3D Procedural Cave Generation

**October 2023**

**By**

**Louie Waterfall**

**Student number 202115597**

**Word count: 2265**

Contents

[1. Project background and purpose 3](#_Toc149171028)

[1.1. Introduction 3](#_Toc149171029)

[1.2. Objectives 3](#_Toc149171030)

[1.3. Scope 3](#_Toc149171031)

[1.4. Deliverables 3](#_Toc149171032)

[1.5. Assumptions 4](#_Toc149171033)

[2. Project rationale and operation 5](#_Toc149171034)

[2.1. Project benefits 5](#_Toc149171035)

[2.2. Project operation 5](#_Toc149171036)

[2.3. Options 5](#_Toc149171037)

[2.4. Risk analysis 5](#_Toc149171038)

[2.5. Resources required 6](#_Toc149171039)

[3. Project methodology and outcomes 7](#_Toc149171040)

[3.1. Initial project plan 7](#_Toc149171041)

[3.1.1. Tasks and milestones 7](#_Toc149171042)

[3.1.2 Schedule Gantt chart 8](#_Toc149171043)

[3.2. Project control 9](#_Toc149171044)

[3.3. Project evaluation 9](#_Toc149171045)

[4. References 10](#_Toc149171046)

# Project background and purpose

## Introduction

This project will focus on the procedural generation of caves within computers, it will explore methods of generating terrain in order to procedurally generate a cave system. Using a graphics API such as OpenGL in C++ to render and generate a procedural cave. It will compare different methods of procedural generation, their different purposes as well as their suitability for this project, what procedural generation is will also be researched and its differences compared to random generation.

## Objectives

This project aims to develop a working program that implements procedural terrain generation for caves and displays it through the use of a 3D environment. This will be useful as a project for developing the research of procedural generation of caves specifically rather than just procedural terrain generation in 3D. The motivation of this project is to further advance procedural generation in computer games/programs and more specifically procedural cave generation.

The primary objectives of this project are, these are sorted in order of priority, as are the secondary objectives:

* Implement realistic procedural generation of a cave system using L-Systems.
* Implement a level of realism inside the rendered cave for the walls/floor/ceiling using Perlin Noise.
* Render procedurally generated cave in 3D, this includes shadows, lighting and texturing to make the cave comparable to a real-world environment.
* Implement basic collision detection for the cave walls, floor and ceiling in order to be able to move through the cave without exiting the bounds.
* Create a program with user controls in order for the 3D environment to be observed properly.

If there is enough time during the project, the secondary objectives:

* Create alternative first person character perspective that allows the player to walk around in the cave system.
* Implement procedural terrain generation in 3D using Cellular Automata and compare to other methods used.
* Implement procedural generation at runtime to make the 3D world “infinite” and generate as the user moves, this includes making the environment into “chunks” to aid performance.
* Add extra graphical enhancements to the 3D rendering such as more advanced lighting, advanced texturing, shadows etc.

## Scope

The purpose of this project is to delve into the 3D aspects of procedural cave generation, there are lot of similarities between 2D and 3D procedural generation but only 3D will be involved with this project. There are also a lot of different methods of procedurally generating terrain in computer graphics, the ones being looked into with this project will be Perlin Noise, L-Systems and optionally Cellular Automata. As mentioned, this project is about procedural cave generation in 3D, normal world terrain generation is similar to cave generation, but this will not be included in the project. Procedural generation is often used in video games, this project is focused on the technical side and will not use the procedural generation to create a game.

## Deliverables

The main deliverable will be a program that implements procedural cave generation within a 3D environment that is rendered using a graphics API, it will have user input for moving around the environment. This deliverable will be met if the final program displays a procedurally generated cave within a 3D environment that can be viewed by the user, the cave must be rendered properly and use shadows and lighting to make the cave generation visible. The generated cave must be the same each time based on the same input seed; it must not be random.

## Assumptions

One of the big assumptions when starting a coding project is assuming what technology/methods can be used to achieve what you want. With this project I want to procedurally generate a 3D cave system, having superficially researched methods to do this I have assumed that the methods of generation I’ve chosen are good for my project. Further down the line I might realise that they don’t generate the kind of cave generation I want, for example I assume L-Systems Trees would help generate a tree like structure, but maybe I would need to use a different method for this.

Another big assumption is compatibility of different technology/methods, at this stage I’ve chosen what I intend to use and research further, I assume these will all work well together to create what I want. It may be that these do not work well together and/or actually conflict with each other. For example L-System Trees and Perlin Noise may not be good to use in conjunction with each other. The actual technology such as the language, 3D graphics API may also be an issue later down the line if they don’t work well together.

# Project rationale and operation

## Project benefits

This project aims to further my knowledge of procedural generation in computers and 3D graphics, specifically for use in video games or real world imaging. It aims to research the implementation of a procedural cave generation system and to research the different methods of that generation, it also aims to make comparisons between available methods. This all benefits my knowledge of the topic and how to make use of it, whilst this project focuses on research, it is not necessarily new research, this project will likely not enhance public research of the topic. There is already tons of research carried out by teams of people on this topic, this project is unlikely to touch on any of that, additionally this technology is already in use by many popular video games, it isn’t new in 3D/2D graphics, it has existed for a long time, this project will just be touching the surface on the field.

