Word2Minecraft: Generating 3D Game Levels through Large Language Models

THESIS

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

MASTER OF SCIENCE (Computer Engineering)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Shuo Huang

May 2025

Word2Minecraft: Generating 3D Game Levels through Large Language Models

THESIS

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

MASTER OF SCIENCE (Computer Engineering)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Shuo Huang

May 2025

Approved:

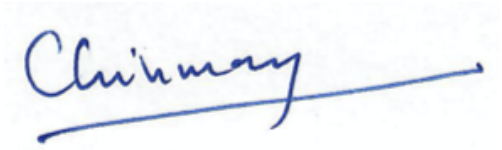
Department Chair Signature

April 28, 2025

Date

Approved by the Guidance Committee:

Major: MS Computer Engineering



Chinmay Hedge
Associate Professor of
NYU Tandon School of Engineering
04/15/2025
Date

Julian Togelius
Associate Professor of
NYU Tandon School of Engineering
04/16/2025
Date

Chen Feng
Associate Professor of
NYU Tandon School of Engineering
04/16/2025
Date

Microfilm or other copies of this thesis are obtainable from

UMI Dissertation Publishing

ProQuest CSA

789 E. Eisenhower Parkway

P.O. Box 1346

Ann Arbor, MI 48106-1346

Vita

Shuo Huang was born in Fuzhou, Fujian, China.

0.1 Education

New York University, New York, NY, USA, *M.S. in Computer Engineering*, Sep 2023 – May 2025

Southern University of Science and Technology, Shenzhen, China, *B.E. in Computer Science and Engineering*, Sep 2019 – Jun 2023

0.2 Thesis

Since September 2024, I have been working on my graduation thesis under the supervision of Professors Julian Togelius and Chinmay Hegde. During this period, I have conducted my research using the resources provided by the Game Innovation Lab.

ABSTRACT

Word2Minecraft: Generating 3D Game Levels through Large Language Models

by

Shuo Huang

Advisor: Prof. Chinmay Hegde, Ph.D.

Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science (Computer Engineering)

May 2025

We present *Word2Minecraft*, a system that leverages large language models to generate playable game levels in Minecraft based on structured stories. The system transforms narrative elements—such as protagonist goals, antagonist challenges, and environmental settings—into game levels with both spatial and gameplay constraints. We introduce a flexible framework that allows for the customization of story complexity, enabling dynamic level generation. The system employs a scaling algorithm to maintain spatial consistency while adapting key game elements. We evaluate *Word2Minecraft* using both metric-based and human-based methods. Our results show that GPT-4-Turbo outperforms GPT-4o-Mini in most areas, including story coherence and objective enjoyment, while the latter excels in

aesthetic appeal. We also demonstrate the system's ability to generate levels with high map enjoyment, offering a promising step forward in the intersection of story generation and game design. We open-source the code at https://github.com/JMZ-kk/Word2Minecraft/tree/word2mc_v0

Table of Contents

Vita iv
0.1 Education iv
0.2 Thesis iv
Abstract v
List of Figures ix
List of Tables x

1 Introduction **1**

2 Background **3**

3 Methodology **6**
3.1 Story generation using LLMs 6
3.2 Initial map generation 7
3.3 Adaptive tile scaling for enhanced realism 7
3.4 Sub-map generation 8

4 Experiments **13**
4.1 Metric-based story coherence evaluation 14
4.1.1 Direct coherence evaluation 14

4.1.2 Reconstructed story-based similarity analysis 14
4.2 Human-based story coherence evaluation 15
4.2.1 User study for story coherence of the main map 15
4.2.2 User study for story coherence of the sub-map 15
4.2.3 User study for level similarity 16
4.3 Diversity 17
4.4 Enjoyment 18
4.4.1 Map enjoyment 18
4.4.2 Objective enjoyment 19
4.5 Aesthetic and functionality 20
4.6 Ablation Study 21

5 Conclusions and Future Work **24**

A Important LLM prompt **26**
A.1 Story generation 26
A.2 Character extraction 26
A.3 Tile extraction 27
A.4 Tile-character mapping generation 27
A.5 Goal extraction 27
A.6 Goal classification 27
A.7 Walkable tile extraction 28
A.8 Objective tile extraction 28
A.9 World generation 28
A.10 Objective Placement 29
A.11 Scaling tiles selection 29

List of Figures

2.1 A flowchart depicting the complete *Word2Minecraft* pipeline. The LLM is used at every stage of the process. 4

3.1 Tile type array representation: 0 represents walkable tiles, 1 represents unwalkable tiles, 2 represents objective-related tiles, 3 represents tiles need to be scaled, and 4 represents already scaled tiles. This map will be updated by Algorithm 1. 10

3.2 LLM-generated buildings, each expanded from a tile labelled "Illusory Object Tile." 11

3.3 Examples of objective-oriented quests. 12

4.1 Reconstructed story-based similarity analysis 15

4.2 Ablation study 22

List of Tables

4.1 Metric-based story coherence evaluation (Mean & Standard Deviation) 14
4.2 Human-based story coherence evaluation 17
4.3 Metric-based tile variety evaluation (Mean & Standard Deviation) . 17
4.4 Metric-based enjoyment evaluation (Mean & Standard Deviation) . 20
4.5 Human-based Aesthetic and functionality evaluation 21
4.6 Ablation study results 23

Chapter 1

Introduction

Procedural content generation (PCG) is a widely used technique in game development that reduces the manual effort required for content creation while providing dynamic and diverse gameplay experiences. Story-driven level generation [1], a subfield of PCG, focuses on designing game levels where the narrative plays a central role in shaping their structure [2]. However, traditional PCG methods for generating such levels typically rely on predefined rule-based algorithms, which often struggle to produce coherent and contextually meaningful game environments [3, 4].

Recent advances in Large language models (LLMs) have introduced new approaches to game content generation [5]. Given their strong ability to understand and produce human-like text [6], LLMs are well-suited for story generation. Game developers can leverage these models to create diverse narratives and design levels that align with them, highlighting a clear opportunity to bridge story generation with procedural level design.

