Accredited SINTA 2 Ranking
Decree of the Director General of Higher Education, Research, and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online at: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 9 No. 2 (2025) 416 - 424 e-ISSN: 2580-0760

Combining the Cellular Automata and Marching Square to Generate Maps

Viore[1], Wirawan Istiono[2]*
[1,2]Department of Informatics, Faculty of Engineering and Informatics, Universitas Multimedia Nusantara, Indonesia
[1]viore@student.umn.ac.id, [2]wirawan.istiono@umn.ac.id

Abstract

As computer technology advances, one of the entertainment media that has emerged is video games. The development of a video game is becoming more expensive and labor-intensive as technology itself continues to grow. One of the characteristics of a game as an entertainment medium is its replay value, which refers to the fact that the subject matter can be played more than once. Automating content through the use of procedural content generation is done with the goal of lowering expenses and reducing the amount of labour that is required. This research has two goals: designing and developing a Maze Game using the Procedural Content Generation method with the Cellular Automata and Marching Square algorithms, and determining the level of player satisfaction with the games developed using the Game User Experience Satisfaction Scale (GUESS) method. This research will utilize Cellular Automata and the Marching Square algorithm as a method for generating 3D game shapes through Procedural Content Generation. After the game has been developed, it will be performed by players, and the Game User Experience Satisfaction Scale will be used to measure the user experience. The result for overall satisfaction, based on the responses of 25 respondents, is 83.14%. Cellular Automata was effectively implemented to generate the map, while Marching Square was used to generate the 3D mesh, albeit with isolated rooms and graphical errors.

Keywords: Cellular Automata; Combining Algorithms; Marching Square; Procedural Content Generation; Video Game

How to Cite: Viore and W. Istiono, "Combining the Cellular Automata and Marching Square to Generate Maps", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 9, no. 2, pp. 416 - 424, Apr. 2025.
DOI: https://doi.org/10.29207/resti.v9i2.6241

1. Introduction

Computer technology is not only used for labor but also for recreation, such as video games, in the current era. The genre of a video game can distinguish the manner in which it interacts with participants. In their function as entertainment, games contain something known as replay value, which determines whether or not the game can be played repeatedly. The design of game levels is a key component that can enhance the potential for a game to be played multiple times [1], [2]. Repetitive value, denoting the capacity to engage in the same content on multiple occasions, is a fundamental attribute of a game functioning as an entertainment medium [3], [4]. The implementation of procedural content generation processes is driven by the objective of cost reduction and labour efficiency in content automation [5], [6]. Procedural Content Generation (PCG) is a technique used to reduce the expense and time required for level design [7], [8].

In addition to providing opportunities for developers, 3D permits participants to act more freely and think creatively. PCG is a technique for automatically generating a variety of content using a set of algorithms [9], [10]. The earliest use of PCG in video games is in Roguelike games, where the majority of the content is random.

One of the algorithms that can be used in PCG level design is Cellular Automata (CA), which is used in this study, where the application works from a cell that influences its neighboring cells until a space is formed. Cellular Automata (CA) is an algorithm used in procedural content generation (PCG), operating from a single cell that influences its neighbors to form chambers based on predefined rules [11]. Maps generated by CA typically have a natural, cavernous appearance. Previous research by Earle, Snider, Fontaine, and Togelius compared the efficiency of map construction using CA and random generation, showing that CA is more performance-efficient [12],[13]. This study also incorporates the Marching Squares algorithm, a technique utilized for generating 2D contours. Furthermore, the Marching Squares algorithm

Received: 17-12-2024 | Accepted: 14-03-2025 | Published Online: 20-04-2025

is employed to create a 3D wall model based on the generated map [14].

The difference with the previous study is that in this study, the map will be generated using CA, and the 3D structures will be generated using the Marching Square (MS) algorithm. This research is developing a horror-themed game in which the objective is to acquire a number of items while avoiding enemies. The game is designed as a single-player experience with a first-person perspective. The player will navigate a maze-like map generated using Cellular Automata and Marching Square. When a player collects a certain number of items without coming into contact with an enemy, an exit will materialize for them to reach in order to win the game. The game will then be evaluated using the Game User Experience Satisfaction Scale questionnaire to determine the validity of the Cellular Automata and Marching Square-programmed game.

2. Research Methods

The research process for creating Maze Games involves several steps, including literature review, game design, game production, game testing, assessment, and reporting. During the literature review stage, extensive research is conducted to find references, theories, and in-depth material on topics relating to the creation of the game labyrinth. Specifically, the focus is on Procedural Content Generation, Cellular Automata, and Marching Square. During the planning stage of the Game, the process involves the creation of a flowchart and a mockup.

During the subsequent stage, the game production process commenced in unity using C-Sharp (C#) and incorporated PCG Cellular Automata as planned. During the subsequent phase, known as the Game testing stage, the game that has been developed is thoroughly examined to identify any flaws, bugs and to ascertain whether it has achieved the intended objectives. Tests are conducted to evaluate the outcomes of generated maps and the positioning of opponents or foes. The test results are then verified to ensure they are not identical or random. This testing is carried out utilizing Black box testing.

