



DESIGN AND BUILD VR HORROR GAMES WITH PROCEDURAL CONTENT GENERATION USING CELLULAR AUTOMATA ALGORITHM

**Muhammad Dzulfikar Ramadhan Wibawanto¹, Angga
Aditya Permana^{2*}, Adhi Kusnadi³**

*Corresponding author
dzulfikar@student.umn.ac.id¹
angga.permana@umn.ac.id^{2*}
adhi.kusnadi@umn.ac.id³

^{1,2,3}Departement of Informatic, Universitas Multimedia
Nusantara, Kabupaten Tangerang, Indonesia

^{1,2,3}Scientia Boulevard Gading, Curug Sangereng, Serpong,
Kabupaten Tangerang, Banten 15810

Article history:

Received Juli 18, 2023

Revised August 22, 2023

Accepted August 31, 2023

Keywords:

Cellular Automata;

Horror Game;

Procedural Content;

Generation.

Abstract

The development of video games is very fast according to data obtained by ESA in 2021, about 226 million people on American played video games and in Indonesia 105 million people play video games and this will continue to increase, it is predicted that in 2025 video players gaming in Indonesia will reach 127 million people, this causes the need for game content continues to increase while for making game content it is not something that cheap, one way to reduce costs in making content in the game is to implement procedural content generation, procedural content generation is one way of creating content game automatically by using an algorithm. This research aims to design and build a virtual reality horror game using use cellular automata algorithm (CA), CA algorithm in research This research is used for map making, besides that this research also aims to measure the level of player satisfaction with the game that has been made. Game created using the Unity 3D game engine and the C# programming language. After game has been successfully built, will measure the level of satisfaction through cakes. questionnaire using GUESS-18, based on tests that have been carried out with GUESS-18, the results obtained are 77.1 percent, which can be concluded that the level of player satisfaction is categorized as good.

1.0 INTRODUCTION

The development of computer technology is very fast to this day, in this day computers are not only used to do work but are also used for entertainment. One of the entertainment media that has developed because of computer technology is video games, video games is an interactive entertainment media that is played through computers, consoles, games, and also phones [1]. According to a survey conducted in 2021 by the ESA approximately 226 million in America play video games [2] and in Indonesia 105 million people play video games and this will continue to increase, it is predicted that in 2025 video players gaming in Indonesia will reach 127 million people [3][4].

Virtual reality games with horror genre, first person shooter genre and survival genre is the most popular genre [5], not only that, those genres has also become a popular genre in

the game streaming community, can be seen out of the many famous streamers who chose to show a fearful reaction their tan when playing virtual reality horror games, an example is the famous game streamer pewdiepie get more than 16 million views on youtube at the same time playing his first VR horror game [6]. In a game, game content is an important thing so that keep players playing [7][2], however to create game content manually are very expensive while the demand for game content continues to increase [8]. One way to solve this problem is to implement transforming procedural content generation into the game have been created [9].

Procedural content creation is a method for creating content game automatically by using an algorithm to determine reduce the cost of making games [10]. PCG can create game content that adapts to player preferences and enhances gaming experience therefore PCG has become very popular in applications game making [11]. PCG can create all content that affects gameplay other than non-player characters (NPCs), such as maps, levels, dialogues, characters, rule sets and weapons [12][13].

One of the algorithms that can be applied to create a procedural map is cellular automata (CA), not only maps CA usually use in games to model environmental systems such as heat, fire, rain, water flow, pressure, and explosion [14]. A simple CA algorithm is evaluated on an infinite cave game, generating playable and well-designed tunnel-based maps. It also has very low computational cost, permitting real time content generation, and the proposed map representation provides sufficient flexibility with respect to level design [7], [15]–[21]. To the best of the authors' knowledge there is no study reporting the use of CA for generating complete horror games. Due to the high interest of gamers towards horror games based on VR technology, and it's also expensive and takes a long time to create content for a game. Then the design of a VR-based horror game is carried out using procedural content generation methods and CA algorithms to make content creation faster and cheaper in a game.