Minecraft [7], a widely popular 3D sandbox game, serves as an ideal plat-

form for procedural level generation. However, translating stories into structured and playable Minecraft levels remains a challenging problem, requiring an intelligent mapping between language and level design. To address this challenge, we present *Word2Minecraft*, an LLM-based system for generating Minecraft levels from structured stories—sequences of events constrained by spatial and gameplay requirements. The system first produces narratives and then translates them into playable game levels. To ensure spatial consistency, *Word2Minecraft* incorporates a scaling algorithm that dynamically adjusts the size of key game elements.

Both metric-based methods and human-based methods are used to assess *Word2Minecraft*. We test our system using two different LLMs, GPT-4-Turbo and GPT-4o-Mini, to compare their performance in story coherence, diversity, map enjoyment, objective enjoyment, aesthetic and functionality. Results indicate that GPT-4-Turbo outperforms GPT-4o-Mini in story coherence, diversity, objective enjoyment, and functionality. However, GPT-4o-Mini generates levels with greater aesthetic appeal, likely due to its tendency to use a more visually diverse selection of blocks. Both models perform similarly in map enjoyment.

The remainder of this paper is organized as follows. Section II reviews related work. Section III details the *Word2Minecraft* system, covering structured story generation with predefined elements (III-A), the main map generation process (III-B and III-C), and sub-map design (III-D). Section IV outlines the experimental setup, while Section V presents the results. Finally, Section VI concludes the paper and discusses directions for future work.

Chapter 2

Background

As a system that starts from a prompt, extracts all relevant information, and generates a playable game in Minecraft, *Word2Minecraft* builds on prior research in generating games from stories. Ammanabrolu et al. [8] employ knowledge graphs to determine object placement, while Fan et al. [9] train a neural network on the Light dataset [10] to position objects within a game. Hartsook et al. [4] apply search-based optimization [11] to construct a graph that dictates object placement based on story elements. Our work extends Word2World [12], a system that generates a story using an LLM and extracts the necessary information to create simplistic 2D role-playing games. We advance this approach by adapting the generated game into a complex 3D Minecraft environment, addressing limitations in Word2World through a scaling algorithm (discussed in Section **??**) and introducing sub-map generation to better handle different objectives.

Word2Minecraft can be viewed as a system for procedural content generation (PCG) [13], a field that has seen significant advancements through deep learning and evolutionary algorithms [11, 14, 15]. In previous research, level generation has

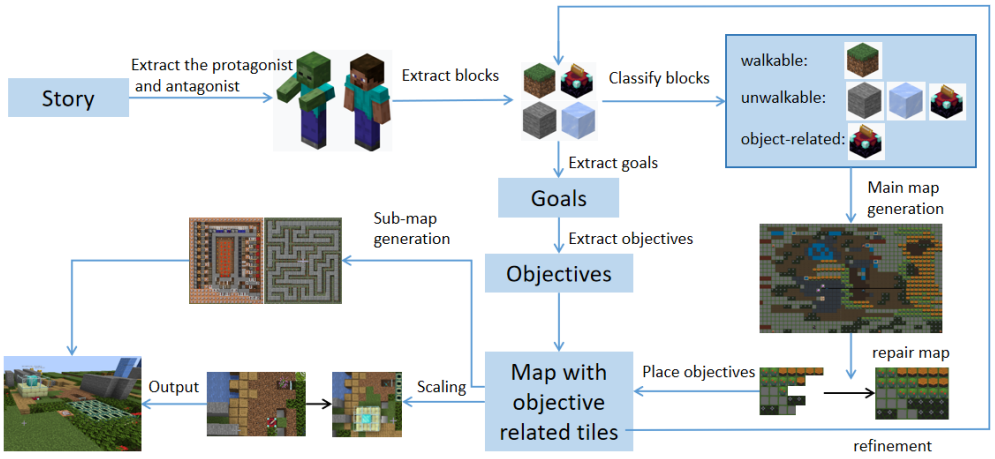


Figure 2.1: A flowchart depicting the complete *Word2Minecraft* pipeline. The LLM is used at every stage of the process.

often been treated as a sequence generation problem, leading to the use of sequence-based models like LSTMs [16]. Building on this, transformer-based architectures [17] have demonstrated strong potential in PCG. For instance, Todd et al. [18] utilize GPT-2 and GPT-3 for Sokoban [19] level generation through supervised learning, while Sudhakaran et al. [20] fine-tune GPT-2 to generate Mario Bros. [21] levels, incorporating novelty search [22] for diversity. Additionally, Nasir and Togelius [23] show that GPT-3 can generate novel levels using human-in-the-loop and bootstrapping methods [24] with minimal human-designed data. Minecraft has also been explored as a 3D PCG environment for LLMs and text-guided generation [25, 26, 27].

Beyond PCG, *Word2Minecraft* falls under the broader scope of automated game generation (AGG) [28]. Other AGG approaches include ANGELINA [29, 30], which uses evolutionary algorithms to create puzzle and platformer games, and Genie/Genie-2 [31, 32], which trains large models on internet videos to generate platformer and 3D games from text prompts. *Word2Minecraft* also aligns with com-

putational game creativity [33, 34], as each stage of game generation—storytelling, character and tile description, and world construction—requires creative decision-making.

Chapter 3

Methodology

This section introduces *Word2Minecraft*, a system that transforms stories into playable Minecraft levels. Each level consists of a main map and multiple sub-maps, generated through a step-by-step process. The following discussion details each stage of this process, with Figure 2.1 providing an overview of the framework.

3.1 Story generation using LLMs

The level generation process begins with story creation. Since LLM-generated stories can vary widely, we impose constraints to ensure they contain the necessary elements for our games. Each story must feature a protagonist pursuing a goal, an antagonist attempting to thwart them, and additional NPCs if required. It should also establish a clear environment and include multiple objectives for the protagonist, one of which must involve defeating the antagonist. In our system, users can customize both the number of story paragraphs and the number of objectives, providing greater control over story complexity.

3.2 Initial map generation

The main game map serves as the foundation for each level. We use the LLM to generate a 2D layout of this map, ensuring it aligns with the narrative environment and objectives. Each tile in the 2D layout represents general objects or terrain features (e.g., trees or rivers), rather than specific Minecraft blocks. In the subsequent step, the LLM translates these general tiles into corresponding Minecraft blocks.

