

Computer Games Development

Project Report

Year IV

[Student Name]

[Student Number]

[Date of Submission]

[Declaration form to be attached]

**Contents**

[Acknowledgements 2](#_Toc54714373)

[Project Abstract 2](#_Toc54714374)

[Project Introduction and/or Research Question 2](#_Toc54714375)

[Literature Review 2](#_Toc54714376)

[Evaluation and Discussion 3](#_Toc54714377)

[Conclusions 3](#_Toc54714378)

[References 4](#_Toc54714379)

[Appendices 4](#_Toc54714380)

# Acknowledgements

I would like to thank the following people who assisted in completing this project including;

John Doe of ACME who kindly agreed to …

I would also like to thank ICME for use of ….

Use this template when writing your research report. As a rule of thumb, the report should be of the order of 10 pages (about 250 words/page).

# Project Abstract

Replace this text with an appropriate Project Abstract.

This section should introduce the problem domain and clearly identify, justify and explain the solution(s) chosen. Care should be taken to ensure that the summary clearly demonstrates the writer’s expert understanding of the problem domain.

The chosen project will be on procedural generation of a maze-like dungeon system that will be made of rooms and will generate when the player enters a cave in the over world. These will generate a 2D array of different values that will represent the different tiles in the dungeon system.

The maze will be generated with a number of different rooms that the player can enter and these can be connected with a number of corridors that connect the rooms.

The procedural generation system will create rooms, corridors, walls and background tiles to create a dungeon system that will be different each time a user enters these dungeons. These will be used in the game as high loot areas and are procedurally generated so as to have a different layout every time a player enters the tunnels.

I used the version in the game as it creates a layout of different characters. These are laid out to be like rooms and corridors that the player can play in. The use of characters allows an handy way to create these dungeons and are laid in a way that makes them easier to work with.

The system creates these rooms and stores them in a vector and these can easily be transferred to a 2D array that I used to create tile maps and allows the system to be created with a tile map and to then draw these with ease.

The tile map class in the first build took a 2D array and in a switch checked the type of value in the array and depending on the value it created a tile with a certain sprite and placed it in the game.

The procedural generation system creates the values in a certain order so as to make it easier to add them to a 2D array form a vector and to then change it from a char to a int and store them in the 2D array. They are then passed to the tile map to create the tiles and to also draw these tiles so as to make it easier to use.

This overall is the main reason that the version of the procedural generation system that I used.

# Project Introduction

Replace this text with an appropriate Project Introduction.

Present relevant background or contextual material and define any terms or concepts when necessary.

Here you present to the audience what you are doing and why it is important. In essence, please provide an introduction to the project, why was it chosen, the potential impact of this research. You should state a research question (if any) and present the project objectives. This will most likely be a concrete question probably from one specific area, such as AI, Networking, Graphics etc.

E.g., Research Question Example (Networking): What is the effect of threshold size in the dead reckoning approach on player performance and player experience?

---------------------------------------------------------------------------------------------------

The project that I have chosen to do this year as the main is a project that will have a dungeon procedural generation system that will be the focus of the project. The reason for this choice is due to an interest to learn these systems and create a game that utilises a procedural generation system

The game will be designed like the original Legend of Zelda and will be based on a map that the players can explore. The overworld will be the main world that the players can explore and there will be enemies here along with a town where the players can buy better gear and upgrades. This map will use a tile map to generate the world and will also have different terrains with this map.

The procedural generation will be used in the creation of dungeons that the player can find in the overworld and then will explore to find the end where a boss might be there for them to fight. The boss could then drop loot that will make the player have different experience as they play more.

The goal is to give an understanding of procedural generation to myself and anyone who may find this report helpful in the area of procedural generation for the different dungeons that can come from such a system.

My hope is to create a cool dungeon system that will be made of different designs. The dungeon will have walls doors and floors that will make up the dungeon and the system will generate a possible layout for these.

My goal is to do research on the procedural generation of tunnels and to learn the best and quickest way to generate the tunnels in the game so as to make a quick and cool system that can generate the tunnels.