2.0 THEORETICAL

2.1. Horror Game

Horror game is a genre in video games that focuses on horror aspects and is usually designed to frighten the player, unlike most video game genres which are usually classified based on gameplay, horror games are almost always based on narrative or visual presentation using various types gameplay [22].

2.2. Procedural Content Generation

Procedural content generation is a method for creating game content automatically using an algorithm to reduce the costs of creating games [10]. The content that can be created using PCG is very varied, for example terrain, textures and even high-level assets such as storyline and characters. PCG not only has the potential to create basic games developed by developers but can also provide endless content to extend the life of a game, apart from that PCG can also be used to create game content that can adapt to player preferences and improve playing experience [11].

2.3. Cellular Automata (CA)

CA are algorithm that can be applied to create maps procedurally. It also used in games for modeling environmental systems such as heat and fire, rain and water flow, pressure and explosion [21]. There are a few webpages that propose the use of CA on small single grids providing no reliable evaluation of the algorithms. This paper provides a CA-based algorithm for horror game.

3.0 METHODOLOGY

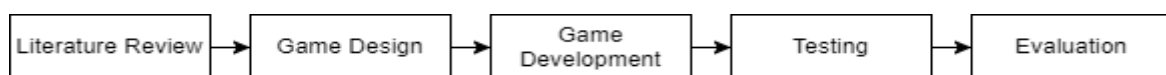


Figure 1. Methodology Research

At the methodology stage, we carried out five stages, namely:

3.1. Literature Review

At this stage, information will be collected on theories which will later be used in conducting research. The theories needed in conducting this research are related to procedural content generation as well as an in-depth understanding of the cellular automation algorithm and the

Game User Experience Satisfaction Scale testing method to measure the level of player satisfaction with the games that are made.

3.2. Game Design

The following are the formal elements and dramatic elements used in the game.

1) Formal Element

a) Player

The game is a single player game that can only be played by players one person.

b) Objectives

The objective of the game is to complete quests that are given the NPC that is looking for medicinal plants that are in a cave that filled with so many zombies.

c) Procedures

- When the player runs the game, a splash screen page will appear.
- After the player passes the splash screen page, a page will appear a menu that contains 4 options, namely start, how to play, credit and quit.
- If the player selects start then the game will start and will a prologue page appears containing the story of the game.
- If the player presses start on the prologue page, the player will be navigate to the city.
- In the city players can talk to NPCs to take quests and open the closed door.
- After the door is opened the player can enter the door and will be directed to the scene containing the map created using PCG.
- To complete the game the player must complete a quest that given is to find 3 medicinal plants.
- If the player's blood is 0 then the player will be directed to the game over page which contains the retry and quit options, where if the player choose retry then the player will repeat the game, if player choose quit then the player will be directed to the main menu page.

d) Rules

- Players must use Google Cardboard to run the game.
- Players can use VR Controller or Joystick as input in the game.
- Players cannot exit when already in the game.
- Players must collect 3 medicinal plants to complete the game.
- Player will die if the blood has reached 0.
- Enemy will die if the blood has reached 0.
- If the player manages to collect 3 plants then the player wins.

e) Resources

- Player Health: Shows the blood indicator owned by the player, where the blood will decrease if hit by an enemy attack and will increase if the player takes the health packs that are in the game.
- Player Ammo: Shows the bullet indicator the player has, where the bullet will decrease if the player shoots and the number of bullets can be added by taking the item Ammo.
- Quest: Shows the progress of the quest that the player is running.

f) Conflict

Players must find medicinal plants on the map while defending their lives from being chased by zombies.

g) Boundaries

Players can only walk around on the map that has been created, as well as players can't exit in the middle of the game if the game has already started.

h) Outcome

The player win if the player has managed to collect 3 medicinal plants.