The LLM extracts key elements from the story—including basic information about the protagonist and antagonist, required tiles, walkable tiles, and objectives—to generate the tile-based main map layout. Additionally, it creates a tile-character mapping for the next step, assigning a unique character to each tile type. Based on this information, the LLM generates an initial world layout represented as a tile-based grid. It then positions the protagonist, antagonist, and interactive objects on the map, assigning each objective to its corresponding location. Finally, the map is padded to form a rectangle.

After generating the world, both the LLM and traditional evaluation methods are used to assess its navigability, balance, and overall quality. Walkable paths are evaluated using the A* algorithm [35] to ensure that all objectives are reachable. During refinement, previous iterations of generated maps serve as references, enabling the model to learn from past designs.

3.3 Adaptive tile scaling for enhanced realism

The initial map generation suffers from a fatal scaling issue that significantly limits realism: each element, regardless of its real-world dimensions, is represented

by a single tile. This limitation affects both aesthetics and realism, as large objects such as houses or mountains are unnaturally compressed. To address this challenge, we introduce a scaling algorithm that selectively expands certain tiles beyond their original size, enhancing both visual realism and the integrity of the game world.

The first step in this process involves using the LLM to analyze the story and main map, identifying which tiles should be expanded and determining their target sizes. Each tile is categorized as walkable, unwalkable, objective-related, or already scaled, as depicted in Figure 3.1. Next, we apply Algorithm 1 to select the highest-scoring valid tile for expansion. A tile is considered valid if scaling it would not overlap objective-related tiles or previously expanded tiles. The score for each candidate tile is calculated based on the summed occurrence frequencies of tiles it would cover in the original map, following the principle of "rarity adds value": covering scarce tiles could disrupt the environment coherence, whereas overlapping common tiles like grass minimally impacts the overall map integrity.

These expanded tiles are then replaced with larger structures generated by the LLM, taking into account the story context and the descriptions of the original tiles. We note that a key property of the LLM is its ability to introduce diversity; even for identical tile types, it produces stylistically consistent yet visually distinct buildings, as illustrated in Figure 3.2.

3.4 Sub-map generation

Prior work [36] categorizes game quests into three fundamental types: place-oriented, time-oriented, and objective-oriented. We further divide objective-oriented quests into five specific types: "Defeat the Enemy", "Chat with NPC", "Exit Maze",

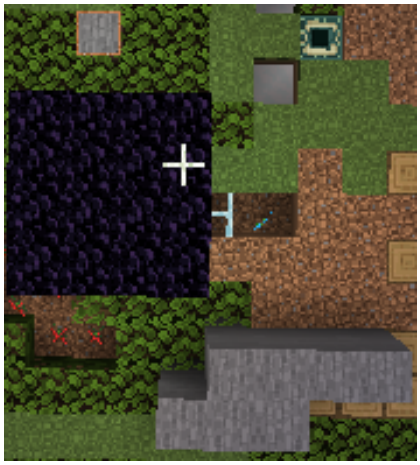
Algorithm 1 Scaling algorithm

Data: Main map layout M, List of tiles requiring scaling C, Dictionary of scaling sizes $D = \{c_i \mapsto s_i\}$, where c_i is a tile character and s_i is its scaling size

```
for i = 1 to |M| do
    for j = 1 to |M[0]| do
        if M[i][j] in C then
            t ← M[i][j] // Target tile
            s ← D[t] // scaling size
            best_m ← −1, best_n ← −1 // Best top-left corner
            best_score ← 0
             for m = i − s + 1 to i do
                for n = j − s + 1 to j do
                    score ← calculate score for s × s block with (m, n) as top-left
                     corner if score > best_score then
                        best_score ← score
                         best_m ← m
                         best_n ← n
                    end
                end
            end
            if best_m! = −1 then
                update the map
            end
        end
    end
end
```

0	2	0	0	0	0	3	0
0	0	0	0	0	0	0	0
4	4	4	4	0	0	0	0
4	4	4	4	0	0	0	0
4	4	4	4	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1
0	0	0	1	1	1	0	0

(a) Tile type array



(b) Corresponding Minecraft level

Figure 3.1: Tile type array representation: 0 represents walkable tiles, 1 represents unwalkable tiles, 2 represents objective-related tiles, 3 represents tiles need to be scaled, and 4 represents already scaled tiles. This map will be updated by Algorithm 1.

"Survive Waves of Enemies", and "Collect Items", as illustrated in Figure 3.3.

However, it is challenging to directly implement all objectives within the main map due to constraints in size and complexity. For instance, maze navigation requires structured layouts incompatible with the open-world style of the primary map. To address this, we introduce tailored sub-maps specifically for the "Exit Maze", "Survive Waves of Enemies", and "Collect Items" objectives using the following process:

1. **Objective Position Generation**: We prompt the LLM to identify a representative tile and determine its position on the main map for each objective.

2. **Position Adjustment**: Since LLM-generated coordinates may sometimes be invalid (e.g., placed within obstacles or mismatched tile types), we employ a BFS algorithm to locate the nearest valid position, and place a portal at that location instead.

