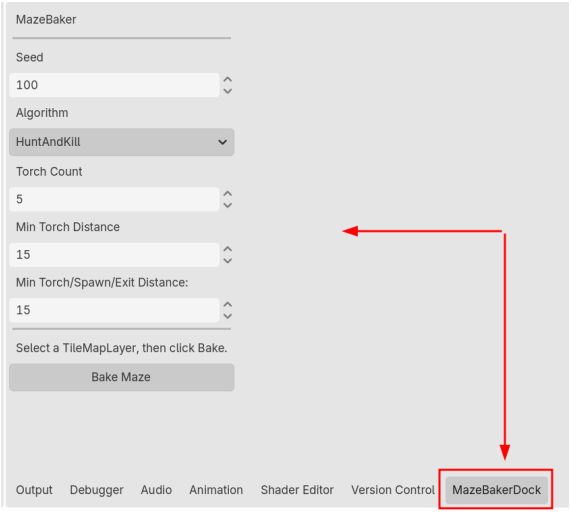
For the handcrafted levels to be derived from their procedural counterparts, designers needed to be able to view the generated maze and modify it directly — adding or removing walls, repositioning torches, and adjusting the start and exit positions. A maze painted fresh each time the game launched could not be edited in this way. The rendering had to happen at design time so that the output could be saved as a static, editable scene file. As the generation pipeline was custom-built, no native Godot tooling could expose the procedurally generated mazes at design time — a

custom plugin was therefore the only viable approach. This crucial insight motivated the development of an internal tool: the Maze Baker.

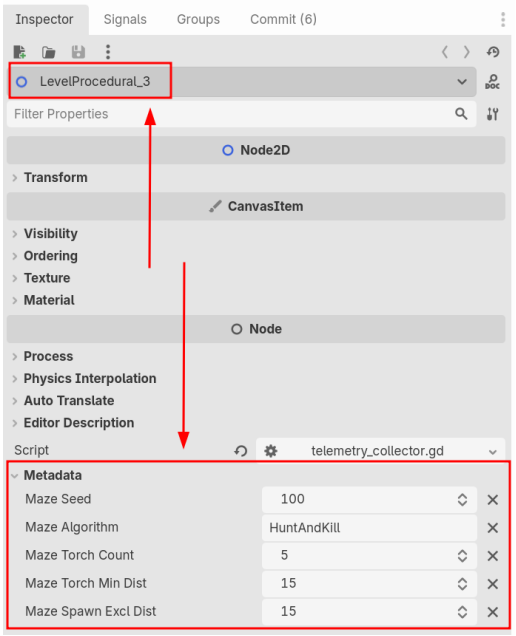
4.2.2 The Maze Baker

The Maze Baker is a custom editor plugin developed specifically for this project, accessible as a dock panel directly within the Godot editor. Rather than generating and rendering mazes at runtime, it exposes the full generation pipeline to the designer at design time. The designer specifies a seed, selects an algorithm, and configures generation parameters — including torch count, the minimum distance between placed torches, and a separate exclusion zone radius around the start and exit positions within which no torches may be placed. A single button press then runs the `ProceduralMazeGenerator` and `TileMapPainter` in sequence, painting the generated maze into the TileMapLayer of the currently open scene and places a start marker, exit portal, and torches at their generated positions.

The result is a fully rendered, static maze scene that exists as an ordinary Godot scene file — one that can be opened, inspected, and edited directly in the editor like any other level. A designer can add or remove walls, move the start and exit positions, reposition torches, all while seeing the complete maze layout in front of them. The generation parameters — seed, algorithm, torch count, and placement distances — are also written as metadata on the scene's root node, ensuring that the exact configuration that produced any given maze can always be recovered and reproduced.



(a) Maze Baker Plugin UI, accessed through the bottom navigation bar



(b) Inspecting a scene's metadata shows what settings were used to generate it

Figure 4.7: Custom-built tools for Godot.

This tool resolved the runtime editing problem and became central to the research methodology. By entering the same seed that had been used for a procedural level, a designer could produce an identical structural starting point and then apply deliberate modifications to create the handcrafted counterpart — with full visibility into the layout and complete control over every tile. The Maze Baker, therefore, transformed what had been a technical constraint into a methodological asset: a reproducible, inspectable, and editable foundation for the handcrafted level design process.

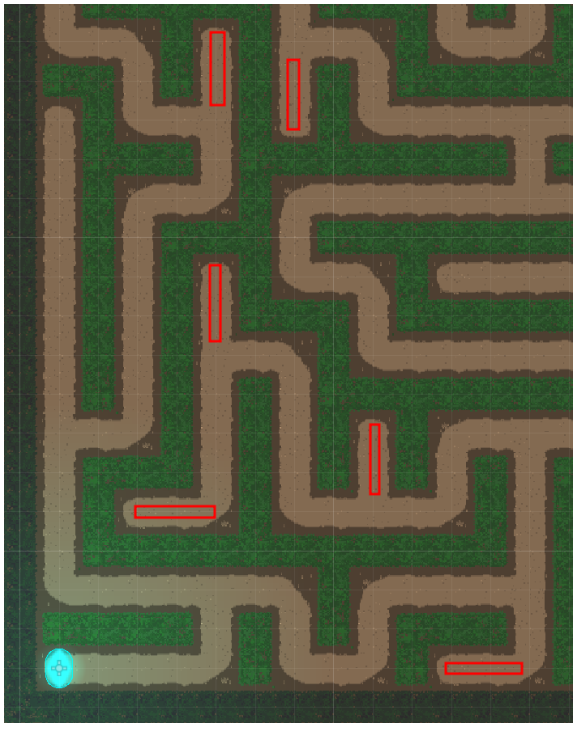
4.3 Procedural generation

The procedural generation system is responsible for producing the maze layouts used in both the procedural and handcrafted levels. All six mazes in the study were generated using this system: the three procedural levels by using the output directly, and the three handcrafted levels by using the output as a structural baseline for subsequent manual modification. The generation process follows a consistent sequence: a random number generator is seeded, the chosen algorithm carves paths through the grid, the outer boundary is replaced with impassable walls, and start and exit positions are assigned, as described in Section 4.3.3. Torch placements are determined last, after the maze structure is finalized, and the method of their placement will be described in more detail in Section 4.3.4.

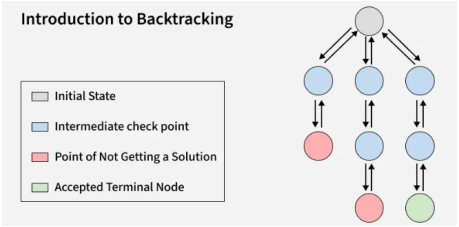
4.3.1 Algorithms

Hunt-and-Kill [28] was selected as the algorithm used for all six study mazes, a representative example of which is shown in Figure 4.10. Internal playtesting revealed that its output struck the most suitable balance for the intended gameplay experience—the mazes it produced did not present an obvious solution path for the player to follow and exhibited sufficiently subtle structural repetition that it was unlikely to be perceived as a characteristic of algorithmic generation. Two additional algorithms were implemented and evaluated: Recursive Backtracking [29] and Prim's algorithm [30]. While all three produce a solvable, fully connected maze (meaning there is always at least one navigable path between any two points), significant differences in structural character were observed during internal testing, and both alternatives were found to generate layouts less suitable for the intended gameplay experience, for the reasons described below.

Recursive Backtracking is a depth-first algorithm. Starting from an initial cell, it randomly selects an unvisited neighbour, carves a passage between them, and continues from the new position. When no unvisited neighbors remain, it backtracks until it finds a cell that has them. This process continues until the entire grid has been visited. The result tends to be a maze with long, winding corridors and relatively few branches and a strong sense of direction, but limited navigational variety [29]. However, during internal playtesting, mazes produced by this algorithm made the solution path readily apparent: dead ends were short enough that players could typically identify them without entering, allowing navigation with minimal active decision-making rather than genuine spatial reasoning.



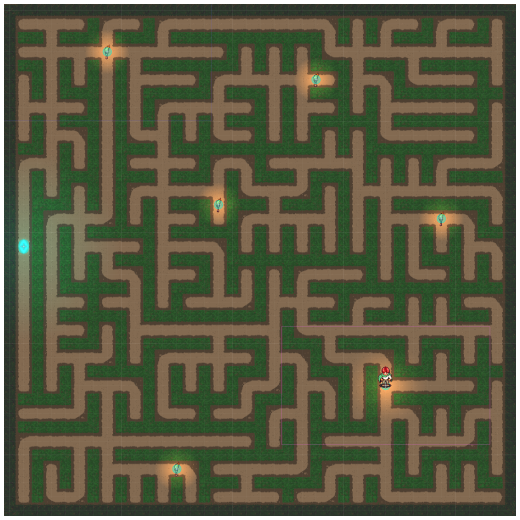
(a) Typical behavior of our Recursive Backtracking implementation, which generated very shallow branches - highlighted in red.



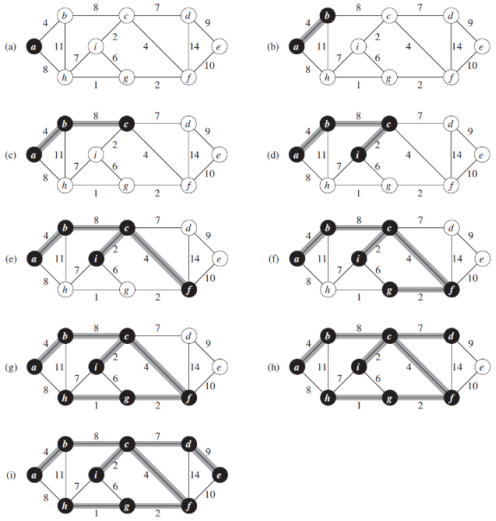
(b) Visualization of the recursive backtracking algorithm [29].

Figure 4.8: Recursive backtracking algorithm.

Prim's algorithm approaches maze generation as a minimum spanning tree problem. Beginning from a single cell, it maintains a frontier list of candidate walls on the boundary between visited and unvisited areas, and repeatedly selects one at random to carve through [30]. This produces a more uniformly distributed maze with shorter corridors and more frequent branching. However, mazes produced by this algorithm exhibited a high degree of visual repetition in internal testing, with frequent short branches creating a characteristic fork-like pattern across the grid. As with Recursive Backtracking, the resulting layouts made the solution path straightforward to identify, and internal playtesters rarely encountered dead ends that required meaningful backtracking.



(a) A maze from the prototype, procedurally generated using Prim's algorithm.

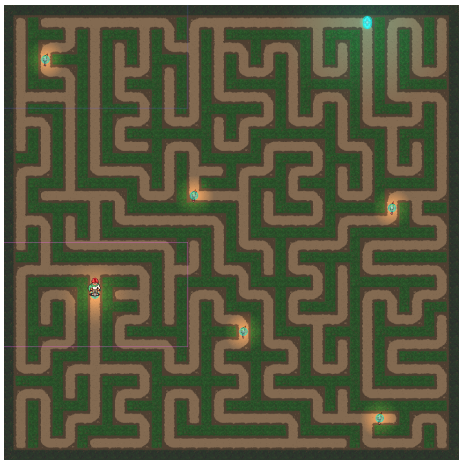


(b) Visualization of Prim's algorithm [30].

