Procedurally Generated Content in Games and its Effect on Player Experience

Bachelor’s thesis in Computer Science and Engineering

Noah Andreasson
Mårten Granath
Gabriel Johnsén
David Memedov
Bisher Suhail
Edgar Young

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2025

Bachelor's thesis 2025

Procedurally Generated Content in Games and its Effect on Player Experience

Noah Andreasson
Mårten Granath
Gabriel Johnsén
David Memedov
Bisher Suhail
Edgar Young

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2025

Procedurally Generated Content in Games and its Effect on Player Experience
Noah Andreasson Mårten Granath Gabriel Johnsén David Memedov Bisher Suhail Edgar Young

Supervisor: Aris Alissandrakis, Department of Computer Science and Engineering
Examiners: Patrik Jansson and Arne Linde, Department of Computer Science and Engineering
Graded by teacher: Jasmina Maric, Department of Computer Science and Engineering

Bachelor's Thesis 2025
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: PCG text in the form of graphs, made in Lucid (lucid.co)

Typeset in LaTeX
Gothenburg, Sweden 2025

Procedurally Generated Content in Games and its Effect on Player Experience
Noah Andreasson, Mårten Granath, Gabriel Johnsén, David Memedov, Bisher Suhail, Edgar Young

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

Abstract

This thesis investigates the relationship between algorithmic complexity, procedural content complexity, and user experience for procedural content generation (PCG) in video games. It examines whether increased algorithmic sophistication leads to more varied and higher-quality content, and how such content compares to handcrafted levels in terms of player experience.

A prototype game was developed in Unity using C# to compare two PCG approaches, a basic random walk algorithm and a more advanced graph rewriting system. The generated game levels were evaluated using both player-centric and technical methods. Player experience was assessed through Think-Aloud play-testing, semi-structured interviews, the Game Experience Questionnaire (GEQ), and gameplay data analysis. Algorithmic complexity was determined using Halstead complexity measurements and Cyclomatic Complexity.

Results indicate that while the graph rewriting algorithm produced more diverse and engaging level structures than the random walk approach, it also introduced greater development complexity and reduced maintainability. Additionally, gameplay issues arose in procedurally generated content that were not present in handcrafted levels. These findings highlight the trade-offs between PCG complexity, content control, and content variety.

Ultimately, the thesis emphasises the need for controlled iteration, player-focused evaluation, and careful variable isolation in PCG system development. It provides actionable insights for game developers on when and how to apply procedural generation effectively while balancing content quality, development time, and system maintainability.

Keywords: Procedural Content Generation, PCG, User Study, Game Experience Questionnaire, Think-Aloud, Algorithmic Complexity, Halstead's Complexity Metrics, Graph Rewriting, Random Walk, Design Constraints in Games

Procedurally Generated Content in Games and its Effect on Player Experience
Noah Andreasson, Mårten Granath, Gabriel Johnsén, David Memedov, Bisher Suhail, Edgar Young

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

Sammandrag

Denna uppsats undersöker relationen mellan algoritmisk komplexitet, procedurell innehållskomplexitet och användarupplevelse inom procedurell innehållsgenerering i datorspel. Den undersöker ifall ökad algoritmisk komplexitet leder till mer varierat och högkvalitativt innehåll, och hur sådant innehåll jämför sig med handgjorda banor när det gäller användarupplevelse.

En spelprototyp utvecklades i Unity med användning av C# i syfte att jämföra två PCG-metoder, en grundläggande slumpvandringsalgoritm samt ett mer avancerat *graph rewriting*-system. De banor som genererats utvärderades genom både spelarfokuserade och tekniska metoder. Spelarens upplevelse uppskattades genom *Think-Aloud*-speltester, semistrukturerade intervjuer, *Game Experience Questionnaire* (GEQ) och analys av data insamlat under spel. Algoritmisk komplexitet bestämdes genom Halstead's komplexitetsmått och Cyklomatisk komplexitet.

Resultaten visar att medan *graph rewritning*-algoritmen gav mer varierade och engagerande banor än slumpvandrings-metoden, så medförde den också högre utvecklingskomplexitet och minskad underhållbarhet. Dessutom uppstod problem med spelupplevelsen i genererade banor som inte fanns i handgjorda banor. Dessa resultat belyser kompromisserna mellan procedurell innehållsgenereringskomplexitet, kontroll över innehållet och dess variation.

Slutligen betonar uppsatsen behovet av kontrollerad iteration, spelarfokuserad utvärdering och noggrann variabel isolering i systemutveckling av procedurell innehållsgenerering. Den ger spelutvecklare praktiska insikter om när och hur man kan tillämpa procedurell innehållsgenerering på ett effektivt sätt och samtidigt balansera innehållskvalitet, utvecklingstid och systemets underhållbarhet.

Nyckelord: Procedurell Innehållsgenerering, PCG, Användarstudie, Game Experience Questionnaire, Think-Aloud, Algoritmisk komplexitet, Halstead's Komplexitetsmått, Graph Rewriting, Slumpvandring, Design-begränsningar i Spel

Acknowledgements

Above all, we would like to offer our expressed gratitude to our supervisor and mentor for his continued support, wealth of insights and many anecdotes that pulled us through the execution and authoring of our bachelor's thesis. It would have been a bleak prospect to pen this academic paper without his high expectations and critical feedback. You've helped us grow as academics and people both. For the second time, thank you for everything, Aris Alissandrakis.

We will also take this opportunity to show our gratitude to the people who have given their time to participate in our user studies. Without your contributions, this paper would have never been published with any semblance of substance, weight or necessary credibility.

- Noah Andreasson, Mårten Granath, Gabriel Johnsén, David Memedov, Bisher Suhail, Edgar Young, Gothenburg, June 2025

Contents

List of Figures **xiii**

List of Tables **xv**

1 Introduction **1**
1.1 Background . . . 1
1.1.1 Related Work . . . 3
1.2 Purpose . . . 3
1.3 Problem Statement . . . 4
1.4 Scope & Limitations . . . 4
1.4.1 Scope . . . 4
1.4.2 Limitations . . . 5
1.5 The Game . . . 6
1.6 Ethical & Societal Considerations . . . 8
1.6.1 Participatory Measures . . . 8
1.6.2 Academic Measures . . . 9
1.6.3 Societal Assessment . . . 9

2 Method **11**
2.1 Overview . . . 11
2.2 Test Case . . . 11
2.2.1 Within-Subject Case . . . 12
2.3 Data Collection . . . 12
2.3.1 Demographic Questionnaire . . . 13
2.3.2 Game Experience Questionnaire . . . 13
2.3.3 Game Data Statistics . . . 13
2.3.4 Think-Aloud . . . 13
2.3.5 Post Interview . . . 13

3 Implementation **15**
3.1 Overview . . . 15
3.2 Workflow . . . 15
3.3 Core Game System Implementation . . . 16
3.4 General Design . . . 16
3.5 Visual Design . . . 17

3.6 Handmade Level Design . . . 17
3.7 Procedural Content Generation System . . . 19
3.7.1 System Requirements . . . 19
3.7.2 System Overview . . . 19
3.7.3 Iteration One Implementation . . . 21
3.7.4 Iteration Two Implementation . . . 22
3.8 Algorithmic Complexity . . . 24

4 Result **29**
4.1 Overview . . . 29
4.2 Demographics Results . . . 30
4.2.1 Outliers . . . 31
4.3 First Iteration . . . 31
4.3.1 Common Interview Responses . . . 31
4.3.2 Noteworthy Interview Responses . . . 32
4.3.3 Common Think-Aloud Topics . . . 32
4.3.4 Noteworthy Think-Aloud Topics . . . 32
4.3.5 Game Experience Scores . . . 33
4.3.6 In-Game Data Statistics . . . 35
4.4 Second Iteration . . . 35
4.4.1 Common Interview Responses . . . 35
4.4.2 Noteworthy Interview Responses . . . 36
4.4.3 Common Think-Aloud Topics . . . 36
4.4.4 Noteworthy Think-Aloud Topics . . . 36
4.4.5 Game Experience Scores . . . 37
4.4.6 In-Game Data Statistics . . . 39
4.5 Summary . . . 39

5 Discussion **41**
5.1 Constraints on Procedurally Generated Content . . . 41
5.2 Change in Difficulty between Iterations . . . 42
5.3 Discrepancy between Results & Expectations . . . 42
5.4 Discrepancy between Qualitative & Quantitative Results . . . 43
5.5 Relation between Algorithmic Complexity & User Experience . . . 43
5.6 Relation between Algorithmic Complexity & Game Development . . . 44

6 Conclusion **45**

Bibliography **47**

A Questions **I**

B Data Collection **III**
B.1 First Iteration . . . III
B.2 Second Iteration . . . IX

C Use of Generative AI **XV**

List of Figures

1.1 The initial tutorial area (left), the goblin player character (top right), and the dungeon selection area (bottom right). 6

1.2 The first handmade level (top), the first iteration's procedurally generated level (bottom left) and the second iteration's procedurally generated level (bottom right). 7

1.3 A boss room (top left), a puzzle room (top right) and the player getting hurt by a trap (bottom left and right). 8

3.1 An example of the optional goal pattern found in the game 17

3.2 The player's path is blocked by water (left), player encounters ranged enemy and can see that they can shoot over water (right). 18

3.3 A dungeon with four rooms of types: Start, Boss, Reward, and Exit (left). A graph with four vertices, labelled: START, BOSS, REWARD, and EXIT (right). 19

3.4 A simple graph rewriting rule applied to a basic target graph. 20

3.5 A more complex graph rewriting rule applied to the result of Fig. 3.4. 20

4.1 The boxplots of the means from the PCG dungeon for the first iteration for each GEQ component. From Table B.2 in Appendix B, it can be inferred that P5 is the outlier in competence, P1 in Negative Affect and P6 in Flow as well as Tension/Annoyance. The boxplots were drawn using Matplotlib provided by [64] 33

4.2 The boxplots of the means from the handmade dungeon for the first iteration for each GEQ component. From Table B.1 in Appendix B, it can be inferred that P5 is the outlier in Positive Affect. The boxplots were drawn using Matplotlib provided by [64] 34

4.3 The boxplots of the means from the PCG dungeon for the second iteration. From Table B.7 in Appendix B, it can be inferred that P12 is the outlier in Negative Affect. The boxplots were drawn using Matplotlib provided by [64] 37

4.4 The boxplots of the means from the handmade dungeon for the second iteration for each GEQ component. From Table B.6 in Appendix B, it can be inferred that P12 is the outlier in Positive Affect, Negative Affect and one of the outliers in Competence. Specifically, P12 is the bottom outlier in Competence, while P11 is the top outlier. Lastly, P7 is the outlier in Tension/Annoyance. The boxplots were drawn using Matplotlib provided by [64] 38

List of Tables

3.1 The Halstead Volume, Cyclomatic Complexity and Lines of Code for iteration one for the layout and content layer as well as combined. . . . 25
3.2 The Halstead Volume, Cyclomatic Complexity and Lines of Code for iteration two for the layout and content layer as well as combined. . . . 26
3.3 The Halstead Difficulty, Halstead Effort and Projected Hours for the first iteration. 27
3.4 The Halstead Difficulty, Halstead Effort and Projected Hours for the second iteration. 27

4.1 A summary of the data collected from the demographic questionnaire, which is presented in columns two through six. The # symbol is used instead of "number of" and exp. is short for experience. In the seventh column, there is information about the play-tester's gameplay session. The format used in the column is N(D), where N is the iteration number and D is the dungeon they first played. To further emphasise who played which one first, grey colour is used to indicate when the PCG dungeon was played first. The specific questions used in the questionnaire are shown in Table A.2 in Appendix A. 30
4.2 The means and standard deviations of each component from the game experience questionnaire for the first iteration. They are written in the format of M ± SD, where M is the mean and SD is the standard deviation. 33
4.3 The simpler game data collected during the play-test sessions for the first iteration. HM is the handmade dungeon, while PCG is the procedurally generated dungeon. The term (first) is used to indicate which dungeon was first played. Additionally, when PCG was the first dungeon played, grey colour is used. 35
4.4 The means and standard deviations of each component from the game experience questionnaire for the second iteration. They are written in the format of M ± SD, where M is the mean and SD is the standard deviation. 37
4.5 The simpler game data collected during the play-test sessions for the second iteration. HM is the handmade dungeon, while PCG is the procedurally generated dungeon. The term (first) is used to indicate which dungeon was first played. Additionally, when PCG was the first dungeon played, grey colour is used. 39

4.6 The difference between the handmade and PCG dungeon in both iterations. In the first two columns, a positive value indicates that the PCG dungeon scored higher, whereas a negative indicates a higher score for the handmade one. In the last column, the change from iteration one to iteration two is shown. 40

A.1 All questions used in the post interview during the play-test sessions. Depending on the dungeon order, the handmade dungeon or the PCG dungeon might be first. I
A.2 All questions used in the demographic questionnaire along with what type of question they were. II

B.1 The individual play-tester's means for each component in the Game Experience Questionnaire for the handmade dungeon in the first iteration. Grey colour is used to show which play-tester did the PCG dungeon first. III
B.2 The individual play-tester's means for each component in the Game Experience Questionnaire for the PCG dungeon in the first iteration. Grey colour is used to show which play-tester did the PCG dungeon first. III
B.3 The individual answers provided in the game experience questionnaire in the first iteration for the handmade dungeon. Grey colour is used to show which play-testers did the PCG dungeon first. III
B.4 The individual answers provided in the game experience questionnaire in the first iteration for the PCG dungeon. Grey colour is used to show which play-testers did the PCG dungeon first. IV
B.5 The collected data associated with weapon statistics for the first iteration. In the second column, HM is the handmade dungeon and PCG is the procedurally generated dungeon. Next to the abbreviation, (first) is used to denote which dungeon was first played. To further emphasise, grey colour is used to show which play-testers did the PCG dungeon first. V
B.6 The individual play-tester's means for each component in the Game Experience Questionnaire for the handmade dungeon in the second iteration. Grey colour is used to show which play-tester did the PCG dungeon first. IX
B.7 The individual play-tester's means for each component in the Game Experience Questionnaire for the PCG dungeon in the second iteration. Grey colour is used to show which play-tester did the PCG dungeon first. IX
B.8 The individual answers provided in the game experience questionnaire in the second iteration for the handmade dungeon. Grey colour is used to show which play-testers did the PCG dungeon first. IX
B.9 The individual answers provided in the game experience questionnaire in the second iteration for the PCG dungeon. Grey colour is used to show which play-testers did the PCG dungeon first. X

B.10 The collected data associated with weapon statistics for the second iteration. In the second column, HM is the handmade dungeon and PCG is the procedurally generated dungeon. Next to the abbreviation, (first) is used to denote which dungeon was first played. To further emphasise, grey colour is used to show which play-testers did the PCG dungeon first. XI

1 Introduction

1.1 Background

The video game industry is growing larger with each year that passes. In 2022, it reached $182 billion in sales [1], with approximately 50 games released every day [2] on Steam, one of the largest gaming platforms. The market's size indicates that the demand for high-quality video game content has never been higher.

Naturally, the scope of games being produced has also increased [3]. Therefore, methods and tools to aid in production are highly sought after, not only by the gaming industry, but also indirectly by the players that expect high quality standards. One of these methods is Procedural Content Generation (PCG).

PCG is a method that generates content algorithmically [4]–[6] as opposed to creating it manually, referred to as handmade content. In other words, it is the process of taking an input of data and applying an algorithm in order to generate a different output of data. When applied within the context of video games, the output often takes the form of game content such as world generation, dynamic enemy behaviour or randomised item properties.

One of the major benefits of using PCG is the possibility of generating a wealth of variety [7]. This is the case because algorithms can continually be prompted with new input data to generate more content.

In contrast to procedurally generated content, the variety gained from handmade content lies in the purposeful design behind each piece of content. The ability of the human mind to understand the relations and context between different game elements allows for content that is not only non-repetitive, but also feature elements that complement, contrast, or juxtapose each other in ways that are difficult to encode in an algorithm.

The automation of PCG makes it far more efficient at generating content compared to manual methods. For example, it can speed up creation of large game levels as well as provide variation to game systems such as the non-playable characters. Variety is closely tied to novelty of user experience [8], a positive trait of gameplay elements that developers often want to evoke. This is one of the reasons why PCG is so prevalent in the industry today.

Procedural generation has been used in video game development as far back as 1980, with games such as Beneath Apple Manor, where the method was applied to generate game levels [9]. Despite being a rudimentary implementation by modern standards, the resulting variety is demonstrative of what can be achieved even with a simplistic design.

To the present day, technology has taken huge strides to the point that entire worlds and universes can be procedurally generated. This can be seen in more recent examples of market success, such as Minecraft and No Man's Sky, respectively. Minecraft has managed to sell over 300 million copies [10], and No Man's Sky was the top download in August 2016 on the PlayStation store [11]. Minecraft procedurally generates an infinite world that consists of different biomes with accompanying creatures and resources [12]. On the other hand, No Man's Sky used PCG for many of their game systems such as generating planets, fauna, creatures, items, objectives, resources and solar systems [13]. Both introduce more complex implementations of PCG to yield a variety of features on a far larger scale as compared to Beneath Apple Manor. Nevertheless, the fundamental goal of producing variety and a sense of novelty has not changed.

However, there are major risks of using procedural generation, including the creation of recognisable patterns and homogenous content. This is because rule-sets are prone to create patterns, and if the user can recognise them, then the generated content will become predictable. This is generally not desirable, since it might affect the user experience negatively due to the sense of novelty being lost.

There are hybrid approaches that utilise both procedural and manual methods to create content. For instance, a system where a game level is generated with variety tuned by a developer. These hybrid approaches can be used to retain the benefit of variety provided by procedural methods and the thoughtful design that is provided by manual methods.

It should be noted that implementing systems for procedural generation can be far from trivial, sometimes requiring years of refinement and skilled labour to properly develop. Depending on the specific algorithm used, it can end up taking more time to create a system that is able to procedurally generate specific features as compared to making it manually [5]. Therefore, it is imperative that developers weigh the time it takes to implement a feature procedurally and what they may stand to gain in terms of scalability and quality when compared to a handmade design.

A noteworthy example of the pitfalls of procedural generation is the notoriously bad launch of the previously mentioned No Man's Sky. The developers over-promised in terms of game features, and the game was also released prematurely due to rushing to meet deadlines [14]. Some of the core PCG systems were not sufficiently polished and had a large amount of bugs on release, subsequently causing a lot of frustration within the player base [15]. Moreover, even though the scale of the game was undeniably big, the game felt repetitive and lacking in both variety and content [16], underlining the importance of implementing PCG properly. These issues have since been addressed by the continuous updates and support by the developers [17].

1.1.1 Related Work

Previous research on the subject over the last five years has been plentiful, which is an indication of its contemporary relevance. Various PCG techniques and problem statements have been investigated. One example is a study that investigated the integration of large language models with procedural landscape generation to empower ordinary users to generate landscapes [18].