2) Dramatic Element

a) Challenge

The challenge in this game is how the player can keep his life while looking for medicinal plant items while being chased by zombies.

b) Play

In this game, players are given the freedom to explore against the map that was built while looking for medicinal plants needed by NPCs.

c) Premise

A government soldier named Joe who was sent by the government to solve the problem of the Berun village.

d) Character

The character in this game is a government soldier who was sent to a village to help Dr. Sisca in solving the plague problem there.

e) Story

In 1980 in the village of Berun there was a dangerous epidemic where this plague could cause death to anyone who was infected. This plague can only be cured using the solanum plant which is processed into medicine where the existence of this plant is only in the cave of Mordor there are many zombies. Then there is a doctor from the village named Dr. Sisca called the institution government to send someone to help. Doctor is looking for the solanum plant, and a soldier named Joe was sent to help Dr. Sisca in finding the solanum plant.

4. Flowchart

The following are flowcharts used in making the game

1) Map Generator

The map generator flowchart explains the process of map generation. The process begins with initializing the variables that will be used in making the map. The variables used in the map creation include width, height which is used to define map width and height, seed variable that is used for coding on map, the fillPercent variable used for the percentage of filling the map, as well as variable map which has an array data structure that is used to represent cells in the CA algorithm.

After initializing the variable, the next step is initializing the value of the cell with a random value that is obtained from the seed.GetHashCode() function, after random cells have been generated, run the rules() function to determine the value of a cell based on its neighbors and run the spawn() function to do a spawn prefab against a cell with a value of 1.

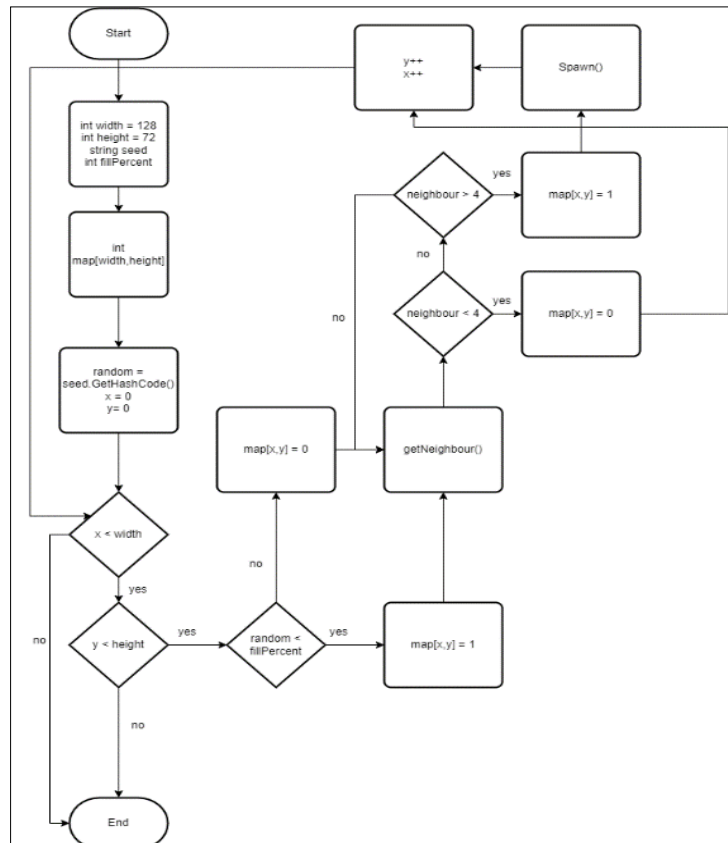


Figure 2. Map Generator Flowchart

2) Main Menu

The main menu flowchart explains the player flow on the main menu page, where on the main menu the player can choose 4 menu options.