Next, the evaluation stage determines the quality of the maps developed using the CA and MS algorithms and draws conclusions based on the evaluation of user opinions using the Game User Experience Satisfaction Scale. A total of 25 participants participated in the game and provided feedback through a Google form. The form included questions designed according to the rules of GUESS. The GUESS was chosen because the GUESS has advantages, such as [15], [16]:

Quantitative Measurement: GUESS provides a quantitative measurement of user satisfaction through structured questions and rating scales. This allows for more objective evaluation compared to qualitative methods alone.

Standardized Metrics: GUESS offers standardized metrics that can be used across different games and contexts, facilitating comparisons and benchmarking. This consistency aids in identifying trends and areas for improvement.

Comprehensive Feedback: The questionnaire covers various dimensions of the gaming experience, including gameplay, aesthetics, controls, and overall enjoyment. This comprehensive feedback helps developers understand users' preferences and areas needing enhancement.

Efficiency: GUESS can be administered to a large number of users efficiently, especially in online surveys or post-play sessions. This scalability allows for gathering feedback from a diverse user base, enhancing the representativeness of the data.

Iterative Improvement: By regularly administering the GUESS questionnaire at different stages of game development, developers can track changes in user satisfaction over time and iteratively improve the game based on feedback.

User-Centric Design: GUESS emphasizes the importance of user experience in game design and development, encouraging developers to prioritize aspects that contribute to user satisfaction and engagement.

The final stage is the writing of a research report, which at this stage writing a report on how the implementation of Procedural Content Generation Cellular Automata and Marching Square on the Maze Game and an explanation of how the algorithms and assets used in the creation of the game work. Through the Game User Experience Satisfaction Scale is collected the level of player satisfaction after the game is collected in the test by either player until completion or not.

2.1. Cellular Automata for Generate Map

Cellular Automata on PCG is used to create maps like dungeons. A dungeon map has a room that has a door that leads to another neighboring room, and from its formation, there is a portion that is considered to be an inaccessible area [17]. In the formation of maps, there are several parameters for the limitation point of generating maps, which are as follows [18], [19].

Percentage of stone cells or areas inaccessible; The number of generations of Cellular Automata that will run; The neighbor will be a stone; Number of neighbor cells.

In an empty base room with a grid X x Y, cells have two states of emptiness and stone. The way it works is as follows [20]: The grid of the empty room will be given several cells randomly with a probability of 0.5 (r) that the cell will be a stone; Then Cellular Automata will be applied to the grid with the rule of a cell will be a stone in the next step if the five cells around it are stones and vice versa if the number of stones around it is not up to five then the cell will become free space.

From the set of stone cells that have formed, the edge area that comes into contact with the free space cell will be called a wall or wall. In this research, Cellular Automata (CA) is used to create a map resembling a dungeon. CA uses a set of rules to determine the generation of the map [21], including the percentage of cells that are active or inaccessible, the number of CA iterations that will be performed, the neighbourhood value that designates an active cell, and the number of neighbor cells. CA begins with a single cell, which then checks the neighbouring cells; if more than four neighbouring cells are active, the initial cell is transformed into an active cell, whereas if fewer than four neighbouring cells are active, the initial cell remains inactive. CA will repeat this step until the entire grid is covered and each cell is verified. Figure 1 is a flowchart illustrating how CA operates. Where in the flowchart in Figure 1 shows how to change a cell from a floor to a wall and vice versa if it meets a provision or rule. First, a loop will be run for all cells and a 3x3 check will be carried out on the cells surrounding the cell to be checked, then wallCount will be added to map[neighbourX, neighbourY], if map[neighbourX, neighbourY] is 1 then the cell is a wall. After checking the neighboring cells, it will be determined if the wallCount is less than 4, then the cell is still considered a floor, and vice versa; if the wallCount is more than 4, then the cell will be made a wall. This process is carried out for all map cells.