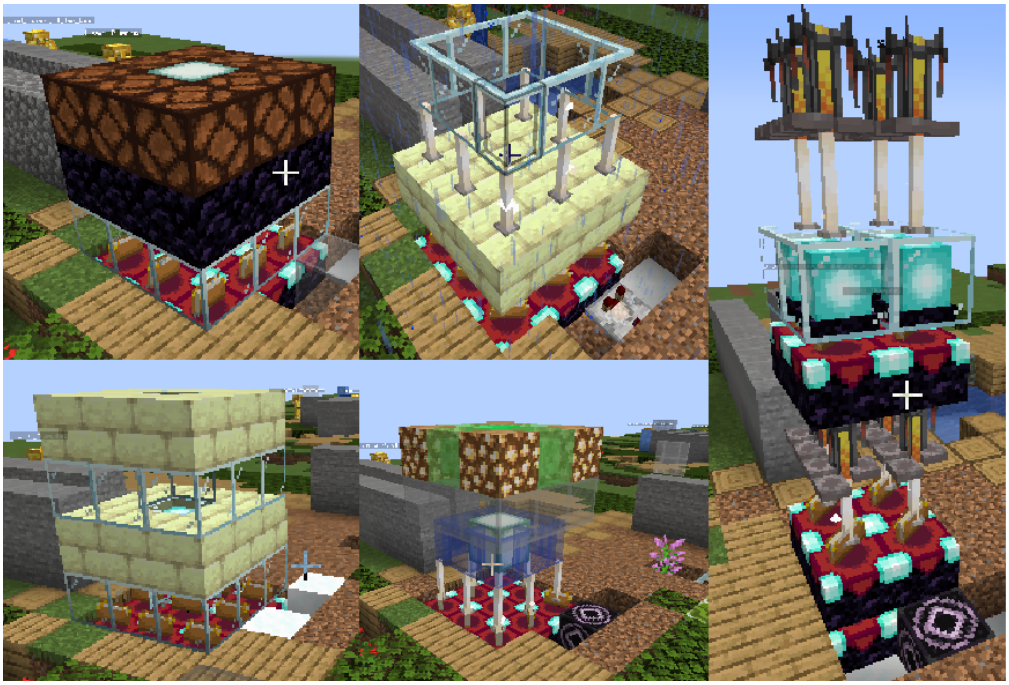
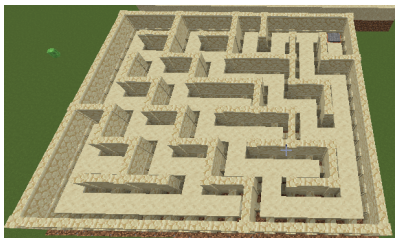


Figure 3.2: LLM-generated buildings, each expanded from a tile labelled "Illusory Object Tile."

3. **Sub-map Generation**: We generate structured sub-maps tailored to specific objectives. These sub-maps connect directly to the main map via the previously placed portals.

(a) Learn Old Farming Techniques from Señora Miro ("Chat with NPC")



(b) Navigate through the Sand Serpent Dunes ("Exit Maze")

(c) Collect samples from the Cactus Gates ("Collect Items")

(d) Dodge attacks from local wildlife ("Survive Waves of Enemies")

(e) Defeat Vorath to secure the Bloom of Night ("Defeat the Enemy")

Figure 3.3: Examples of objective-oriented quests.

Chapter 4

Experiments

For metric-based evaluation, we conduct experiments using two language models—GPT-4-Turbo and GPT-4o-Mini—to evaluate *Word2Minecraft*. Each model generates 30 game levels, which are then assessed based on story coherence, diversity, enjoyment, aesthetics, and functionality. For the enjoyment metric, we additionally compare these results against those produced by an evolutionary algorithm as a baseline.

We adopt hyperparameters similar to those used by Nasir et al. [12], configuring the game levels with a story comprising 4–5 paragraphs and assigning the protagonist 8 objectives to complete each game. The A* algorithm is limited to 1000 iterations when searching for paths to all objectives in the initial map.

For human-based evaluation, we conduct a user study to further assess GPT-4-Turbo's performance regarding story coherence, visual appearance, and functionality. A total of 17 participants with basic knowledge of Minecraft and Game AI took part in this study.

4.1 Metric-based story coherence evaluation

Story coherence measures how well the generated levels match their original stories. In the metric-based story coherence evaluation, we employ both direct coherence evaluation and reconstructed story-based similarity analysis.

4.1.1 Direct coherence evaluation

We provide the LLM with the original story, the tile-character mapping, and the character-based tile encoding of the level's main map. The model is then prompted to assess the coherence between the generated map and the story.

4.1.2 Reconstructed story-based similarity analysis

We extract the block-level representation of the level generated within Minecraft as a JSON file. Next, we prompt the LLM to generate a new story based on these extracted blocks. To quantify coherence, we compute the cosine similarity between the original and reconstructed stories using `text-embedding-3-large` [37], a high-dimensional embedding model that captures semantic closeness as shown in Figure 4.1.

Table 4.1 shows that GPT-4-Turbo has higher average coherence score and lower standard deviation, which implies that it is more stable and performs better in maintaining story coherence.

Table 4.1: Metric-based story coherence evaluation (Mean & Standard Deviation)

Model	Direct Coherence	Reconstructed Story Coherence
GPT-4-Turbo	84.47 ± 8.87	0.90 ± 0.01
GPT-4o-Mini	71.75 ± 13.99	0.84 ± 0.04

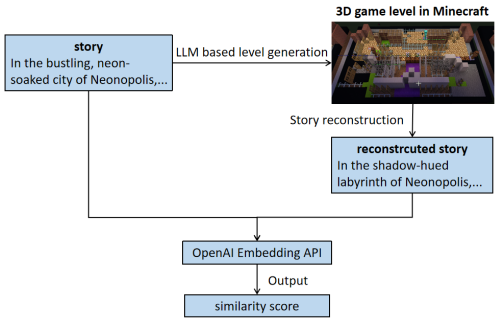


Figure 4.1: Reconstructed story-based similarity analysis

4.2 Human-based story coherence evaluation

4.2.1 User study for story coherence of the main map

We designed two questions to evaluate GPT-4-Turbo's ability to generate levels consistent with given stories. In each question, participants were presented with a short story and two top-down images of Minecraft levels: one generated from the provided story and another from a different story with a similar theme (e.g., both set in a forest). Participants were asked to identify which level matched the given story. This design specifically ensures that participants cannot rely solely on visual cues, as both maps share similar thematic elements, encouraging them to consider the deeper narrative and structural coherence instead. The accuracy of participants' choices shows how well our generated levels maintain coherence with the original story.

4.2.2 User study for story coherence of the sub-map

When evaluating the coherence between a sub-map and its corresponding objective, we offer participants the description of the objective, relevant part in the story, and two sub-maps. Both sub-maps are generated from different objectives of

this story, whose similar block compositions make this task challenging. Therefore, a high accuracy in this task would further indicate a strong alignment between the sub-map and its objective.