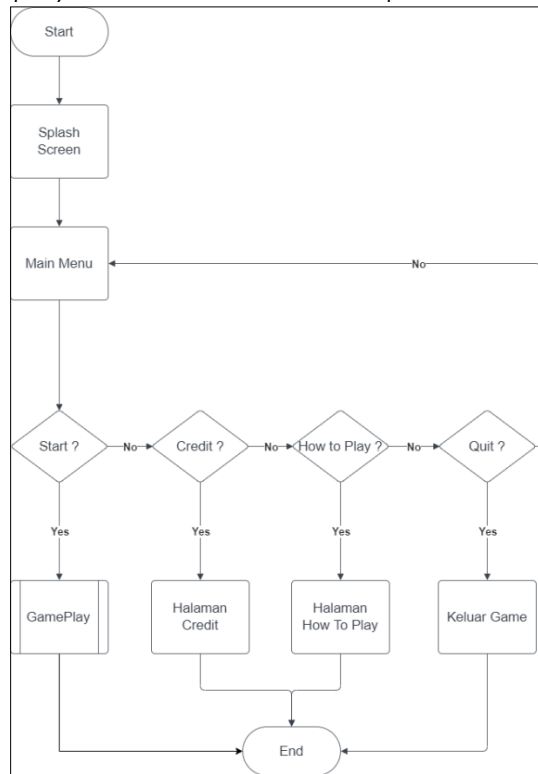


Figure 3. Main Menu Flowchart

3) Player

The player flowchart explains the player flow in the game. In the game players can move, shoot, reload, and pick items.

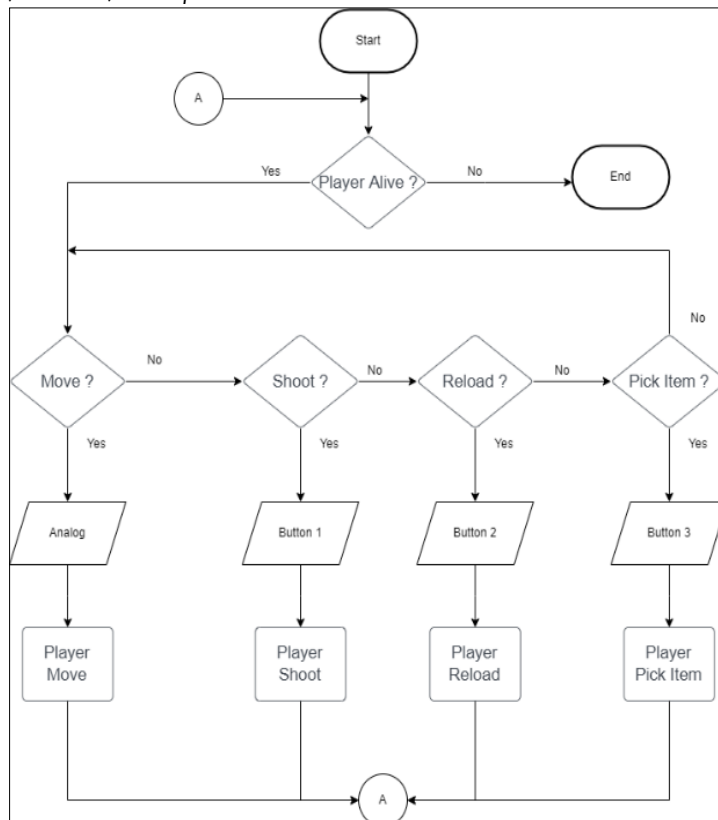


Figure 4. Player Flowchart

4) Enemy

The enemy flowchart describes the enemy flow in the game. Enemy will look for the player when it is first spawned, if the player is in the enemy's attack range, the player will be attacked.