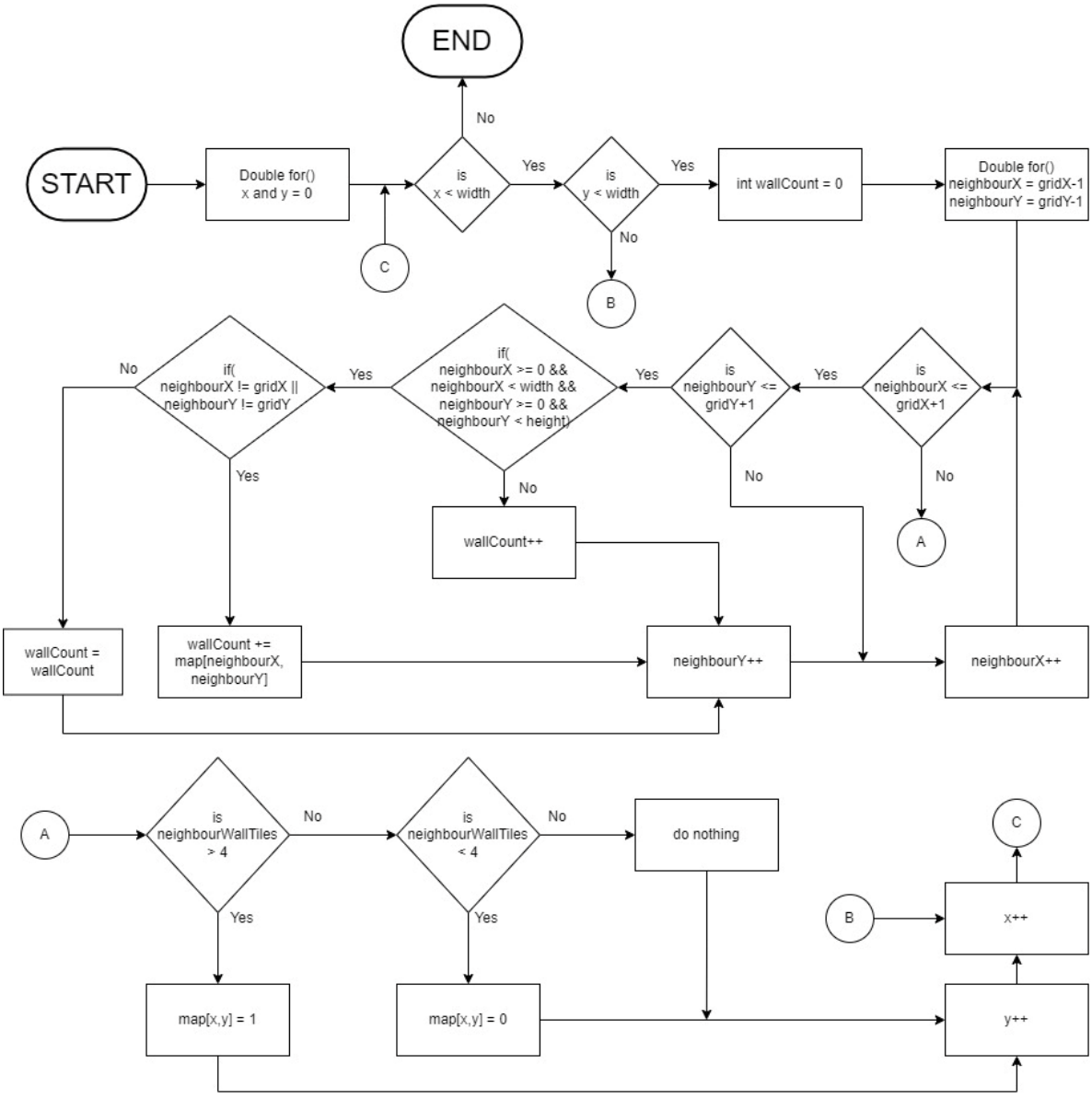

Figure 1. Flowchart for Convert Wall with Cellular Automata

2.2. Create Mesh on Map

Marching Square (MS) is an algorithm typically used to create maps that generate 2D contours. MS uses a configuration square where each corner can be referred to as a cell and has an active or inactive state [22].

The configuration of a square is depicted in Figure 2, with the upper left corner labeled 1 and moving clockwise to 4. A configuration can be calculated with each cell represented in binary based on the active cells; for instance, if cells 1 and 3 are active, the binary configuration is 1 0 1 0, and the calculation is 8 + 0 + 2 + 0 with 10 as the configuration [23].

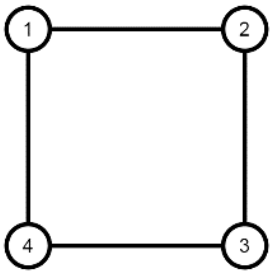

Figure 2. Square Configuration in Marching Square

There are a total of 16 configurations in Figure 3, including the state where all cells are inactive, which is configuration number 1. As depicted in Figure 2, a line is constructed based on the configuration number to form a shape connecting other active cells between squares. Figure 4 shows GenerateMesh, which works to create a mesh on maps that have been created using Cellular Automata. First, in initialization for wall height, nodeCountX, nodenCountY, mapWidth, and mapHeight then used a loop to give positions for each controlNodes as well as controlNode conditions when active or not. After that, it will be initialized square to give the box based on controlNodes created using the loop. After that, it will use Marching Square to determine the value of the box to form a triangle, then after that, it is the formation of the mesh.