In other cases, studies examine the effects of procedurally generated content on player experience, as does this thesis. Relevant examples include [19] and the earlier paper it cites [20], where both use quantitative methods to compare handmade content with procedurally generated content. In the first study, they compare levels using A/B testing, while in the second, specific graphic props are compared. Another example is [21], where they examine how procedural generation can be used to balance difficulty, and subsequently analyse the user experience using scores. Finally, there is the study [22], which examines how PCG can be used for creating dungeons using an evolutionary algorithm.

To summarise, the mentioned works investigate handmade content and procedurally generated content in similar domains to this thesis. However, among these specific examples of recent research, none of them have used a mix of quantitative and qualitative approaches to data collection, as described in Section 2.3.

1.2 Purpose

The purpose of this thesis is to investigate procedurally generated content in the context of video game design, using handmade content as a benchmark for comparison. The investigation will be evaluated using user experience and algorithmic complexity as metrics to gauge efficacy of design, offer greater comprehension of procedural generation as a tool in the video game industry, and broaden empirical evidence on how it can be utilised.

The thesis specifically examines how select gameplay elements impact user satisfaction based on qualitative and quantitative data derived from user studies. The impact will be studied with the intention of exploring the differences between procedurally generated and handmade content, in addition to evaluating positive and negative aspects of PCG.

The examined relation between procedurally generated and handmade content will be used to draw conclusions when and where one or the other may be more appropriate. Additionally, the data collected may give further insight into how procedural generation can be used to achieve desired user experiences and the underlying reasons as to why this is the case. Ultimately, the intention is to offer developers a better foundation to make informed decisions in the design phase of creating or updating a game.

1.3 Problem Statement

Is there a relationship between the complexity of procedural content generation algorithms, the content they generate, and user experience? The initial hypothesis was that one way to improve user experience is adding more features to the game, which in turn requires increasing the complexity of the underlying systems.

The reason why this problem is worth investigating is because of the potential gain in making educated decisions upon a game's design process and ultimately its development. The way that games are currently being produced relies on guessing beforehand what methods and design choices to use, depending on the size and scope of the game. Typical considerations are development cost and time, as well as the theme and content of the game [23], [24].

This could potentially save companies money and further increase the quality of new video games. This would be a beneficial outcome for not only the companies but also for the consumers.

To better investigate the problem statement, the following auxiliary questions were identified:

1. How can a robust comparison between procedurally generated content and handmade content be executed?
2. What methods of data collection, and what types of questions to ask users will give a sufficient amount of unbiased data to be able to draw data-backed conclusions?
3. Which metrics exist for measuring the complexity of software systems, and how accurate are they?

1.4 Scope & Limitations

The presented problem is substantial and would require larger sample sizes, and more prototypes to compare, than this thesis can feasibly provide to make an exhaustive analysis. For this reason, research will be conducted under an appropriate scope to ensure completion is possible within the given time frame. Additionally, there are recognised limitations that will constrain the credible and comprehensive nature of the study as a whole.

1.4.1 Scope

A single game will be made following the format of a rogue-lite dungeon crawler. Rogue-lite means that the game will not completely erase a player's progress after the player is defeated. It also means the game relies heavily on procedural generation for re-playability. Dungeon crawler means that the player has to navigate and complete dungeon levels in order to progress through the game [25]. The format choice was made on the basis that it inherently lends itself to the use of procedurally generated gameplay mechanics, as PCG and dungeons have a long and close history [6].

A number of constraints will be made to the formulation of the game to ensure ease of implementation and minimal distractions during play-testing. Firstly, the game's environment will have two dimensions (2D). Secondly, the aesthetic complexity of the game will be confined to pixel art. Thirdly, the content will be limited to solely include the parts of a game necessary for comparison and analysis.

Moreover, only a few selected game mechanics will be subject to the process of procedural generation. These were selected on the basis of suitability to the chosen game format and prevalence of use in the industry. These game mechanics are heavily coupled, which means they will not be tested in isolation. The following list is the selection:

- Level layout.
- Level content (room type, object placement).

The procedural algorithms that will be studied are graph grammar and a special case of graph grammar, namely random walk (see Section 3.7.2). Additionally, the focus of the research will be limited to procedural generation during runtime.

1.4.2 Limitations

The choice of algorithms was constrained by development time and the feasibility of implementing them as runtime systems. Also, since the number of game mechanics that can be tested within the given time is limited, conclusions about PCG as a whole cannot be made.

An iterative approach to procedurally generate levels was used and given the time constraints, the amount of iterations was also limited. No more than two iterations were developed, one of low complexity and one of high complexity relative to each other. To be able to draw meaningful comparisons between iterations, one handmade level was created to serve as a baseline for both iterations.

Another limitation pertaining to available resources is the number of participants in user studies. Challenges such as the inability to recruit large numbers, the need for convenience sampling, and logistical constraints such as session length, limit the possible number of participants. As a result, the findings should be considered with caution until further research is conducted with larger user groups, which would provide more robust results.

On a similar note, attempts were made to recruit a diverse group of participants. However, due to limited outreach, only certain demographics ended up in the user studies. For instance, participants only included males in their twenties.

As for the handmade content, a limitation is the group's inexperience of video game content creation. This could cause the handmade game to be a low bar to pass, which might skew the end results in favour of PCG. The opposite is also true, where the group's inexperience of developing PCG algorithms could skew the end result to favour handmade content.

Since the game will be in 2D as described in Section 1.4.1, conclusions about PCG in 3D cannot be confidently made, although some may be applicable to both.

1.5 The Game

The game revolves around a goblin character that raids castles in search of loot, a classic premise for a dungeon crawler where hordes of knights and mages are continually fought to reach the final depths of castle dungeons. The player starts in an outdoor tutorial level to learn the game's basics. Once the tutorial is completed, they proceed to the castle dungeons.

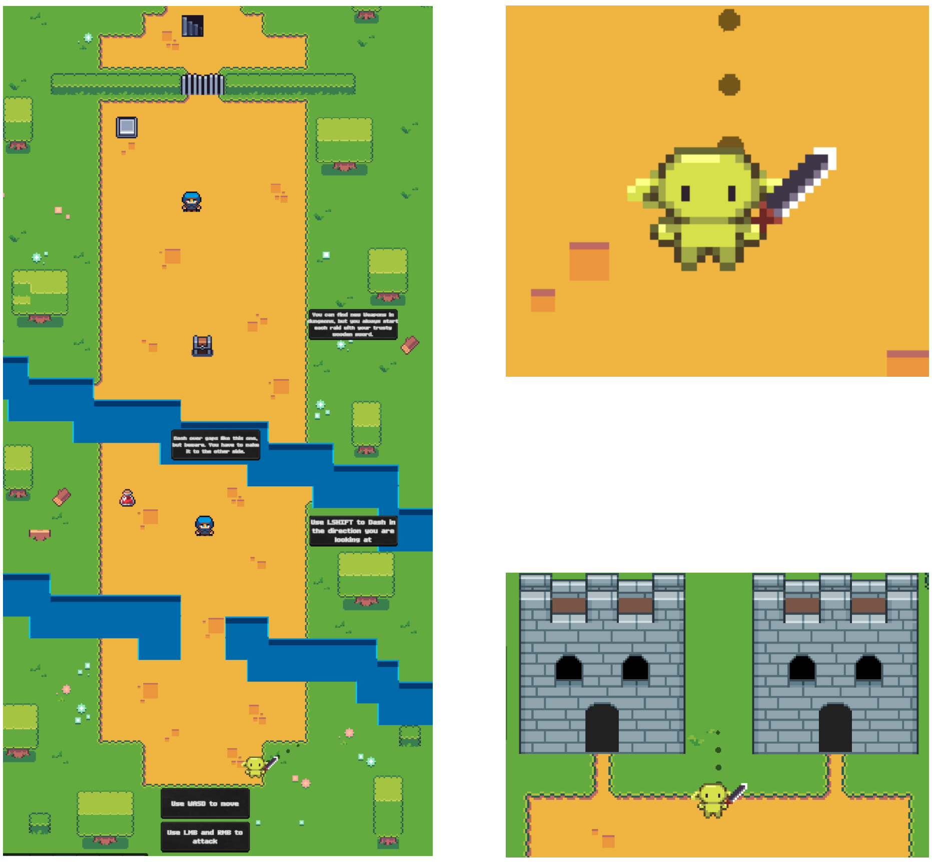

Figure 1.1: The initial tutorial area (left), the goblin player character (top right), and the dungeon selection area (bottom right).

The dungeons are presented as two separate small castles, side by side. One dungeon consists of procedurally generated content, while the other is comprised of handmade content. Each dungeon is divided into three consecutive levels with a successive increase in difficulty. In order to complete the game, the player has to complete both dungeons.

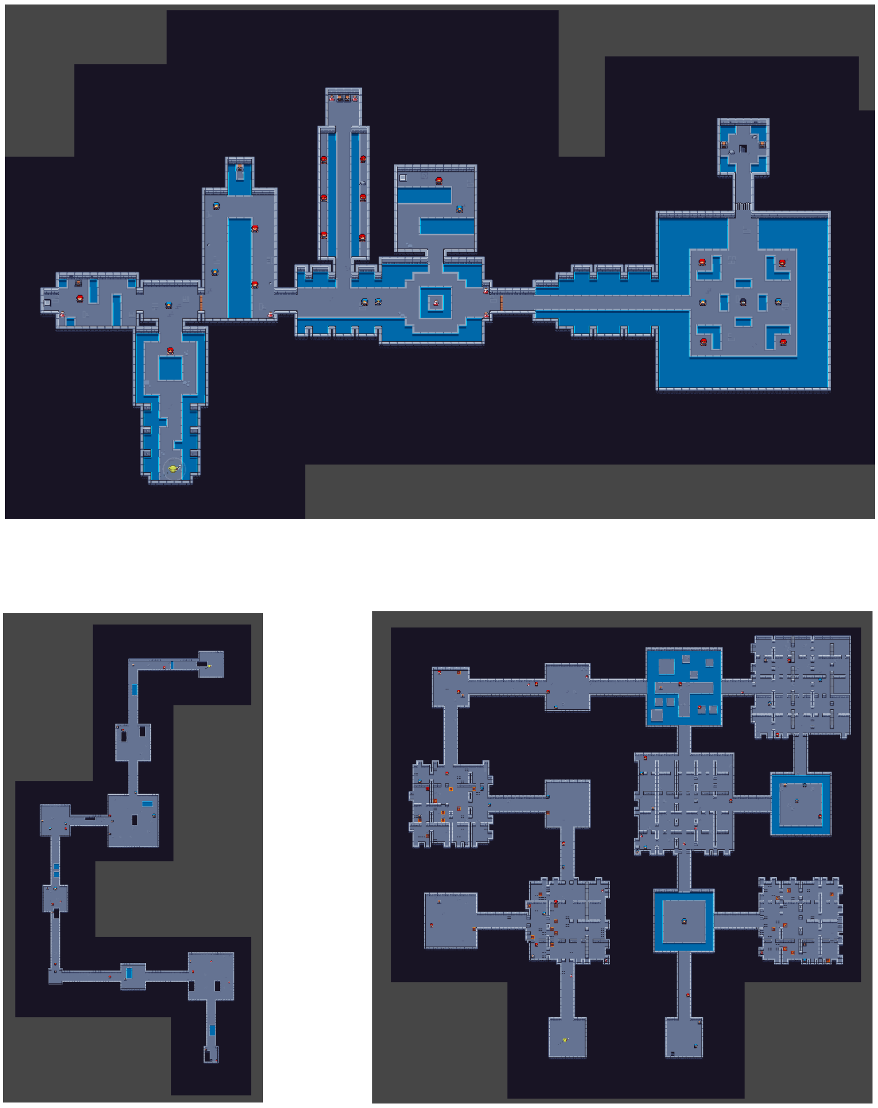

Figure 1.2: The first handmade level (top), the first iteration's procedurally generated level (bottom left) and the second iteration's procedurally generated level (bottom right).

As mentioned in Section 1.4.1, the game follows a simple rogue-lite format and features puzzles, traps and boss fights. In addition to this, the difficulty of the game is balanced by continuously improved weapon drops. These game elements were identified as the conventions of the genre, hence their inclusion.

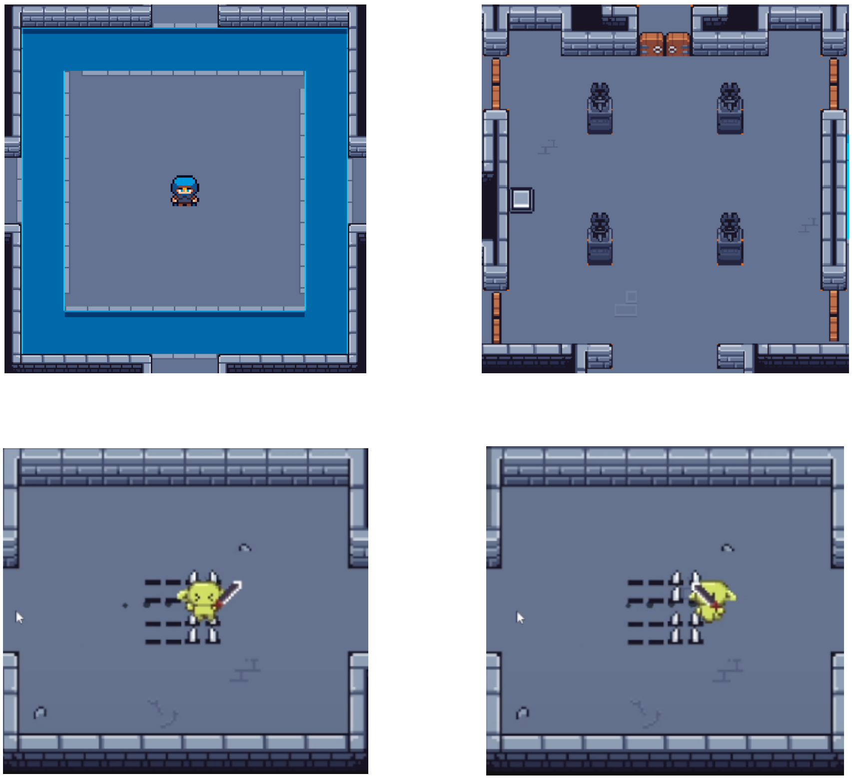

Figure 1.3: A boss room (top left), a puzzle room (top right) and the player getting hurt by a trap (bottom left and right).

1.6 Ethical & Societal Considerations

This thesis followed ethical guidelines to ensure that academic and participatory integrity is upheld, moreover, it considers its research in a societal context in order to pilot future research and use responsibly.

1.6.1 Participatory Measures

The results of this thesis are based on a user study, making it essential to ensure the integrity of the participants. One facet of participant integrity is user privacy, which relates to personal data and personally identifiable information [26], and is defined as the protection of personal information and data collected from individuals [27]. In order to achieve this, all personal data collected prior to and during user study sessions were kept confidential.

User responses were securely stored on Google Drive, a trusted platform that uses encryption and other security standards to store information [28], [29]. Furthermore, access to sensitive documents was only granted to the research team and relevant parties charged with supervisory duties to avoid internal information leaks.

All participants of the user studies were informed that collected data would be anonymised prior to publication. In addition, written consent was required when personal information was gathered with the guarantee that it would be deleted upon project completion. This ensured continued user privacy and freedom of participation.

1.6.2 Academic Measures

Academic integrity has been considered throughout this thesis. All sources and tools used have been credited and disclosed. Disciplinary rigour has been taken into account on multiple levels, as explained throughout this section. Still, it should be noted that research has been done by students with limited domain expertise, and it should be evaluated with this in mind.

To facilitate critical examination, all relevant data used in this thesis can be found in the appendix in its original format, see Appendix B. All questions asked in the demographics questionnaire and during the interviews are also compiled in Appendix A. In addition to this, all code used to create the game and analyse the data has been published in a repository on GitHub [30] for greater transparency.

To reduce bias in the user studies, care was taken to ensure that any questions posed were neutral and did not lead participants to expected answers. During user study sessions, an approach of minimum interference was adopted for the Think-Aloud sessions. This meant that the interviewer provided minimal help and guidance. See Section 2.3.4 for more details.

1.6.3 Societal Assessment

From a societal point of view, procedural generation is often used to create large amounts of content. This implies an automation of the responsibilities and duties that a developer traditionally may have had. This may negatively affect employment rates throughout the industry.

By the same coin, automation may drastically lighten workloads and provide economic benefits. This may contribute to improved working conditions for developers and may drive down the prices of games, making them more available to consumers. It may also incite companies to heighten productivity and make more games overall. Something that may counteract the previously mentioned negative effects.

2

Method

2.1 Overview

A user study was conducted to investigate player experience with procedurally generated content. The focus was on a single game element, generated using algorithms of varying complexity and composition. In order to accommodate multiple algorithms, an iterative approach was chosen. Specifically, each iteration of the user study served to test one algorithm applied to a single game element.

Six people were selected to participate in each iteration of the user study, as per the recommendations of Steve Bromley [31]. Nonetheless, It is clear that more subjects would have been preferable to strengthen the credibility and statistical significance of any quantitative data collection [32]. However, this was simply not feasible given project time and outreach. Thus, conclusions drawn from the latter warrant further research to prove reliable.

The target user for the study was intentionally broad and open to any age group, gender identity and level of experience. The main reason for this was to accurately represent the diverse games market [33] and to increase the pool of participants. The limited outreach of the project group should be kept in mind when evaluating demographic coverage [34]. Furthermore, the people who play-tested the game were intentionally interviewed by someone they did not know prior, rather than by the person who recruited them, to avoid potential bias.

2.2 Test Case

The test case was divided into two parts that each participant had to finish. The first part was an entirely handmade level that acted as a control for the second part, which was similarly comprised of handmade content with the exception of the level layout that was procedurally generated. This element was selected due to its prevalence in games, in particular because level layout is often subject to be procedurally generated [35], [36].

The levels are designed with simplicity and swiftness in mind. Simplicity is important to minimise the effect of learned behaviour, moving from one dungeon to the next. A simple game generally has a low skill threshold, which lessens the impact that player skill has on user experience. Swiftness aims to expedite the user study sessions to be respectful to the participant's time and to maintain their attention during the whole session.

2.2.1 Within-Subject Case

All participants were tried against both handmade and procedurally generated levels, which constitute a within-subject design [37] for the test case as a whole. Besides allowing for a better comparison, it made more efficient use of the small pool of participants [37], which was suitable considering project group outreach.

One of the major problems with within-subject design is introducing bias through learned behaviour [37]. This occurs in the game as the subject plays one level after the other, both sharing the same or similar gameplay. To mitigate the effect of learned behaviour, a tutorial was introduced to ensure the skill threshold of all participants before they progressed to the handmade and procedurally generated level. Moreover, the order in which the participants were exposed to the levels was divided evenly to further counterbalance this effect. For each iteration, three people played the handmade dungeon first, while three others played the procedurally generated dungeon first.