# Literature Review

Replace this text with an appropriate Literature Review.

The literature review places your research in context. You aren’t the first person to investigate or research a particular topic. Present a short literature review with the following goals:

* Give the reader a good overview of the key concepts;
* Describe the most relevant work (in your own words) that other people have done in this area;
* Use proper academic writing with references.
* Show how the existing work influenced your project.

----------------------------------------------------------------------------------------------------------------

# The main goal is to create a procedural generation system within a Legend of Zelda style of game that will generate a dungeon when the player enters the caves / dungeons from the over world.

The goal is to create a system that will have different layouts for the dungeon and each time there will be a different style and layout to the dungeon.

The dungeon will also have loot for the player to use elsewhere in the game and they can also have bosses that the player can fight. These can be unique bosses such as bosses that can be only defeated a certain way.

The dungeon can also be maze like to allow the player to get lost but there could also be a way to create a way to allow the player to keep track of their position and to make it easier to get through the maze.

There is also a state machine to be put in to allow the animation of the character and the different states that the player may be in. These can be walking to attacks that the player can use in the game.

The state machine will also manage the player movement and the different states that the player will have such as if the player attacks an enemy and various other.

# Evaluation and Discussion

Replace this text with Results and Discussion.

Describe the results using diagrams such as graphs etc. as appropriate, and discuss what the results mean.

Example: Results indicate that once the threshold gets over a certain point it significantly reduces player performance and player experience

## Project Milestones

### ***The Game Loop***

The first mile stone was to make a game will a game loop and at the very least to have a player character that the player can control and move around. Also needed to animate the player and to get some the states for the player in.

This was to be done by the week of the 16 November 2020. The player moves and is animated using a Finite State Machine and the tile map for the overworld has been started.

More was done than what was expected and the tile map code is also nearly done as all it needs is the positions of the tiles and to also add more tiles to the world. At the moment there is only grass, dirt, water but more are too added to the tile map.

The tile map will be generated when the game starts and will only be done once.

The animations and the player movement were the focus at the start and they have a very basic movement but it works for now as the goal was to get the Finite State Machine working and that was achieved.

### ***The Dungeon***

Started the procedural generation of the dungeon system that will generate a text-based system that represents the dungeon. This was started on the week of the 1st December 2020 and has been made to generate a dungeon.

The start of the dungeon was completed and generating a dungeon each time the game was run. The dungeon has some issues that has to be worked on and these are the likes of the game not generating a dungeon every time the game is run and this will have to be fixed as the project continues.

The goal is to get the procedural generation completed and generating a dungeon for the game as soon as possible and to have it so there is at least a system to show the different way the dungeon is laid out.

There has also to be a bit of work to be done to the finite state machine so as to fix any issues that the state machine has on it. There are issues with it that need to be fixed.

### ***The Dungeon Decorations***

The next major milestone was to create the was to decorate the dungeon and to get there being basic decorations in the rooms. This was to be done after the unique rooms were created but were done without an order at first.

The goal was to create different rooms such as a boss room, worship room etc and then to create the unique rooms such as the library and the coffin rooms that will have different decorations from the other rooms. Then decorate the remainders with random decorations and this would be done last to the remaining rooms.

However, because this was started first all the rooms looked the same when decorated. This convinced me that the different and unique rooms were needed to fill the level out and make it so that there were different looks to some of the rooms.

Each room would have the same decorations and these were things like potions, money, skeletons that will make up the decorations of each of the rooms. Some were given a spawn along the walls and others were given random spawns in the room so as to make it that they didn’t spawn near the walls of each of the rooms. These were completed by 1st February 2021.

### ***Dungeon Generation Refactor***

There were a lot of issues however decorating the dungeon on the one layer and so it was going to be needed to be redesigned. The decorations would need to be stored separate from background tiles and so once the layout of the room was created the vector was copied and used to determine where tiles were and what type was at certain positions.

The background tiles and the décor tiles were stored separate and so it was easy to pass these to the tile map and to then generate the map based on these. The background was rendered first and the decorations was then rendered after to allow the decorations to render before the background tiles.