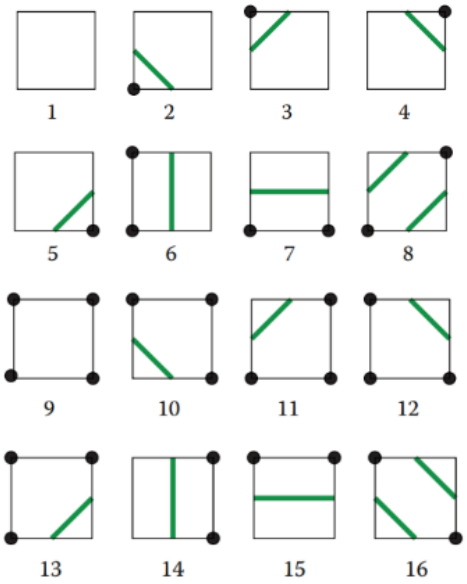

Figure 3. Configurations in Marching Square

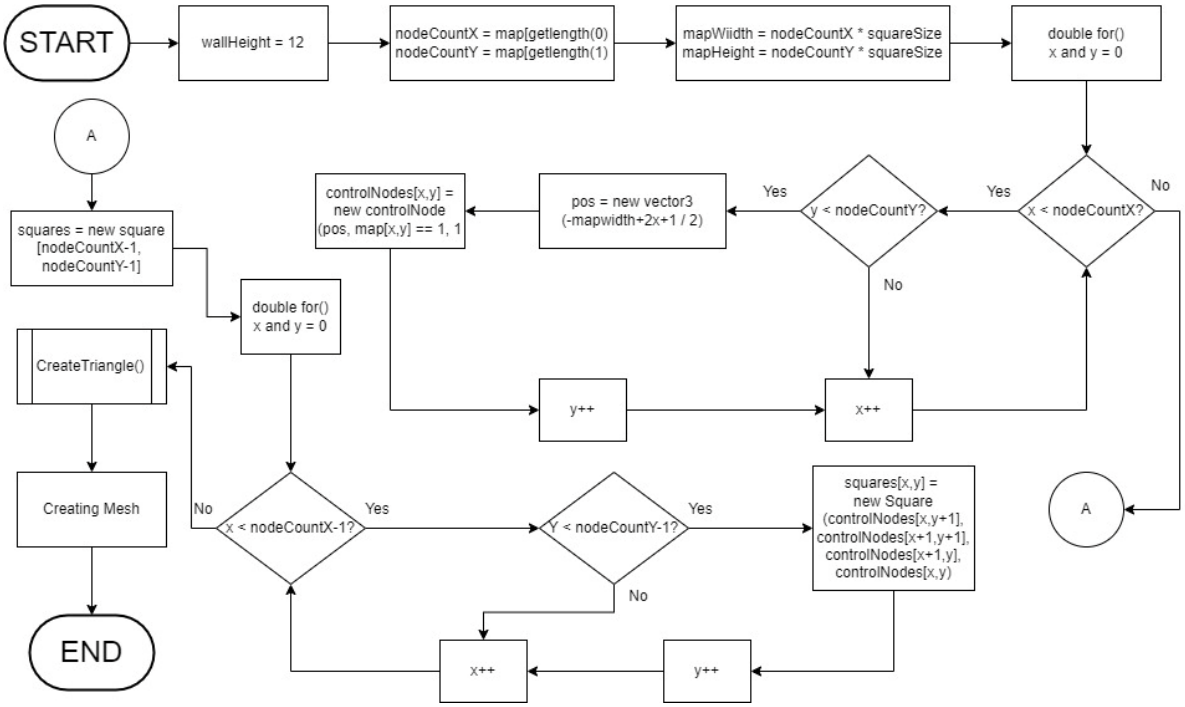

Figure 4. Flowchart Generate Mesh

2.3. Marching Square for Create Triangle on Map

Figure 5 shows the flowchart of Marching Square. First, the configuration initialization is performed, and then the boxes that have been formed from the previous controlNodes are checked for an active condition with a maximum value of 15. Once the configuration value is obtained, the switch will be used with 15 possibilities to connect nodes. For example, a configuration with a value of 1 or case 1 is obtained from a binary value of 0 0 0 1 and in the box, then the fourth sequence is the bottom left controlNode. The Marching Square rule for the bottom left configuration is to connect the nodes from the middle left, bottom left and middle bottom.