2.3 Data Collection

For each participant studied, a protocol was followed to guarantee the rigour of empirical procedure. The procedure was broken down into five parts that correspond to different sources of data collection, as shown below:

- **Quantitative Data Collection:**
  - Demographic Questionnaire
  - Game Experience Questionnaire
  - Game Data Statistics
- **Qualitative Data Collection:**
  - Think-Aloud
  - Post Interview

The breakdown of data collection was deliberately balanced between quantitative and qualitative methods, in an attempt to provide grounds for more holistic and analytical conclusions. Additionally, using a mixed-methods approach can counteract the disadvantages each of them have [38].

The combination of methods was also chosen to cross-validate different results and to facilitate unexpected discoveries. As penned by Garnham [39], the quantitative and qualitative data can describe different but complementary perspectives.

2.3.1 Demographic Questionnaire

Before conducting the user study, a demographic questionnaire was sent to every interested play-tester. In it, there were questions regarding their gaming experience and background (see Table A.2 in Appendix A). The primary reasons for making them fill out this questionnaire was to assess their gaming experience. This data served to inform the arguments discussed in Section 5.

2.3.2 Game Experience Questionnaire

During the play-test session, the play-testers were asked to fill out a Game Experience Questionnaire after each completed dungeon. The Game Experience Questionnaire is a set of 33 items, where for each item, the play-testers mark, on a scale from 0 to 4, how much they agree with the statements about their experience [40]. The scores can afterwards be translated into overall scores for certain components. Each item in the questionnaire belongs to one of the following seven components: "Competence", "Sensory and Imaginative Immersion", "Flow", "Tension/Annoyance", "Challenge", "Negative affect", and "Positive affect". The results from the questionnaires are described in Sections 4.3.5 and 4.4.5.

2.3.3 Game Data Statistics

While the play-tester played through the dungeons, in-game data was automatically collected. For instance, how long each play-tester spent in a dungeon was gathered. The data collected is shown in Tables 4.3, B.5, 4.5, and B.10 later in Section 4. This provided a way of obtaining more data without manually taking notes during the sessions.

2.3.4 Think-Aloud

During the play-testing sessions, the play-testers were asked to Think-Aloud. Think-Aloud is a tool for testing usability [41], where the participants are encouraged to say what they are thinking while performing representative tasks. This method is useful to get the immediate thoughts from the play-tester.

2.3.5 Post Interview

After each play-tester had finished playing through the entire game, six questions were asked in the format of a semi-structured interview. A semi-structured interview is a qualitative research method with a pre-determined set of open questions, allowing the interviewer to further delve into specific topics or answers in greater depth [42]. First, the play-tester was asked whether they had a hunch of what the difference between the two dungeons were. After that, the real purpose of the study was revealed to them, and at that point they were asked additional questions.

They were questioned on whether knowing how the dungeons were made affected the game experience or their previous thoughts. Lastly, they were asked about which dungeon was their favourite. For the exact question protocol used, see Table A.1 in Appendix A.

3

Implementation

3.1 Overview

Although the thesis is primarily concerned with comparing the game experience between procedurally generated content and handmade content, a lot of thought and time went to creating the game that would serve as the platform for the user studies. This section will describe both the design process and relevant game mechanics, to better provide a background for what the users were presented with in the user studies, as described in Section 2.

3.2 Workflow

For the creation of the game, there was a specific workflow and tools used. The workflow can be described as agile, since many agile processes were used. As written by Christopher Fox [43], incremental design is a key practice used in extreme programming and other agile methodologies. In this project, an incremental approach was used by utilising feature branches. The project had a main branch, a development branch and many feature branches.

Each feature branch was responsible for one feature or bug fix. Sometimes these were larger, while other times they were a simple one line change. All feature branches had to be approved by at least one other member before merging it to development. Due to this reviewing process, a lot fewer bugs ended up in development than what otherwise would have. The development branch had many pull requests merged into it. The main branch on the other hand was only merged into when an iteration was considered done and ready for its play-test sessions. Before merging into main, even more meticulous testing was done, as catching the bugs before the play-test sessions was a high priority. If any issues or bugs happened during a play-test session, it could influence the result of the study. Note, despite this carefulness, one play-test session encountered a game-breaking bug never seen in other sessions. For more details, see Section 4.2.1.

In combination with the feature branching process, a Kanban board was utilised. According to Max Rehkopf [44], a Kanban board represents an agile project management tool to aid in task visualisation, limiting features to be done, and optimising efficiency or flow. It was used to concretise what should be done next in the project, which was achieved by having issues about various features and bugs to be fixed. Each issue was assigned a priority from P0 to P2 and also a time estimate in the Fibonacci scale. For instance, an issue about fixing a simple bug was typically assigned P0 and estimate 1, while creating a whole new system could be estimated as a 3 or 5 with varying priority.

Nearing the end of the development of the handmade parts of the game, pilot play-testing sessions were made. These were conducted for mainly two reasons. Firstly, to catch any bugs that could happen during an actual play-test session. Secondly, getting some feedback of the game in order to make it more enjoyable. The second reason might sound unnecessary at first, but using a fun and meaningful game is vital for the study. If the core parts of the game are not perceived as fun, then play-testing sessions will feel more artificial. In addition, the feedback will likely be more about how the game should be improved rather than a play-tester's experience.

The game was made using Unity and Git. Unity is a game engine [45], and Git is a version control system [46].

3.3 Core Game System Implementation

The game features a rudimentary implementation of a combat system consisting of weapons along with a dash and an accompanying stamina system for movement. Furthermore, the game also implements an interaction system to allow the opening of doors and chests in addition to the equipping of weapons and using health potions.

The reason why the core game systems are of a simple nature is not to distract players from playing and experiencing the game level, which is the purpose of the game. The concrete implementation of the core game system can be found in the project's GitHub repository [30], where playable builds are also available. The repository referenced here only contains the game logic, missing any graphical assets and engine assets required to compile the game. This is because a majority of the graphical assets were used under a licence, and are not permitted to be shared.

3.4 General Design

As mentioned in Section 1.4.1, the study did not use an existing game. Instead, one was made specifically for this study, and this had its advantages.

Firstly, the created game was customised to make the play-testing sessions easier to conduct. The most obvious example of this is in the second area of the game, as seen in Fig. 1.1 (bottom right). The player is prompted with a choice of which dungeon to play through, but not knowing which is which, as both look identical. This choice was very deliberate since it allowed the procedural generation and the handmade dungeons to be tested in isolation, but still hiding the study's objective for the play-testers.

Secondly, specialised in-game statistics were able to be automatically collected, as is shown in Tables 4.3 and 4.5.

Thirdly, creating a specialised game allowed it to be limited in ways described in Section 1.4. In turn, this enabled the focus of the game to have relevant content for the user studies.

3.5 Visual Design

When choosing which tiles and sprites to use for the game, assets that were simple and easily readable were prioritised. That way, the player would not be distracted from the study by looking at the assets or trying to understand what an entity is. An asset pack was purchased online that fit the criteria, and then some enemy sprites were recoloured, giving each enemy type its own colour. This made what type of enemy the player is facing obvious. Lastly, the ground textures were simple, making the entities easily distinguishable.

3.6 Handmade Level Design

When constructing levels for the handmade part of the game, multiple different gameplay patterns were followed, to try and make the levels as good as possible. An apparent example of this is the use of the Boss Monsters pattern. A boss monster is, according to Staffan Björk [47], a more powerful enemy than the player usually faces and can be used as a good end point for levels. All the three handmade levels end with a more challenging encounter featuring a boss monster. For instance, the second level can end with an encounter against a tank enemy, i.e. a stronger enemy the player has not faced before.

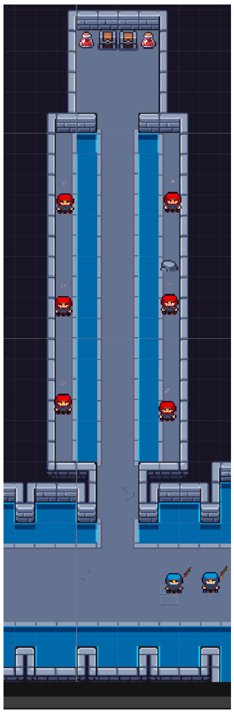

Figure 3.1: An example of the optional goal pattern found in the game

Another pattern used throughout all the levels is the optional goal pattern. The optional goal pattern is a goal or challenge, where the player does not have to do to complete the level, according to Staffan Björk [48]. The optional goal could be resources to help the player, a side quest, or just a fun bonus challenge [48]. The most prominent example of an optional goal in the handmade levels is a room in the first level (see Fig. 3.1). The player can choose to enter the room where they will face a group of ranged enemies shooting at them and some rewards afterwards, but the player can beat the level without entering the room at all.

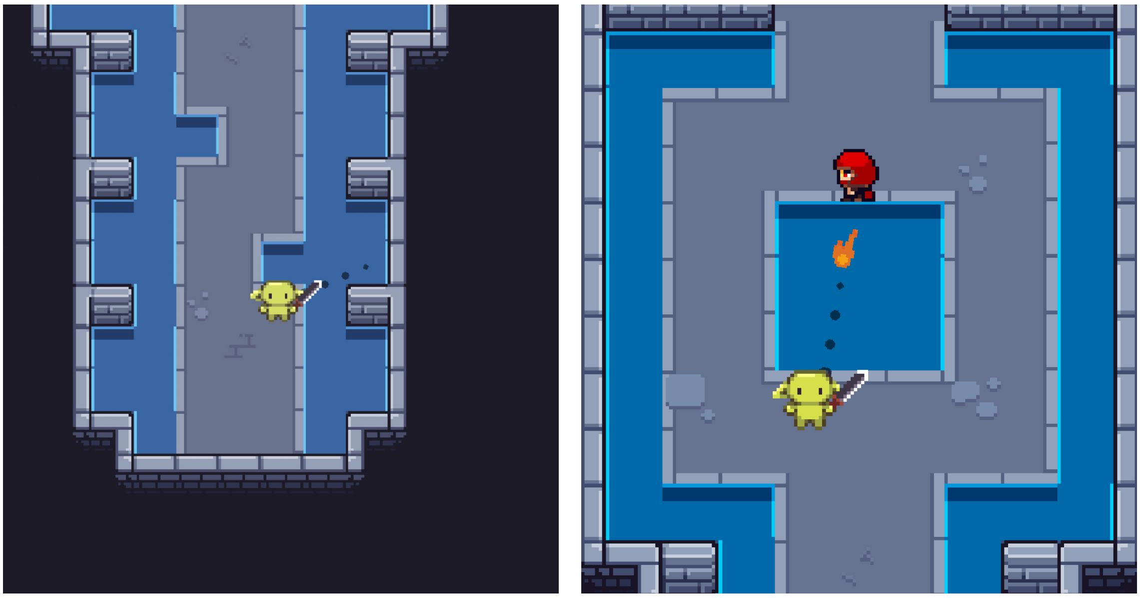

Figure 3.2: The player's path is blocked by water (left), player encounters ranged enemy and can see that they can shoot over water (right).

When building the levels, new mechanics and enemies were gradually introduced to let the player become comfortable with them before entering challenging scenarios. Even though, this is not a game design pattern, it is a very common practice used in many popular video games. Ernest Adams and Joris Dormans [49, p. 31] writes that slowly introducing mechanics in simple chunks is a great way to teach the mechanic to the player without overwhelming them. This design technique was used to introduce many of the mechanics, with the introduction of water tiles in level one serving as a noteworthy example.

The water tiles are a type of tile that the player can dash, but not walk over. Additionally, enemies can shoot projectiles over them. In the beginning of the level, the player's path is blocked by the water, letting them understand that they cannot walk over it, but they can dash over it. Subsequently, they face one ranged enemy, making them understand that enemies can shoot over water (see Fig. 3.2 (right)). Following this, they get their real challenge using the water mechanic. This way, the player gets to know all the aspects of the water tiles in their own time, before they get placed in a challenging combat scenario utilising water tiles.
Similarly, abyss tiles serve the exact same purpose as water tiles but at a different appearance for some graphical variety.

These are just the most conspicuous examples of patterns and common design decisions that were used during the creation of the handmade levels, but even more were followed. In addition, the levels were play-tested and iterated on multiple times to ensure optimal quality within the project's deadline.

3.7 Procedural Content Generation System

3.7.1 System Requirements

To ensure that the implementation of the PCG system would have the necessary properties and features required to ensure a successful user study, a couple of requirements were specified.

To be able to fairly compare the handmade and PCG levels, the levels generated by the system must:

- contain a subset of the features of the handmade levels.
- be of similar length to the handmade levels.
- provide a similar challenge as the handmade levels.

Since the aim is to develop multiple iterations with increasing complexity, the system must be able to be extended to generate levels of varying complexity. The expected development window for each iteration was about a week, so the system must be able to be extended rapidly.

3.7.2 System Overview

The procedural content generation system is structured into two primary layers, the layout layer and the content layer.

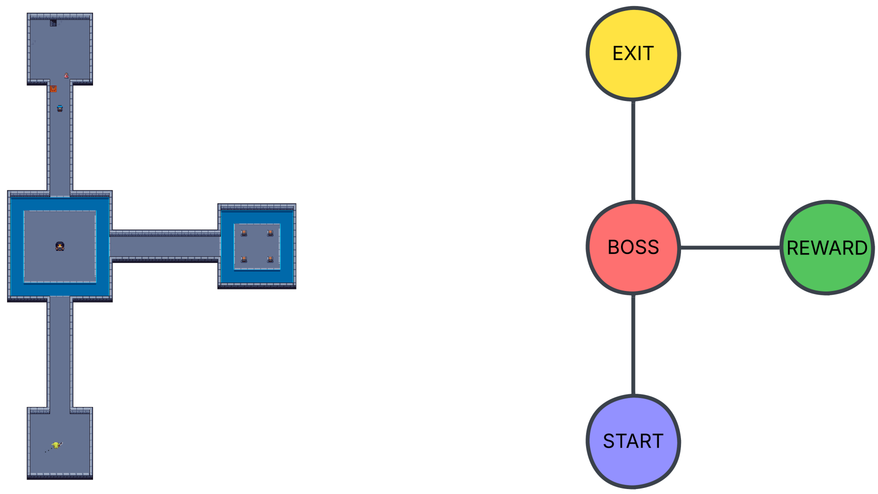

Figure 3.3: A dungeon with four rooms of types: Start, Boss, Reward, and Exit (left). A graph with four vertices, labelled: START, BOSS, REWARD, and EXIT (right).

The responsibility of the layout layer is to determine the locations of and relationships between rooms. It decides the overarching structure and composition of the level, dictating which rooms are placed, where they are placed, and how they are connected. The structure of a dungeon can be represented as a graph, where each room is a vertex, and each corridor between rooms is an edge between vertices, as seen in Fig. 3.3. The basis of the layout layer is graph rewriting, which performs all of these functions in a flexible and extendable way.

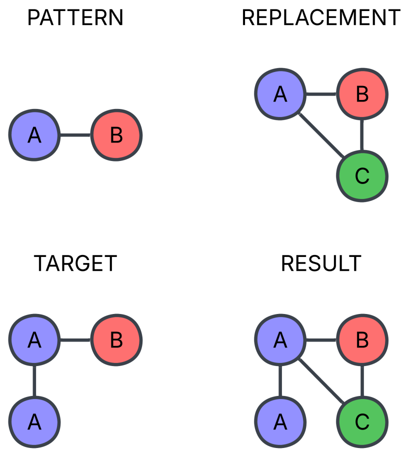

Figure 3.4: A simple graph rewriting rule applied to a basic target graph.

It does so through the use of a grammar, which is a set of rewrite rules that when applied to a graph will transform it [50]. A rule is of the form $L \rightarrow R$, where L is the pattern graph, and R is the replacement graph. When a rule is applied to a target graph, it tries to match the pattern graph to a label and edge preserving isomorphic sub-graph of the target graph, and if it finds a matching sub-graph, replace it with the replacement graph. In simple terms, a label and edge preserving isomorphism of the pattern graph is a sub-graph in which the structure, the vertices and how they are connected, and the labels, the types or properties of vertices, exactly match those of the pattern graph.

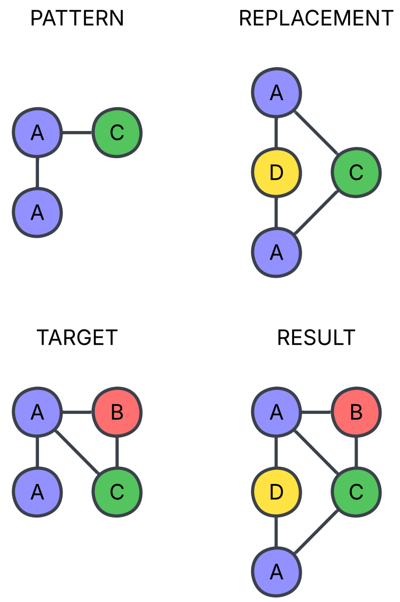

Figure 3.5: A more complex graph rewriting rule applied to the result of Fig. 3.4.

As illustrated in Figs. 3.4 and 3.5 graph rewriting can, when applied iteratively, create complex graphs with a small set of rules. The size, label variety, and edge to vertex ratio of the transformed graph depends on which rules are employed and how many replacements are made.

From this follows that a layout layer incorporating graph rewriting can generate dungeons of varying complexity. Additionally, by randomly selecting rules from the grammar, the system accomplishes its purpose of providing a new level each time.

The content layer, on the other hand, is responsible for the concrete realisation of each room and their connections. Based on the room type and its position, this layer draws the room and connects it to adjacent rooms via corridors that are generated according to their relative world positions. In addition, this layer also handles object placement and depending on the room type, various game objects are placed. These objects include: decorations, enemies, traps, obstacles, health potions, and loot chests.

Both layers were iteratively developed. In the first iteration, a minimum viable product (MVP) was developed for each of the two layers to establish a working baseline whilst still ensuring some base level of quality. In the second iteration, the layout layer was significantly extended by developing an entire new module for the graph grammar, while the content layer was extended with a greater amount of room types.

3.7.3 Iteration One Implementation

The aim of iteration one was to serve as a working baseline, as mentioned in the previous paragraph. To accomplish this a very simple idea was followed, namely: a set of interconnected rooms filled with enemies, loot chests and health potions. The last room should also contain an exit which transports the player to the next procedurally generated dungeon. This set of features were deemed essential to mimicking the experience of the handmade level. With this in mind, the layout and content layers were developed.

The first algorithm used for the layout layer was a simple implementation of a random walk algorithm [51]. A random walk in our context functions as follows: for each step after the first, choose one random direction, excluding the direction from which you just came, and place a room there. This process is repeated until a set number of rooms have been placed. Random walk can be considered a special case of graph rewriting, where only one rule exists: match on the exit, and replace it with a room vertex connected to a new exit vertex. Since our implementation of graph rewriting tries to match all rotations and mirrors of the pattern graph in a random order, this will lead to the new exit vertex being placed in a random direction.

Despite this, it was decided during implementation that a stand-alone random walk algorithm would be developed. This was due to delays in development of the graph rewriting system, which was instead developed in parallel and integrated in the second iteration. This saved valuable time and upon completion of the graph rewriting system the two were verified to be functionally equivalent in performing the random walk generation.