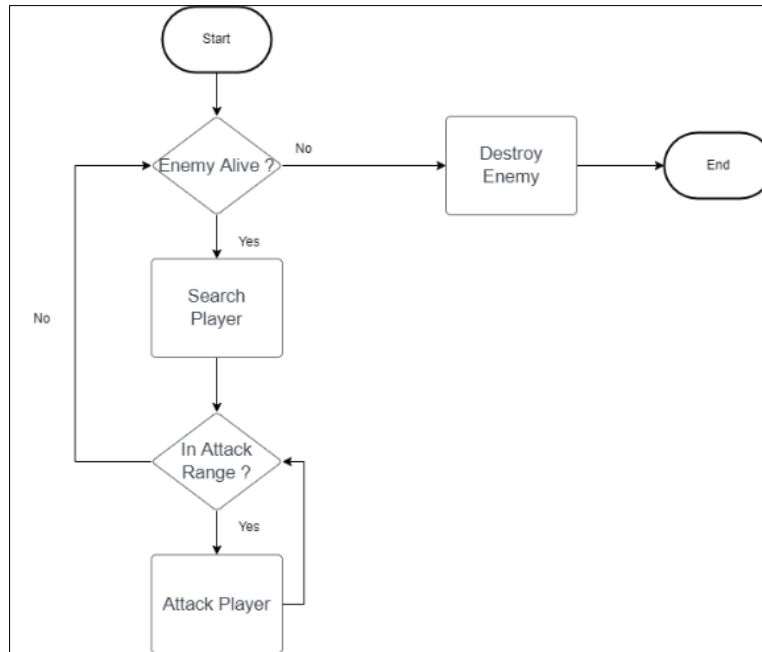



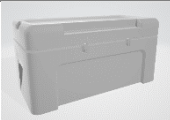



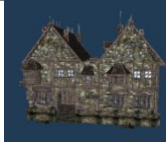

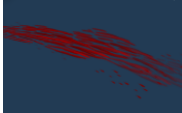

Figure 5. Enemy Flowchart

5. Asset

The following table are assets that used in making a game.

Table 1. Assets

No	Name	Image	Description	Source
1	3D Model NPC		NPCs residing in city	assetstore.unity.com
2	3D Model Ammo		Items to add ammo	assetstore.unity.com
3	3D Model Flower		Quest items for finish the game	assetstore.unity.com
4	3D Model Healthpacks		Items to add player blood	assetstore.unity.com
5	3D Model Pistol		The gun used by player	assetstore.unity.com

6	3D Model Town		City used in game	assetstore.unity.com
7	3D Model Zombie		Enemy in game	assetstore.unity.com
8	Blood Effect		Blood effect if enemy is shot	Assetstore.unity.com
9	MuzzleFlash Effect		Shot effect when Player shooting	assetstore.unity.com
10	Gun Reload Sound Effect	-	Sound when reloading	youtube.com
11	Gun Shot Dry Fire Sound Effect	-	Sound when the player shoots but the bullet runs out	youtube.com
12	Cinematic Trailer Background Music	-	Background music on game	youtube.com
13	Realistic Gunshot Sound Effect	-	Sound when player shoots	youtube.com
14	Zombie Sound Effects (free)	-	Zombie sound	youtube.com

3.3. Game Development

At this stage, the game will be created using the Unity game engine and the C# programming language according to the design that was created at the design stage.

3.4. Testing

Testing will be carried out by giving respondents a game that has been created to play, after the game has been played the respondent will be asked to fill out a questionnaire that has been created in accordance with the Game User Experience Satisfaction Scale method where the questionnaire that has been filled in by the players will later be used to measure the level of player satisfaction with games built with procedural content generation using cellular automata, the number of samples to be taken in the game testing process is 30 samples in accordance with the combination research method carried out by Sugiono .

3.5. Evaluation

Conclusions and evaluations will be carried out when the testing phase is complete, at this stage conclusions will be drawn and evaluation of the games that have been made based on the questionnaire data obtained.

4.0 RESULTANTS

4.1. Map Generator Implementation

GenerateMap function is code snippet to do map creation. In making map, the map will be filled with cells that have random state after that will be checked against the neighbors that are owned to determine the state of each cell in the next generation or iteration and then spawn prefab will be done. RandomFillMap function is code snippet that works to do filling the map with cells randomly based on the value of the randomFillPercent variable, the process of filling in the map randomly is done by comparing the pseudoRandom value with randomFillPercent, if the pseudoRandom value is less than randomFillPercent then map[x][y] will be worth 1 if pseudoRandom is more than randomFillPercent then map[x][y] will be 0. The

result of the randomFillMap function where the map will be filled by cells that have random states.



Figure 6. RandomFillPercent Result

GetNeighbour function is a code that serves to calculate the total neighbors of a cell by comparing the value of the neighbor variable with the grid variable. Rules function is used to determine states of cell based on its neighbourhood, if neighbor of cell is more than 4 then the state cell will be 1, if the neighbor is less than 4 then the state cell will be 0. Map result after GetNeighbour function and the Rules function is executed.