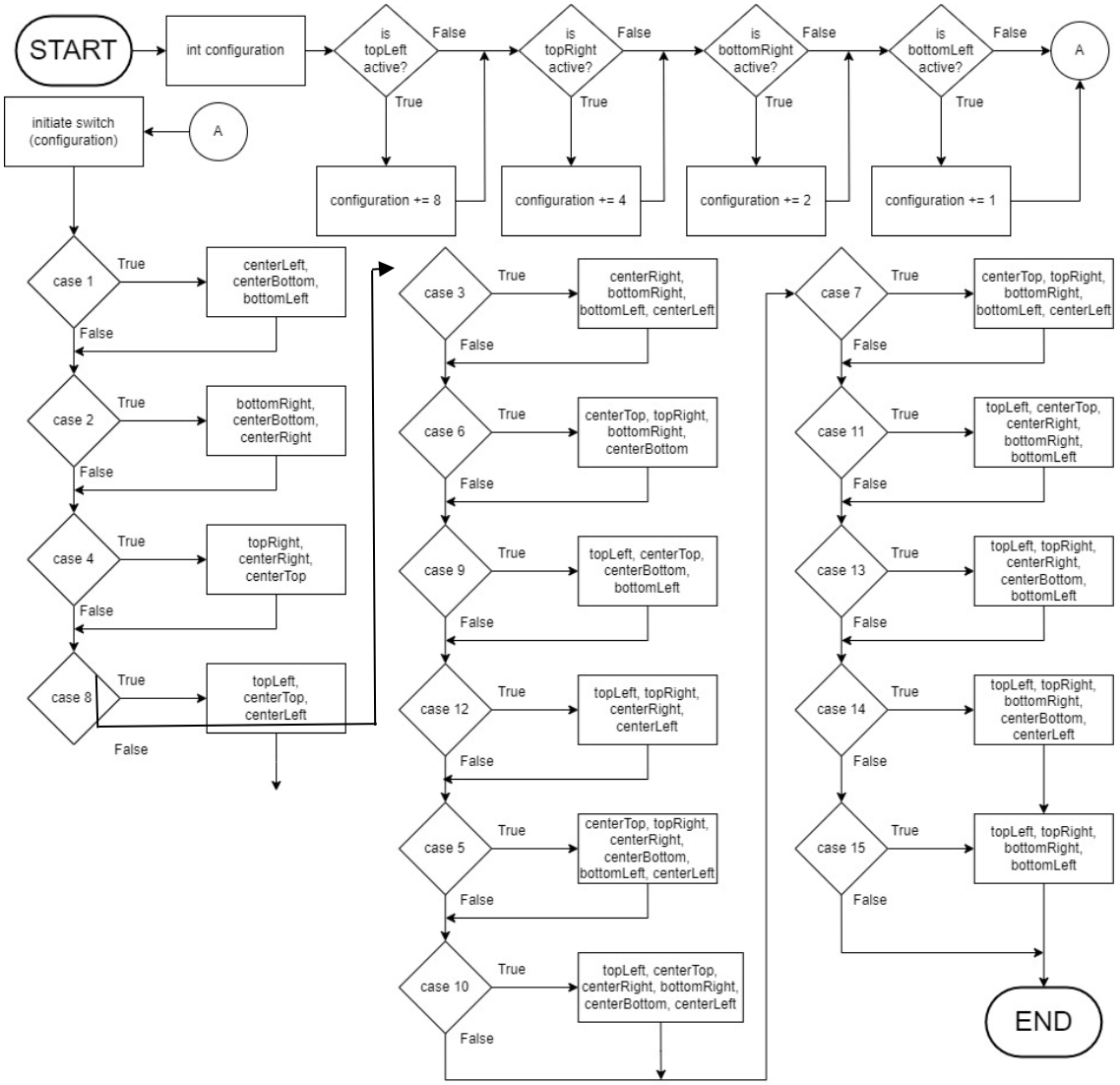

Figure 5. Flowchart for Create Triangle on Map with Marching Square Configuration

3. Results and Discussions

The game in this paper is made with a specification of RTX 3050 laptop, i7-12650H and 16 gb ram. Using Unity Engine and C3 language to implement the algorithms. Size of the map used to make the map is 164x92 unity unit. The app is designed using Unity 2021 and is intended for Windows operating systems. The game uses Procedural Content Generation Cellular Automata and the Marching Square algorithm to create random maps. While game content consists of players, items, and enemies in the process of their random positioning based on maps created by Cellular Automata and Marching Square. Figure 6 is an initial map display that has not been initialized by GenerateMap.

Figure 7 depicts three random maps generated by Cellular Automata and Marching Squares. The result of each map generation is different, meaning that the same map will never appear twice. Random adversaries, items, and the player appear on the map, which the player must navigate.

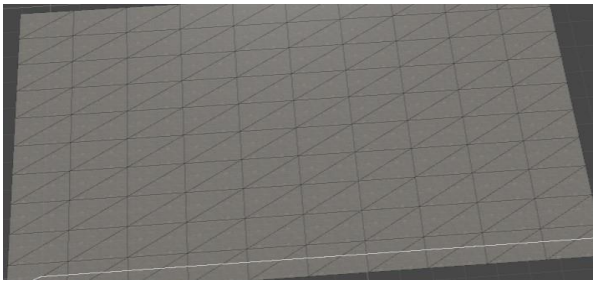

Figure 6. Example of map created using Cellular Automata and Marching Square

Game creation focused on the generation of maps using Cellular Automata and Marching Square. After that the map is generated in the input of the gameplay element is the creation of the player character that will be controlled by the player, items and obstacles in the form of enemies. Once the gameplay is integrated with the map generator, it will be tested by several players and requested feedback in the form of a questionnaire using the GUESS or Game User Experience Satisfaction

Scale method. The game was tested by 25 players who gave feedback through Google Forms with questions made based on guess rules.

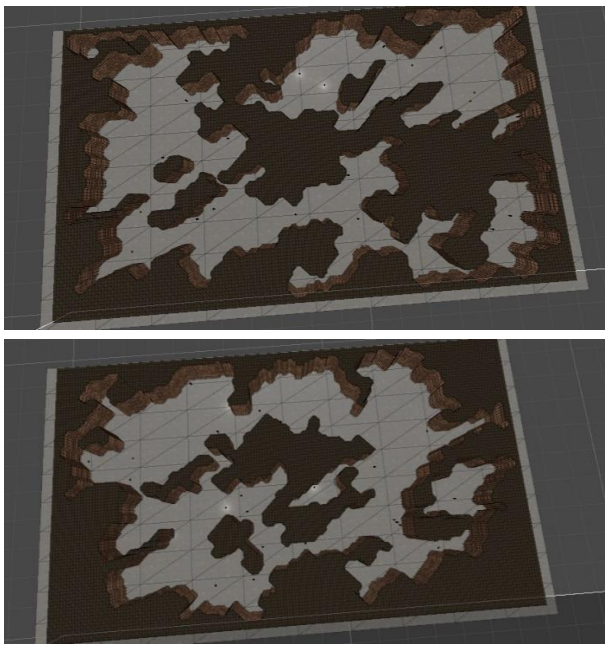

Figure 7. Example of map created using Cellular Automata and Marching Square

The questionnaire is selected based on the needs of the game, where the nine factors of the rating category are taken five factors will then be selected again questions related to the game. The factors used are 1 Usability/Playability factor with 11 questions, 4 Enjoyment factor with 4 questions, 5 Creative Freedom factor with 5 questions, 6 Audio Aesthetics factor with 4 questions and 7 Personal Gratification factor with 5 questions totalling 29 questions. Other factors, such as the second factor, Narratives are not used because the game has no narrative and does not need to be tested. Here is the average calculation of the factors taken.