4.2.3 User study for level similarity

We maintain that the levels generated from the same story should follow a consistent distribution in terms of style and structure. In this experiment, each participant is presented with a base level and three candidate levels, among which only one level is generated from the same story as the base level.

The results are shown in Table 4.2. In both themes, sub-maps achieve higher coherence scores (0.82) compared to main maps (0.59 for forest and 0.47 for desert). A possible explanation is that main maps are designed to represent the entire story, which contains fewer details. In contrast, sub-maps focus on a specific objective, and participants only need to read a small portion of the story rather than processing the entire narrative.

We also find that the desert theme main map shows lower coherence than the forest one. This is due to the limited block variety in desert maps, where sand and stone dominate both answer choices, making it difficult to distinguish. Additionally, the correct map was smaller and structurally simpler compared to the distractor, which may also lead to the low accuracy in this map.

Overall, the level similarity accuracy is moderate (0.59), likely because perceived similarities in top-down views are influenced more by overall visual themes rather than specific narrative coherence.

Table 4.2: Human-based story coherence evaluation

Question	Accuracy
Forest theme main map	0.59
Forest theme sub map	0.82
Desert theme main map	0.47
Desert theme sub map	0.82
Level similarity	0.59

4.3 Diversity

Diversity is a crucial aspect of game-level design [38], as it enriches gameplay by providing players with varied experiences. In our experiments, we evaluate diversity through tile variety, a key metric that measures environmental richness. Specifically, we quantify tile variety using Shannon Entropy, defined as follows. Given a map containing $N \in \mathbb{N}_+$ distinct tile types, the Shannon Entropy (H) is calculated as shown in Equation (1), where p_i represents the proportion of the i^{th} tile type:

$$H = -\sum_{i=1}^{N} p_i \log_2 p_i. \tag{4.1}$$

Results from our evaluation, shown in Table 4.3, indicate that maps generated by GPT-4-Turbo exhibit greater diversity, reflected by higher tile variety and Shannon entropy, compared to those created by GPT-4o-Mini. This demonstrates that GPT-4-Turbo generates more varied and visually interesting game environments.

Table 4.3: Metric-based tile variety evaluation (Mean & Standard Deviation)

Metric	GPT-4-Turbo	GPT-4o-Mini
Tile Type Number	18.23 ± 5.70	12.95 ± 2.04
Shannon Entropy	4.06 ± 0.49	3.58 ± 0.33

4.4 Enjoyment

We measure the enjoyment of the levels from two aspects: map enjoyment and objective enjoyment. We also employ an evolutionary algorithm (EA) as the baseline to compare the LLM's performance with traditional PCG. For a fair comparison, the chromosome in the EA is set to a 15×15 grid-based map—the smallest size among the LLM-generated maps. The fitness is the average of the shortest paths from the starting position to all objective tiles (ASPAO)

4.4.1 Map enjoyment

Playability is the cornerstone of map enjoyment in any game. Although the A* algorithm is used to ensure that the protagonist can access all objectives, scaling can block some original paths. We applied a BFS check after scaling to verify that all objective-related tiles on the map remain connected by at least one valid path.

As long as the map remains valid, we assume that a high proportion of unwalkable tiles relative to the total area indicates high map enjoyment, as a greater density of unwalkable tiles forces players to strategize their movement and navigate dynamic obstacles. Therefore, we use the valid unwalkable tiles ratio (VUTR) instead of the unwalkable tiles ratio (UTR) to represent map enjoyment.

We define $\mathcal{A}_m$ as the area of map m and $\mathcal{U}_m$ as the area of unwalkable tiles in the map m. Assuming we have M maps, UTR and $VUTR$ can be modeled as (2) and (3), where $C_m \in \{0, 1\}$ denotes the validity of map m, with $C_m = 1$ if map m is valid.

$$UTR = \sum_{m=1}^{M} \frac{\mathcal{U}_m}{\mathcal{A}_m} \tag{4.2}$$

$$VUTR = \frac{\sum_{m=1}^{M} C_m \cdot \mathcal{U}_m}{\sum_{m=1}^{M} \mathcal{A}_m} \tag{4.3}$$

4.4.2 Objective enjoyment

While a high proportion of unwalkable tiles generally represent enjoyment, this metric may fail in certain extreme cases. For instance, if a large number of unwalkable tiles are set at the edges of the map while the objectives are concentrated at the center, the actual challenge for players may remain low.

The average of the shortest paths from the starting position to all objective tiles (ASPAO) among valid maps is a key metric for evaluating the objective enjoyment of a game level, as the player is teleported back to the starting position after completing an objective. A higher average distance suggests a more complex level, requiring players to traverse longer distances to reach objectives. ASPAO can be modeled in (4) and (5), where $d(s, o)$ is the shortest distance from the starting point s to objective tile o, and $P(s, o)$ is the set of shortest paths.

$$ASPAO = \frac{1}{|O|} \sum_{o \in O} d(s, o) \tag{4.4}$$

$$d(s, o) = \min_{\pi \in P(s,o)} |\pi| \tag{4.5}$$

We compare the enjoyment levels of the two LLM models and EA, with the results presented in Table 4.4. We observe that EA achieves the best performance across all metrics, which is expected since it optimizes solely for ASPAO rather than trying to make a balance between ASPAO and other metrics like story coherence. This direct optimization also improves the final VUTR by increasing the ratio of

unwalkable tiles, thereby enhancing level playability. These findings highlight the significant gap between LLM and EA in terms of enjoyment.

Additionally, we find that GPT-4o-Mini achieves a higher playability score (0.87) compared to GPT-4-Turbo (0.82) while maintaining a lower UTR. This suggests that GPT-4-Turbo adopts a more aggressive scaling strategy when generating levels, potentially blocking important paths and increasing the proportion of unreachable objectives. As a result, the VUTR remains similar between the two models, indicating comparable map enjoyment. The higher ASPAO of GPT-4-Turbo further suggests that its generated objectives are more engaging.