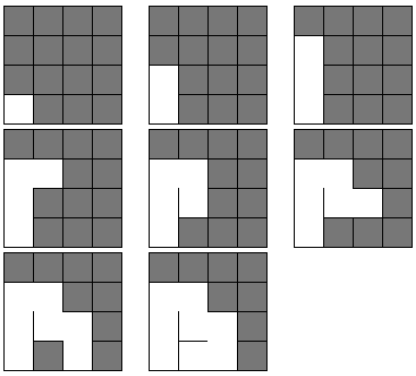
Figure 4.9: Mazes generated using our implementation of Prim's algorithm, showcasing heavy repetition of a fork-like pattern.

Hunt-and-Kill combines a random walk phase with a systematic scanning phase. The algorithm carves passages randomly until it reaches a dead end, then scans the grid row by row to find an unvisited cell adjacent to an already carved path, and resumes carving from there [28]. Unlike the other two algorithms, this produces structurally deep and winding dead ends with minimal apparent repetition observable to the player, characteristics that neither Recursive Backtracking nor Prim's reliably exhibited. During internal playtesting, mazes produced by Hunt-and-Kill consistently led players into extended dead ends, generating observable frustration. Within the context of the prototype's mechanics, this was precisely the intended effect. A player confronted with a deep dead end faces a meaningful decision: spend a torch charge to burn through the wall and immediately correct their path at the cost of reduced future visibility, or backtrack and preserve their charges. Either outcome carries a lesson: torch charges are a finite and valuable resource, and visibility suffers directly as a result of spending them. The frustration of a deep dead end therefore serves as a natural incentive to seek out ground torches proactively as the player progresses through the maze.

This decision-making is the core tension the game was designed around.



(a) Maze from the prototype, generated using the Hunt-and-kill algorithm.



(b) Step by step visualization of the Hunt-and-kill algorithm - in order of left to right, top to bottom.[28].

Figure 4.10: Hunt-and-kill algorithm.

The integration of Hunt-and-Kill marked a turning point in the development of the prototype as the game finally exhibited the gameplay character described in the Game Domain Description (see Appendix B). Given the time and resource constraints of the project, no further algorithms were pursued beyond this point, though the authors acknowledge that a considerably wider range of maze generation algorithms exists in the literature. For the purposes of this study and the gameplay experience it required, Hunt-and-Kill proved well-suited to both the research requirements and the intended design vision.

4.3.2 Seeds and determinism

The custom maze generation pipeline is seeded, which enables a procedurally generated maze to be reproduced, even though the underlying algorithm would otherwise produce a different maze each time it was applied. This is achieved through a seeded random number generator: the seed is an initial value that fully determines the sequence of random decisions made during the procedural generation process. The same seed, algorithm, and configuration will always produce the same maze, without exception. This determinism proved valuable at two distinct stages of the project.

During development, an initial batch of approximately twenty procedural mazes were generated and carefully evaluated before three seeds were selected for use in the study. The seeded approach made this process straightforward because rather than saving each candidate maze as a separate scene file, any maze could be regenerated at any time simply by reusing its seed, making it trivial to revisit and compare candidates from the entire batch throughout the selection process.

The selection process was not motivated by a need to avoid structurally poor layouts as none of the initial batch produced obviously problematic results, but rather to ensure that the chosen mazes were representative of what the algorithm typically produces. A purely random selection of seeds was considered but rejected, as it could not guarantee that the chosen mazes would fairly represent the algorithm's output or be sufficiently comparable to their handcrafted counterparts. Given the time constraints of the project and the team's experience with procedural generation, manually evaluating and verifying a set of candidate seeds was considered the most responsible approach — ensuring that neither the limitations of the algorithm nor the experience of the developers would be obviously reflected in the mazes presented to participants.

For the research methodology, determinism was equally essential. Each handcrafted level was derived from a procedural counterpart by entering the same seed into the Maze Baker, producing an identical structural starting point before any manual modifications were introduced. Without deterministic generation, reproducible paired baselines would not have been possible and without paired baselines, the designer bias mitigation described in Section 3.3.1 would have been significantly weakened.

4.3.3 Start and exit positions

The start and exit positions of each maze are determined procedurally from a predefined set of candidate positions. All candidate positions are defined at odd coordinates on both axes, and no runtime verification of these positions is required. This is because all three generation algorithms share a fundamental structural property: they start at coordinate (1,1) and advance in steps of exactly two tiles. Since adding an even number to an odd number always produces an odd number [31], the algorithm can only ever visit positions where both X and Y are odd. Furthermore, all three algorithms are guaranteed to produce a fully connected maze: meaning

every odd, odd coordinate in the grid is visited without exception. Any candidate position defined at odd-odd coordinates is therefore mathematically certain to be a carved path cell, regardless of which seed or algorithm is used.

This step-of-two property is fundamental to how grid-based maze generation works — by advancing two tiles at a time, the algorithm always leaves a wall tile between any two carved cells, which is precisely what creates the corridor-and-wall structure of the maze. If the algorithm advanced one tile at a time, every adjacent tile would be carved, and the result would be a single open room rather than a navigable maze.

This same structural property determines the minimum path lengths visible in the maze. Any corridor segment connecting two cells consists of the two cell tiles at either end: both at odd, odd coordinates, and the single carved wall tile between them — producing a minimum corridor length of three tiles. Dead ends are the one exception since they connect to the maze on only one side: a player entering a dead end walks through one carved wall tile and one dead end cell before hitting a wall — exactly two tiles. This distinction was consciously preserved during the handcrafting process, where all manual modifications to the maze layout were made in accordance with the same odd-coordinate grid structure. This ensured that the handcrafted levels remained structurally indistinguishable from their procedural counterparts at the tile level, meaning a player navigating either type of level encounters the same minimum path lengths and the same dead-end behavior.

To select the final start and exit positions, the candidate list is shuffled using a Fisher-Yates shuffle [32] driven by the seeded random number generator rather than Godot's built-in shuffle, which uses a global random number generator and would produce different results regardless of the configured seed. The start position is drawn as the first candidate from the shuffled list. The exit position is then selected as the first remaining candidate that satisfies a minimum Manhattan distance [24] threshold from the start, ensuring the two positions are never placed too close together. If no candidate meets this threshold, the furthest available candidate is used as a fallback. Together, these constraints ensure that every maze presents the player with a meaningful navigational challenge regardless of which seed is used.

Because the entire process uses the same seeded random number generator as the carving algorithm, start and exit positions are fully deterministic — the same seed always produces the same pair. As discussed in Section 4.3.2, this was essential for the paired level methodology: when the Maze Baker was used to produce a handcrafted baseline from a procedural seed, the start and exit positions were identical to those of the procedural counterpart, giving the handcrafted designer a defined structural starting point that could then be deliberately modified.

4.3.4 Torch placement

Ground torch positions are determined using a custom seeded placement algorithm that approximates the spatial distribution properties of Poisson disk sampling [33]: a distribution method that produces naturalistic, evenly spread placement by guaranteeing a minimum distance between any two placed items. Rather than implementing Bridson's algorithm directly, a simpler greedy sequential approach was used, as described below. This avoids the visible clustering that can arise from pure random scatter, producing a distribution that resembles how a human designer might intuitively space items across a space.

All eligible path tiles, excluding the start and exit positions, are first collected and shuffled using the same seeded Fisher-Yates algorithm [32] used to select the start and exit positions. The system then iterates through the shuffled candidates in order, accepting each one only if it satisfies a minimum distance constraint from all previously placed torches. This greedy sequential approach produces results that approximate Poisson disk sampling without requiring multiple sampling attempts per point.

The need for this approach over purely random placement arose after initial random scatters tended to produce visible clustering patterns: areas of high torch density alongside areas with none, which may be perceived by players as a characteristic of algorithmic rather than human placement. By ensuring a minimum distance between all torch positions, the distribution character is kept consistent across both level types, minimizing the risk that torch placement alone acts as an unintended cue to which type of level the player is navigating. The seeded implementation ensures that the same seed always produces the same torch positions, consistent with the determinism of the generation pipeline as a whole, as described in Section 4.3.2. In the Maze Baker, an additional exclusion zone parameter prevents torches from being placed in close proximity to the start and exit positions, ensuring that torch availability does not trivialize the opening or closing moments of each maze.

4.4 Handcrafted design philosophies

Each of the three handcrafted mazes was designed around a distinct underlying philosophy — a deliberate design intent intended to guide the player in specific ways that a procedural algorithm, by its nature, cannot produce. While the procedural mazes generated valid, solvable layouts that supported the core gameplay loop, they could not embed intentional player steering: the placement of torches, the structure of paths, and the arrangement of dead ends all emerged from the algorithm's logic rather than from a considered understanding of how a player would move through the maze. The handcrafted mazes were therefore designed to exploit precisely this capability. Each maze was conceived with a specific behavioral expectation in mind, anticipated to manifest as measurable differences in the collected telemetry.

The starting position, determined during the handcrafting process, gave the designer knowledge of the direction from which the player would first encounter each part of the maze. This made it possible to reason about when and how the player would encounter specific torches, dead ends, branching paths, and to design around those encounters deliberately. Three distinct philosophies emerged from this process.

During internal playtesting, it was observed that players consistently attempted to collect as many torches as possible and were notably reluctant to spend charges even when doing so would have been strategically beneficial. This hoarding tendency informed the design of all three handcrafted mazes: torch placement became the primary tool for steering player movement, and wall burns could be incentivized by placing a torch visibly on the other side of a wall, offering a net charge-neutral exchange that made the decision to burn feel justified rather than costly.

A secondary consideration in the design of the handcrafted mazes was the degree to which each maze would be perceptibly handcrafted through play. The visual and structural constraints described in Section 3.3.1, shared tile textures, consistent path lengths, and identical dead-end behavior, ensured that no surface-level visual cue could distinguish a handcrafted tile from a procedural one. The detectability gradient operated at a different level entirely: the navigational topology of the mazes. The first maze introduces deliberate torch positioning that, while frustrating, could be attributed to a specific seed. The second maze introduces islands: closed loops disconnected from the solution path, a topological feature that the procedural algorithm cannot produce. In practice, the limited visibility mechanic makes this nearly impossible to detect during play without extensively mapping the maze and the distinction is most apparent when the full topology is revealed in post-play analysis. Nevertheless, the feature is present and constitutes a verifiable structural difference between the two level types. The third maze places the player on an island at spawn, a structural deviation so fundamental that any attentive player is likely to recognize it as deliberate (unless it was the first maze that they played, since the order of levels was randomized, as described in Section 3.3.1). This escalating design strategy raised an implicit question alongside the primary research question: at what point does the handcrafted nature of a level become perceptible to the player through play, even when it remains visually indistinguishable?