Table 1 shows a question on factor 1, Usability/Playability, that serves to measure the level of player satisfaction with the ease of the game, such as clear lenses, UI and controls that are easy to understand by the player. Table 1 shows a question on factor 1, Usability/Playability, that serves to measure the level of player satisfaction with the ease of the game, such as clear lenses, UI and controls that are easy to understand by the player. Table 1 also shows the results of the sum of the values of each selected anchor, along with the total and average anchor values of the Usability/Playability factors. And the average of all questions is 83.84%.

Table 1. List of Usability/Playability Questions

No	Question List	1	2	3	4	5	6	7	Sum	Average (Total/175)
1	I find it easy to learn how to play the game	0	0	2	3	3	6	11	146	0,834285714
2	Game control is clear	0	0	1	3	5	10	6	142	0,811428571
3	I always know how to finish goals in the game.	0	0	2	0	9	5	9	150	0,822857143
4	I think the interface of the game is easy to use	0	0	1	1	4	10	9	144	0,857142857
5	I don't need to follow a long tutorial instructions just to play the game	0	1	0	2	6	8	8	141	0,822857143
6	The menu of the game is easy to use.	0	0	1	4	6	6	8	154	0,805714286
7	I think gamenya taught me about its control smoothly.	0	0	0	2	3	9	11	148	0,88
8	When I finish a goal, I always know the next goal.	0	1	0	2	4	8	10	153	0,845714286
9	I think the gamenya gave me the information needed to complete the objectives in the game	0	0	0	2	2	12	9	150	0,874285714
10	I think the information on the game (e.g., the message on the screen) is clear	0	0	1	3	2	8	11	150	0,857142857
11	I feel confident in playing the game.	1	1	0	1	6	7	9	142	0,811428571

Table 2 shows questions on factor 4 Enjoyment which serves to measure the level of player satisfaction with the player's pleasure after playing the game and Table 2 also shows the results of the total value of each selected anchor along with the total anchor value and the average of the Enjoyment factor, where the *a*verage of Enjoyment is 81.86%.

Table 2. Enjoyment question table

No	Question list	1	2	3	4	5	6	7	Total	Average (total/175)
12	I think the game is fun	0	0	0	2	8	4	11	149	0,851428571
13	I enjoy playing games	0	0	0	4	5	6	10	147	0,84
14	I seem more likely to recommend this game to others	0	0	1	2	8	8	6	141	0,805714286
15	If there is a chance, I want to play the game again	0	1	1	2	9	6	6	136	0,777142857

Table 3 shows questions on factor 5, Creative Freedom, which serves to measure the level of satisfaction of players with the freedom of players when playing games. Table 3 also shows the results of the sum of the values of each selected anchor, along with the total and average anchor values of the Creative Freedom factor, where the average yield of Creative Freedom is 81.83%.

Table 3. Creative Freedom question table

No	Question list	1	2	3	4	5	6	7	Total	Average (total/175)
16	I feel like I can be more imaginative by the game	0	1	0	2	6	8	8	144	0,822857143
17	I feel given enough freedom to do as I please	0	1	0	1	7	11	5	142	0,811428571
18	I can explore things in games	0	0	1	2	6	8	8	145	0,828571429
19	My curiosity increases after playing the game	0	0	1	2	5	10	7	145	0,828571429
20	I think the game is unique or original	0	0	1	4	5	9	6	140	0,8

Table 4 shows questions on factor 6 Audio Aesthetics which serves to measure the level of player satisfaction with the sound effects in the game and also Table 4 shows the results of the sum of the values of each selected anchor along with the total and average anchor values of the Audio aesthetics factor, where the average result of Audio Aesthetics is 83.71%.

Table 4. Audio Aesthetics question table

No	Question list	1	2	3	4	5	6	7	Total	Average (total/175)
21	I enjoy the sound effects in games	0	0	0	3	4	10	8	148	0,845714286
22	I enjoy the music in games	0	0	0	3	5	10	7	146	0,834285714
23	I think game audio (such as sound effects and music) adds to my gaming experience	0	1	0	5	3	7	9	142	0,811428571
24	I feel that the game's audio matches the atmosphere of the game	0	0	0	2	4	11	8	150	0,857142857

Table 5 shows questions on factor 7 Personal Gratification which serves to measure the level of satisfaction of players with challenges in games that can give players a sense of success while playing, and also Table 5 shows the results of the sum of the values of each selected anchor along with the total and average anchor values of the Personal Gratification factor with the average result of Personal Gratification being 84.46%.