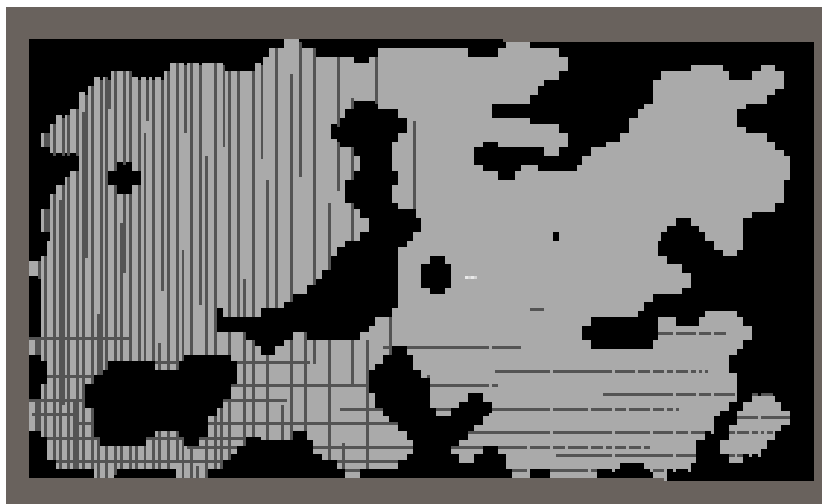


Figure 7. Cellular Automata Result



Figure 8. Prefab To Fill Map

Final result of the map that has been spawned Prefab.

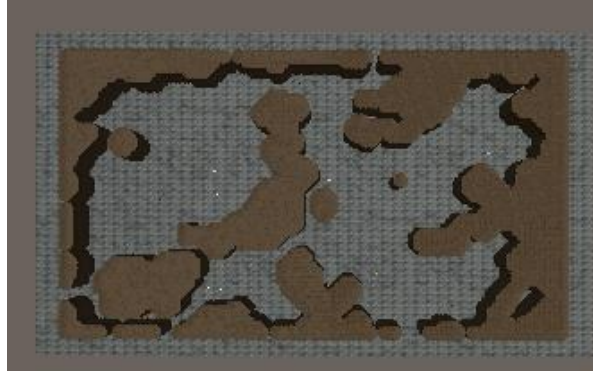


Figure 9. Final Result Map

4.2. Horror Game VR Implementation

Implementation result of splash screen where on this page there is the title of the game and the unity logo below it. After the splash screen page, players will be directed to the main page menu.



Figure 10. Splash Screen Implementation

Implementation result of the main menu page where on this page players can choose 3 options, namely start, credit, how to play, and quit



Figure 11. Main Menu Implementation

Implementation result of the credit page where on this page there is the name of the game maker.

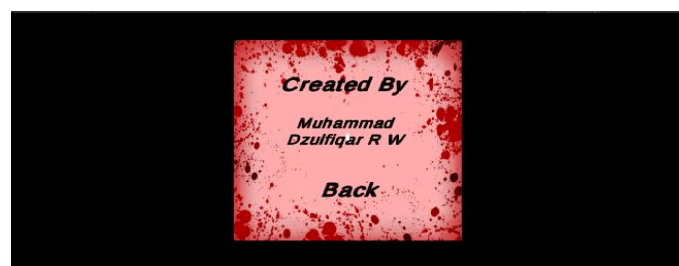


Figure 12. Credit Implementation

Implementation results from the how to play page where on this page there is information on the controller used for playing the games.



Figure 13. How To Play Implementation

Implementation result of the prologue page where this page will appear when the player selects start on the main menu page, on this page contains the story of the game.



Figure 14. Prolog Implementation

When the player has finished reading the prologue and presses the start button then the player will be directed to the city and told to talk to the NPC.



Figure 15. NPC Implementation

When the player has finished talking to the NPC, the gate will open and there is an arrow above it indicating the player must enter to the gate that has been opened.