4.4.1 Handcrafted 1 — Convincingly conventional

The first handcrafted maze was designed around torch placement as its primary challenge. Every torch was positioned just beyond a wall at the precise location where the player would first encounter it, presenting an immediate decision: burn through the wall to reach it directly, or navigate the surrounding corridors to find an alternative path to collect the torch. What was not apparent from the point of first encounter was that the detour to each torch was considerably longer than it appeared, making the pursuit of any torch costly regardless of the approach taken.

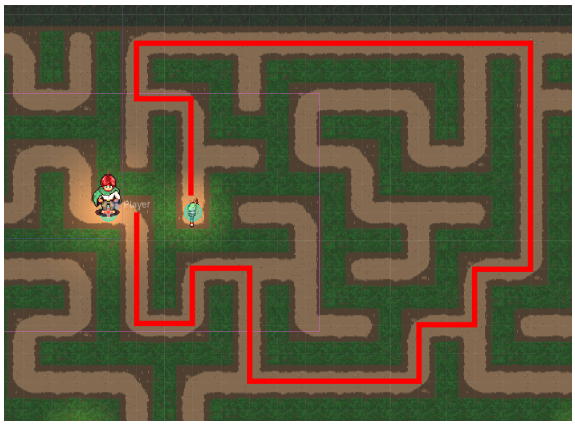


Figure 4.11: Long detour from when the player first encounters the torch to its actual position.

This design was not intended to produce a single predictable outcome, but rather to create meaningful variance in player behavior depending on individual strategy. A player who recognized the torches as detour traps and avoided them would cover less distance but suffer a net charge loss over time, navigating with less visibility. A player who pursued torches via the detour would cover considerably more distance but achieve a net charge gain, maintaining better visibility throughout. A player who burned walls to reach torches directly would achieve a net charge-neutral exchange — but might also inadvertently create a shortcut toward the exit in doing so. The maze was therefore expected to produce high variance across participants in both distance covered and torches collected, rather than a consistent directional shift in either metric.

4.4.2 Handcrafted 2 — Deceptively divided

The second handcrafted maze was built around a long, winding solution path as its primary structural feature. The solution path connected the start to the exit, but its winding structure traversed the entire maze before doing so — a deliberate design choice intended to frustrate players into seeking shortcuts. A player who then grew impatient would likely spot a nearby torch and burn through a wall to reach it, only to find that the torch had been placed on an island: a closed loop of path tiles with no connection to the rest of the maze. The torch offered a net charge-neutral exchange at best, and escape required either backtracking to the solution path or spending a second charge to burn out, compounding the resource cost and adding to the frustration.

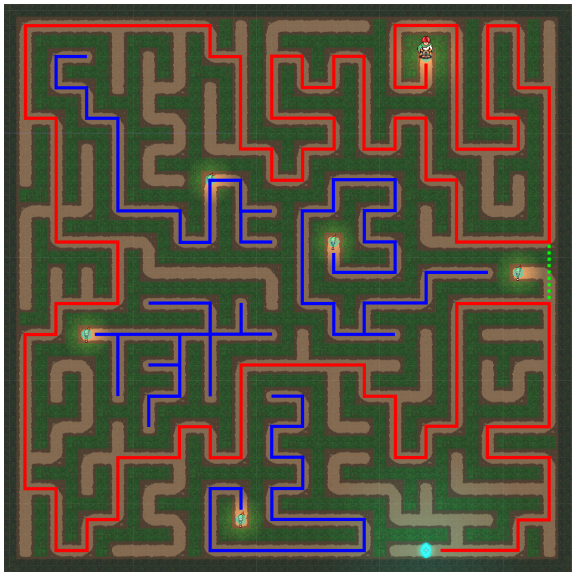


Figure 4.12: Red = solution path; Blue = Isolated islands with torches; Dashed green = immediate shortcut.

The exception was the first torch the player would encounter, placed close to the starting position but still requiring a wall burn to access it. Unlike the island torches, this one was not a closed loop—burning through to reach it placed the player considerably further along the solution path, a shortcut that significantly reduced the remaining distance to the exit. This was not information available to the player at the time of the decision, but for a player willing to immediately use the wall-burning mechanic, it was an unknowing lucky break.

This created an unusually wide expected range of outcomes: players who followed the solution path without deviation would travel the full winding distance to the exit; those who eventually took the bait of the island torches would accumulate additional detour distance and on top of that, without a net charge gain; and those who immediately took the shortcut at the first torch encountered would complete the maze considerably faster than either group. During the design phase, this maze was therefore expected to produce the largest interval in completion times of the three handcrafted levels.

4.4.3 Handcrafted 3 — Obvious to the observant

The third handcrafted maze was designed as a deliberate outlier — the only maze in the study that was structurally unsolvable without wall destruction. No uninterrupted path existed between the start and exit positions, meaning players were required to burn through at least three walls to complete the level.

The starting position was placed in a small island in one of the bottom corners of the maze. From this starting island, two torches were visible—deliberately positioned so that burning the wall separating the player from either torch would yield an immediate charge replacement, resulting in a net charge-neutral cost.

(a) Western **(b)** Northern

Figure 4.13: Torches visible to the player from the starting island.

These two torches were further placed on the same island, so that regardless of which direction the player chose to break out of the starting island, both could be collected without requiring additional wall burns. This gave the player a net charge gain after escaping the starting position — a deliberate safeguard against hard-locking, a state in which the player has exhausted all torch charges, cannot obtain more, but still requires walls to be destroyed in order to reach the exit, and is therefore unable to complete the maze.

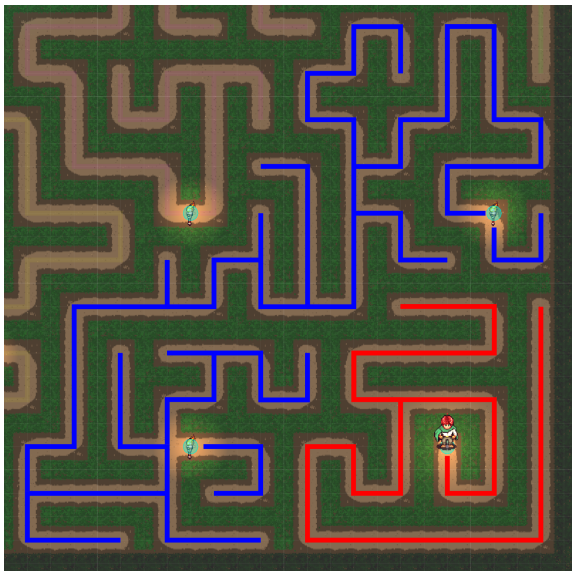
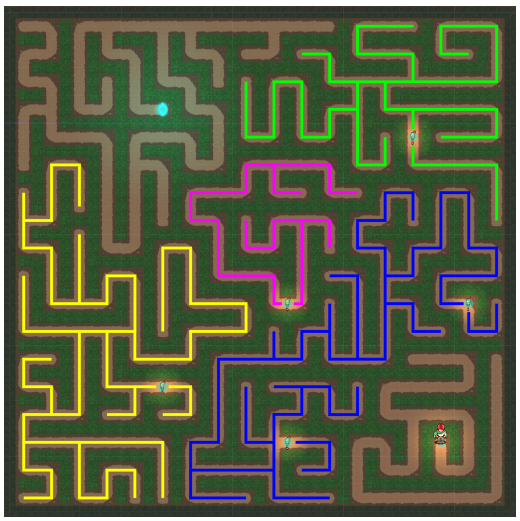
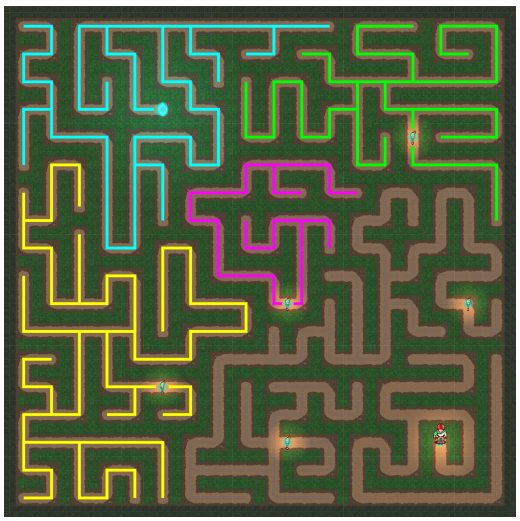


Figure 4.14: From the player's spawn island, represented in red, both torches on island 2, represented in blue, are visible.

Upon reaching the second island, the player could see three additional torches, each requiring a wall burn to reach and each situated on its own isolated island, offering no net charge gain. However, each of these three islands was positioned close enough to the exit that the faint blue glow of the exit marker would be visible from within them — providing the player with a directional hint and the confidence to commit to a wall burn despite the absence of a visible torch to offset the cost.



(a) From the second island represented in blue, the player is again guided by visible torches placed on the next islands, represented in yellow, magenta, and light green.



(b) From each of the three islands represented by yellow, magenta and green, there will be a section from where the exit is visible. The island of the exit is represented by cyan.

Figure 4.15: Level 6: Handcrafted 3.

This maze deviates most significantly from all others in the study, including the procedural levels — it is the only maze that is unsolvable without wall destruction, and the only one in which a hard-lock state is genuinely possible. It was therefore expected to be the maze most likely to be perceived by players as a distinctly handcrafted experience, as its topology could not plausibly emerge from an algorithm operating without design intent.

4.5 Maze comparison

All procedural mazes and their handcrafted equivalents are shown visually in Appendix A.1. Each procedural baseline is included in the appendix with the handcrafted level that was produced using the same seed. This comparison shows how the handcrafted levels introduce intentional modifications to topology, torch placement, and navigation flow while maintaining the same visual language and tile-level limitation.

4.6 Telemetry and data collection

To supplement the survey data collected during playtesting, a custom telemetry system was developed and integrated directly into the prototype. At game launch, the playtest coordinator is prompted to enter a session ID through a dedicated UI panel. The same session ID is recorded on the participant's survey, enabling game metrics and survey responses to be linked to a specific participant during analysis. The session ID also allows a play session to be resumed after an unintended interruption, such as a game crash, without losing previously collected data.

All levels, including the tutorial, were tracked with the single exception of the Level Hub. Telemetry data from the tutorial was retained as a potential diagnostic tool during analysis — a participant who spent significantly more time in the tutorial may have been more thorough in learning the mechanics, potentially explaining faster completion times in the mazes, while a participant who rushed through it may have entered the mazes unprepared, which could manifest as abnormally long completion times or higher wall destruction counts.