Table 5. Personal Gratification Question Table

No	Question list	1	2	3	4	5	6	7	Total	Average (total/175)
25	I feel tense whether I can finish the game or not	0	0	1	3	2	9	10	149	0,851428571
26	I feel successful when I pass challenges in games	0	0	0	2	3	10	10	153	0,874285714
27	I want to do my best while playing	0	0	0	4	3	9	9	148	0,845714286
28	I focus a lot on my performance when I play the game	0	0	0	5	4	5	11	147	0,84
29	I feel my skills improve as I complete challenges in the game	0	0	0	4	6	9	6	142	0,811428571

Based on the Questioner results of the five factors which have a total of 29 questions, it can be calculated that the average level of player satisfaction with the Maze Game built using PCG Cellular Automata and Marching Square is 83.14%, Which shows that respondents are very satisfied with the results of game creation using Procedural Content Generation (PCG) through the application of Cellular Automata and Marching Squares.

The findings that can be derived from the study conducted to build the Maze Game using Procedural Content Generation Cellular Automata, and the Marching Square algorithm are as follows:

Cellular Automata is employed to generate a 2D map, with a predetermined size, wherein active cells are randomly distributed and referred to as walls. Subsequently, in the next generation, a filtering process is conducted to identify walls that are deemed too small and isolated rooms. Subsequently, the map's shape is defined by the inclusion of supplementary walls along its edges. Once the 2D map shape has been constructed successfully, the Marching Square algorithm will be employed to create a mesh for the map, enabling its conversion into a 3D representation. The Marching Square algorithm will generate a triangular shape within a box composed of four cells. This will result in a three-dimensional wall mesh that appears more polished and less box-like. The creation of a three-dimensional wall is achieved by the utilization of the Marching Square algorithm, which identifies a line within the triangular region located at the boundary of the wall. The Maze Game is designed with a singular purpose and features barriers in the form of diamonds and zombies. However, it incorporates dynamic components such as ever-changing map positions and shapes in each generation.

The satisfaction of players with the Maze Game, which was created using PCG Cellular Automata and Marching Square, was assessed using the Game User Experience Satisfaction Scale. This scale includes five factors: Usability/Playability, Enjoyment, Creative Freedom, Audio Aesthetics, and Personal Gratification. These factors were chosen based on the specific requirements of the game. The game has received a high degree of player satisfaction, with the majority of

players strongly agreeing with it. The average total game satisfaction level is 83.14%. Based on these findings, it can be inferred that the respondents strongly agree and derive pleasure from the outcomes of creating games employing Procedural Content Generation (PCG) with the utilization of Cellular Automata and Marching Squares.

4. Conclusions

The reason this game is in 3d rather than 2d is to enable the element of horror by utilizing the dread of the unknown due to the game's first-person perspective. Creating 3D barriers requires the use of Marching Square in order to be rendered in 3D. While Cellular Automata correctly generates the map, there are instances in which a room is generated without an exit, isolating it. As a result, an identifier is required as the next stage to improve map generation in order to identify isolated rooms and build a connecting bridge between them. Next are the 3d walls created with Marching Squares; while it has been successful, the wall contains a graphical flaw where the lighting does not render properly on one side; to solve this issue, another algorithm called Cubical Marching Squares could be used, as it can create a more sophisticated looking 3d shape. The Game User Experience Satisfaction Scale measures player satisfaction with PCG Cellular Automata and Marching Square's Maze Game using five factors: Usability/Playability, Enjoyment, Creative Freedom, Audio Aesthetics, and Personal Satisfaction, which were chosen based on the game's requirements. The average user satisfaction with the game is 83.14%, which is Strongly Agree, indicating that users can accept a PCG labyrinth game employing CA and MS. This research cannot generate big maps since tiny and medium rooms will be isolated or inaccessible, causing things, gamers, and enemies to spawn in isolated rooms. In addition, this research has a graphical flaw that limits lighting to one direction.

References

[1] R. Adellin, C. T. Khuan, and L. D. Gertrude, "Conceptual Framework Puzzle Game with High Replayability," *J. Phys. Conf. Ser.*, vol. 1228, no. 1, 2019, doi: 10.1088/1742-6596/1228/1/012070.

[2] J. Sampurna and W. Istiono, "Virtual Reality Game for Introducing Pencak Silat," *Int. J. Interact. Mob. Technol.*, vol. 15, no. 1, pp. 199–207, 2021, doi: 10.3991/IJIM.V15I01.17679.

[3] A. Khalifa, P. Bontrager, S. Earle, and J. Togelius, "PCGRL: Procedural content generation via reinforcement learning," *Proc. 16th AAAI Conf. Artif. Intell. Interact. Digit. Entertain. AIIDE 2020*, pp. 95–101, 2020, doi: 10.1609/aiide.v16i1.7416.

[4] Y. Bai, Y. Wang, Y. Tong, Y. Yang, Q. Liu, and J. Liu, "Boundary Content Graph Neural Network for Temporal Action Proposal Generation," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12373 LNCS, pp. 121–137, 2020, doi: 10.1007/978-3-030-58604-1_8.