Table 4.4: Metric-based enjoyment evaluation (Mean & Standard Deviation)

Metric	GPT-4-Turbo	GPT-4o-Mini	EA
Playability	0.83 ± 0.37	0.87 ± 0.35	**1 ± 0**
UTR	0.18 ± 0.16	0.14 ± 0.12	**0.40 ± 0.02**
VUTR	0.11 ± 0.13	0.12 ± 0.11	**0.40 ± 0.02**
ASPAO	22.41 ± 9.28	10.05 ± 5.66	**33.4 ± 1.85**

4.5 Aesthetic and functionality

To evaluate the aesthetic and functionality of levels generated by GPT-4-Turbo, we use GPT-4o-Mini as a baseline. We record a 2-3 minute gameplay video showcasing a generated level for each model. Participants are then asked to compare the two videos and rank the levels based on aesthetic appeal and functionality separately.

The results in Table 4.5 show that GPT-4-Turbo outperforms GPT-4o-Mini in functionality. However, participants perceive GPT-4o-Mini as having better aesthetics. We believe this may be due to GPT-4o-Mini selecting blocks with

weaker story coherence, leading to more contrasting block choices. For example, in a forest-themed level, GPT-4o-Mini incorporated lava blocks, resulting in a more colorful and visually striking map, which could appeal to participants' aesthetic preferences.

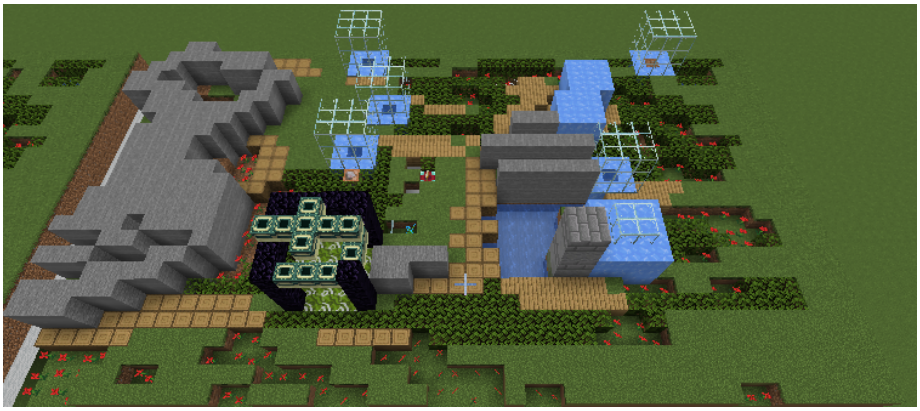
Table 4.5: Human-based Aesthetic and functionality evaluation

Metric	GPT-4-Turbo	GPT-4o-Mini
Aesthetic	4	**13**
Functionality	**11**	6

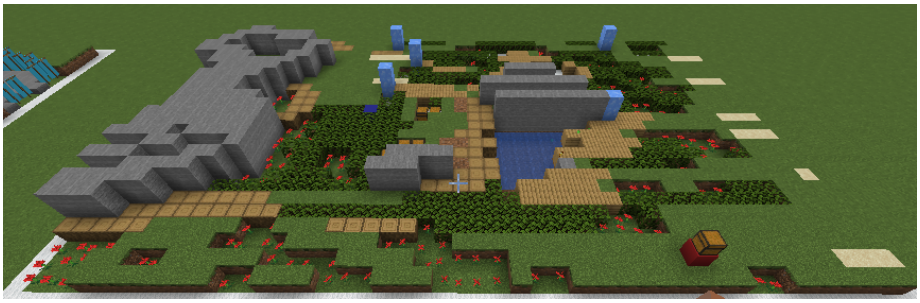
4.6 Ablation Study

We conduct an ablation study to determine whether the scaling algorithm or the LLM's block selection plays a more critical role in the post-processing phase after generating the initial map. We prepare four levels that share the same initial main map but differ in their post-processing methods: A (GPT-4-Turbo with scaling), B (GPT-4-Turbo without scaling), C (GPT-4o-Mini with scaling), and D (GPT-4o-Mini without scaling), as shown in Figure 4.2. Participants are asked to rank these levels based on their preference. The composite score S is calculated as a weighted average of the votes across different rankings, providing an overall assessment of the results. Given k ranking categories, where each rank i has a corresponding vote count R_i and weight w_i, the composite score S can be expressed as (6), where $\mathcal{N} := \sum_{i=1}^{k} R_i$ is the total number of votes. We set $k = 4$ and $w_i = k - i$ in our experiments.

$$S = \frac{\sum_{i=1}^{k} w_i R_i}{\mathcal{N}} \tag{4.6}$$



(a) Map A: GPT-4-Turbo with scaling



(b) Map B: GPT-4-Turbo without scaling

(c) Map C: GPT-4o-Mini with scaling

(d) Map D: GPT-4o-Mini without scaling

Figure 4.2: Ablation study

As shown in Table 4.6, Map A achieves the best performance. Additionally, we observe that Map B has a higher composite score than Map C, indicating that the LLM's block selection has a greater impact on the final quality of post-processed maps than the scaling algorithm.

Table 4.6: Ablation study results

Map	Composite score
Map A	**3.24**
Map B	2.76
Map C	2.29
Map D	1.35

Chapter 5

Conclusions and Future Work

In this paper, we introduce *Word2Minecraft*, a novel system that utilizes LLMs to generate structured and playable Minecraft levels from stories. Our approach addresses key challenges in story-driven PCG by incorporating a scaling algorithm to preserve spatial consistency and sub-map generation to support diverse objectives. Through both metric-based and human-based evaluations, we found that GPT-4-Turbo produces levels with higher story coherence, diversity, and functionality, while GPT-4o-Mini excels in aesthetic appeal. Our ablation study further highlighted the importance of block selection by the LLM in determining the final quality of generated levels.

In the future, we aim to improve level integration by merging all objectives into a single continuous map, rather than relying on portals to connect sub-maps. This will require a more advanced scaling algorithm. Additionally, our current maps assume uniform height for all walkable tiles, which does not fully utilize Minecraft's action space, including jumping and climbing mechanics. We plan to introduce more complex terrain variations to enhance verticality and traversal dynamics,

making levels more engaging and challenging.