The following metrics are recorded automatically for each completed level:

- `session_id`: Unique identifier used to link game data to survey data
- `level_display_name`: Level name displayed to the participant in the UI (Level 1-6)
- `level_name`: Internal level name hidden from the participants (Level_1-Handcrafted_2)
- `timestamp`: The date and time at which the level was completed
- `time_seconds`: Total time spent in the level from entry to exit

- `torches_picked_up`: Number of ground torches collected during the run
- `walls_destroyed`: Number of hedge walls burned through during the run
- `distance_tiles`: Total distance walked within the maze

Each entry is appended to a per-session JSON file immediately upon level completion, ensuring that data from previously completed levels is preserved in the event of a crash. The separation between display names and internal research names means the data files can be analysed directly by level type without requiring a conversion table, while ensuring that naming conventions remain invisible to participants during play. The analysis of this data is discussed in Chapter 5.

5. Results

The gameplay metrics analyzed in this chapter were collected through the telemetry system described in Section 4.6. These metrics include completion time, movement distance, torches collected, and walls destroyed. Since the core mechanics described in Section 4.1 are built around limited visibility, torches collected, and wall destruction, these measurements are used to interpret how players interacted with the different maze structures.

This chapter provides the playtesting results from the 26 participants. The findings are categorized into gameplay and questionnaire data. The gameplay data includes completion time, movement distance, torches collected, and walls destroyed. The questionnaire data focuses on how players rated the various levels, specifically in terms of structure, randomization, challenge, success, and design quality.

The results are presented in relation to the research questions defined in Section 1.2. The gameplay data is used to address how level generation affects player behaviour. The questionnaire data is used to examine whether players perceive differences between level types and how they interpret structure and intentionality.

5.1 Gameplay

5.1.1 Performance

The first section of the results discusses gameplay performance. Completion time was measured to determine how long players (n=26) required to complete each level. This indicates how tough, complex, or time-consuming each maze was to navigate.

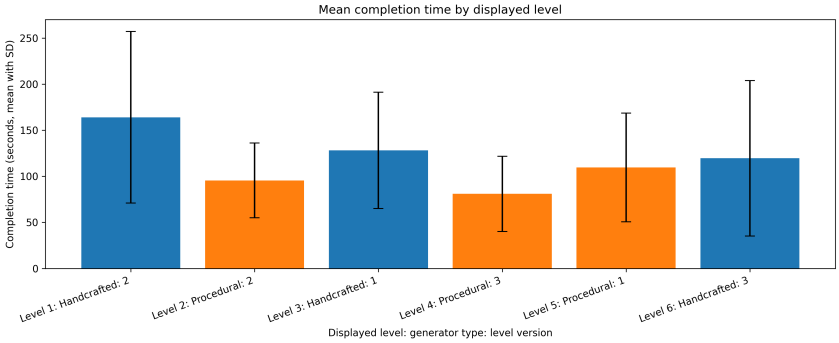


Figure 5.1: Mean completion time for each maze level.

Figure 5.1 depicts the mean completion time for each level. The findings demonstrate that completion times varied among individual levels. In general, handcrafted levels took longer to complete than procedurally generated levels. These results indicate that level type influenced player behavior, and the handcrafted levels in the

study required more time and movement, which suggests an increased need for exploration and decision-making. However, the data demonstrate that there was variance between individual levels, implying that level type alone cannot explain the entire difference.

When the levels are organized by type (see Figure 5.2), the distinction becomes evident. For example, *Level 1: Handcrafted 2* was purposefully created to be longer than the previous mazes while also offering an early shortcut to the end. This is shown in the completion time data, where this maze demonstrates the most diversity across players. This suggests that most players took the longer intended route, while a few players found and used the shortcut. Figure 5.5 is further supported by the fact that handcrafted levels had a larger mean number of demolished walls than mechanically generated levels. This shows that customized layouts encouraged players to employ torches more actively as navigation tools rather than as visibility resources. This, together with the completion time and mobility data, suggests that handcrafted levels increased engagement and decision-making throughout gameplay.

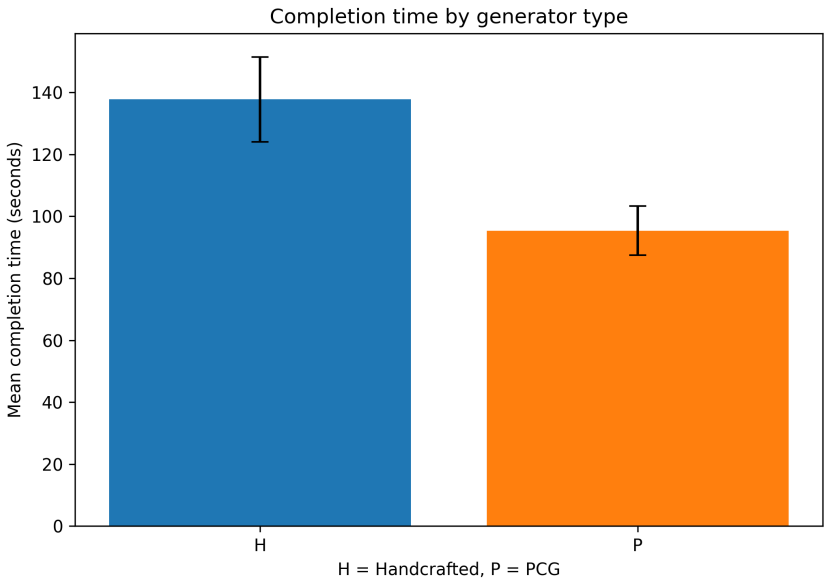


Figure 5.2: Mean completion time grouped by level generation type.

One possible explanation is that handcrafted levels demanded additional investigation, backtracking, or decision-making before players discovered the exit. The procedural levels may have resulted in more direct courses or layouts that were easier to understand. However, this finding should be regarded cautiously, as completion time can also be influenced by player familiarity, level order, and individual player skill.

This result should be evaluated in reference to the implementation given in Chapter 4. The handcrafted mazes were not produced as individual levels, but rather from procedurally generated baselines using the same seed and then manually adjusted. As a result, the longer completion times indicate that the manual alterations

incorporated during handcrafting had an impact on navigation, rather than being caused by various visual styles or basic mechanics.

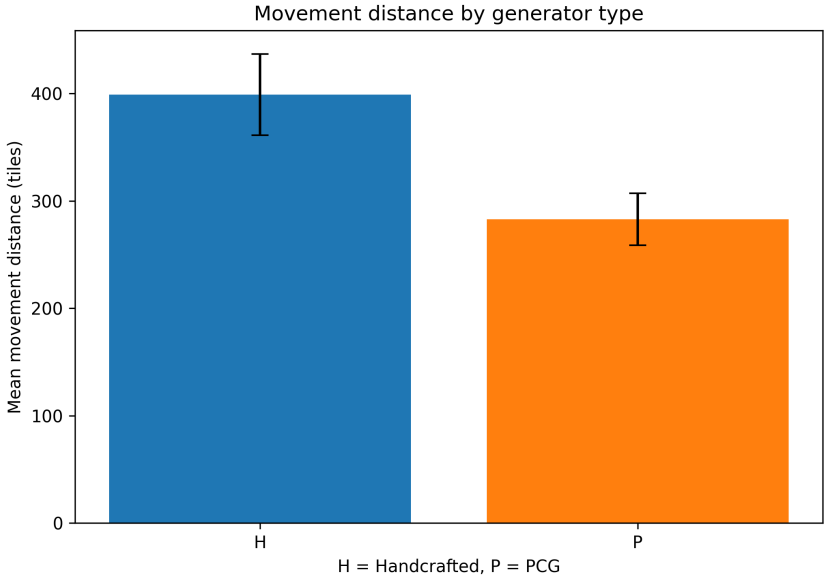


Figure 5.3: Mean movement distance by level generation type.

The movement distance data (how much players walked around the mazes) follows a similar pattern. As seen in Figure 5.3, participants traversed an average of 454.0 tiles in handcrafted levels versus 282.9 tiles in procedurally generated levels, a clear significant difference. This compliments the completion-time findings, as players spent more time and progressed further through customized levels.

The increased movement distance in handcrafted levels could imply that players explored more of the maze, took longer paths, or had to retrace more frequently. This shows that the handcrafted levels posed a more difficult navigation challenge. In contrast, shorter movement distances on procedural levels in the study indicate that the players were able to reach the exit with less exploration.

5.1.2 Game Mechanics

The second set of gameplay results addresses player engagement with key game mechanisms. Torches and destructible hedge fences are the prototype's main mechanics. Torches restore visibility while also being used to burn down the hedge walls to establish shortcuts, at the cost of player visibility. These mechanisms are crucial because they demonstrate how players interact with the level structure during gameplay.

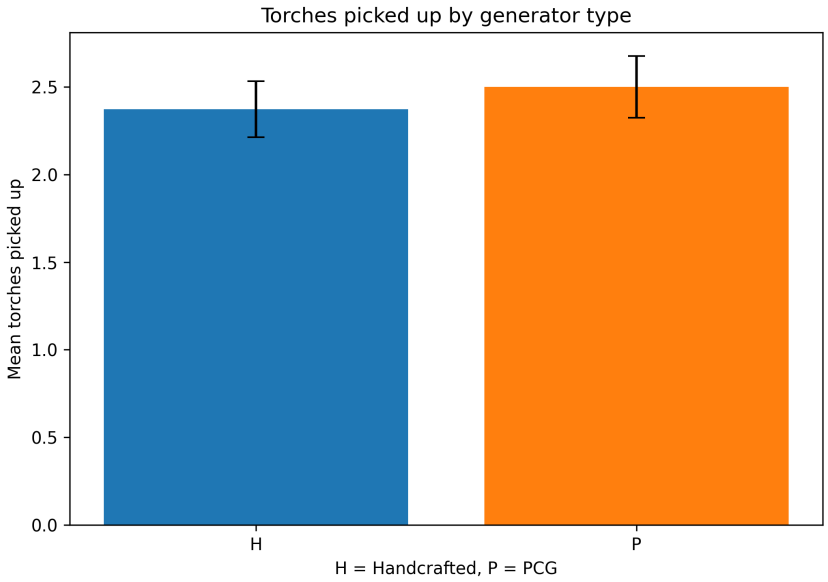


Figure 5.4: Mean number of torches picked up, grouped by level generation type.

Figure 5.4 depicts the mean number of torches picked up. There was no real difference between the two level generation types. This implies that torch gathering was essentially the same across both conditions.