[5] D. A. Ramadhan and A. D. Indriyanti, "Procedural Content Generation pada Game World Exploration Sandbox Menggunakan Alogoritma Perlin Noise," *J. Informatics Comput. Sci.*, vol. 4, no. 01, pp. 86–91, 2022, doi: 10.26740/jinacs.v4n01.p86-91.

[6] H. Juwiantho, L. Liliana, and M. Budiono, "Procedural Content Generation pada Game Tower Defense menggunakan Perlin Noise dan Algoritma Floyd Warshall," *J. Animat. Games Stud.*, vol. 9, no. 1, pp. 11–28, 2023, doi: 10.24821/jags.v9i1.8100.

[7] A. Gambi, M. Mueller, and G. Fraser, "Automatically testing self-driving cars with search-based procedural content generation," *ISSTA 2019 - Proc. 28th ACM SIGSOFT Int. Symp. Softw. Test. Anal.*, no. September, pp. 273–283, 2019, doi: 10.1145/3293882.3330566.

[8] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman, "Leveraging procedural generation to benchmark reinforcement learning," *37th Int. Conf. Mach. Learn. ICML 2020*, vol. PartF16814, pp. 2026–2034, 2020.

[9] N. A. Barriga, "A Short Introduction to Procedural Content Generation Algorithms for Videogames," *Int. J. Artif. Intell. Tools*, vol. 28, no. 2, 2019, doi: 10.1142/S0218213019300011.

[10] S. Risi and J. Togelius, "Increasing generality in machine learning through procedural content generation," *Nat. Mach. Intell.*, vol. 2, no. 8, pp. 428–436, 2020, doi: 10.1038/s42256-020-0208-z.

[11] K. Vayadande, R. Pokarne, M. Phaldesai, T. Bhuruk, T. Patil, and P. Kumar, "Simulation of Conway'S Game of Life Using Cellular Automata," *Int. Res. J. Eng. Technol.*, no. March, pp. 327–331, 2022, [Online]. Available: www.irjet.net

[12] S. Earle, J. Snider, M. C. Fontaine, and J. Togelius, *Illuminating Diverse Neural Cellular Automata for Level Generation*, vol. 1, no. 1. Association for Computing Machinery, 2022.

[13] W. Istiono, "Bilingual Color Learning Application as Alternative Color Learning for Preschool Student," *Int. J. Interact. Mob. Technol.*, vol. 16, no. 5, pp. 224–233, 2022, doi: 10.3991/ijim.v16i05.28319.

[14] B. Lopez, J. Munoz, F. Quevedo, C. A. Monje, S. Garrido, and L. Moreno, "4D Trajectory Planning Based on Fast Marching Square for UAV Teams," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 6, pp. 5703–5717, 2024, doi: 10.1109/TITS.2023.3336008.

[15] W. A. Rohmah and W. Apriyandari, "Implementation of the Algorithm Fisher Yates Shuffle on Game Quiz Environment," *J. Informatics Telecommun. Eng.*, vol. 4, no. 1, pp. 161–172, 2020.

[16] J. R. Keebler Assoc, W. J. Shelstad, D. C. S. Google, B. S. Chaparro, and M. H. Phan Google, "Validation of the GUESS-18: A Short Version of the Game User Experience Satisfaction Scale (GUESS)," *J. Usability Stud.*, vol. 16, no. 1, pp. 49–62, 2020.

[17] O. Tekik, E. Surer, and A. Betin Can, "Verifying Maze-Like Game Levels With Model Checker SPIN," *IEEE Access*, vol. 10, no. May, pp. 66492–66510, 2022, doi: 10.1109/ACCESS.2022.3185109.

[18] Z. Wu, Y. Mao, and Q. Li, "Procedural Game Map Generation using Multi-leveled Cellular Automata by Machine learning," *ACM Int. Conf. Proceeding Ser.*, no. October 2021, pp. 168–172, 2021, doi: 10.1145/3500931.3500962.

[19] A. Gharaibeh, A. Shaamala, R. Obeidat, and S. Al-Kofahi, "Improving land-use change modeling by integrating ANN with Cellular Automata-Markov Chain model," *Heliyon*, vol. 6, no. 9, p. e05092, 2020, doi: 10.1016/j.heliyon.2020.e05092.

[20] J. Öhman, "Procedural Generation of Tower Defense levels," pp. 1–10, 2020, [Online]. Available: http://www.ep.liu.se/.

[21] A. Sabanovic and A. Khodabakhshi, "Evolved cellular automata for 2D video game level generation," 2022.

[22] P. Chen, Y. Huang, E. Papadimitriou, J. Mou, and P. van Gelder, "Global path planning for autonomous ship: A hybrid approach of Fast Marching Square and velocity obstacles methods," *Ocean Eng.*, vol. 214, no. February, p. 107793, 2020, doi: 10.1016/j.oceaneng.2020.107793.

[23] J. Muñoz, B. López, F. Quevedo, S. Garrido, C. A. Monje, and L. E. Moreno, "Gaussian processes and Fast Marching Square based informative path planning," *Eng. Appl. Artif. Intell.*, vol. 121, no. July 2022, p. 106054, 2023, doi: 10.1016/j.engappai.2023.106054.