This made also made it easy to change tiles for the decorations without the need to change the tiles for the background and so allowed these to be kept separate. If a tile was to be checked then the decorations vector can be checked and changed if needed.

When these were then passed and used by the tile map, they were cleared to stop them taking up space in memory as they were no longer needed once the tile map was created in the tile map class. This were completed by 29th February 2021.

### ***Dungeon Generation Different Rooms***

The next types of room that were created were rooms that had unique decorations in them and rooms that were completely different form the original rooms.

The rooms that were completely different were rooms such as a boss room. Worship room and a statue room.

the boss room had a hole in the middle, skeletons and a health that the player can pick up and use. The hole is where the boss can spawn and the skeletons can be of pass heroes that were killed by the boss. This can be done to give it a story and to make exploring the dungeon cool and fun.

The statue room can have a statue of the boss that can act as a warning to the player of the boss and this can also be cool and would add to the story of a game.

These rooms will be generated in the dungeon and will be of different sizes and place at different areas in the dungeon. These can be generated and add to the cool look of the dungeon.

These rooms will be generated at the beginning and they will be got from the rooms vector and these will be given a certain size to be. They can be generated as many times as you want as long as they are within the size required. This were completed by 15th March 2021.

#### **Unique Rooms**

These rooms will be generated after the different rooms and will be made of different décor but will not give the player any information.

These will simply make the dungeon seem different and will make it so there are different room that will make up the dungeon. These will have unique and similar decorations to the other rooms in the dungeon and are there to make the dungeons rooms feel different

These can include rooms like the Feast room, the library, prison etc. These are generated after the boss rom and the before the room decorations are added to each of the other rooms.

They will not have any story to them and are there to simply fill up the dungeon. They were completed by 25th March 2021.

#### **Halls Rooms**

These were decorated will a small number of decorations and were made to have decorations to make them fell like they are not empty and to give colour to them as well. They were now able to have plants, money and potions in them to fill the rooms and to make it feel less empty.

#### **Other Decorations**

There were also other types of decorations added to the dungeon generator such as the ability to have chair next to potin tables and to also have chairs around the feast table.

There also exits / spawns in the walls of some of the rooms. These were in all rooms but were removed from the boss and different looking rooms.

There were also decorations added to the walls and they were the likes of lamps, pictures and chains to make the wall have decorations and to feel interesting looking.

The lamps can be used to create lights to the room and these can have their positions stored in a vector to be used by other objects.

There were also traps that can harm the player and these were set up as triggers and the tiles can be tagged as obstacles if you wanted to create impassable walls and make some of the decorations as also impassable. These can have colliders added to them to do this and can be used as well for a pathfinder. These can be stored and passed around if they are needed by say pathfinders or colliders etc.

***Procedural Generation Examples***

During the research of the project, I came across a number of different dungeon generation articles that talked about and gave examples of the use of procedural generation in games and more precisely the use of it to generate dungeons.

(Gamasutra The Art and Business of Making Games, 2015) talks about the use of procedural generation and how they use it to create random dungeons and how they their algorithm works. They talk about using the algorithm to randomly generate dungeons using normal distribution in the creation of the dungeons.

(Gamasutra The Art and Business of Making Games, 2015) also talk about the use of tiles in the creation of the dungeon and how they have to keep all room positions and the sizes of these to a certain multiple.

In their example the tile size is 4 so the size they have to keep to is a multiple of 4.

(Gamasutra The Art and Business of Making Games, 2015) next part is to separate the rooms as they were all generated close together and have to be moves away from one another. They use a physics engine to do this over a separation steering behaviour and so they separate the rooms so as to not have these over lapping.

(Gamasutra The Art and Business of Making Games, 2015) designs their rooms in a way that makes them ideal to the type of gameplay that they have in the game and so as to make it more ideal to the game they are making.

They also prevent the generation and separating becoming too tall by generating the rooms positions in a very narrow strip that they can separate from better and not as tall as the other way they did it. They then selected what were rooms and what were not rooms based on how big the size of the rooms was and did this by selecting rooms with a width greater than 30.