This approach results in a linear room layout, where each room has exactly one entry and one exit, except for the first and last rooms. This forms a path graph, a graph in which all vertices, in this case rooms, are connected by edges, in this case corridors, in a chain, like beads on a string [52]. Through user testing, it was determined that a level consisting of ten rooms would have a length similar to the average between the three levels of the handmade dungeon.

The content layer is responsible for both local room realisation and tile painting, as well as stochastic object placement. Based on the information relayed by the layout layer, a room is generated at a given world position using a predefined list of room sizes. The room is then rendered with floor tiles, walls, water, and abyss tiles. In this process, random pits of water and abyss tiles are placed throughout the room, while ensuring that the level can be completed.

Corridors are generated between rooms based on the relative positions of the adjacent rooms, ensuring connectivity that follows the path structure defined by the layout layer.

Finally, the room positions that contain floor tiles are passed to the stochastic object placement system, which places various intractable or decorative objects. Object placement depends on several factors:

- A spawn chance (probability of appearance).
- A maximum spawn count per object type.
- The current level iteration (the number of times the dungeon has been generated, which is used for difficulty scaling).
- Whether the specific room floor position is already occupied by another object.

Together, iteration one of the procedural generation system produces a path graph of ten rooms filled with randomly placed objects, including enemies, and environmental hazards such as water and abyss pits. An example is seen in Fig. 1.2 (bottom left).

3.7.4 Iteration Two Implementation

For the second iteration, a more sophisticated approach was made. Not only was the ambition to extend the amount of room variations, but also to enable rooms to be positioned within a context, meaning that related rooms could be placed adjacent to each other. A relevant example that is used in the game is a loot room after a boss room, to reward the player for defeating a challenging enemy. This in turn enables the PCG system to mimic thoughtful design and give rooms an increased sense of purpose, similar to how the handmade level was designed. With this in mind, the layout and content layers were developed.

This time graph rewriting was fully implemented for the layout layer. To constrain the generation enough to be comparable with the gameplay experience of the handmade level, the rewriting was performed in three steps. The starting graph contained two vertices, an Entrance and an Exit, with an edge between them. The grammar of the first rewriting step contained a single rule, described above in the random walk example. The stop condition, after which no more rules would be applied and the next step would start, was when the graph contained 10 vertices. In essence, at the end of this step, the structure of the level resembled the first iteration.

The purpose of the second rewriting step was to add branching paths to the graph. The grammar contained two rules, one which would create a branch, and one which would create a second, longer path forward. The stop condition of this step was for the graph to have 14 vertices. This step aimed to enable the player to explore the level in a non-linear fashion.

The third rewriting step contained rules that replaced basic room vertices with other vertex types, described in the section about the content layer below. This introduced the first use of rules for contextual placement of vertices, with one rule which would replace the last room before the exit with a Boss room, and one rule which would create a branch from a Boss room that consisted of a Loot room. This is a significant difference from the first iteration, as the intentional placement of these rooms gives them an increased purpose. The stop condition for this step was that 14 rules would be applied.

The stop conditions were tuned to try to match the length of the dungeon to the handmade one. The length of the level was verified by expert testing by the development team, and only a small amount of user testing due to time constraints.

The content layer for iteration two, similarly to iteration one, consisted of room generation and tile painting, as well as stochastic object placement. However, this time, the solution was significantly extended to support a greater variety of room types.

Just like in iteration one, the layout layer relayed information to the content layer specifying a world position, but this time, since there are now more types of room than just one, a room type is also specified. The different room types are:

- A water room.
- An island room.
- A corridor room (i.e., an extended corridor).
- A narrow abyss room.
- A ruin room.
- A trap room.
- A maze room.
- A boss room.
- A loot room.

Naturally, each room has different characteristics, and to stay consistent with that, the object placement system was handled differently for some of the rooms. For instance, the loot room always spawns four loot chests to reward the player for reaching it. Additionally, the boss room always spawns a boss and that boss is of a difficulty consistent with the player's progress. Lastly, the trap room spawns a greater number of traps. To accommodate the increasing difficulty of the different rooms, as well as acknowledge the observations made during iteration one play-test sessions (see Section 4.3.4), the health potions' spawn chance was increased in all rooms.

A big challenge when designing the algorithms, which draw out the rooms, was ensuring that the level could be completed, especially for the maze room. The solution to this problem was to ensure that a path filled with floor positions always exists from the corridor all the way to the middle of the room for each different corridor leading from and to the room. In this way, a traversable path was essentially carved out into the room, which ensured that the room could be completed for all different room types. This was then masked in some of the rooms by making them large enough to not see the next corridor from the entry.

To summarise, this system is capable of generating rooms which are not linearly connected and are able to be contextually generated, causing more coherency. Not only that, but it also offers a greater variety, effectively ironing out the repetitiveness of iteration one. A total of 14 rooms could now be generated of different types, filled with randomly and some non-randomly placed objects as well as environmental hazards. One example of its generation is shown in Fig. 1.2 (bottom right).

3.8 Algorithmic Complexity

Complexity can be used as a metric to better understand the differences between separate algorithms. There are various types of complexity and many ways to compute them. The focus of this thesis revolves around software complexity, which characterise how difficult the code in a module is to understand, maintain and modify [53]. A module is defined as a piece of code that provides a specific function [54].

Three metrics of software complexity will be considered to present a holistic view of the code in relevant modules. These are as follows:

- Halstead Volume
- Cyclomatic Complexity
- Lines of Code

Halstead volume is a metric of the constituent parts that comprise a module, largely based on the operators and operands used in the module [55]. It offers an estimate of complexity related to the size and properties of a program module and can be calculated as follows [56]:

$$HV = (N_1 + N_2) * log_2((n_1 + n_2)) \tag{3.1}$$

Where:

- n_1 (number of distinct operators)
- n_2 (number of distinct operands)
- N_1 (total number of operators)
- N_2 (total number of operands)

Cyclomatic Complexity is a metric of linearly independent decision paths through a program, it offers an estimate for how stable a program is and how difficult a program is to understand [57]. Stability refers to runtime consistency [58], while independent decision paths equate to the number of logical branches present in a piece of code [57]. To evaluate cyclomatic complexity, the following formula is used [59]:

$$CC = 1 + DP \tag{3.2}$$

Where:

- CC (Cyclomatic Complexity)
- DP (Decision Paths)

Lines of code is a simple metric that measures the size of a program module [55]. Note that it does not include comments or blank lines of code. While this metric is closely associated with Halstead Volume [55], it is worth noting that it can offer further nuance in how concentrated the operators and operands of the program are.

Table 3.1: The Halstead Volume, Cyclomatic Complexity and Lines of Code for iteration one for the layout and content layer as well as combined.

Layers	**Halstead Volume (HV)**	**Cyclomatic Complexity (CC)**	**Lines of Code**
Layout	2 244	7	68
Content	17 224	73	412
Combined	20 350	79	480

Table 3.2: The Halstead Volume, Cyclomatic Complexity and Lines of Code for iteration two for the layout and content layer as well as combined.

Layers	**Halstead Volume (HV)**	**Cyclomatic Complexity (CC)**	**Lines of Code**
Layout	23 431	102	641
Content	41 879	145	935
Combined	70 741	246	1576

Moreover, this thesis considers two other metrics of complexity to quantify how difficult modules are to develop, leading to estimates of both development time and effort. These metrics include:

- Halstead Difficulty
- Halstead Effort

Halstead Difficulty is a measure of how hard it is to write and understand a piece of code [56], [60], [61], that is largely based on the same variables as Halstead Volume. It is calculated as follows [56], [60], [61]:

$$D = \frac{n_1}{2} * \frac{N_2}{n_2} \tag{3.3}$$

Where:

- D (Halstead difficulty)
- n_1 (number of distinct operators)
- n_2 (number of distinct operands)
- N_2 (number of total operands)

Halstead Effort is a measure of the mental effort required to develop and understand a piece of software [60], [62], it takes into consideration both Halstead Difficulty and Halstead Volume. It is calculated as follows [56], [61], [62]:

$$E = D * HV \tag{3.4}$$

Where:

- E (Halstead effort)
- D (Halstead difficulty)
- HV (Halstead volume)

Halstead effort can in turn be used to approximate the amount of hours required to program a piece of code. A formula for calculating the value in seconds is presented by Hariprasad *et al.* [56] and Coimbra *et al.* [61]. The original formula has 18 in the denominator, but it can be modified to give the number of hours using the formula below:

$$T_{hours} = \frac{E}{S * 60 * 60} \tag{3.5}$$

Where:

- T_{hours} (time required to write a program in hours)
- E (Halstead effort)
- S (Stroud number, i.e., mental discriminations per second, in this case 18 [63])

Table 3.3: The Halstead Difficulty, Halstead Effort and Projected Hours for the first iteration.

Layers	**Halstead Difficulty** (D)	**Halstead Effort** (E)	**Projected Hours** (T_{hours})
Layout	45.5	102 178.8	1.6
Content	129.4	2 228 825.8	34.4
Combined	135.9	2 765 450.3	42.7

Table 3.4: The Halstead Difficulty, Halstead Effort and Projected Hours for the second iteration.

Layers	**Halstead Difficulty** (D)	**Halstead Effort** (E)	**Projected Hours** (T_{hours})
Layout	139.9	3 277 572.4	50.6
Content	208.8	8 745 682.9	135.0
Combined	236.5	16 728 484.9	258.2

The code for the calculations can be found in the project's GitHub repository [30].

4

Result

4.1 Overview

This section describes the results gathered from the conducted user study. There are individual presentations of the data from each iteration to provide a basis for discussion in Section 5. In the following text, there are subsections that present results gathered from the different data collection methods. First, there will be descriptions of common as well as noteworthy themes and topics found using the two qualitative techniques, namely the interview and the think-aloud method. Then, the results of the quantitative methods will be presented using boxplots and tables. However, prior to describing the results from the data collection methods, the results from the demographic questionnaire will be presented in Table 4.1.

There are three kinds of tables shown in each iteration. Firstly, in Tables 4.2 and 4.4, there are means and standard deviations shown for each component of the game experience questionnaire. For every component, there is the handmade dungeon's mean and standard deviation next to the procedurally generated dungeon's. For information about the individual play-testers' scores per item, they are shown in Tables B.3, B.4, B.8, and B.9 in Appendix B.

Secondly, Tables 4.3 and 4.5 show the game data collected during the first iteration and second iteration, respectively. To keep the data anonymous, their names are displayed as PX. For example, P1 from iteration one refers to the same person. In other words, it is possible to connect the statistics of a certain play-tester with their qualitative responses. In the second column, there is an abbreviation indicating which dungeon the data was collected from. The numbers in the time spent column are in seconds and rounded to the nearest integer. All other columns are counts, conveying simple details about the game played. Some describe the player's play style, such as death count, while other counts are more focused around how many resources were provided for the player. For instance, the number of chests were always 71 in the handmade dungeon, while it could vary for the procedurally generated ones.

Thirdly, in addition to the game data described above, more specific data was collected for each weapon that the players used in the two dungeons. This data is found in Tables B.5 and B.10 and are similar to the tables described before, with their names replaced with PX. For each weapon the tables specifically show the time when it was first equipped in seconds rounded to the nearest integer and also how many times that weapon was equipped in which dungeon according to the second column.

Alongside the Tables 4.2 and 4.4, boxplots are also provided. The Figs. 4.2, 4.1, 4.4, and 4.3 illustrate the boxplots of the component scores for a dungeon in an iteration.

4.2 Demographics Results

As mentioned in Section 2.3.1 there was a demographic questionnaire (see Table A.2 in Appendix B) sent to the play-testers before conducting the sessions. In Table 4.1 the results are summarised and grouped based on iteration as well as what dungeon was first played.

Table 4.1: A summary of the data collected from the demographic questionnaire, which is presented in columns two through six. The # symbol is used instead of "number of" and exp. is short for experience. In the seventh column, there is information about the play-tester's gameplay session. The format used in the column is N(D), where N is the iteration number and D is the dungeon they first played. To further emphasise who played which one first, grey colour is used to indicate when the PCG dungeon was played first. The specific questions used in the questionnaire are shown in Table A.2 in Appendix A.

Play-tester	**Video game exp.**	**Identifies as gamer?**	**# years playing games**	**# hours spent playing per week**	**Was "Dungeon crawlers" a favourite genre?**	**Gameplay Info**
P1	4	Yes	15	8	No	1(PCG)
P2	4	No	22	40	Yes	1(PCG)
P3	4	Yes	15	20	No	1(HM)
P4	4	Yes	23	30	No	1(PCG)
P5	3	Yes	14	32	No	1(HM)
P6	4	Yes	20	20	No	1(HM)
P7	4	Maybe	13	2	Yes	2(PCG)
P8	3	Yes	22	5	Yes	2(HM)
P9	4	Maybe	20	15	No	2(HM)
P10	4	Yes	20	15	No	2(HM)
P11	3	Yes	16	30	Yes	2(PCG)
P12	4	Yes	20	20	No	2(PCG)

4.2.1 Outliers

Among the participants, there were two clear outliers. To begin, when the play-tester P2 played through the game, a bug that had never appeared before occurred. During the session, the player character kept moving in one direction, independent of what the play-tester did. To counter this bug, they had to move in the opposite direction, as a consequence their dashing and movement ability was impaired. This created a somewhat different experience than the rest, but regardless, the play-tester managed to complete both of the dungeons. According to the play-tester themselves, the scores they gave were not affected.

The other outlier was P12. They said that they were a game developer and described how that shaped their critical perspective based on what a developer would be looking for in a play-test session. An example of their behaviour is them suddenly pausing the game during the session, opening the settings menu from the pause menu, and pressing the ESC key to test if the pressed key would go back to the pause menu. They expected this result, since pressing the ESC key in the pause menu resumes the game.

4.3 First Iteration

In general, for iteration one, it is worth mentioning that play-testers P1, P2, and P4 played the procedurally generated dungeon first, while P3, P5 and P6 played through the handmade dungeon first. The motivation to have some play-testers play one before the other is mentioned in Section 2.2.1.

4.3.1 Common Interview Responses

As mentioned earlier in Section 2.3.5, there were some interview questions considered. When the play-tester was asked whether the experience of the game was affected by being handmade or procedurally generated, the common answer was that it was not. The play-testers were indifferent to whether the content was procedurally generated or handmade. To exemplify, here is an answer said by P5: "I probably wouldn't care, if it is the same quality, if it's handmade or generated with code. Uh, I liked the first one more [referring to the handmade dungeon], so I don't know if knowing that changes much". Similarly, knowing whether it was handmade or procedurally generated had little effect on their earlier questionnaire responses, as most play-testers said it did not matter.

In addition to more direct answers to the interview questions, other notable comments are worth addressing. To begin, in the cases where the procedural dungeon was first played, every play-tester thought it served as an extension to the tutorial, where the handmade game then acted as the actual game. As an example, P1 said: "For a tutorial level, like that. I think it could, it could actually be a cool concept to have the first level, which is supposed to be a little more simple, be procedurally generated."

Most of the participants said that they didn't know what procedurally generated content means after the design choice for each dungeon was revealed to them; only P1 and P2 knew about procedural generation. Lastly, when asked about their favourite dungeon, it was unanimously the handmade dungeon for the first iteration.

4.3.2 Noteworthy Interview Responses

Among those who expressed reservations about PCG, P2 emphasised that procedural generation must feel as purposeful as handmade content, ideally to the point where it is impossible to tell whether it was created by a developer or generated by a system. The design of certain rooms in the handmade dungeon was praised by them. They were uncertain about changing their answers to the questionnaire questions, as their opinions did not reflect the reservations they expressed after the reveal.

P2 also remarked that they would not have enjoyed the PCG dungeon as much as they did if they started with the handmade dungeon first, due to how underwhelming the PCG levels were compared to the handmade's. P4 shared the same sentiment when it comes to the order of dungeons, though they were neutral towards the concept of procedural generation after it was explained to them during the interview questions.

4.3.3 Common Think-Aloud Topics

For many play-testers during the first iteration, their comments focused on the general experience they had. It ranged from how they perceived the weapon system, both aesthetically and mechanically, to enjoying the game's soundtrack and its complementary effect on the overall experience.

Many play-testers were commenting on the game difficulty and their enjoyment of how entertaining the challenge was for them. Few of them even regarded the frustration and annoyance they felt as a positive sign of a captivating gameplay. P6 specifically said: “You should be annoyed and irritable in a good game, I would say”.

4.3.4 Noteworthy Think-Aloud Topics

During almost all of the play-test sessions in the first iteration, there was a clear lack of health potions due to many play-testers dying multiple times during the PCG dungeon. To exemplify, P6 explicitly commented on this issue, as they said: “Maybe you wanted it to be hard. So, it's up to you how hard you want it to be. In my opinion, some more HP pots here and there, because as I said, I don't feel like I play that poorly”.

In the first iteration, both P1 and P3 noticed that killing enemies had no purpose when playing the PCG dungeons. This conclusion was even more prevalent in iteration two (see Section 4.4.3).

4.3.5 Game Experience Scores

Table 4.2: The means and standard deviations of each component from the game experience questionnaire for the first iteration. They are written in the format of M ± SD, where M is the mean and SD is the standard deviation.