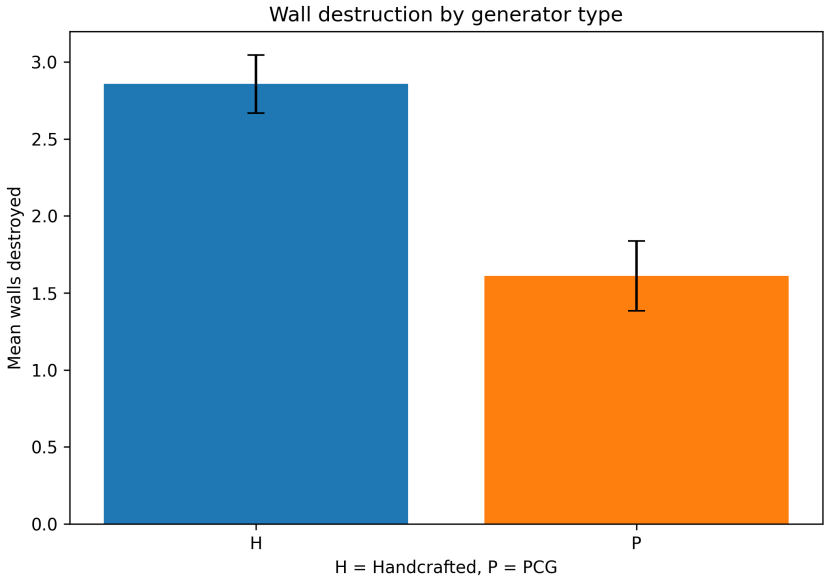


Figure 5.5: Mean number of wall destructions, grouped by level generation type.

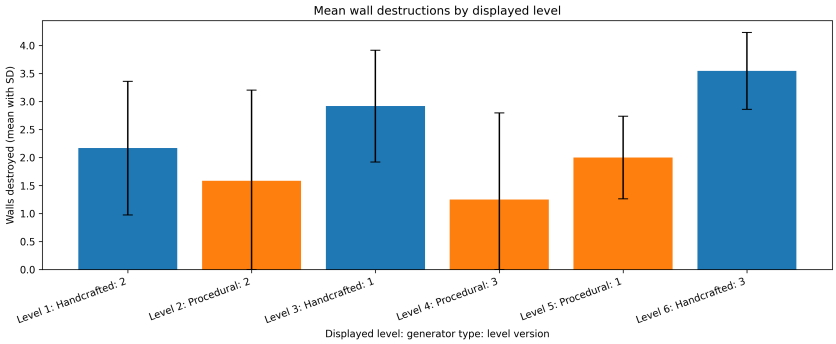


Figure 5.6: Mean number of wall destructions for each individual level.

However, a noticeable difference was observed in terms of wall destruction, as Figure 5.5 illustrates. This result should be interpreted with caution since wall destruction was not equally optional at all levels. As described in Section 4.4.3, *Level 6: Handcrafted 3* required wall burning in order to be completed, while the other mazes could be completed without using this mechanic.

For this reason, the wall destruction was also examined on a level-specific basis, shown in Figure 5.6. This makes it possible to see whether the grouped difference between handcrafted and procedurally generated levels is distributed across several handcrafted levels or mainly caused by *Level 6: Handcrafted 3*.

Destroying a hedge wall creates a shortcut, but it also reduces the player's light radius and therefore limits visibility. Wall destruction is therefore both a liberating action and a trade-off. If the level-specific data shows that several handcrafted levels had higher wall destruction counts, this may indicate that handcrafted layouts more often encouraged players to create alternative routes. If the difference is mainly concentrated in *Level 6: Handcrafted 3*, the result should instead be understood as a consequence of that level's specific design, rather than as a general property of all handcrafted levels.

This distinction was further supported by qualitative observations during playtesting. Two game developers who participated in the study independently identified *Level 6: Handcrafted 3* as the most memorable level. One participant, upon spawning on the starting island, immediately recognized that the maze was unsolvable without wall destruction, remarking that it was the first level they had encountered that forced deliberate use of the mechanic. In a post-session discussion, they described it as the most enjoyable level precisely because it required strategic thinking about where to spend torch charges, rather than simply navigating forward until the exit was found. Notably, the two participants held opposing views on difficulty: one found the maze among the more challenging, while the other considered it relatively easy — illustrating how individual player background and experience can produce divergent perceptions of the same level design.

5.2 Questionnaire results

The questionnaire results provide information about how participants perceived the levels after playing, individually and overall. These findings are particularly significant for the thesis's hypothesis, which argues that players will view handcrafted levels as more cohesive and deliberate than procedurally created levels. This section addresses the research questions of whether players perceived differences between level types and how they evaluated structure and intentionality. These results are used to evaluate the hypothesis.

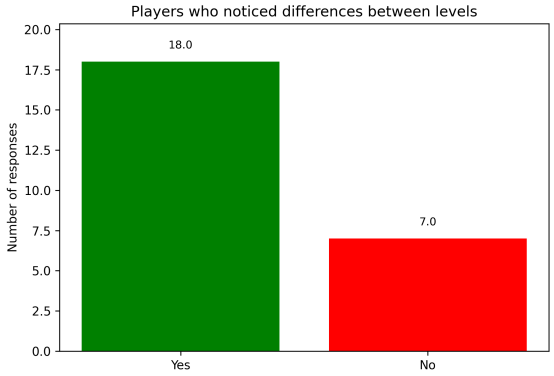


Figure 5.7: Number of players who noticed differences between levels (overall).

As Figure 5.7 illustrates, the majority of participants detected some difference between levels. 18 responses indicated that players observed a difference, while the remaining seven said they did not. This implies that most players noticed the variations between levels, even if they weren't told beforehand that they were either handcrafted or procedurally generated.

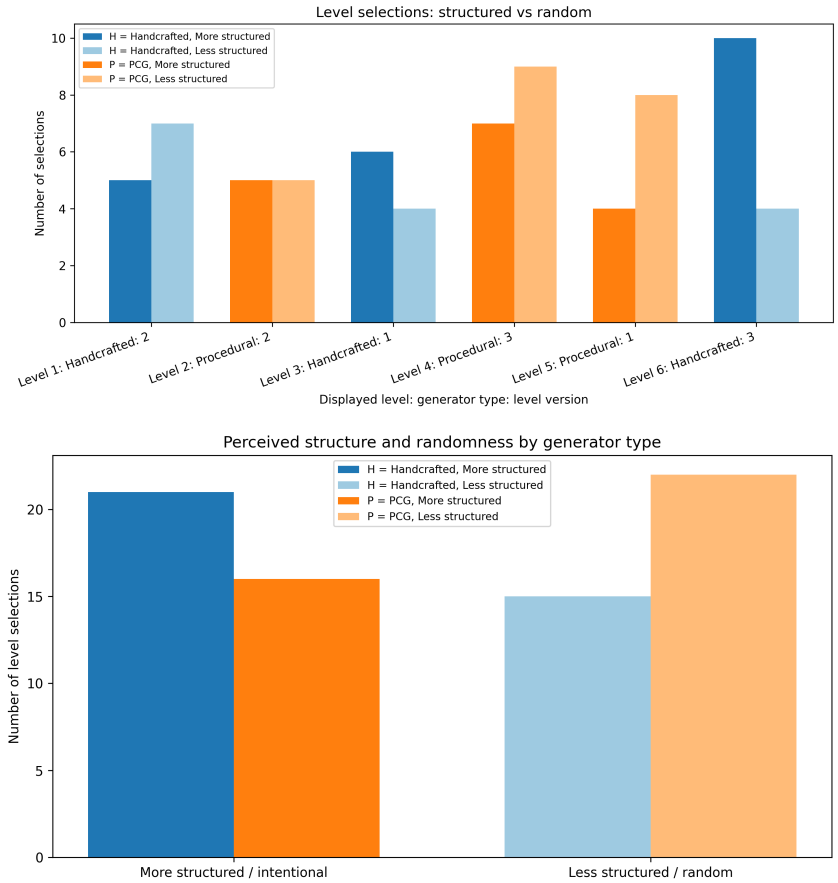


Figure 5.8: Maze perceived structure and randomness for each individual maze (top) and grouped by level generation type (below).

The plots in Figure 5.8 indicate how handcrafted and procedurally generated levels were perceived as more structured or random. The handcrafted levels were more often selected as 'more structured or intentional,' while the procedurally generated levels were more often selected as 'less structured or random'. This supports the hypothesis that handcrafted levels would more often be perceived as coherent and deliberately designed.

However, the level-specific results show that this distinction was not equal across all levels. *Level 6: Handcrafted 3*, corresponding to the third handcrafted maze (see Section 4.4.3), stands out as the clearest example of a level perceived as structured. This is not surprising when compared with the implementation described in Section 4.4.3, since this maze was intentionally designed around island-based progression and mandatory wall destruction. Its structure, therefore, contained clearer signs of deliberate design than the other mazes.

A possible theory is that handcrafted levels provide the designers more direct control over the player's route. The positioning of corridors, dead ends, torches, and exits in custom levels might be modified in order to provide a clearer feeling of progression. This is relevant to game design principles, where the level area serves as both a

visual environment and a structure for player movement and experience. Jenkins contends that game design shapes experience, designing locations, but the MDA framework explains how mechanics and level structure can generate player-facing dynamics such as exploration, challenge, and uncertainty [9], [13].

The level-based results in Figure 5.8 (top) indicate that the distinction between handcrafted and procedural design is clearer. For example, *Level 6: Handcrafted 3* was seen as more organized, implying that its layout provided players with clearer navigation clues or a more readable path through the maze. *Level 1: Handcrafted 2* produced a more mixed response, indicating that even a handcrafted design can appear less intentional if the route involves many confusing paths or if the player's orientation is not strongly steered.

The procedural levels also varied. *Level 5: Procedural 1* was more frequently perceived as less structured or random, which may indicate that its generated layout had more irregular branching or fewer readable navigation cues. At the same time, *Level 4: Procedural 3* was perceived as structured by some, showing that procedural generation was still capable of producing layouts that some players experienced as organized.

This is significant because procedural generation should be judged not only by whether it generates playable levels but also on the type and variety of level structures it produces. Smith and Whitehead define this as a generator's expressive range: the variety and style of levels can make [34]. In this project, the procedural generator was able to build workable maze layouts. The findings indicate that certain generated levels lacked the same visible design purpose as handcrafted levels. As a result, the key distinction was not procedural versus handcrafted, but how effectively each level communicated direction, tempo, and purpose to the player.