(Gamasutra The Art and Business of Making Games, 2015) then uses the Delaunay Triangle + Graph top take in points and then gives back triangles. These were then added to a graph that they had in their project and they also gave the rooms ids. They use the graph to create a minimum spanning tree. This is used to make sure that all their rooms are reachable and that they are not connected to one another and give them a good distance between them.

(Gamasutra The Art and Business of Making Games, 2015) then create hallways that connect the nodes in the graph and they add these if they are close enough on the horizontal and the vertical and add hallways if they are. These start as lines. Any rooms that are colliding with the lines are then added o a structure that holds these rooms. They then add one tile sized grid cells that make up any missing parts.

(Gamasutra The Art and Business of Making Games, 2015) example is a very sophisticated one that had a lot of work in it and ultimately wasn’t used as a reference when I was creating the game.

The example was a bit too complicated for myself and I believed that I would have spent a lot of time simply learning how this works. It was however a good read and allowed me to learn about this approach to procedural generation.

(BlackThornprod, 2018) version of procedural generation that I came across was interesting and was created with unity and involved generating a maze with a beginning and a end to the maze. He created rooms by placing a room and doing a number of checks.

The first he did was to check whether there was a door in the 4 directions. Then placed a position marker in that direction to determine where the position of the new room was. Then choose a door to place and made sure there was a door on the new room that was in the opposite direction of the door that was placing a room in the direction. then if the new room had a door in the right direction it was placed.

In essence it would create the rooms till it reached the end and made sure all doors on each of the rooms had a room it led to and so there were no doors into the void.

(BlackThornprod, 2018) stores the rooms in a array in a script and accesses the rooms form the array depending on the room that’s needed e.g. if the current room has a door that down then he needs a room from the top door array to get a room that has a door at the top. There can be a number of rooms in this array such as a room with doors in all directions, or left, or right, or down etc. They however have one thing in common and that’s there is a door that is at the top of the room. He does a check like this for all the doors that are placed as the script is run.

To prevent a room spawning on another room he adds a box collider to prevent it spawning a new room. The collider is a trigger and when there is a collision then the new room is destroyed if there was a room placed and then there will be no rooms that will spawn on top of already existing rooms.

(BlackThornprod, 2018) then fixes issues with doors having openings into the void and does a check to make sure that the spawner that has a bool that says nothing was spawned and the other spawner that was destroyed also has a spawner that’s false then he creates a bit of wall to close the room. These he calls closed rooms. He then has a problem with there being closed rooms spawning on the start room then he fixes this by destroying these using a script to destroy them

Another thing is that he can have certain rooms spawn more often than others and he does this by adding more of the type he wants more of in the array that it belongs to.

(BlackThornprod, 2018) then creates an exit and does this by creating a list. He adds the rooms to the list as they are created and this means that the last room in the list is the exit room and creates a boss in this last room that he has in the game. This makes it easy to know where the last room is and makes it easier to spawn a boss.

In the end I thought this was an interesting way to do the procedural generation in my own game but due to the lack of corridors I did not use it. I also had planned to use a tile map class early and this was not using a tile map. I preferred a system that would generate the values for the walls, floors and corridors using numbers that can be stored in 2D array and used to create a tile map. I thought it was interesting but chose not to do mine this way.

**Major Technical Achievements**

What are your major technical achievements?

**Project Review**

What went right? What went wrong? What (if anything) is still outstanding/missing (i.e., still left to do)? If starting again, how would you approach this project differently? What advice would you have for someone attempting a similar project in the future? Were your technology choices the right or wrong ones? If you chose the wrong technology, provide justifications for why you think this. What were the implications of your technology choices?

# Conclusions

summarise your work and findings.

**Future Work**

Indicate what might be some next steps to try (if a student next year was going to undertake a project in this area what might be an interesting thing for him/her to examine?).

# References

# Appendices

Replace this text with Appendices.

This might include ethics application and other relevant material e.g. copy of any questionnaires used.