GEQ Components	Handmade	PCG
Competence	3.33 ± 0.69	3.47 ± 0.35
Sensory and Imaginative Immersion	2.58 ± 0.68	2.47 ± 0.74
Flow	3.17 ± 0.87	2.90 ± 0.95
Tension/Annoyance	0.56 ± 0.50	0.28 ± 0.39
Challenge	2.10 ± 0.80	1.73 ± 0.58
Negative affect	0.25 ± 0.39	0.13 ± 0.21
Positive affect	3.83 ± 0.23	3.70 ± 0.28

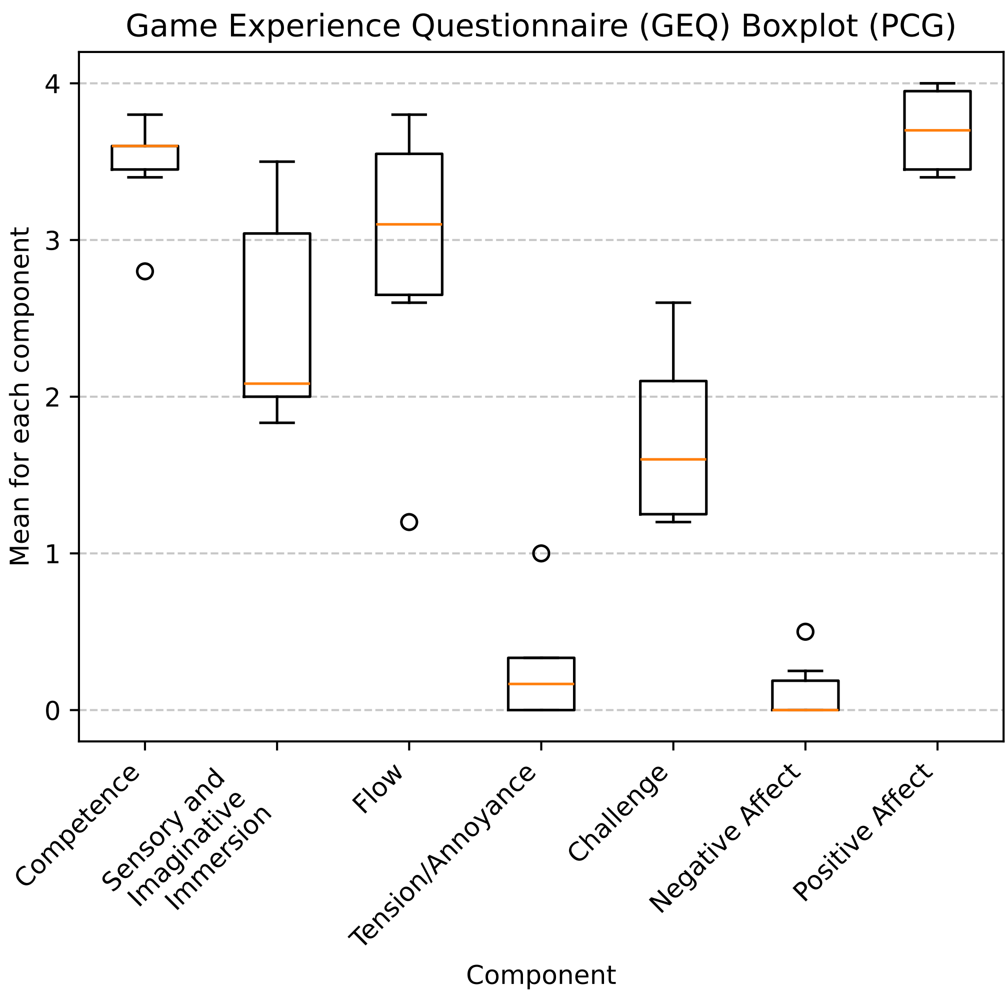

Figure 4.1: The boxplots of the means from the PCG dungeon for the first iteration for each GEQ component. From Table B.2 in Appendix B, it can be inferred that P5 is the outlier in competence, P1 in Negative Affect and P6 in Flow as well as Tension/Annoyance. The boxplots were drawn using Matplotlib provided by [64]

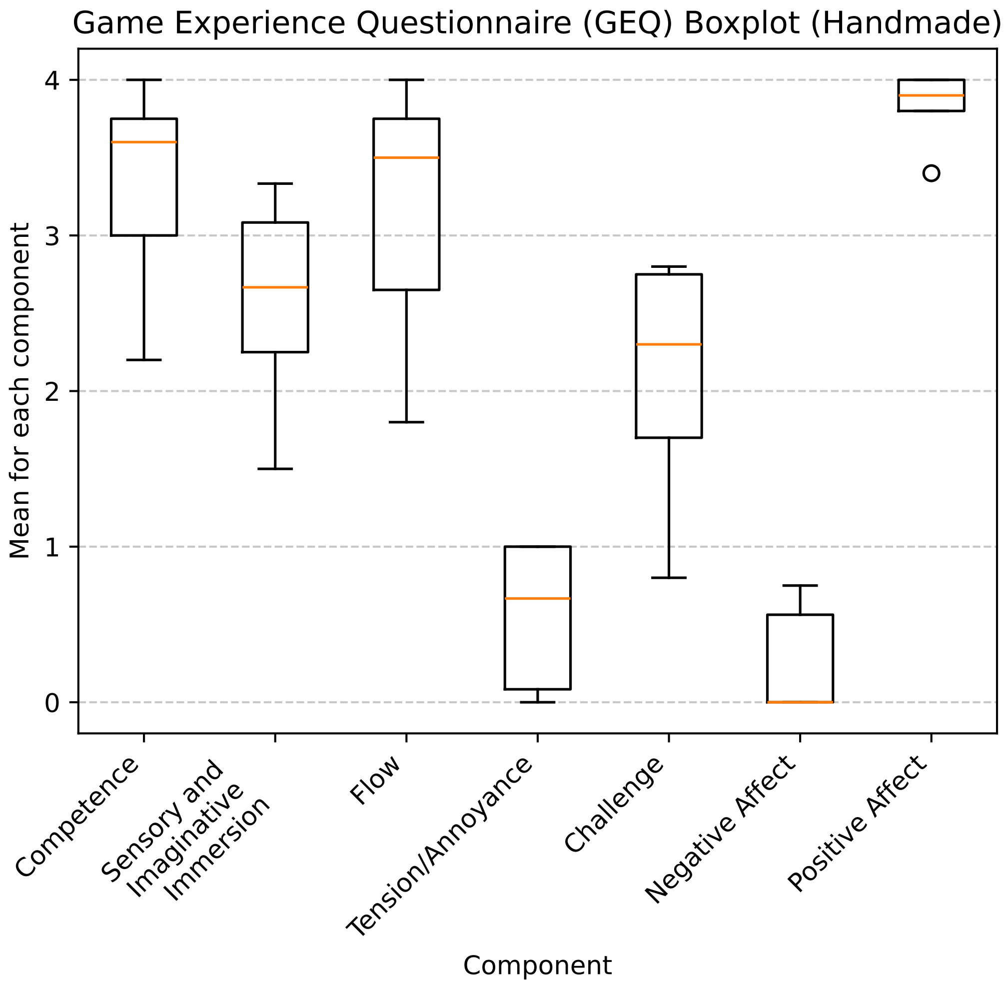

Figure 4.2: The boxplots of the means from the handmade dungeon for the first iteration for each GEQ component. From Table B.1 in Appendix B, it can be inferred that P5 is the outlier in Positive Affect. The boxplots were drawn using Matplotlib provided by [64]

4.3.6 In-Game Data Statistics

Table 4.3: The simpler game data collected during the play-test sessions for the first iteration. HM is the handmade dungeon, while PCG is the procedurally generated dungeon. The term (first) is used to indicate which dungeon was first played. Additionally, when PCG was the first dungeon played, grey colour is used.

Play-tester	Dungeon Type	Death Count	Time Spent	Enemy Death Count	Enemy Spawn Count	Chest Spawn Count	Health Potion Spawn Count
P1	HM	10	856	56	71	21	16
P1	PCG(first)	2	738	63	63	20	12
P2	HM	11	994	60	71	21	16
P2	PCG(first)	6	861	57	57	16	8
P3	HM(first)	3	657	57	71	21	16
P3	PCG	0	540	60	60	22	13
P4	HM	3	798	53	71	21	16
P4	PCG(first)	3	891	65	65	19	9
P5	HM(first)	4	655	59	71	21	16
P5	PCG	0	580	59	59	24	11
P6	HM(first)	5	947	62	71	21	16
P6	PCG	2	664	58	58	23	11

4.4 Second Iteration

In the same manner as the first iteration, the play-testers in the second iteration were also given different orders in which to play the dungeons. P7, P11, and P12 all played through the procedurally generated dungeon first, while P8, P9 and P10 played the handmade one first. A detail not visible in the results is that some play-testers did not formulate a lot of thought when they played through the game. This was an issue for this iteration, which caused less data to be gathered from the Think-Aloud method.

4.4.1 Common Interview Responses

In the second iteration, the answers to the interview questions mostly showed indifference towards the nature of how the both dungeons were created once again. A few play-testers perceived the PCG dungeon and its randomised nature in a more positive tone after disliking the experience while playing through the dungeon. A couple of participants noted that they would have been a bit harsher in their questionnaire answer would they have known that it was procedurally generated.

Dungeon preference was not as unanimous as the previous iteration, as P10 preferred the PCG dungeon over the handmade because of the increased difficulty. The rest of the participants preferred the handmade dungeon, either because of the particular setup of certain rooms or because it was concise and easy to progress through.

Most participants preferred the PCG dungeon as their second dungeon for various reasons, including both those who played it first and those who played it second. Either because the PCG dungeon was more difficult than the handmade one, and doing them in a different order would have upset the natural progression of difficulty, or because the controlled placement of enemies and player weapons is preferable as the player is getting familiar with the controls and combat.

4.4.2 Noteworthy Interview Responses

P8 was the only participant who actively tried to explore the entire procedurally generated dungeon. During their answer to the question about their thoughts on the PCG dungeon now knowing how it is made, they note how the PCG dungeon displayed many glaring issues in the game that were either not obvious or encountered during the handmade dungeon. Their gripes during the play-test session were about how enemies' health do not reset after the player dies, enemies can attack through wall tiles, how some weapon classes outshine others, and the occasional movement jitter that the enemies exhibit when trying to find their path towards the player.

4.4.3 Common Think-Aloud Topics

In this iteration, play-testers were requesting additional features that they thought were crucial for the game. For example, a mini-map and checkpoints to help with traversing the PCG dungeon were suggested, since re-spawning from the beginning of the level and having to go through the maze or trap rooms was deemed too punishing.

Play-testers were also critical to certain game mechanics while going through the procedurally generated dungeon. Among these were, the lack of progression when killing enemies, the underwhelming rewards in the form of random weapons in chests as they are progressing through the level, and the effectiveness of the spear weapon and how steep the difference is between it and other weapons.

4.4.4 Noteworthy Think-Aloud Topics

Play-tester P12, the game developer (see Section 4.2.1) with the outlying questionnaire answers (see Figs. 4.4 and 4.3), demonstrated a high level of elaboration during the session. They connected observations with previous assessments and actively probed for typical bugs and shortcomings that are common for projects of this calibre. Their reoccurring notion was that there was no reason for killing enemies in the PCG dungeon, yet there was reason in the handmade dungeon, as progression was locked behind clearing certain enemies.

4.4.5 Game Experience Scores

Table 4.4: The means and standard deviations of each component from the game experience questionnaire for the second iteration. They are written in the format of M ± SD, where M is the mean and SD is the standard deviation.

GEQ Components	**Handmade**	**PCG**
Competence	2.67 ± 0.39	2.43 ± 0.79
Sensory and Imaginative Immersion	2.39 ± 0.98	2.42 ± 1.00
Flow	2.60 ± 1.00	2.60 ± 0.93
Tension/Annoyance	0.17 ± 0.28	1.22 ± 0.78
Challenge	1.53 ± 0.68	1.20 ± 1.02
Negative affect	0.67 ± 0.97	0.96 ± 0.89
Positive affect	3.17 ± 0.61	3.17 ± 0.72

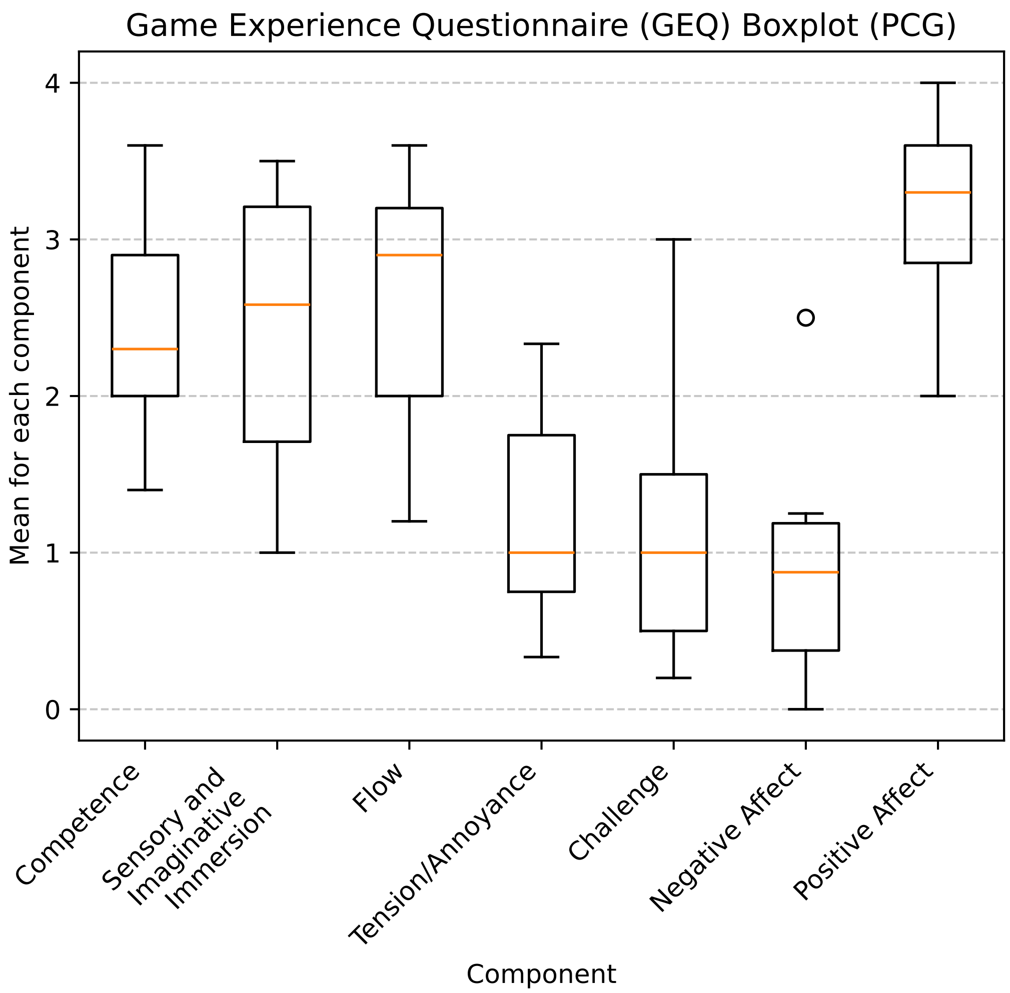

Figure 4.3: The boxplots of the means from the PCG dungeon for the second iteration. From Table B.7 in Appendix B, it can be inferred that P12 is the outlier in Negative Affect. The boxplots were drawn using Matplotlib provided by [64]

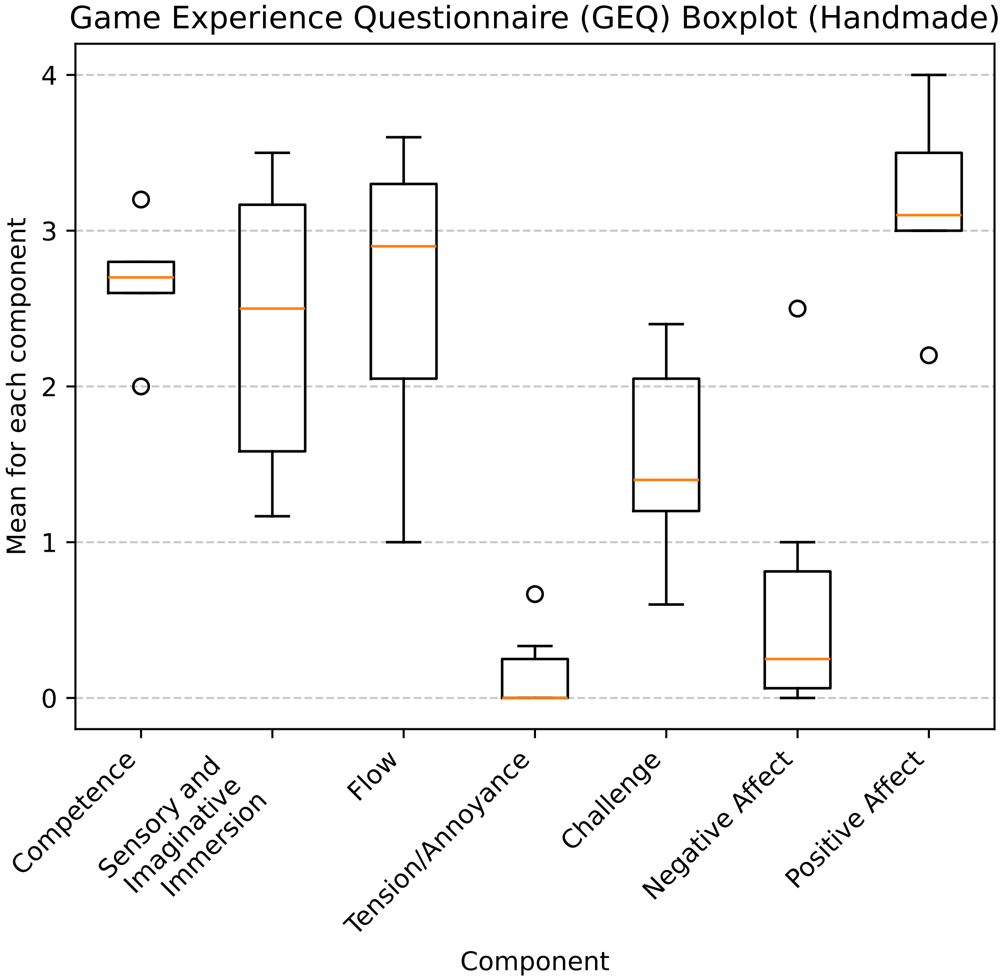

Figure 4.4: The boxplots of the means from the handmade dungeon for the second iteration for each GEQ component. From Table B.6 in Appendix B, it can be inferred that P12 is the outlier in Positive Affect, Negative Affect and one of the outliers in Competence. Specifically, P12 is the bottom outlier in Competence, while P11 is the top outlier. Lastly, P7 is the outlier in Tension/Annoyance. The boxplots were drawn using Matplotlib provided by [64]

4.4.6 In-Game Data Statistics

Table 4.5: The simpler game data collected during the play-test sessions for the second iteration. HM is the handmade dungeon, while PCG is the procedurally generated dungeon. The term (first) is used to indicate which dungeon was first played. Additionally, when PCG was the first dungeon played, grey colour is used.

Play-tester	Dungeon Type	Death Count	Time Spent	Enemy Death Count	Enemy Spawn Count	Chest Spawn Count	Health Potion Spawn Count
P7	HM	4	654	38	71	21	16
P7	PCG(first)	11	1396	71	99	39	19
P8	HM(first)	5	760	55	71	21	16
P8	PCG	4	1676	89	93	44	26
P9	HM(first)	5	593	47	71	21	16
P9	PCG	7	1530	89	95	36	26
P10	HM(first)	5	605	59	71	21	16
P10	PCG	6	1009	77	89	46	24
P11	HM	4	725	54	71	21	16
P11	PCG(first)	15	1805	88	99	45	19
P12	HM	5	784	40	71	21	16
P12	PCG(first)	8	1302	62	98	39	19

4.5 Summary

To summarise the result section, there were both qualitative and quantitive data collected. The order in which play-testers played the dungeons affected their experience in both iterations. For the first iteration, the PCG dungeon felt like a tutorial, while playing the handmade dungeon first for the second iteration was perceived more natural. The latter indicates that the opposite was true for the second iteration. In other words, the handmade dungeon was experienced as a tutorial. Additionally, for the second iteration, many play-testers suggested new features while playing the PCG dungeon.

When it comes to quantitative data, a very important detail is the time spent in the dungeons. When looking at how long people spent in the PCG dungeon in the second iteration, it is significantly longer than the rest. The opposite was true in the first iteration, where the time spent in the procedurally generated dungeon was often less than in the handmade one. In addition, when looking at the deaths from each iteration, this followed a similar pattern. In most cases, play-testers died more often in the PCG dungeon in the second iteration, but less in the first iteration.