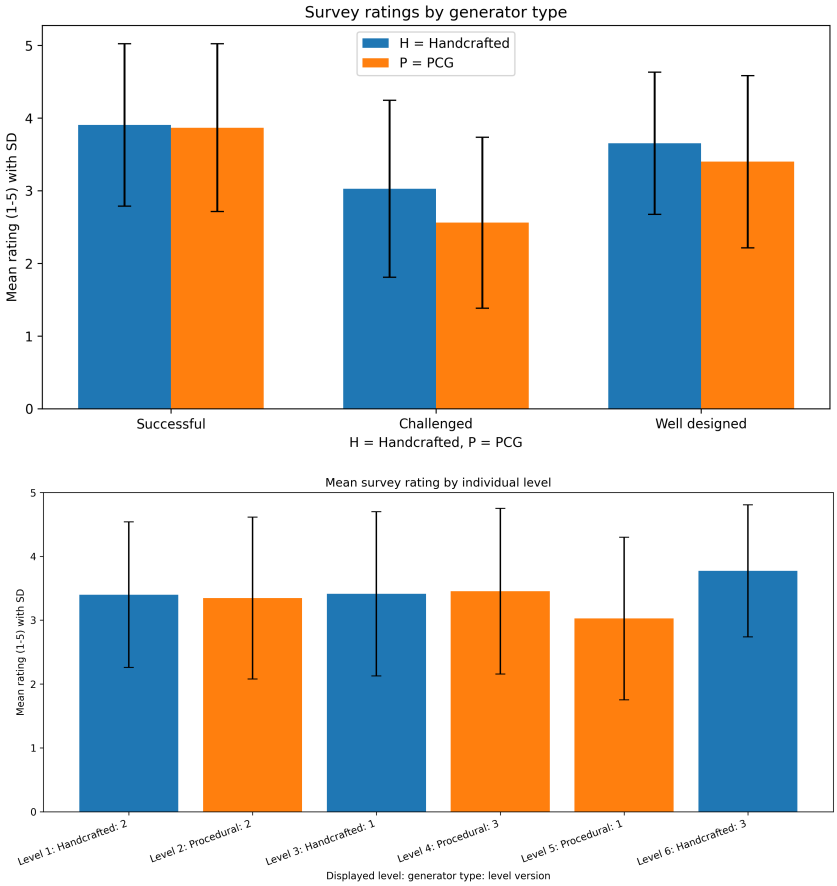


Figure 5.9: Mean ratings for how successful players felt, how challenged they were, and how well designed they thought the maze they just played was, grouped by level generation type (top). Mean rating for individual levels (bottom).

Figure 5.9 (top) illustrates the average questionnaire rating for successful completion, challenge, and perceived design excellence. Both level types earned comparable ratings for successful completion, with handcrafted levels scoring only marginally higher. This shows that participants felt capable of completing both handcrafted and procedurally generated levels.

The challenge rating indicates a larger variance. The tested handcrafted levels had a higher average challenge rating than the procedurally generated levels. This is consistent with gameplay data, which shows that the handcrafted levels took longer to complete and require higher movement distances. In total, these results indicate that handcrafted levels were perceived as more challenging.

Handcrafted levels were likewise relatively higher in terms of being well-designed. This provides credence to the theory that players evaluated handcrafted levels as more intentional. However, the difference was not significant, indicating that the procedurally generated levels were not considered poorly planned in general. Instead, they were slightly lower in perceived design quality while being enjoyable and intelligible.

Figure 5.9 (bottom) displays a more comprehensive overview of the average survey ratings for each level. This graph indicates that ratings varied not only depending on whether a level was handcrafted or procedurally generated but also among individual levels. *Level 6: Handcrafted 3* had the highest mean rating, while *Level 5: Procedural 1* scored the lowest. However, certain procedurally generated levels, such as *Level 4: Procedural 3*, were rated equivalent to or higher than some handcrafted levels. This suggests that the procedural generation approach used in this study was able to generate levels that were enjoyed by players, even though handcrafted levels were scored slightly higher overall.

This indicates that the distinction between handcrafted and procedural levels should not be seen as absolute. The findings indicate a general trend in which handcrafted levels were viewed more positively, while particular procedural levels may still do well depending on their layout and gameplay experience.

6. Discussion

The data reveals clear differences between the two level types. The results show that handcrafted levels required more exploration and interaction from the player. This suggests that players spent more time exploring and engaging with their surroundings in handcrafted levels. In contrast, procedurally generated levels were completed faster and needed less movement on average. This could indicate that the produced levels were easier to navigate or had more direct routes to the exit.

The questionnaire results partially support the hypothesis that players will regard handcrafted levels as more coherent and deliberate than procedurally generated levels. Players more often associated handcrafted levels with clear structure and intentional design, whereas procedurally generated levels were more commonly linked with randomness or reduced structure. Handcrafted levels were also rated significantly higher. At the same time, the distinction was not absolute, as some procedurally created levels were seen as structured. This demonstrates that procedural generation does not necessarily result in poor or random-feeling design, but rather that the quality of the created levels is heavily influenced by the generator's design and parameters.

As a result, the study indicates that procedural generation did affect user experience in this prototype, but not primarily by rendering the levels unplayable or unsuccessful. Instead, the difference was in how players advanced through the levels, how long they stayed in them, how often they used mechanisms like wall removal, and how deliberate the level layout appeared to them. This suggests that level structure influenced the player experience in *Light the Way*.

The handcrafted levels were not designed according to a single method. Instead, the three levels followed distinct underlying philosophies, where the degree of perceptible handcrafted topology increased across the set. The first level used more limited changes that could still be read as a procedurally generated layout, while the later levels relied more on islands, planned sightlines, torch visibility, and resource-based decisions. This distinction is important when interpreting the results. The level with the most deliberate use of navigational topology, *Level 6: Handcrafted 3*, was also the one that appeared to create the strongest sense of meaning for players. This suggests that handcrafted design had the greatest effect when it shaped the player's choices through intentional player steering, not only by changing the maze layout itself. The findings therefore support the hypothesis that handcrafted levels will be perceived as more unified and deliberate than procedurally generated ones.

The procedural levels were not seen negatively in all aspects. They received similar ratings for successful completion and somewhat lower ratings for perceived design quality. This shows that the procedural generation approach was capable of producing playable levels that worked within the game. However, unlike the handcrafted levels, the procedural levels were less frequently viewed as carefully planned.

The findings show that procedural generation can be effective for creating playable maze levels, but that more design control may be required if the goal is to build levels that players view as extremely cohesive and carefully constructed.

One significant takeaway from the research is that procedural content creation can be effective for creating playable maze levels, but it does not eliminate the need for design control. This is consistent with earlier research, which suggests that PCG is most effective when designers can direct the generation process using rules, restrictions, and parameters [10], [12]. In this experiment, the generator was able to create functioning and playable levels. However, the findings suggest that a hybrid approach may be more suitable than relying on procedural generation alone. Procedural generation can provide a useful structural baseline, while handcrafted refinement can add clearer direction, pacing, and a stronger sense of intentional design.

The experiment also demonstrates that level structure has a significant impact on player experience. In *Light the Way*, minor variations in maze layout influenced completion time, movement distance, wall destruction, and player perception. This relates to the MDA framework, in which mechanisms like mobility, lighting, torches, and wall destruction generate gameplay dynamics that alter the player's experience [13]. In this prototype, level design was more than just a technological structure; it also influenced how players perceived challenge, exploration, and orientation.

While the study provides useful insights, there are several limitations that should be acknowledged. The number of participants was minimal, so the findings should be considered as descriptive trends rather than statistically confirmed conclusions. The study also examined one prototype, one set of handcrafted levels, and one implementation of procedural generation. As a result, the findings should not be interpreted as a broad assessment of all procedural generation systems. A different generator, parameters, or game genre may yield different results.

Future work could continue exploring a hybrid workflow between procedural generation and handcrafted level design. This approach is comparable to the design direction used in games such as the *Remnant* series, where procedural systems are combined with authored tiles, events, and points of interest to balance replayability with intentional structure [6]. The results of this project suggest a similar direction: procedural generation can provide reproducible filler sections, while handcrafted sections can improve pacing, navigation, and add a sense of intentional design.

Such an approach could be implemented in two ways: either through a more complex generative algorithm with additional parameters designed to mimic human design decisions, or through a simpler system that procedurally assembles a set of pre-authored, handcrafted sections. The latter is likely the more practical direction: a complex algorithm capable of replicating intentional player steering, deliberate sightlines, and resource trade-offs would require more development time and careful parameter tuning, whereas a library of verified handcrafted sections keeps a human designer in the loop and avoids this complexity. One concrete example of a mechanic that could reinforce designer intentionality in such sections is the stone wall,

a tile type described in the Game Domain Description (see Appendix B), but not implemented in this prototype due to time constraints. Unlike hedge walls, stone walls cannot be destroyed by the player, meaning a designer could use them to enforce specific routes and prevent unintended shortcuts. Given that the study found players responded most strongly to levels where navigation felt deliberate and intentional, stone walls could serve as a targeted tool for embedding that intentionality into specific sections of an otherwise procedurally generated layout.

Future testing could also involve more participants, more levels, and deeper statistical analysis to strengthen the findings. It would also be useful to collect more qualitative feedback from players, for example, written explanations of why they perceived certain levels as structured, random, clear, or confusing. This could make it easier to connect player experience with specific level design choices.

7. Conclusion

This thesis studied the use of procedural content generation in a maze-based 2D game prototype, as well as how procedurally generated levels differ from handmade levels in terms of user experience. The project concentrated on the *Light the Way* prototype, in which players explore dark maze surroundings, collect torches, manage limited visibility, and look for an exit. The primary comparison was between handcrafted levels and levels created using procedural maze-generating algorithms.

The results indicate that both handcrafted and procedurally generated levels were playable in the prototype. Players were able to complete both sorts of levels, and the procedural levels received comparable ratings for success. This implies that the procedural generation algorithms were capable of creating viable maze layouts that supported the fundamental gameplay loops of exploration, navigation, and level progression.

The results show clear differences between handcrafted and procedurally generated levels in the tested prototype.

- RQ1: Do players perceive differences between handcrafted and procedurally generated levels?

The results show that most participants were able to perceive differences between the levels, even though they were not informed about how the levels were created. This indicates that variations in level structure were noticeable to players.

- RQ2: How do players perceive the structure and intentionality of handcrafted levels compared to procedurally generated levels?

The results reveal a distinction between handcrafted and procedurally generated levels. Handcrafted levels were more frequently seen as structured and deliberate, while procedurally generated levels were more often linked with randomness or lower structure.

These findings support the project's hypothesis, which states that players tend to perceive handcrafted levels as more cohesive and deliberate than procedurally generated levels.

- RQ3: How does level type influence player behavior and interaction during gameplay?

Handcrafted levels took longer to complete, required longer movement distances, and led players to use torches to destroy walls and create shortcuts more often. Procedurally generated levels, on the other hand, were completed faster and required less movement.

This pattern was particularly obvious on *Level 1: Handcrafted 2*. This layout was intended to require players to navigate across a substantial part of the maze before

reaching the exit, as shown in Figure 5.1, where it has the highest mean completion time. On the other hand, the level had an immediate shortcut near the maze's start. This is evident in the wide range of completion times: most players do not appear to have discovered the shortcut, while a few players completed the level substantially faster by using it.

These findings indicate that handcrafted levels were not only more time-consuming but also resulted in more diverse player behaviour. In this scenario, the design provided both a long intended route and a much shorter alternate route, depending on whether the player discovered and used the shortcut.

Overall, the study demonstrated that procedural generation is a promising technique for producing maze-based levels, particularly when the goal is to increase variation while reducing manual-level building work. However, the findings indicate that handcrafted design continues to have advantages in terms of perceived intentionality and structure. The best approach may thus be to view procedural generation as a tool that can help designers when combined with defined design goals and strict control over the generated output, rather than as a replacement for handcrafted design.