Our work demonstrates the potential of LLMs in procedural content generation, particularly in translating narratives into structured, playable game environments. By further exploring hybrid approaches that combine LLMs with traditional PCG techniques, we aim to push the boundaries of story-driven level generation, making AI-generated game worlds more immersive, coherent, and engaging for players.

Appendix A

Important LLM prompt

A.1 Story generation

Write a 4-5 paragraph story which has characters including the protagonist trying to achieve something and the antagonist wanting to stop the protagonist. There should be 8 objectives for the protagonist in the story. One of them should be to defeat the antagonist somehow. You should describe the environments that those objectives happen. You can add some NPCs in the story.

A.2 Character extraction

Let's use the above story to create a 2D game. Write a specific description of each character which can be used as a prompt to generate sprites for the characters.

A.3 Tile extraction

Create an exhaustive list of tiles needed to create the environment. Some tile can occupy more than one space.

A.4 Tile-character mapping generation

Imagine each tile maps to an alphabet or a character. For environment, use alphabets and for characters use special characters. Create it in a single Python Dictionary style. Return only and only a Python Dictionary and nothing else in your response. Don't return it in a Python response. Names should be the Keys and alphabets or characters should be the Values. Protagonist should always strictly be '@' and the antagonist should always strictly be '#'.

A.5 Goal extraction

What is the main goal for protagonist of the story? What are the small goals for protagonist to achieve the main goal of the story? Also create rewards and penalties based on the goals for protagonist. Create a score for each reward or penalty.

A.6 Goal classification

You will be given a list of objectives: objective_list and each objectiveś goal belongs to at least one of the following categories: 'Defeat the enemy', 'Chat with NPC', 'Navigate the maze to find an exit', 'Survive waves of enemies', 'Collect

Items'. If an objective's goal does not fit into any of these categories, classify it as 'Other'. Only one onjective belongs to 'Defeat the enemy'. Your task is to match each objective to its corresponding category and return a python dictionary where the key is objective and the value is the corresponding category (or 'Other').

A.7 Walkable tile extraction

Considering the above goals that you extracted from the story and the following tileset create a Python list of the walkable tiles in the 2D tilemap world. Only return the walkable tiles.

A.8 Objective tile extraction

Considering the above goals that you extracted from the story and the following tileset, create a Python list of the characters of the object tiles that can be interacted with in the 2D tilemap world. Only return the object tiles that can be interacted with.

A.9 World generation

Using the following tile to character mapping:{tile_map_dict}. Create an entire world on a tile-based grid. Do not create things that would need more than one tile. Also, following characters are important to place:{important_tiles_list}, walkable tiles:{walkable_tiles_list}. Do not place the protagonist, the antagonist and the interactive objects of the story right now. Only create the world right now. Create it is a string format with three backticks to start and end with (“‘) and not in a

list format."

A.10 Objective Placement

You are a great planner in 2D game. You plan objectives for the protagonist of the game. All objectives should match the goals extracted from the story. One of them should be to defeat the antagonist somehow. Other objectives can be: finding the exit of a complex labyrinth, finding a chest in the map, surviving waves of enemies, or gathering some items in the map. Return a Python dictionary of the objective as the key and a tile that achieves the objective and the position of the tile. For example 'Objective': ['A', 6, 1]. Only return a Python dictionary. Do not return a python response.

A.11 Scaling tiles selection

Given the story, a 2D map {tile_map}, and a dictionary des2not where each key is a tileś description and each value is the notation for that tile in the map, identify which tile notations in the map need to be scaled? A tile is considered scaled if it should occupy more than one grid cell, such as a ́house ́tile. # Please avoid selecting adjacent tiles of the same type and frequent tiles. Please avoid scaling # and @ Return a Python list of tiles, formatted as a list of scaled tile notations (e.g., [a,b]])'

Bibliography

[1] H. Jenkins, "Game design as narrative architecture," *Computer*, vol. 44, no. 3, pp. 118–130, 2004.

[2] M. Cook, M. Eladhari, A. Nealen, M. Treanor, E. Boxerman, A. Jaffe, P. Sottosanti, and S. Swink, "Pcg-based game design patterns," *arXiv preprint arXiv:1610.03138*, 2016.

[3] M. Dahrén, "The usage of pcg techniques within different game genres," 2021.

[4] K. Hartsook, A. Zook, S. Das, and M. O. Riedl, "Toward supporting stories with procedurally generated game worlds," in *2011 IEEE Conference on Computational Intelligence and Games (CIG'11)*. IEEE, 2011, pp. 297–304.

[5] M. F. Maleki and R. Zhao, "Procedural content generation in games: A survey with insights on emerging llm integration," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 20, no. 1, 2024, pp. 167–178.

[6] P. J. Wang and M. Kreminski, "Guiding and diversifying llm-based story generation via answer set programming," *arXiv preprint arXiv:2406.00554*, 2024.

[7] S. C. Duncan, "Minecraft, beyond construction and survival," *Well Played: a journal on video games, value and meaning*, vol. 1, no. 1, pp. 1–22, 2011.

[8] P. Ammanabrolu, W. Cheung, D. Tu, W. Broniec, and M. Riedl, "Bringing stories alive: Generating interactive fiction worlds," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, no. 1, 2020, pp. 3–9.

[9] A. Fan, J. Urbanek, P. Ringshia, E. Dinan, E. Qian, S. Karamcheti, S. Prabhumoye, D. Kiela, T. Rocktaschel, A. Szlam *et al.*, "Generating interactive worlds with text," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 02, 2020, pp. 1693–1700.

[10] J. Urbanek, A. Fan, S. Karamcheti, S. Jain, S. Humeau, E. Dinan, T. Rocktäschel, D. Kiela, A. Szlam, and J. Weston, "Learning to speak and act in a fantasy text adventure game," *arXiv preprint arXiv:1903.03094*, 2019.

[11] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne, "Search-based procedural content generation: A taxonomy and survey," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 3, pp. 172–186, 2011.

[12] M. U. Nasir, S. James, and J. Togelius, "Word2world: Generating stories and worlds through large language models," *arXiv preprint arXiv:2405.06686*, 2024.