Then for the game experience scores, there were means and standard deviations provided in the Tables 4.2 and 4.4. To provide a new perspective, the means are compared in Table 4.6. From Table 4.6 it can be observed that the Tension/Annoyance was the component to increase the most from iteration one to two. In addition, when comparing the Tables 4.2 and 4.4, there were noticeably higher scores given for both dungeons in the first iteration as compared to the second. When looking specifically at the handmade dungeon, which stayed the same, the Negative affect was the only component to increase while the rest, Positive affect, Flow, Competence, Challenge, Tension/Annoyance, as well as Sensory and Imaginative Immersion all decreased.

Table 4.6: The difference between the handmade and PCG dungeon in both iterations. In the first two columns, a positive value indicates that the PCG dungeon scored higher, whereas a negative indicates a higher score for the handmade one. In the last column, the change from iteration one to iteration two is shown.

GEQ Components	**Iteration one difference (D1)**	**Iteration two difference (D2)**	**Change between iterations (D2-D1)**
Competence	0.13	-0.23	-0.37
Sensory and Imaginative Immersion	-0.11	0.03	0.14
Flow	-0.27	0.00	0.27
Tension/Annoyance	-0.28	1.06	1.33
Challenge	-0.37	-0.33	0.03
Negative affect	-0.13	0.29	0.42
Positive affect	-0.13	0.00	0.13

5
Discussion

Based on the results gathered, there are many points to discuss. The following subsections will highlight key points.

5.1 Constraints on Procedurally Generated Content

In iteration two, quantitative data from the GEQ in Tables 4.2, 4.4, and 4.6 suggest increased frustration and annoyance towards the procedurally generated dungeon when compared to both the handmade dungeon as well as the PCG dungeon in the first iteration. Qualitative data from Think-Aloud and interviews suggest that players had a hard time navigating the PCG dungeon. For example, players felt that adding a mini-map and checkpoints would improve their experience. As seen in Table 4.5, time spent in the PCG dungeon ranged from two to three times the time spent in the handmade dungeon. There are several possible contributing factors.

To begin, enemy count, which was about 25% higher because of the increased number of rooms (see Table 4.5), is certain to increase the play time as more time is spent combating them, which is shown in the enemy death count column of the same table.

On average, the player also died about 50% more frequently in the second iteration of the PCG dungeon. The most likely explanation is the presence of more enemies and traps that are harder to avoid than the carefully placed ones in the handmade level.

Another possibly impactful factor is the non-linearity introduced by the branching present in the second iteration PCG dungeon. According to Tables 4.3 and 4.5, playtesters spent two to three times the amount of time in the second PCG dungeon compared to the first PCG dungeon, which was similar to the handmade dungeon in terms of time spent. In the first iteration, each level was completely linear, with only one possible path to take. The second iteration had many branching and cyclic paths, which appears to have hindered navigation, as indicated by the mini-map request.

The second iteration had 14-16 rooms, compared to 10 in the first iteration. This would, even when discarding the idea of mis-navigation, introduce a significant difference in time simply because it takes longer to walk through the level, assuming that the player explores some of the branches and does not take the optimal path.

Logically, adding more rooms, branches and dangers can only increase the time spent in the level, so one can rightfully ask, why did the length of the optimal path not decrease in the second iteration? In hindsight, it is obvious that keeping the 10-room optimal path would affect the issues demonstrated by the results. This decision was, as mentioned in Section 3.7.4, made based on expert testing by the developers, in which the length of the dungeon was deemed similar to the handmade dungeon. The issue with this method of evaluation is of course the fact that the developers, in addition to high skill in the game, had knowledge about patterns of the generation that could have eased their navigation through the level.

The issues mentioned above stem from a failure to correctly constrain the generated content. As described by Kenny Backus [65], limiting what is procedurally generated is critical for the game experience. The constraints laid out by the system requirements outlined in Section 3.7.1 were not satisfied, as both length and difficulty of the second iteration PCG dungeon exceeded that of the handmade dungeon. The failure was less so in the ability to constrain these properties, and more so in the use of expert testing as the evaluation method when tweaking the parameters of the system.

5.2 Change in Difficulty between Iterations

As per Section 4.4.1, play-testers were of the opinion that playing the handmade dungeon before the PCG dungeon in the second iteration would provide a better experience than the reverse order. The given reasons hint at the handmade dungeon providing a better introduction to the game mechanics. This can be explained by the slow introduction of new features, as described in Section 3.6. The opposite was true for the first iteration, where players regarded it as a tutorial level, because of its low difficulty and lack of features when compared to the handmade dungeon.

5.3 Discrepancy between Results & Expectations

Contrary to what the quantitative data might show at first, the second iteration was more successful than the first. This can be inferred from the qualitative data. In many cases, as previously mentioned in Section 5.1, there were comments about faults in the game. The fact that these appeared implies that the second iteration had more depth to it, as it was no longer perceived as a tutorial as it was in iteration one (see Sections 4.3.1 and 4.4.1). To continue this trend, future iterations could continue to improve.

Furthermore, as mentioned in Section 4.5, the scores differed between the people from iteration one and two. The scores from the components, Positive affect, Flow, Sensory and Imaginative Immersion can be thought of as positive traits of the game. All of these traits had lower scores in the second iteration. Additionally, as mentioned in Section 4.3.3, many play-testers in iteration one enjoyed the difficulty, where some described how experiencing frustration and annoyance is a positive trait that games ought to have. This resulted in higher challenge as well as Tension/Annoyance scores for iteration one. Based on how the specific scores were changed, it can be inferred that the people from iteration one were less critical. Surprisingly, as seen in Table 4.1, they had on average more hours per week, indicating a higher skill level and experience with video games.

Another perspective worth considering is the possibility of one property affecting the player's judgement of the others. For example, due to the annoyance felt in the second iteration, which was the most significant change from the first iteration (see Table 4.6), the ability of the play-testers to accurately judge whether they felt the game was more challenging or not could have been affected. This would then explain the discrepancy between the expectation that the second iteration, due to its increased length, enemy count, and the introduction of traps, would be perceived as more difficult than the first iteration, which was perceived as a tutorial (see Section 4.3.1) by the play-testers. Since several variables were changed, it is possible that one influenced the other.

5.4 Discrepancy between Qualitative & Quantitative Results

As mentioned in the previous section, the qualitative data for the second iteration makes it seem better, but do not correlate with the quantitative data as GEQ components like Tension/Annoyance and Negative affect are much higher for the second iteration (see Table 4.4). Looking at the game data in Table 4.5 compared to Table 4.3, it is possible to see that the time spent in the PCG dungeon is much higher in iteration two than it was in the first iteration. For many play-testers, a lot of this extra time was spent walking around rooms they had already completed to try to find where they were supposed to go or backtracking to where they were after death, which could be contributing to the play-testers feeling annoyed.

As noted under Section 4.4.4, the play-tester P12 was able to find many faults with the game, but they were specifically mentioned during the PCG dungeon, even though both dungeons use the same game systems.

5.5 Relation between Algorithmic Complexity & User Experience

When comparing the algorithmic complexity of the two iterations, the data shows no direct correlation with user experience. All metrics point to the complexity between iterations increasing, as can be seen in Tables 3.1 and 3.2. The Halstead Volume of the content layer in iteration two was 2.4 times higher, and the layout layer was 10.4 times higher than in iteration one. However, as discussed in Section 5.3, this increase in complexity did not correspond to a proportional improvement in user experience.

This discrepancy highlights a key point, namely that users do not interact with the algorithm itself, but with its generated output. As explored in Section 5.1, user experience in the second iteration was likely influenced by insufficient constraints, particularly the lack of control over level length. Because the complexity of the algorithm is abstracted away from the player, any potential relationship between algorithmic complexity and user experience is indirect and obscured by other factors.

While the addition of new features in the second iteration led to an increase in algorithmic complexity, the data does not support a corresponding improvement in user experience. Had a clear link been established between the new features and enhanced user experience, one might conclude that the added complexity was justified. In the absence of such evidence, no such conclusion can be drawn.

5.6 Relation between Algorithmic Complexity & Game Development

The results of the complexity analysis suggest that algorithmic complexity significantly affects development effort. Developer time tracked during the project aligns reasonably well with the Halstead Effort metric estimates (see Tables 3.3 and 3.4), though the actual hours were somewhat lower. This suggests that Halstead metrics can provide a reasonable approximation of the time required to implement an algorithm.

If developers can make informed estimates of the Halstead parameters, they can generate structured and transparent predictions about development time. Compared to expert judgment alone, this formalised approach introduces a shared framework that enhances communication and planning within development teams. Such methods are especially effective when used in combination with expert assessment [66].

Overall, the Halstead metrics offered a fairly accurate description of algorithm size, implementation effort, and difficulty. As such, they proved to be a valuable tool for assessing how algorithmic complexity impacts the game development process.

6

Conclusion

This study explored the relationship between PCG and player experience, highlighting the importance of careful design of constraints and evaluation in procedural generation systems. The results from the second iteration demonstrated that while increased algorithmic complexity can lead to richer and more challenging content, an overly complex and ill-constrained result also risks overwhelming the player if not properly balanced with navigational support and difficulty management. The increased frustration and annoyance reported in the second PCG iteration, compared to the handmade dungeon and the first iteration, stemmed not only from objective metrics such as higher enemy counts and longer playtime but also from structural design issues like excessive branching and unclear progression.

A major theme across the findings is the discrepancy between qualitative and quantitative data. Although numeric scores indicated a decline in player experience, the qualitative feedback suggested a more engaged and critical player response, an indicator of greater depth and potential for improvement. However, the small sample size limits the statistical strength of the conclusions and reinforces the need to prioritise qualitative findings in studies with limited participation. Additionally, the study highlighted how play order, player familiarity, and multiple changing variables can influence perception and complicate interpretation of difficulty and enjoyment metrics.

Importantly, this study could not establish any direct relationship between algorithmic complexity and user experience. Although the second iteration was significantly more complex, exhibiting 2.4 times the Halstead Volume in the content layer and 10.4 times in the layout layer, the user experience did not improve accordingly. This suggests that added complexity alone does not guarantee improved gameplay. The absence of constraints, namely on the level length, likely had a greater impact on perceived experience than the complexity of the procedural generation logic.

However, the complexity analysis did yield valuable insights into the development process. The Halstead Effort metric aligned closely with actual development time, suggesting that algorithmic complexity is a useful predictor of implementation cost. Developers who estimate Halstead parameters can make structured and reasonably accurate forecasts of development effort, supporting better planning and communication. Thus, while algorithmic complexity may not correlate with user experience, it remains a critical factor in project scoping and team coordination.

The methodology used in this thesis proved to be an appropriate way to compare procedurally generated content and handmade content. Using these methods, future work could explore additional iterations that address the shortcomings identified in Section 5.1. For example, simply reducing the number of rooms in the second iteration could help isolate the effects of added features such as room types and traps, that were likely overshadowed by excessive level length. Further experiments could also explore more constrained versions of the generation system that interpolate between the two existing iterations to isolate individual variables.

Implementing in-game navigational aids such as mini-maps or path markers, as suggested by participants, would be possible features to test for future research. Additionally, future iterations could benefit from evaluating generation quality through user testing rather than expert play, mitigating developer bias. Finally, expanding the number of participants would allow for more robust statistical conclusions and a better understanding of how different player demographics interpret procedural generation's complexity and challenge.

In summary, this study underscores the potential of PCG to create engaging gameplay and illustrates the need for deliberate design constraints when increasing the complexity of procedural algorithms for content generation. It also highlights the importance of balanced data collection for user experience evaluation and iterative refinement.

Bibliography

[1] L. Wooldridge, *Games market enters a new era as pandemic years reset growth dynamic*, 16th May 2023. [Online]. Available: https://www.ampereanalysis.com/insight/games-market-enters-a-new-era-as-pandemic-years-reset-growth-dynamic (visited on 17/04/2025).

[2] SteamDB, *Steam game release summary by year*, 2025. [Online]. Available: https://steamdb.info/stats/releases/ (visited on 17/04/2025).

[3] R. Elliott, *Game budgets have gone out of control – let's shrink the scope*, 11th Apr. 2024. [Online]. Available: https://www.midiaresearch.com/blog/single-player-game-budgets-and-scopes-have-spiralled-out-of-control-its-time-to-shrink-the-scope (visited on 02/06/2025).

[4] J. Togelius, E. Kastbjerg, D. Schedl and G. N. Yannakakis, "What is procedural content generation? mario on the borderline," in *PCGames '11: Proceedings of the 2nd International Workshop on Procedural Content Generation in Games*, Association for Computing Machinery, Jun. 2011, pp. 1–6, ISBN: 9781450308724. DOI: 10.1145/2000919.2000922.

[5] Kenny, *Procedural generation: An overview*, 19th Feb. 2021. [Online]. Available: https://kentpawson123.medium.com/procedural-generation-an-overview-1b054a0f8d41 (visited on 17/04/2025).

[6] R. V. D. Linden, R. Lopes and R. Bidarra, "Procedural generation of dungeons," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 6, pp. 78–89, 1 Mar. 2014, ISSN: 1943068X. DOI: 10.1109/TCIAIG.2013.2290371.

[7] N. Gupta, *Is procedural content generation (pcg) a game-changing tool for advanced game design?* 23rd Feb. 2024. [Online]. Available: https://www.seamedu.com/blog/procedural-content-generation-tool-for-advanced-game-design (visited on 17/05/2025).

[8] C. Dunson, *The importance of variety in gaming*, 8th Dec. 2013. [Online]. Available: https://www.gamedeveloper.com/design/the-importance-of-variety-in-gaming (visited on 17/04/2025).

[9] J. Aycock, "Procedural content generation," in *Retrogame Archeology: Exploring Old Computer Games*, J. Aycock, Ed. Springer International Publishing, 2016, pp. 109–143, ISBN: 978-3-319-30004-7. DOI: 10.1007/978-3-319-30004-7_6.

[10] A. Parrish, *Minecraft has sold over 300 million copies*, 15th Oct. 2023. [Online]. Available: https://www.theverge.com/2023/10/15/23916349/m

inecraft-mojang-sold-300-million-copies-live-2023 (visited on 17/04/2025).
[11] S. Arif, *No man's sky was august's top download on the playstation store*, 13th Sep. 2016. [Online]. Available: https://www.vg247.com/no-mans-sky-was-augusts-top-download-on-the-playstation-store (visited on 17/04/2025).
[12] Mojang Studios, *What is minecraft? build, discover realms & more*, 2025. [Online]. Available: https://www.minecraft.net/en-us/about-minecraft (visited on 17/04/2025).
[13] Hello Games, *No man's sky*, 2025. [Online]. Available: https://www.nomanssky.com/ (visited on 17/04/2025).
[14] K. Orland, *How sky-high hype formed a storm cloud over no man's sky's release*, 10th Aug. 2016. [Online]. Available: https://arstechnica.com/gaming/2016/08/how-sky-high-hype-formed-a-storm-cloud-over-no-mans-skys-release/ (visited on 17/04/2025).
[15] S. Saed, *No man's sky concurrent player numbers down by 90% on pc, less than two weeks from release*, 25th Aug. 2016. [Online]. Available: https://www.vg247.com/no-mans-sky-concurrent-player-numbers-down-by-90-on-pc-less-than-two-weeks-from-release (visited on 17/04/2025).
[16] B. Hillier, *The internet's list of everything missing from no man's sky*, 17th Aug. 2016. [Online]. Available: https://www.vg247.com/everything-missing-from-no-mans-sky-list (visited on 17/04/2025).
[17] S. Shahzad, *5 failed video games that had a resurgence*, 14th Apr. 2025. [Online]. Available: https://comicbook.com/gaming/news/5-failed-video-games-that-had-a-resurgence/ (visited on 05/05/2025).
[18] J. H. Liu, S. K. Zhang, C. Zhang and S. H. Zhang, "Controllable procedural generation of landscapes," in *MM 2024 - Proceedings of the 32nd ACM International Conference on Multimedia*, vol. 24, Association for Computing Machinery, Oct. 2024, pp. 6394–6403, ISBN: 9798400706868. DOI: 10.1145/3664647.3681129.
[19] L. Rodrigues, R. Bonidia and J. Brancher, "Procedural versus human level generation: Two sides of the same coin?" *International Journal of Human-Computer Studies*, vol. 141, pp. 1–11, Sep. 2020, ISSN: 1071-5819. DOI: 10.1016/J.IJHCS.2020.102465.
[20] 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 (CIE)*, vol. 15, pp. 1–15, 2 Apr. 2017, ISSN: 15443981. DOI: 10.1145/2974026.
[21] T. Pal and S. D. Bit, "Design and implementation of a survival game towards achieving user's satisfaction by providing balanced difficulty," *Entertainment Computing*, vol. 52, pp. 1–13, Jan. 2025, ISSN: 1875-9521. DOI: 10.1016/J.ENTCOM.2024.100899.
[22] L. T. Pereira, P. V. de Souza Prado, R. M. Lopes and C. F. M. Toledo, "Procedural generation of dungeons' maps and locked-door missions through an evolutionary algorithm validated with players," *Expert Systems with Applica-*

tions, vol. 180, pp. 1–15, Oct. 2021, ISSN: 0957-4174. DOI: 10.1016/J.ESWA.2021.115009.

[23] N. Stefyn, *How video games are made: The game dev process*, 9th May 2022. [Online]. Available: https://www.cgspectrum.com/blog/game-development-process (visited on 17/04/2025).

[24] R. Bramble, *What are the main stages of game development?* 10th May 2023. [Online]. Available: https://gamemaker.io/en/blog/stages-of-game-development (visited on 17/04/2025).

[25] N. Brewer, *Going rogue: A brief history of the computerized dungeon crawl*, 7th Jul. 2016. [Online]. Available: https://insight.ieeeusa.org/articles/going-rogue-a-brief-history-of-the-computerized-dungeon-crawl/ (visited on 17/04/2025).

[26] A. Wójtowicz and W. Cellary, "New challenges for user privacy in cyberspace," in *Human-Computer Interaction and Cybersecurity Handbook*, A. Moallem, Ed. CRC Press, Oct. 2018, pp. 77–96, ISBN: 9781315184319. DOI: 10.1201/B22142-4.

[27] Samhub, *What is user privacy?* 25th Oct. 2024. [Online]. Available: https://samhub.io/what-is-user-privacy/ (visited on 18/05/2025).

[28] ManagedMethods, *How safe is google drive?* 22nd Aug. 2024. [Online]. Available: https://managedmethods.com/blog/how-safe-is-google-drive/ (visited on 18/05/2025).

[29] E. Schwenke, *Is google drive secure? learn about google drive security*, 18th Nov. 2024. [Online]. Available: https://www.mimecast.com/blog/google-drive-security-guide/ (visited on 01/06/2025).

[30] N. Andreasson, M. Granath, G. Johnsén, D. Memedov, B. Suhail and E. Young, *Datx11 bachelor project scripts*, commit hash: 905eafff957a52c12bb782c8992f5ae5a5a14b67, 2025. [Online]. Available: https://github.com/Figer-op/DATX11-Bachelor-Project-Scripts.