8. Use of AI Tools

During the writing process, AI-based tools were used to varying degrees depending on the chapter. For chapters such as Results and Conclusion, AI assistance was limited to language enhancement — improving sentence structure, finding suitable synonyms, and suggesting alternative formulations to make academic writing clearer and more coherent, and was never used to draw conclusions from the data. For technically dense chapters such as Implementation, where complex concepts had to be explained concisely across many pages and subsections, AI was additionally used to evaluate and refine the ordering of subsections, ensuring that concepts, problems, and solutions were introduced in a natural sequence for the reader. The content of each subsection was always determined by the authors. This was an iterative process involving multiple revisions, with all writing directions, arguments, and conclusions remaining the authors' own. AI functioned as a tool to support expression and structure, not as a source of ideas or conclusions.

For the development of the game prototype, AI tools were used in a syntactical rather than semantic capacity — assisting with the correct expression of solutions in GDScript rather than generating the solutions themselves. All design decisions, including those directly relevant to the research question — such as the methodology for mitigating designer bias and the deliberate variation of global topology between level types — were conceived and made exclusively by the authors. All game assets were either produced by the research team or sourced under licenses permitting their use in this project.

References

[1] Statista, *Daily time spent playing console games worldwide, by country*, https://www.statista.com/statistics/1263080/daily-time-console-games-country/, Accessed: 2026-04-13, 2024.

[2] Statista, *Video games - statistics & facts*, https://www.statista.com/topics/868/video-games/, Accessed: 2026-04-13, 2026.

[3] D. P. Bourscheid, A. V. Steil, and E. da Silva Paixão, "Crunch on video game production: Practices to avoid for healthier workplaces in game development," in *Proceedings of the 20th International Conference on the Foundations of Digital Games*, 2025, pp. 1–5. DOI: 10.1145/3723498.3723730.

[4] Mojang Studios. "Minecraft Official Site," Accessed: May 12, 2026. [Online]. Available: https://www.minecraft.net/en-us.

[5] 2K Games and Gearbox Software. "Latest News | Borderlands 4," Accessed: May 12, 2026. [Online]. Available: https://borderlands.2k.com/borderlands-4/news/.

[6] Fextralife Wiki. "Remnant 2 Wiki," Accessed: May 12, 2026. [Online]. Available: https://remnant2.wiki.fextralife.com/Remnant+2+Wiki.

[7] N. Andreasson, M. Granath, G. Johnsén, D. Memedov, B. Suhail, and E. Young, "Procedurally generated content in games and its effect on player experience," Chalmers University of Technology, Bachelor Thesis, 2025. [Online]. Available: https://hdl.handle.net/20.500.12380/310844.

[8] G. N. Yannakakis and J. Togelius, "Experience-driven procedural content generation," *IEEE Transactions on Affective Computing*, pp. 147–161, 2011. DOI: 10.1109/T-AFFC.2011.6.

[9] H. Jenkins, *Game design as narrative architecture*, Essay / manuscript PDF, 2003.

[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*, 1:1–1:22, 2013. DOI: 10.1145/2422956.2422957.

[11] O. Korn, M. Blatz, A. Rees, J. Schaal, V. Schwind, and D. Görlich, "Procedural content generation for game props? a study on the effects on user experience," *Computers in Entertainment*, 1:1–1:15, 2017. DOI: 10.1145/2974026.

[12] R. van der Linden, R. Lopes, and R. Bidarra, "Designing procedurally generated levels," in *Artificial Intelligence in the Game Design Process 2: Papers from the 2013 AIIDE Workshop*, AAAI Press, 2013, pp. 41–47. DOI: 10.1609/aiide.v9i3.12592. [Online]. Available: https://ojs.aaai.org/index.php/AIIDE/article/view/12592.

[13] R. Hunicke, M. LeBlanc, and R. Zubek, *MDA: A formal approach to game design and game research*, Paper developed for the Game Design and Tuning Workshop, Game Developers Conference, San Jose, 2004.

[14] R. Budiu, *Between-subjects vs. within-subjects study design*, https://www.nngroup.com/articles/between-within-subjects/, Nielsen Norman Group, Accessed: 2026-04-14, 2023.

[15] A. M. Connor, T. J. Greig, and A. M. Kruse, "Evaluating the impact of procedurally generated content on game immersion," *The Computer Games Journal*, pp. 209–225, 2017. DOI: 10.1007/s40869-017-0043-6.

[16] J. Taylor, T. D. Parsons, and I. Parberry, "Comparing player attention on procedurally generated vs. hand crafted sokoban levels with an auditory stroop test," Laboratory for Recreational Computing, Department of Computer Science & Engineering, University of North Texas, Tech. Rep., 2015.

[17] Godot Engine, *Introduction — godot engine documentation*, https://docs.godotengine.org/en/stable/about/introduction.html, Accessed: 2026-04-13, 2026.

[18] M. Games. "Make a 2d action & adventure rpg in godot 4 - tutorial series," Accessed: May 18, 2026. [Online]. Available: https://www.youtube.com/playlist?list=PLfcCiyd_V9GH8M9xd_QKlyU8jryGcy3Xa.

[19] M. Games. "2d action-adventure rpg assets," Accessed: May 18, 2026. [Online]. Available: https://michaelgames.itch.io/2d-action-adventure-rpg-assets.

[20] rone3190. "Torch 32x32 animated," Accessed: May 18, 2026. [Online]. Available: https://rone3190.itch.io/torch-32x32-animated.

[21] T. Ferreira. "Bubble explosion," Accessed: May 18, 2026. [Online]. Available: https://opengameart.org/content/bubble-explosion.

[22] artisticdude. "Fire & evil spell," Accessed: May 18, 2026. [Online]. Available: https://opengameart.org/content/fire-evil-spell.

[23] QuadSpinner. "Gaea discord community." Official Gaea community and support page, Accessed: May 16, 2026. [Online]. Available: https://quadspinner.com/Support/Community.

[24] GeeksforGeeks. "Manhattan distance." Last updated: 26 August 2025, Accessed: May 16, 2026. [Online]. Available: https://www.geeksforgeeks.org/data-science/manhattan-distance/.

[25] G. Walker. "Wang tiles: Blob tileset." Mirror of Guy Walker's Stagecast, hosted by BorisTheBrave, Accessed: May 6, 2026. [Online]. Available: https://www.boristhebrave.com/permanent/24/06/cr31/stagecast/wang/blob.html.

[26] H. Wang, "Proving theorems by pattern recognition — ii," *Bell System Technical Journal*, 1961. DOI: 10.1002/j.1538-7305.1961.tb03975.x.

[27] michagamedev. "Autotiling in godot 3.0," Accessed: May 18, 2026. [Online]. Available: https://michagamedev.wordpress.com/2018/02/24/181/.

[28] J. Buck, *Maze generation: Hunt-and-kill algorithm*, https://weblog.jamisbuck.org/2011/1/24/maze-generation-hunt-and-kill-algorithm, Accessed: 2026-04-13, 2011.

[29] J. Buck, *Maze generation: Recursive backtracking*, https://weblog.jamisbuck.org/2010/12/27/maze-generation-recursive-backtracking, Accessed: 2026-04-13, 2010.

[30] J. Chen, "The analysis and application of prim algorithm, kruskal algorithm, boruvka algorithm," *Applied and Computational Engineering*, 2023. DOI: 10.54254/2755-2721/19/20231012.

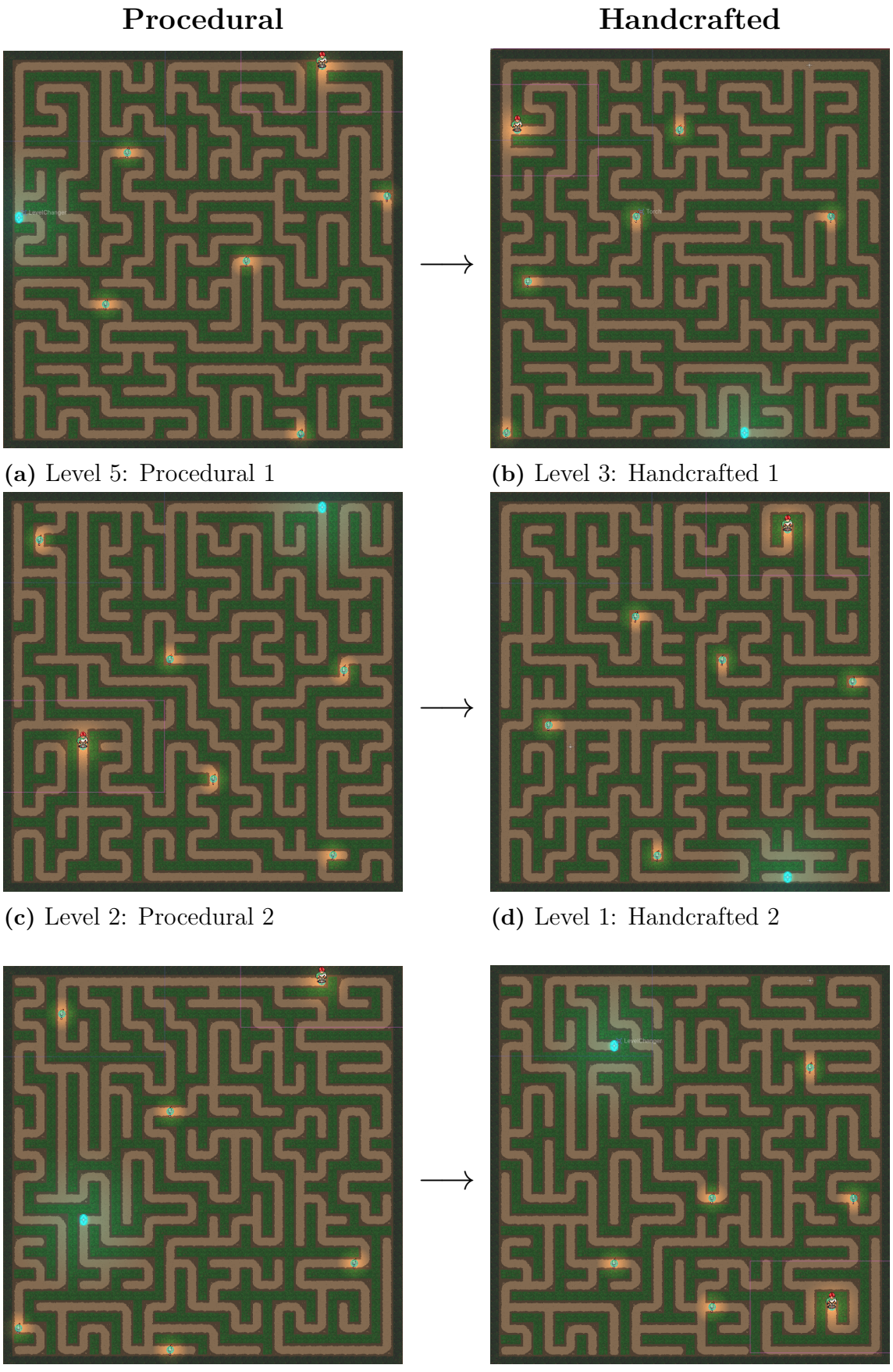
[31] K. H. Rosen, *Discrete Mathematics and Its Applications*, 8th. McGraw-Hill Education, 2019, ISBN: 978-1-259-67651-2.