[13] N. Shaker, J. Togelius, and M. J. Nelson, "Procedural content generation in games," 2016.

[14] A. Summerville, S. Snodgrass, M. Guzdial, C. Holmgård, A. K. Hoover, A. Isaksen, A. Nealen, and J. Togelius, "Procedural content generation via machine learning (pcgml)," *IEEE Transactions on Games*, vol. 10, no. 3, pp. 257–270, 2018.

[15] J. Liu, S. Snodgrass, A. Khalifa, S. Risi, G. N. Yannakakis, and J. Togelius, "Deep learning for procedural content generation," *Neural Computing and Applications*, vol. 33, no. 1, pp. 19–37, 2021.

[16] A. Summerville and M. Mateas, "Super mario as a string: Platformer level generation via lstms," *arXiv preprint arXiv:1603.00930*, 2016.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[18] G. Todd, S. Earle, M. U. Nasir, M. C. Green, and J. Togelius, "Level generation through large language models," in *Proceedings of the 18th International Conference on the Foundations of Digital Games*, 2023, pp. 1–8.

[19] Y. Murase, H. Matsubara, and Y. Hiraga, "Automatic making of sokoban problems," in *PRICAI'96: Topics in Artificial Intelligence: 4th Pacific Rim International Conference on Artificial Intelligence Cairns, Australia, August 26–30, 1996 Proceedings 4*. Springer, 1996, pp. 592–600.

[20] S. Sudhakaran, M. González-Duque, M. Freiberger, C. Glanois, E. Najarro, and S. Risi, "Mariogpt: Open-ended text2level generation through large language models," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[21] J. Togelius, E. Kastbjerg, D. Schedl, and G. N. Yannakakis, "What is procedural content generation? mario on the borderline," in *Proceedings of the 2nd international workshop on procedural content generation in games*, 2011, pp. 1–6.

[22] J. Lehman and K. O. Stanley, "Novelty search and the problem with objectives," *Genetic programming theory and practice IX*, pp. 37–56, 2011.

[23] M. U. Nasir and J. Togelius, "Practical pcg through large language models," in *2023 IEEE Conference on Games (CoG)*. IEEE, 2023, pp. 1–4.

[24] R. R. Torrado, A. Khalifa, M. C. Green, N. Justesen, S. Risi, and J. Togelius, "Bootstrapping conditional gans for video game level generation," in *2020 IEEE Conference on Games (CoG)*. IEEE, 2020, pp. 41–48.

[25] M. Awiszus, F. Schubert, and B. Rosenhahn, "Wor (l) d-gan: toward natural-language-based pcg in minecraft," *IEEE Transactions on Games*, vol. 15, no. 2, pp. 182–192, 2022.

[26] M. Beukman, M. Fokam, M. Kruger, G. Axelrod, M. Nasir, B. Ingram, B. Rosman, and S. James, "Hierarchically composing level generators for the creation of complex structures," *IEEE Transactions on Games*, 2023.

[27] S. Earle, F. Kokkinos, Y. Nie, J. Togelius, and R. Raileanu, "Dreamcraft: Text-guided generation of functional 3d environments in minecraft," in *Proceedings of the 19th International Conference on the Foundations of Digital Games*, 2024, pp. 1–15.

[28] M. J. Nelson and M. Mateas, "Towards automated game design," in *AI* IA 2007: Artificial Intelligence and Human-Oriented Computing: 10th Congress*

of the Italian Association for Artificial Intelligence, Rome, Italy, September 10-13, 2007. Proceedings 10. Springer, 2007, pp. 626–637.

[29] M. Cook and S. Colton, "Multi-faceted evolution of simple arcade games," in *2011 IEEE Conference on Computational Intelligence and Games (CIG'11)*. IEEE, 2011, pp. 289–296.

[30] ——, "Angelina-coevolution in automated game design," *Computational Creativity*, vol. 228, 2012.

[31] J. Bruce, M. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps *et al.*, "Genie: Generative interactive environments," *arXiv preprint arXiv:2402.15391*, 2024.

[32] J. Parker-Holder, P. Ball, J. Bruce, V. Dasagi, K. Holsheimer, C. Kaplanis, A. Moufarek, G. Scully, J. Shar, J. Shi, S. Spencer, J. Yung, M. Dennis, S. Kenjeyev, S. Long, V. Mnih, H. Chan, M. Gazeau, B. Li, F. Pardo, L. Wang, L. Zhang, F. Besse, T. Harley, A. Mitenkova, J. Wang, J. Clune, D. Hassabis, R. Hadsell, A. Bolton, S. Singh, and T. Rocktäschel, "Genie 2: A large-scale foundation world model," 2024. [Online]. Available: https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/

[33] C. Browne and S. Colton, "Computational creativity in a closed game system," in *2012 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 2012, pp. 296–303.

[34] A. Liapis, G. N. Yannakakis, and J. Togelius, "Computational game creativity." ICCC, 2014.

[35] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[36] E. Aarseth, "From hunt the wumpus to everquest: introduction to quest theory," in *International Conference on Entertainment Computing*. Springer, 2005, pp. 496–506.

[37] L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, and F. Wei, "Improving text embeddings with large language models," *arXiv preprint arXiv:2401.00368*, 2023.

[38] Y. Li, Z. Wang, Q. Zhang, B. Yuan, and J. Liu, "Measuring diversity of game scenarios," *IEEE Transactions on Games*, 2025.

ProQuest Number: 31999388

INFORMATION TO ALL USERS
The quality and completeness of this reproduction is dependent on the quality
and completeness of the copy made available to ProQuest.

Distributed by
ProQuest LLC a part of Clarivate (2025).
Copyright of the Dissertation is held by the Author unless otherwise noted.

This work is protected against unauthorized copying under Title 17,
United States Code and other applicable copyright laws.

This work may be used in accordance with the terms of the Creative Commons license
or other rights statement, as indicated in the copyright statement or in the metadata
associated with this work. Unless otherwise specified in the copyright statement
or the metadata, all rights are reserved by the copyright holder.

ProQuest LLC
789 East Eisenhower Parkway
Ann Arbor, MI 48108 USA