[31] S. Bromley, *How many players do i need for a playtest*, 24th Sep. 2022. [Online]. Available: https://gamesuserresearch.com/how-many-players-do-i-need-for-a-playtest/ (visited on 01/05/2025).

[32] J. Nielsen, *Quantitative studies: How many users to test?* 25th Jun. 2006. [Online]. Available: https://www.nngroup.com/articles/quantitative-studies-how-many-users/ (visited on 16/04/2025).

[33] A. Fahmy, *Redefining 'gamer:' what today's diverse audience means for brands*, 14th Nov. 2024. [Online]. Available: https://www.activisionblizzardmedia.com/blog/gaming-audiences/what-todays-diverse-gaming-audience-means-for-brands (visited on 16/04/2025).

[34] Eurostat, *Glossary:coverage*. [Online]. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Glossary:Coverage (visited on 16/04/2025).

[35] C. Baker, *'no man's sky': How games are building themselves*, 9th Aug. 2016. [Online]. Available: https://www.rollingstone.com/culture/culture-news/no-mans-sky-how-games-are-building-themselves-104779/ (visited on 01/05/2025).

[36] G. D. Staff, *7 uses of procedural generation that all developers should study*, 1st Jan. 2016. [Online]. Available: https://www.gamedeveloper.com/design/7-uses-of-procedural-generation-that-all-developers-should-study (visited on 01/05/2025).

[37] R. Budiu, *Between-subjects vs. within-subjects study design*, 10th Jul. 2023. [Online]. Available: https://www.nngroup.com/articles/between-within-subjects/ (visited on 16/04/2025).

[38] C. Vinney, *Quantitative vs. qualitative research*, 20th Dec. 2022. [Online]. Available: https://www.uxdesigninstitute.com/blog/quantitative-vs-qualitative-research/ (visited on 16/04/2025).

[39] C. Garnham, *Balancing qualitative and quantitative data in ux research: Pros, cons, and best practices*, 21st Aug. 2023. [Online]. Available: https://dovetail.com/ux/qualitative-and-quantitative-data-in-ux-research/ (visited on 16/04/2025).

[40] W. IJsselsteijn, Y. de Kort and K. Poels, *The Game Experience Questionnaire*. Technische Universiteit Eindhoven, 2013, pp. 1–9. [Online]. Available: https://research.tue.nl/en/publications/the-game-experience-questionnaire.

[41] J. Nielsen, *Thinking aloud: The #1 usability tool*, 15th Jan. 2012. [Online]. Available: https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/ (visited on 27/04/2025).

[42] C. Barclay, *Semi-structured interviews*, Mar. 2018. [Online]. Available: http://sociology.fas.harvard.edu/files/sociology/files/interview_strategies.pdf (visited on 27/04/2025).

[43] C. Fox, "Agile methods in software engineering," *Access Science*, Jul. 2023. DOI: 10.1036/1097-8542.014730.

[44] M. Rehkopf, *What is a kanban board?* [Online]. Available: https://www.atlassian.com/agile/kanban/boards (visited on 30/04/2025).

[45] S. Singh and A. Kaur, "Game development using unity game engine," in *ICAN 2022 - 3rd International Conference on Computing, Analytics and Networks*, Institute of Electrical and Electronics Engineers Inc., Nov. 2022, pp. 1–6, ISBN: 9781665499446. DOI: 10.1109/ICAN56228.2022.10007155.

[46] Git, *1.3 getting started - what is git?* [Online]. Available: https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F (visited on 28/04/2025).

[47] S. Björk, *Boss monsters*, 20th Mar. 2018. [Online]. Available: http://virt10.itu.chalmers.se/index.php?title=Boss_Monsters&oldid=26475 (visited on 15/04/2025).

[48] S. Björk, *Optional goals*, 17th Mar. 2018. [Online]. Available: http://virt10.itu.chalmers.se/index.php?title=Optional_Goals&oldid=26288 (visited on 16/04/2025).

[49] E. Adams and J. Dormans, *Game Mechanics: Advanced Game Design*. New Riders Publishing, 2012, pp. 1–281, ISBN: 0321820274. [Online]. Available: https://books.google.se/books?hl=en&lr=&id=_AzioOtxIdAC&oi=fnd&pg=PR7&dq=slowly+introducing+new+game+mechanics&ots=RqkB-J6-ZL&sig=HqdG7Py7-Mws_0IRvgtJi7IQqHU&redir_esc=y#v=onepage&q&f=false.

[50] J. Engelfriet and G. Rozenberg, "Node replacement graph grammars," in *Handbook of Graph Grammars and Computing by Graph Transformation*, G. Rozenberg, Ed. World Scientific, Feb. 1997, pp. 1–94, ISBN: 9789814498104. DOI: 10.1142/9789812384720_0001.

[51] G. F. Lawler and V. Limic, "Preface," in *Random Walk: A Modern Introduction.* Cambridge University Press, Jun. 2010, vol. 123, pp. ix–xii, ISBN: 9780521519182. DOI: 10.1017/CBO9780511750854.

[52] O. Jones, "Spectra of simple graphs," *Whitman college*, vol. 13, pp. 1–20, 2013.

[53] Mendix, *Software complexity.* [Online]. Available: https://www.mendix.com/glossary/software-complexity/ (visited on 17/05/2025).

[54] Capterra, *Module.* [Online]. Available: https://www.capterra.com/glossary/module/ (visited on 17/05/2025).

[55] T. Gilboy, *Maintainability index - what is it and where does it fall short?* 7th Mar. 2022. [Online]. Available: https://sourcery.ai/blog/maintainability-index/ (visited on 02/05/2025).

[56] T. Hariprasad, G. Vidhyagaran, K. Seenu and C. Thirumalai, "Software complexity analysis using halstead metrics," in *International Conference on Trends in Electronics and Informatics, ICEI 2017*, Institute of Electrical and Electronics Engineers Inc., May 2017, pp. 1109–1113, ISBN: 9781509042579. DOI: 10.1109/ICOEI.2017.8300883.

[57] IBM, *Cyclomatic complexity*, 8th Mar. 2021. [Online]. Available: https://www.ibm.com/docs/en/raa/6.1.0?topic=metrics-cyclomatic-complexity (visited on 17/05/2025).

[58] Repeato, *Understanding when a software product is stable*, Jul. 2024. [Online]. Available: https://www.repeato.app/understanding-when-a-software-product-is-stable/.

[59] A. V. Gorchakov, L. A. Demidova and P. N. Sovietov, "A rule-based algorithm and its specializations for measuring the complexity of software in educational digital environments," *Computers*, vol. 13, p. 75, 3 Mar. 2024, ISSN: 2073-431X. DOI: 10.3390/COMPUTERS13030075.

[60] Y. D. Prabowo, H. L. H. S. Warnas, F. L. Gaol, E. Abdurachman and B. Soewito, "An initial research on halstead's technique for pattern similarity relationship study," in *2018 International Conference on Information and Communications Technology, ICOIACT 2018*, vol. 2018-January, Institute of Electrical and Electronics Engineers Inc., Apr. 2018, pp. 773–777, ISBN: 9781538609545. DOI: 10.1109/ICOIACT.2018.8350771.

[61] R. T. Coimbra, A. M. P. D. Resende and R. Terra, "A correlation analysis between halstead complexity measures and other software measures," in *2018 XLIV Latin American Computer Conference (CLEI)*, Institute of Electrical and Electronics Engineers Inc., Oct. 2018, pp. 31–39, ISBN: 9781728104379. DOI: 10.1109/CLEI.2018.00014.

[62] N. Govil, "Applying halstead software science on different programming languages for analyzing software complexity," in *Proceedings of the 4th International Conference on Trends in Electronics and Informatics, ICOEI 2020*, Institute of Electrical and Electronics Engineers Inc., Jun. 2020, pp. 939–943, ISBN: 9781728155180. DOI: 10.1109/ICOEI48184.2020.9142911.

[63] N. Vansuetendael and D. Elwell, “Software quality metrics,” Computer Resource Management Inc Pleasantville NJ, Tech. Rep., Aug. 1991.

[64] The Matplotlib Development Team, “Matplotlib: Visualization with python,” *Zenodo*, Feb. 2025. DOI: `10.5281/ZENODO.14940554`.

[65] K. Backus, “Managing output,” in *Procedural Generation in Game Design*, T. X. Short and T. Adams, Eds. A K Peters/CRC Press, Jun. 2017, pp. 13–21, ISBN: 9781315156378. DOI: `10.1201/9781315156378-3`.

[66] A. Trendowicz, J. Münch and R. Jeffery, “State of the practice in software effort estimation: A survey and literature review,” in *Software Engineering Techniques*, Z. Huzar, R. Koci, B. Meyer, B. Walter and J. Zendulka, Eds., Springer Berlin Heidelberg, 2011, pp. 232–245, ISBN: 978-3-642-22386-0. DOI: `10.1007/978-3-642-22386-0_18`.

A

Questions

Table A.1: All questions used in the post interview during the play-test sessions. Depending on the dungeon order, the handmade dungeon or the PCG dungeon might be first.

Interview questions
Do you have a hunch of what the difference between the two dungeons were?
The **first** dungeon you played was [handmade/procedurally generated], does it affect the overall game experience for you?
The **second** dungeon you played was [handmade/procedurally generated], does it affect the overall game experience for you?
Knowing that the **first** dungeon was [handmade/procedurally generated], does it affect your previous thoughts? Think about the questionnaires that you did.
Knowing that the **second** dungeon was [handmade/procedurally generated], does it affect your previous thoughts?
Which dungeon did you like the most?

Table A.2: All questions used in the demographic questionnaire along with what type of question they were.

Demographic questions	**Type of question**
What is your name?	Open answer.
What is your age?	Open answer limited to strictly positive integers.
What do you identify as?	Multiple choice, with options “Male”, “Female”, “Prefer not to say”, “Other ...”.
How experienced are you with playing video games?	Scale from 0 to 4.
Do you identify yourself as a gamer?	Multiple choice, with options “Yes”, “No”, “Maybe”.
For how many years have you played games?	Open answer limited to positive integers.
How much do you play per week? (in hours)	Open answer limited to positive integers.
What are your favourite game genres?	Multiple choice with options “First person shooter”, “MOBA”, “Puzzle games”, “Dungeon crawlers”, “Tactical shooters”, “Strategy games”, “Platformers”, “Tower defence”, “Horror”, “Survival”, “Open world”, “MMO”, “Fighting”, “Exploration”, “None of the above”. (Multiple could be selected)
What dates in the next two upcoming weeks (v.16 - v.17) are you available?	Open answer.
Please provide an email in case we want to contact you	Open answer. (Optional)

B

Data Collection

B.1 First Iteration

Table B.1: The individual play-tester's means for each component in the Game Experience Questionnaire for the handmade dungeon in the first iteration. Grey colour is used to show which play-tester did the PCG dungeon first.

GEQ Components	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**
Competence	2.80	3.80	3.60	3.60	2.20	4.00
Sensory and Imaginative Immersion	3.17	3.33	2.83	2.17	1.50	2.50
Flow	4.00	3.80	3.60	3.40	2.40	1.80
Tension/Annoyance	0.33	0.00	1.00	0.00	1.00	1.00
Challenge	2.60	1.60	2.80	2.80	2.00	0.80
Negative Affect	0.00	0.00	0.00	0.75	0.75	0.00
Positive Affect	4.00	4.00	3.80	4.00	3.40	3.80

Table B.2: The individual play-tester's means for each component in the Game Experience Questionnaire for the PCG dungeon in the first iteration. Grey colour is used to show which play-tester did the PCG dungeon first.

GEQ Components	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**
Competence	3.40	3.60	3.60	3.60	2.80	3.80
Sensory and Imaginative Immersion	2.17	3.33	3.50	2.00	1.83	2.00
Flow	3.80	3.60	3.40	2.60	2.80	1.20
Tension Annoyance	0.33	0.00	0.00	0.00	0.33	1.00
Challenge	1.20	1.40	2.60	2.20	1.20	1.80
Negative Affect	0.50	0.00	0.00	0.00	0.25	0.00
Positive Affect	3.60	3.80	3.40	4.00	3.40	4.00

Table B.3: The individual answers provided in the game experience questionnaire in the first iteration for the handmade dungeon. Grey colour is used to show which play-testers did the PCG dungeon first.

Item	P1	P2	P3	P4	P5	P6
I felt content	4	4	4	4	3	3
I felt skilful	4	4	4	4	3	4
I was interested in the game's story	3	2	2	0	0	0
I thought it was fun	4	4	4	4	3	4
I was fully occupied with the game	4	4	3	4	4	4
I felt happy	4	4	3	4	3	4
It gave me a bad mood	0	0	0	0	0	0
I thought about other things	0	0	0	3	0	0
I found it tiresome	0	0	0	0	2	0
I felt competent	3	4	4	4	2	4
I thought it was hard	3	2	3	4	3	1
It was aesthetically pleasing	4	4	4	3	3	3
I forgot everything around me	4	4	4	3	4	2
I felt good	4	4	4	4	4	4
I was good at it	2	4	3	3	2	4
I felt bored	0	0	0	0	1	0
I felt successful	3	4	4	4	1	4
I felt imaginative	3	3	3	2	2	3
I felt that I could explore things	3	4	2	2	2	4
I enjoyed it	4	4	4	4	4	4
I was fast at reaching the game's targets	2	3	3	3	3	4
I felt annoyed	1	0	0	0	2	1
I felt pressured	2	0	4	2	1	2
I felt irritable	0	0	0	0	0	1
I lost track of time	4	4	4	3	0	0
I felt challenged	3	3	4	4	2	1
I found it impressive	3	4	3	3	2	3
I was deeply concentrated in the game	4	4	4	4	3	3
I felt frustrated	0	0	3	0	1	1
It felt like a rich experience	3	3	3	3	0	2
I lost connection with the outside world	4	3	3	3	1	0
I felt time pressure	2	0	0	0	3	0
I had to put a lot of effort into it	3	3	3	4	1	0

Table B.4: The individual answers provided in the game experience questionnaire in the first iteration for the PCG dungeon. Grey colour is used to show which play-testers did the PCG dungeon first.

Item	P1	P2	P3	P4	P5	P6
I felt content	3	3	4	4	4	4
I felt skilful	3	4	4	4	3	4
I was interested in the game's story	1	2	3	0	0	0
I thought it was fun	4	4	3	4	3	4

I was fully occupied with the game	4	4	4	4	3	3
I felt happy	3	4	4	4	3	4
It gave me a bad mood	0	0	0	0	0	0
I thought about other things	0	0	0	0	1	0
I found it tiresome	1	0	0	0	0	0
I felt competent	3	3	3	4	3	4
I thought it was hard	2	1	2	3	1	3
It was aesthetically pleasing	3	4	4	4	4	2
I forgot everything around me	4	4	3	3	3	0
I felt good	4	4	3	4	4	4
I was good at it	3	4	4	3	2	3
I felt bored	1	0	0	0	0	0
I felt successful	4	3	3	4	3	4
I felt imaginative	3	3	4	2	2	2
I felt that I could explore things	1	3	3	2	1	2
I enjoyed it	4	4	3	4	3	4
I was fast at reaching the game's targets	4	4	4	3	3	4
I felt annoyed	1	0	0	0	1	1
I felt pressured	0	0	3	2	1	2
I felt irritable	0	0	0	0	0	1
I lost track of time	3	3	4	2	2	0
I felt challenged	2	3	3	3	1	3
I found it impressive	3	4	4	2	3	3
I was deeply concentrated in the game	4	4	3	3	3	3
I felt frustrated	0	0	0	0	0	1
It felt like a rich experience	2	4	3	2	1	3
I lost connection with the outside world	4	3	3	1	3	0
I felt time pressure	0	0	1	0	2	0
I had to put a lot of effort into it	2	3	4	3	1	1

Table B.5: The collected data associated with weapon statistics for the first iteration. In the second column, HM is the handmade dungeon and PCG is the procedurally generated dungeon. Next to the abbreviation, (first) is used to denote which dungeon was first played. To further emphasise, grey colour is used to show which play-testers did the PCG dungeon first.

Play-tester	**Dungeon Type**	**Weapon Name**	**First Time Equipped**	**Equip Count**
P1	HM	PlayerWoodenDagger	55	3
P1	HM	PlayerWoodenSword	58	1
P1	HM	PlayerWoodenSpear	88	1
P1	HM	PlayerIronDagger	169	7
P1	HM	PlayerIronAxe	172	5
P1	HM	PlayerIronSword	280	1

P1	HM	PlayerGoldenDagger	406	3
P1	HM	PlayerGoldenSpear	515	1
P1	HM	PlayerDiamondSword	662	2
P1	HM	PlayerDiamondDagger	805	1
P1	PCG(first)	PlayerIronSpear	38	8
P1	PCG(first)	PlayerWoodenSword	44	1
P1	PCG(first)	PlayerWoodenSpear	66	1
P1	PCG(first)	PlayerWoodenDagger	112	1
P1	PCG(first)	PlayerGoldenSword	210	7
P1	PCG(first)	PlayerIronDagger	241	1
P1	PCG(first)	PlayerIronAxe	324	1
P1	PCG(first)	PlayerGoldenDagger	401	4
P1	PCG(first)	PlayerDiamondSword	597	1
P1	PCG(first)	PlayerDiamondSpear	616	5
P1	PCG(first)	PlayerDiamondAxe	718	2
P2	HM	PlayerWoodenDagger	36	1
P2	HM	PlayerWoodenSpear	116	1
P2	HM	PlayerIronAxe	297	4
P2	HM	PlayerIronSword	373	2
P2	HM	PlayerGoldenDagger	517	9
P2	HM	PlayerGoldenAxe	545	3
P2	HM	PlayerGoldenSword	574	4
P2	HM	PlayerGoldenSpear	722	1
P2	HM	PlayerDiamondAxe	754	4
P2	HM	PlayerDiamondSword	805	1
P2	HM	PlayerDiamondSpear	912	1
P2	HM	PlayerDiamondDagger	927	2
P2	PCG(first)	PlayerIronAxe	37	7
P2	PCG(first)	PlayerWoodenAxe	94	1
P2	PCG(first)	PlayerIronSword	125	7
P2	PCG(first)	PlayerWoodenDagger	152	3
P2	PCG(first)	PlayerIronDagger	253	1
P2	PCG(first)	PlayerIronSpear	303	1
P2	PCG(first)	PlayerGoldenDagger	453	4
P2	PCG(first)	PlayerGoldenSpear	483	3
P2	PCG(first)	PlayerGoldenSword	631	3
P2	PCG(first)	PlayerGoldenAxe	716	2
P3	HM(first)	PlayerWoodenDagger	42	5
P3	HM(first)	PlayerWoodenSword	46	4
P3	HM(first)	PlayerWoodenSpear	127	5
P3	HM(first)	PlayerIronDagger	196	2
P3	HM(first)	PlayerIronAxe	198	2
P3	HM(first)	PlayerIronSword	294	3
P3	HM(first)	PlayerIronSpear	376	1