[32] D. E. Knuth, *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*, 3rd. Addison-Wesley, 1998, vol. 2, See p. 145.

[33] R. Bridson, "Fast poisson disk sampling in arbitrary dimensions," in *ACM SIGGRAPH 2007 Sketches*, Association for Computing Machinery, 2007. [Online]. Available: https://dl.acm.org/doi/10.1145/1278780.1278807.

[34] G. Smith and J. Whitehead, "Analyzing the expressive range of a level generator," in *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, 2010. DOI: 10.1145/1814256.1814260. [Online]. Available: https://doi.org/10.1145/1814256.1814260.

A. Game Maze Levels



(a) Level 5: Procedural 1

(b) Level 3: Handcrafted 1

(c) Level 2: Procedural 2

(d) Level 1: Handcrafted 2

(e) Level 4: Procedural 3

(f) Level 6: Handcrafted 3

Figure A.1: All mazes used in the prototype. Each row shows a procedurally generated maze (left) and its handcrafted counterpart derived from the same seed (right).

B. Game Domain Description

Note: This Game Design Document (GDD) was created during the pre-production stage to describe the base mechanics, core gameplay loop, and overall vision of the game. One planned feature, stone walls, was ultimately not implemented, as the added complexity needed for the feature to add real value to the research, could not be justified given the development time available. As a result, this GDD does not fully represent the final state of the game, but it does capture our original vision.

Light the Way is a top-down 2D maze game where you navigate a dark hedge maze using a torch with 3 charges. Each charge represents torch intensity: high, medium, or low — which directly controls how far you can see. You may spend a charge to burn through hedge walls as shortcuts, but your light dims with each use. At zero charges, your torch goes out completely, leaving you nearly blind and forced to navigate toward distant torch beacons.

Torches are scattered throughout the maze — always visible even in darkness. Touch one to refill your torch charges, set a respawn point, and receive a directional hint pointing toward the exit.

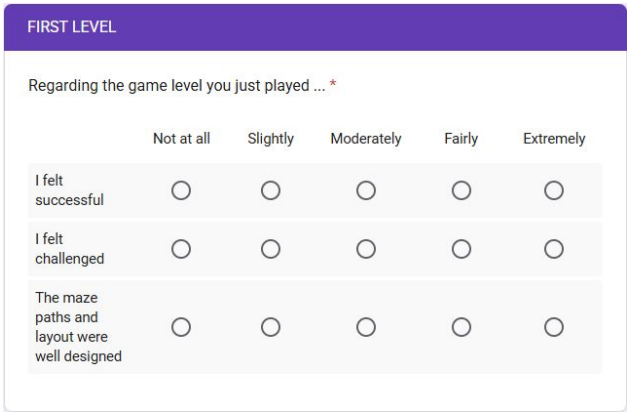
If you find yourself desperate: torch burnt out, wandering in circles, but one wall away from salvation — you can force your way through the thorny hedges at the cost of significant health. It's a last-resort gamble: reach the torch and survive, or go mad in the darkness.

Not all walls can be burned or crawled through. Stone walls are impassable barriers that force you to navigate around them, making maze topology actually matter—you can't just burn straight to the exit.

Both handcrafted and procedurally generated mazes use stone walls to create dead ends, false paths, and forced detours. The question is whether algorithmic placement of these obstacles can match human designers at being cleverly cruel.

The core tension is three-way resource management: do you burn through walls for shortcuts, conserve charges to maintain vision, or risk your health pushing through hedges when desperate? Every choice has consequences.

C. Playtesting Questionnaire



FIRST LEVEL

Regarding the game level you just played ... *

	Not at all	Slightly	Moderately	Fairly	Extremely
I felt successful	○	○	○	○	○
I felt challenged	○	○	○	○	○
The maze paths and layout were well designed	○	○	○	○	○

Figure C.1: Player's brief impression after playing a game level. This same set of questions was repeated after each level, six times in total.

Please indicate how you felt while playing the game. *

	Not at all	Slightly	Moderately	Fairly	Extremely
I was interested in the game	○	○	○	○	○
I felt successful	○	○	○	○	○
I felt bored	○	○	○	○	○
I found the experience impressive	○	○	○	○	○
I lost track of time while playing	○	○	○	○	○
I felt frustrated	○	○	○	○	○
I found the experience tiring	○	○	○	○	○
I felt irritable	○	○	○	○	○
I felt skilful	○	○	○	○	○
I felt completely absorbed	○	○	○	○	○
I felt satisfied with my performance	○	○	○	○	○
I felt challenged	○	○	○	○	○
I had to put a lot of effort into it	○	○	○	○	○
I felt good overall	○	○	○	○	○

Figure C.2: Player's overall impression after playing all game levels.

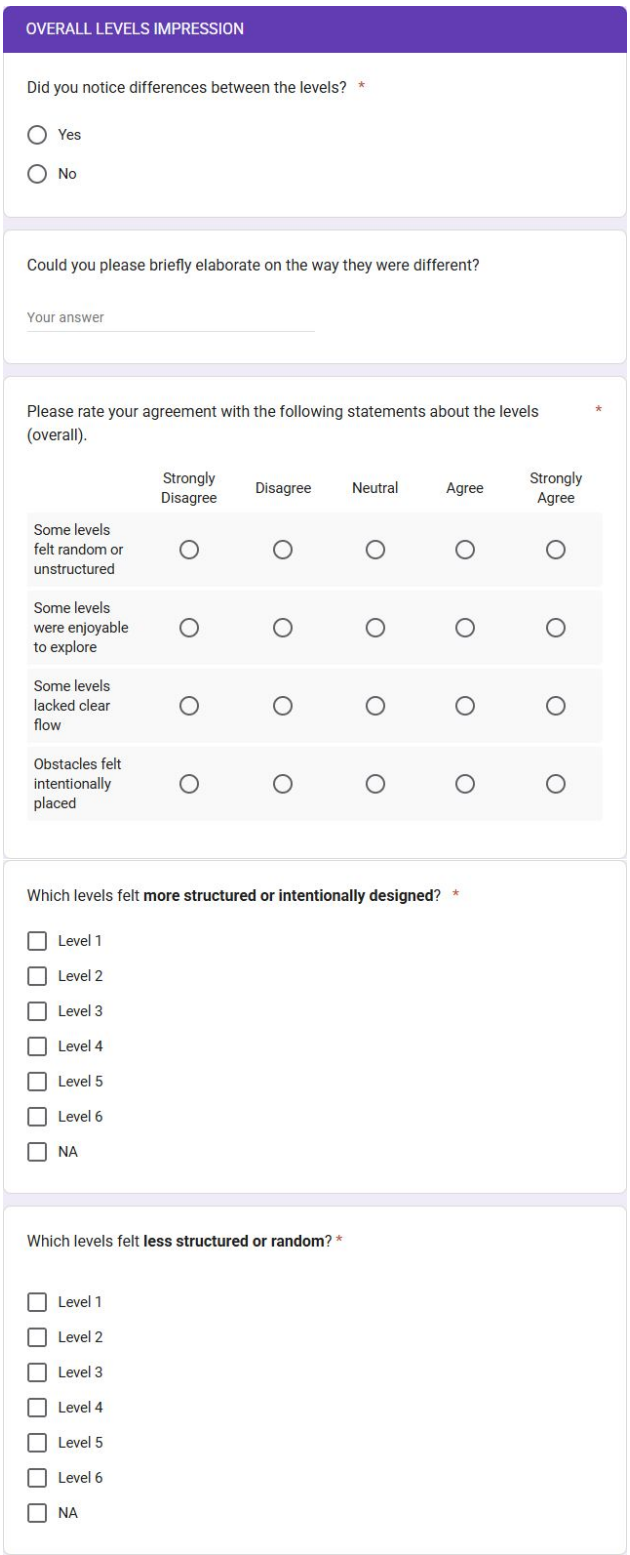


Figure C.3: Additional player impressions after playing all game levels.

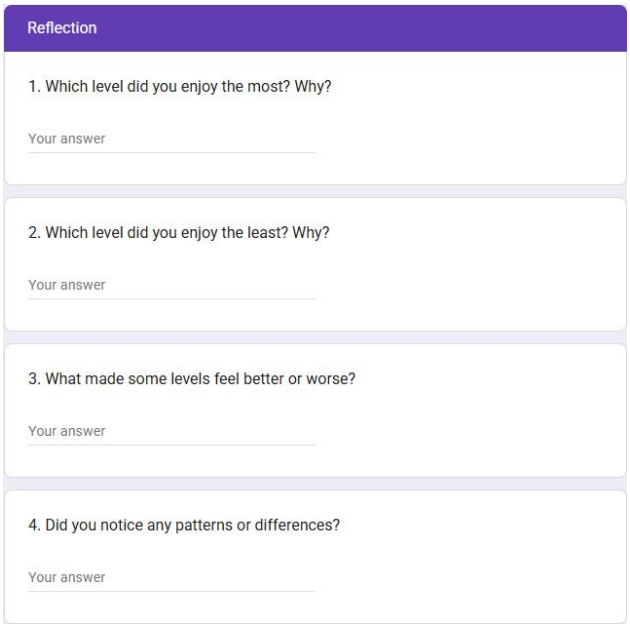


Figure C.4: Additional player impressions after playing all game levels.

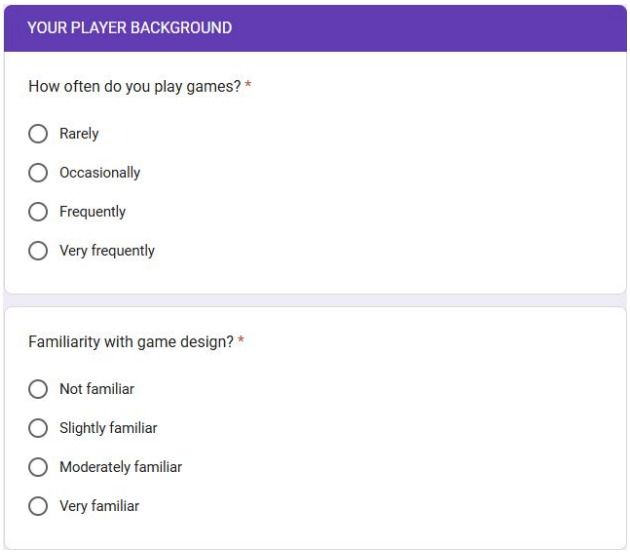


Figure C.5: Demographic information, collected at the very end.