Figure 16. Door Implementation

After the player enters the gate, a landing page will appear to give instruction to player, the instruction will give player information about what plant should player collect.



Figure 17. Instruction Page Implementation

Map results generated by the cellular automata algorithm.

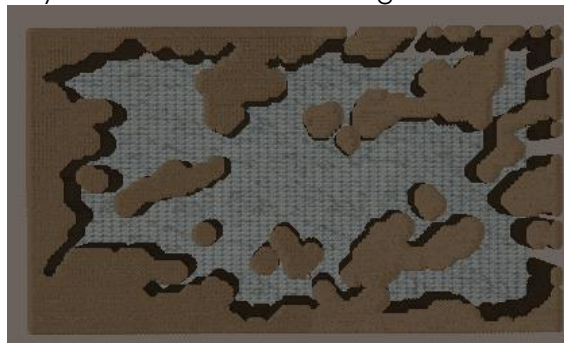


Figure 18. Map Implementation

After the player presses the start button on the instruction page and the map has been created, the game will start.



Figure 19. FPS Horror Implementation

When in the game the player can collect 3 medicinal plants to win the game.



Figure 20. Medicine Flower Implementation

Implementation result of the game over page where this page will appear if the player dies in the middle of the game, on this page the player can choose two options, namely retry to repeat the game and also exit to return to the main menu page.



Figure 21. Game Over Page Implementation

Implementation result of the winning page where this page will appear if the player manages to get to the three flowers that are requested by the NPC on this page there is an epilogue and also an exit button for back to main menu.



Figure 22. Game Finish Page Implementation

4.3. User Satisfaction Test

After the game has been made, testing is carried out by distributing questionnaires that have been made based on GUESS-18 to 32 respondents who have tried the game, the evaluation results from 32 respondents are as follows:

Table 2. Guess-18 result

Construct	Result
Usability/Playability	79.9 % (Good)
Narratives	70.8 % (Pretty Good)
Play Engrossment	74.8 % (Good)
Enjoyment	75.4 % (Good)
Creative Freedom	74.1 % (Good)
Audio Aesthetics	78.3 % (Good)
Personal Gratification	79.3 % (Good)
Visual Aesthetics	84.2 % (Very Good)
Final Result	77.1 % (Good)

Based on the completed questionnaire, the final score for the game that has been made is 77.1%.

5.0 CONCLUSION

Based on the research that has been done, it can be concluded that the cellular automata algorithm has been successfully implemented into a virtual reality horror game using the Unity 3D game engine with the C# programming language. The game that has been successfully built implements several features such as the quest system and shoot system to

support the game, implementation cellular automata algorithm is used to create maps automatically. After the design and development of the game has been completed, 32 respondents were tested using the GUESS-18 method to measure the level of player satisfaction and obtained an average value of about 77.1 percent, it can be concluded that the games that have been built can provide a good experience for players.

ACKNOWLEDGMENTS

The authors would like to thank Universitas Multimedia Nusantara for the support and facilities given for this research project.