P3	HM(first)	PlayerGoldenDagger	442	3
P3	HM(first)	PlayerGoldenSpear	450	5
P3	HM(first)	PlayerDiamondSword	545	1
P3	HM(first)	PlayerDiamondSpear	593	2
P3	HM(first)	PlayerDiamondDagger	604	1
P3	PCG	PlayerIronSpear	13	6
P3	PCG	PlayerWoodenSword	16	1
P3	PCG	PlayerWoodenAxe	53	1
P3	PCG	PlayerIronDagger	69	1
P3	PCG	PlayerGoldenSword	254	2
P3	PCG	PlayerGoldenSpear	292	1
P3	PCG	PlayerDiamondSword	354	4
P3	PCG	PlayerDiamondDagger	470	4
P4	HM	PlayerWoodenDagger	41	3
P4	HM	PlayerWoodenSword	42	4
P4	HM	PlayerWoodenSpear	94	3
P4	HM	PlayerIronDagger	142	4
P4	HM	PlayerIronSword	256	3
P4	HM	PlayerIronSpear	332	1
P4	HM	PlayerGoldenSword	409	2
P4	HM	PlayerGoldenAxe	426	5
P4	HM	PlayerGoldenDagger	466	2
P4	HM	PlayerGoldenSpear	525	1
P4	HM	PlayerDiamondSword	636	2
P4	HM	PlayerDiamondAxe	667	3
P4	HM	PlayerDiamondSpear	741	1
P4	PCG(first)	PlayerWoodenSpear	39	7
P4	PCG(first)	PlayerWoodenSword	44	1
P4	PCG(first)	PlayerIronAxe	92	5
P4	PCG(first)	PlayerIronDagger	114	3
P4	PCG(first)	PlayerIronSpear	200	5
P4	PCG(first)	PlayerWoodenDagger	214	1
P4	PCG(first)	PlayerGoldenSword	336	4
P4	PCG(first)	PlayerIronSword	379	1
P4	PCG(first)	PlayerDiamondSpear	526	4
P4	PCG(first)	PlayerGoldenSpear	588	1
P4	PCG(first)	PlayerDiamondDagger	615	4
P4	PCG(first)	PlayerGoldenAxe	688	1
P5	HM(first)	PlayerWoodenDagger	34	1
P5	HM(first)	PlayerWoodenSword	38	2
P5	HM(first)	PlayerWoodenSpear	69	1
P5	HM(first)	PlayerIronDagger	134	5
P5	HM(first)	PlayerIronAxe	158	3
P5	HM(first)	PlayerIronSword	261	1

P5	HM(first)	PlayerIronSpear	327	1
P5	HM(first)	PlayerGoldenSword	377	2
P5	HM(first)	PlayerGoldenAxe	382	1
P5	HM(first)	PlayerGoldenSpear	391	3
P5	HM(first)	PlayerDiamondSpear	602	2
P5	HM(first)	PlayerDiamondDagger	613	1
P5	PCG	PlayerWoodenSpear	7	2
P5	PCG	PlayerWoodenSword	9	3
P5	PCG	PlayerWoodenDagger	35	2
P5	PCG	PlayerIronAxe	90	2
P5	PCG	PlayerIronSword	121	2
P5	PCG	PlayerWoodenAxe	163	1
P5	PCG	PlayerIronDagger	238	2
P5	PCG	PlayerGoldenAxe	273	5
P5	PCG	PlayerIronSpear	324	1
P5	PCG	PlayerGoldenDagger	370	2
P5	PCG	PlayerGoldenSpear	415	1
P5	PCG	PlayerDiamondDagger	452	2
P6	HM(first)	PlayerWoodenDagger	19	4
P6	HM(first)	PlayerWoodenSword	20	2
P6	HM(first)	PlayerWoodenSpear	69	14
P6	HM(first)	PlayerIronAxe	152	7
P6	HM(first)	PlayerIronDagger	174	4
P6	HM(first)	PlayerIronSword	360	3
P6	HM(first)	PlayerIronSpear	461	1
P6	HM(first)	PlayerGoldenDagger	527	1
P6	HM(first)	PlayerGoldenSpear	530	3
P6	HM(first)	PlayerGoldenSword	533	1
P6	HM(first)	PlayerGoldenAxe	536	1
P6	HM(first)	PlayerDiamondAxe	639	1
P6	HM(first)	PlayerDiamondSpear	880	2
P6	HM(first)	PlayerDiamondDagger	905	1
P6	PCG	PlayerIronDagger	15	1
P6	PCG	PlayerIronSword	25	21
P6	PCG	PlayerWoodenAxe	45	5
P6	PCG	PlayerWoodenSpear	59	7
P6	PCG	PlayerWoodenDagger	77	1
P6	PCG	PlayerIronSpear	248	2
P6	PCG	PlayerGoldenAxe	284	1
P6	PCG	PlayerGoldenSpear	291	2
P6	PCG	PlayerGoldenSword	425	2
P6	PCG	PlayerDiamondSword	555	2

B.2 Second Iteration

Table B.6: The individual play-tester's means for each component in the Game Experience Questionnaire for the handmade dungeon in the second iteration. Grey colour is used to show which play-tester did the PCG dungeon first.

GEQ Components	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**
Competence	2.80	2.80	2.60	2.60	3.20	2.00
Sensory and Imaginative Immersion	1.33	3.50	2.33	3.33	2.67	1.17
Flow	3.60	3.40	3.00	2.80	1.80	1.00
Tension/Annoyance	0.67	0.00	0.33	0.00	0.00	0.00
Challenge	1.20	1.60	2.40	2.20	1.20	0.60
Negative Affect	1.00	0.25	0.00	0.00	0.25	2.50
Positive Affect	3.00	4.00	3.00	3.60	3.20	2.20

Table B.7: The individual play-tester's means for each component in the Game Experience Questionnaire for the PCG dungeon in the second iteration. Grey colour is used to show which play-tester did the PCG dungeon first.

GEQ Components	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**
Competence	2.00	3.60	1.40	2.60	3.00	2.00
Sensory and Imaginative Immersion	1.50	3.33	2.83	3.5	2.33	1.00
Flow	3.20	3.60	3.20	2.60	1.80	1.20
Tension/Annoyance	2.33	1.00	2.00	0.33	1.00	0.67
Challenge	1.20	0.80	3.00	1.60	0.20	0.40
Negative Affect	1.25	0.25	1.00	0.00	0.75	2.50
Positive Affect	3.00	4.00	2.80	3.60	3.60	2.00

Table B.8: The individual answers provided in the game experience questionnaire in the second iteration for the handmade dungeon. Grey colour is used to show which play-testers did the PCG dungeon first.

Item	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**
I felt content	3	4	3	3	3	2
I felt skilful	3	2	4	2	3	2
I was interested in the game's story	0	2	1	3	2	1
I thought it was fun	3	4	3	4	3	2
I was fully occupied with the game	4	4	3	4	3	1
I felt happy	3	4	3	3	3	3
It gave me a bad mood	1	0	0	0	0	1
I thought about other things	0	1	0	0	1	3
I found it tiresome	2	0	0	0	0	3
I felt competent	3	2	2	3	3	1
I thought it was hard	3	1	2	2	3	1

It was aesthetically pleasing	3	4	3	3	3	2
I forgot everything around me	2	3	2	2	2	1
I felt good	3	4	3	4	3	2
I was good at it	2	3	2	2	3	3
I felt bored	1	0	0	0	0	3
I felt successful	3	3	3	3	4	3
I felt imaginative	1	3	2	3	2	1
I felt that I could explore things	1	4	3	4	4	1
I enjoyed it	3	4	3	4	4	2
I was fast at reaching the game's targets	3	4	2	3	3	1
I felt annoyed	1	0	1	0	0	0
I felt pressured	0	2	2	3	0	0
I felt irritable	0	0	0	0	0	0
I lost track of time	4	4	3	3	1	1
I felt challenged	2	4	3	3	2	1
I found it impressive	1	4	3	4	3	1
I was deeply concentrated in the game	4	4	4	3	2	1
I felt frustrated	1	0	0	0	0	0
It felt like a rich experience	2	4	2	3	2	1
I lost connection with the outside world	4	2	3	2	1	1
I felt time pressure	0	0	2	0	0	0
I had to put a lot of effort into it	1	1	3	3	1	1

Table B.9: The individual answers provided in the game experience questionnaire in the second iteration for the PCG dungeon. Grey colour is used to show which play-testers did the PCG dungeon first.

Item	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**
I felt content	3	4	2	3	3	1
I felt skilful	1	4	2	3	3	3
I was interested in the game's story	0	2	1	3	2	0
I thought it was fun	3	4	3	4	4	1
I was fully occupied with the game	4	4	4	3	3	1
I felt happy	3	4	2	4	4	3
It gave me a bad mood	2	0	2	0	0	1
I thought about other things	1	0	0	0	2	3
I found it tiresome	1	1	2	0	1	3
I felt competent	3	4	2	2	3	3
I thought it was hard	2	1	3	3	1	1
It was aesthetically pleasing	3	4	4	3	2	2
I forgot everything around me	3	3	4	2	0	1
I felt good	3	4	4	3	3	3
I was good at it	2	4	1	3	3	2
I felt bored	1	0	0	0	0	3

I felt successful	2	4	2	3	3	1
I felt imaginative	1	2	2	3	2	1
I felt that I could explore things	1	4	4	4	3	1
I enjoyed it	3	4	3	4	4	2
I was fast at reaching the game's targets	2	2	0	2	3	1
I felt annoyed	3	1	2	1	1	1
I felt pressured	1	0	3	0	0	0
I felt irritable	2	0	2	0	0	1
I lost track of time	3	4	3	3	2	2
I felt challenged	3	2	3	3	0	0
I found it impressive	2	4	3	4	3	1
I was deeply concentrated in the game	3	4	3	3	2	1
I felt frustrated	2	2	2	0	2	0
It felt like a rich experience	2	4	3	4	2	1
I lost connection with the outside world	3	3	2	2	2	1
I felt time pressure	0	0	2	0	0	0
I had to put a lot of effort into it	0	1	4	2	0	1

Table B.10: The collected data associated with weapon statistics for the second iteration. In the second column, HM is the handmade dungeon and PCG is the procedurally generated dungeon. Next to the abbreviation, (first) is used to denote which dungeon was first played. To further emphasise, grey colour is used to show which play-testers did the PCG dungeon first.

Play-tester	Dungeon Type	Weapon Name	First Time Equipped	Equip Count
P7	HM	PlayerWoodenDagger	41	1
P7	HM	PlayerWoodenSword	42	1
P7	HM	PlayerWoodenSpear	59	1
P7	HM	PlayerIronAxe	80	1
P7	HM	PlayerIronSword	202	1
P7	HM	PlayerGoldenDagger	246	1
P7	HM	PlayerGoldenAxe	271	1
P7	HM	PlayerGoldenSword	357	1
P7	HM	PlayerGoldenSpear	419	2
P7	HM	PlayerDiamondAxe	485	1
P7	HM	PlayerDiamondSpear	568	4
P7	HM	PlayerDiamondDagger	576	4
P7	PCG(first)	PlayerWoodenAxe	121	1
P7	PCG(first)	PlayerWoodenSword	122	2
P7	PCG(first)	PlayerIronSpear	139	16
P7	PCG(first)	PlayerIronAxe	154	1
P7	PCG(first)	PlayerWoodenDagger	182	3
P7	PCG(first)	PlayerIronDagger	373	2

P7	PCG(first)	PlayerWoodenSpear	392	1
P7	PCG(first)	PlayerGoldenDagger	452	6
P7	PCG(first)	PlayerGoldenSword	524	7
P7	PCG(first)	PlayerGoldenSpear	800	5
P7	PCG(first)	PlayerDiamondSword	944	4
P8	HM(first)	PlayerWoodenDagger	33	2
P8	HM(first)	PlayerWoodenSword	36	1
P8	HM(first)	PlayerWoodenSpear	82	4
P8	HM(first)	PlayerIronAxe	164	5
P8	HM(first)	PlayerIronDagger	170	3
P8	HM(first)	PlayerIronSword	294	3
P8	HM(first)	PlayerGoldenAxe	435	2
P8	HM(first)	PlayerGoldenSword	450	1
P8	HM(first)	PlayerGoldenSpear	567	1
P8	HM(first)	PlayerDiamondSword	610	4
P8	HM(first)	PlayerDiamondSpear	680	1
P8	HM(first)	PlayerDiamondDagger	691	2
P8	PCG	PlayerWoodenSpear	28	8
P8	PCG	PlayerWoodenSword	29	1
P8	PCG	PlayerWoodenAxe	75	1
P8	PCG	PlayerIronDagger	94	3
P8	PCG	PlayerWoodenDagger	431	1
P8	PCG	PlayerIronAxe	650	1
P8	PCG	PlayerIronSword	759	2
P8	PCG	PlayerGoldenDagger	895	22
P8	PCG	PlayerGoldenAxe	1100	4
P8	PCG	PlayerGoldenSword	1208	8
P8	PCG	PlayerDiamondSword	1268	9
P8	PCG	PlayerDiamondSpear	1423	3
P9	HM(first)	PlayerWoodenDagger	21	5
P9	HM(first)	PlayerWoodenSword	21	2
P9	HM(first)	PlayerWoodenSpear	84	4
P9	HM(first)	PlayerIronDagger	191	3
P9	HM(first)	PlayerIronAxe	195	4
P9	HM(first)	PlayerIronSword	255	2
P9	HM(first)	PlayerGoldenAxe	364	1
P9	HM(first)	PlayerGoldenDagger	366	4
P9	HM(first)	PlayerGoldenSpear	369	3
P9	HM(first)	PlayerGoldenSword	373	1
P9	HM(first)	PlayerDiamondSword	454	1
P9	HM(first)	PlayerDiamondSpear	514	2
P9	HM(first)	PlayerDiamondDagger	534	2
P9	PCG	PlayerWoodenSpear	15	3
P9	PCG	PlayerWoodenSword	17	1

P9	PCG	PlayerIronDagger	110	13
P9	PCG	PlayerIronSword	151	4
P9	PCG	PlayerWoodenDagger	157	3
P9	PCG	PlayerIronAxe	766	3
P9	PCG	PlayerIronSpear	882	1
P9	PCG	PlayerGoldenSword	910	3
P9	PCG	PlayerGoldenDagger	988	10
P9	PCG	PlayerGoldenSpear	1010	1
P9	PCG	PlayerGoldenAxe	1130	1
P9	PCG	PlayerDiamondSpear	1275	3
P9	PCG	PlayerDiamondDagger	1477	3
P9	PCG	PlayerDiamondSword	1524	1
P10	HM(first)	PlayerWoodenDagger	27	4
P10	HM(first)	PlayerWoodenSword	29	2
P10	HM(first)	PlayerWoodenSpear	81	1
P10	HM(first)	PlayerIronAxe	129	4
P10	HM(first)	PlayerIronDagger	132	3
P10	HM(first)	PlayerIronSword	229	4
P10	HM(first)	PlayerGoldenSword	338	1
P10	HM(first)	PlayerGoldenSpear	340	2
P10	HM(first)	PlayerGoldenDagger	343	2
P10	HM(first)	PlayerGoldenAxe	346	1
P10	HM(first)	PlayerDiamondSword	430	1
P10	HM(first)	PlayerDiamondSpear	521	2
P10	HM(first)	PlayerDiamondDagger	535	2
P10	PCG	PlayerWoodenSpear	8	3
P10	PCG	PlayerIronSpear	59	9
P10	PCG	PlayerWoodenAxe	73	2
P10	PCG	PlayerWoodenDagger	101	2
P10	PCG	PlayerIronSword	233	5
P10	PCG	PlayerGoldenSword	385	5
P10	PCG	PlayerGoldenDagger	639	3
P10	PCG	PlayerDiamondAxe	700	3
P10	PCG	PlayerGoldenAxe	842	1
P10	PCG	PlayerDiamondSword	958	2
P11	HM	PlayerWoodenDagger	23	5
P11	HM	PlayerWoodenSword	24	2
P11	HM	PlayerWoodenSpear	85	1
P11	HM	PlayerIronDagger	137	5
P11	HM	PlayerIronAxe	140	1
P11	HM	PlayerIronSword	227	2
P11	HM	PlayerGoldenDagger	330	4
P11	HM	PlayerGoldenSpear	492	1
P11	HM	PlayerDiamondAxe	541	1

P11	HM	PlayerDiamondSword	596	5
P11	HM	PlayerDiamondSpear	659	1
P11	HM	PlayerDiamondDagger	670	3
P11	PCG(first)	PlayerIronAxe	18	14
P11	PCG(first)	PlayerWoodenDagger	39	3
P11	PCG(first)	PlayerWoodenAxe	120	1
P11	PCG(first)	PlayerWoodenSpear	143	3
P11	PCG(first)	PlayerIronSword	699	17
P11	PCG(first)	PlayerIronSpear	708	4
P11	PCG(first)	PlayerGoldenDagger	1039	7
P11	PCG(first)	PlayerIronDagger	1272	1
P11	PCG(first)	PlayerDiamondDagger	1310	9
P11	PCG(first)	PlayerDiamondAxe	1377	1
P11	PCG(first)	PlayerGoldenAxe	1591	1
P11	PCG(first)	PlayerGoldenSword	1631	2
P11	PCG(first)	PlayerGoldenSpear	1634	1
P12	HM	PlayerWoodenDagger	54	1
P12	HM	PlayerWoodenSword	56	1
P12	HM	PlayerWoodenSpear	70	4
P12	HM	PlayerIronDagger	112	1
P12	HM	PlayerIronSword	257	2
P12	HM	PlayerGoldenDagger	351	2
P12	HM	PlayerGoldenAxe	382	1
P12	HM	PlayerGoldenSword	412	1
P12	HM	PlayerGoldenSpear	557	2
P12	HM	PlayerDiamondSword	647	1
P12	HM	PlayerDiamondSpear	700	1
P12	PCG(first)	PlayerWoodenAxe	28	16
P12	PCG(first)	PlayerWoodenSword	34	3
P12	PCG(first)	PlayerWoodenDagger	156	3
P12	PCG(first)	PlayerIronSpear	188	12
P12	PCG(first)	PlayerIronAxe	225	6
P12	PCG(first)	PlayerWoodenSpear	379	1
P12	PCG(first)	PlayerGoldenSpear	480	12
P12	PCG(first)	PlayerIronSword	653	6
P12	PCG(first)	PlayerGoldenDagger	784	3
P12	PCG(first)	PlayerGoldenAxe	813	2
P12	PCG(first)	PlayerDiamondAxe	946	1
P12	PCG(first)	PlayerDiamondDagger	1042	1
P12	PCG(first)	PlayerDiamondSpear	1086	1

C

Use of Generative AI

Generative AI was used in the development of the game, as well as in the writing of this report. For development, tools such as ChatGPT and GitHub Copilot were used to debug, generate and refine code. For the report, ChatGPT was used to refine sections of the text as well as aid in our research, all of which was verified with trusted sources.