REFERENCES

- [1] Vuontisjärvi H, "Procedural Planet Generation in Game Development," 2014.
- [2] M. Hendriks, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural content generation for games: A survey," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 9, no. 1, 2013, doi: 10.1145/2422956.2422957.
- [3] C. Fajri, "Tantangan Industri Kreatif-Game Online di Indonesia," *J. ASPIKOM*, vol. 1, no. 5, p. 443, 2012, doi: 10.24329/aspikom.v1i5.47.
- [4] J. Sampurna, W. Istiono, and A. Suryadibrata, "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.
- [5] 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.
- [6] I. Parberry, "Designer Worlds: Procedural Generation of Infinite Terrain from Real-World Elevation Data," *J. Comput. Graph. Tech.*, vol. 3, no. 1, pp. 74–85, 2014, [Online]. Available: <http://jcgt.org/published/0003/01/04/>
- [7] A. A. Permana, A. T. Perdana, and Y. E. Ramadhan, "Mobile Educational Game of Animal Guess in Android Platform," *JIKA (Jurnal Inform.)*, vol. 6, no. 3, p. 317, 2022, doi: 10.31000/jika.v6i3.6811.
- [8] M. Andersson and S. Kvarnström, "Procedural Generation of a 3D Terrain Model Based on a Predefined Road Mesh," 2017, [Online]. Available: <https://gupea.ub.gu.se/handle/2077/53338>
- [9] G. Smith, "The future of procedural content generation in games," *AAAI Work. - Tech. Rep.*, vol. WS-14-16, no. Persson, pp. 53–57, 2014, doi: 10.1609/aiide.v10i3.12748.
- [10] S. Risi, J. Lehman, D. B. D. Ambrosio, and K. O. Stanley, "Automatically Categorizing Procedurally Generated Content for Collecting Games," *Proc. Work. Proced. Content Gener. Games 9th Int. Conf. Found. Digit. Games (FDG-2014)*, 2014, [Online]. Available: http://eplex.cs.ucf.edu/papers/risi_pcg14.pdf
- [11] G. N. Yannakakis and J. Togelius, "Experience-driven procedural content generation (Extended abstract)," *2015 Int. Conf. Affect. Comput. Intell. Interact. ACII 2015*, pp. 519–525, 2015, doi: 10.1109/ACII.2015.7344619.
- [12] S. Putra and W. Istiono, "Implementation Simple Additive Weighting in Procedural Content Generation Strategy Game," vol. 4, no. 12, pp. 9–18, 2022.
- [13] R. Lara-Cabrera, C. Cotta, and A. J. Fernandez-Leiva, "Procedural map generation for a RTS game," *13th Int. Conf. Intell. Games Simulation, GAME-ON 2012*, pp. 53–58, 2012.
- [14] "Game Programming Gems 3.pdf."
- [15] R. Andrian, A. A. Permana, and A. Kusnadi, "Design Adventure Role Playing Game With Procedural Content Generation Using Perlin Noise Algorithm," *J. Theor. Appl. Inf. Technol.*, vol. 101, no. 9, pp. 3653–3665, 2023.
- [16] E. R. T. Harsaya, A. A. Permana, and Y. Khaeruzaman, "Design Application Simple Learn Based on Mobile With Implementation Gamification for Learning Online," *J. Theor. Appl. Inf. Technol.*, vol. 101, no. 6, pp. 2255–2262, 2023.
- [17] M. B. Temuçin, İ. Kocabaş, and K. Oğuz, "Using Cellular Automata as a Basis for Procedural Generation of Organic Cities," *Eur. J. Eng. Res. Sci.*, vol. 5, no. 12, pp. 116–120, 2020, doi: 10.24018/ejers.2020.5.12.2293.
- [18] C. Anagha Zachariah, G. Pankaj Kumar, D. R. Umesh, and M. N. Arun Kumar, "Terrain generation from user text using cellular automata," *Int. J. Recent Technol. Eng.*, vol. 8,

- no. 3, pp. 8041–8045, 2019, doi: 10.35940/ijrte.C6425.098319.
- [19] C. Adams, H. Parekh, and S. J. Louis, "Procedural level design using an interactive cellular automata genetic algorithm," pp. 85–86, 2017, doi: 10.1145/3067695.3075614.
- [20] M. Cook, J. Gow, and S. Colton, "Danesh: Helping bridge the gap between procedural generators and their output," *Proc. 7th Work. Proced. Content Gener.*, pp. 1–16, 2016.
- [21] L. Johnson, G. N. Yannakakis, and J. Togelius, "Cellular automata for real-time generation of infinite cave levels," *Work. Proced. Content Gener. Games, PC Games 2010, Co-located with 5th Int. Conf. Found. Digit. Games*, no. May, 2010, doi: 10.1145/1814256.1814266.
- [22] T. H. Apperley, "Genre and game studies: Toward a critical approach to video game genres," *Simul. Gaming*, vol. 37, no. 1, pp. 6–23, 2006, doi: 10.1177/1046878105282278.