## Project operation

For project management I will use the waterfall model to keep it more simple, as it’s a one person project less management is needed in terms of assigning tasks/meetings etc. I will make a Gantt chart of tasks in order of priority, it’ll be more or less spending time on each task and then seeing if more/less time was needed and then planning for the next task. I will measure the success of the project by evaluating the final product against the SMART objectives defined in this document, if most of the primary objectives are complete then it is a success, if the secondary objectives are started then even more of a success. I will also measure how well the tasks were completed as part of the project planning process.

## Options

Procedural generation in 3D is quite a common feature in video games as well as real world visualisations, this means that there are a lot of options for tools/techniques. For my project I will choose the best techniques by researching what each technique does and whether it is applicable for my use case. As for tools, they are less related to the use case and more about which one is the best, the big thing for this project in tools is rendering the 3D graphics, I’d need to choose the best graphics API and then a language that can use that API.

For procedural generation techniques there are a lot of techniques available, L-System Trees, Simplex Noise, Perlin Noise/Worms, Cellular Automata, Accidental Noise etc.

For tools there are also a lot of options, graphic APIs include OpenGL, DirectX, Vulkan and more, OpenGL. There is also the programming language to consider, C++ is very common in computer graphics, however C#, python etc can also be used, as can many other languages.

## Risk analysis

With a project over this period of time there are a number of risks associated with it, below is my analysis of those risks and mitigations against them. The table displays the risks and the raw impact of this risk, after the mitigation the likelihood and severity make up an impact of that risk with the mitigation in place.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Risk** | **Raw Risk** | **Mitigation** | **Likelihood** | **Severity** | **Impact** |
| 1 | Hard drive disk failure | 10 | Use version control and make development environment available from more than one computer. | 3 | 1 | 3 x 1 = 3 |
| 2 | Illness | 4 | Have extra time in project in case issues happen along the way, don’t fill up whole timeline with tasks. | 3 | 2 | 3 x 2 = 6 |
| 3 | Tasks take more time than estimated | 6 | Try to estimate tasks as best as possible and leave extra time in case tasks take longer. | 4 | 1 | 4 x 1 = 4 |
| 4 | Certain technologies don’t work well together | 8 | Properly research technology beforehand, have alternatives readily available. | 2 | 4 | 2 x 4 = 8 |
| 5 | Inexperience with programming language or technologies | 4 | Assign more time for programming tasks, spare extra time as well. | 2 | 2 | 2 x 2 = 4 |

## Resources required

As this is a widely researched topic there are a lot of technical resources already available as well as research available. In terms of technology the main ones necessary are a graphics API and a graphics framework. They are both readily available and can just be downloaded of the internet, OpenGL is the graphics API that will be used and GLFW is the framework, these are both pretty standard and easy to access. If they for some reason are not available the project can use an alternative API/framework, there are many available out there.

# Project methodology and outcomes

## Initial project plan

## Tasks and milestones

Below is a list of tasks planned for the entire project, it indicates milestones which will be monitored to be able to see the progress of the project. The deliverables mentioned previously are also bound to tasks here.

|  |  |  |
| --- | --- | --- |
| **ID** | **Task** | **Description** |
| 1 | Literature search | Search for relevant literature and read them to understand about the subject and be able to compile a review. |
| 2 | Write literature review | Write the literature review document using researched literature. |
| 3 | Milestone 1 | Literature review completed. Literature review is also a deliverable. |
| 4 | Rendering with OpenGL | Create basic rendering with OpenGL in order to get started with program. |
| 5 | Implement basic texturing | Basic texturing for rendered objects. |
| 6 | Implement basic shadows/lighting | Very basic shadows/lighting for objects in the scene. |
| 7 | Create user controls with camera | Camera and controls in order for scene to be explored. |
| 8 | Milestone 2 | Project rendering and basics finished, ready for procedural algorithms. |
| 9 | Investigate L-Systems | Look into algorithms for L-Systems and make decisions on what is needed. |
| 10 | Algorithm to generate cave tree | Implement algorithm to L-Systems tree for the cave system. |
| 11 | Render generated tree | Render the cave tree generated from the L-System algorithm. |
| 12 | Use Perlin noise for cave rendering | Add terrain variations and randomness to the previously rendered cave using Perlin noise. |
| 13 | Implement collision detection for cave bounds | Collision detection for the procedurally generated meshes. |
| 14 | Milestone 3 | Project implementation completed. Program is deliverable. |
| 15 | Test/evaluate program | Test and evaluate the program based on requirements. |
| 16 | Propose changes | Decide on changes that should be made to the program. |
| 17 | Write project report | Write the final report for the project. |
| 18 | Milestone 4 | Project completed. Report as deliverable. |

## Schedule Gantt chart

For the project task planning a Gantt chart has been made, below are screenshots of the project plan in Microsoft Projects. There are 4 milestones and deliverables for each part of the project and there are 3 sections marked. Weeks are used for the duration of tasks, 1 week assumes that out of 7 days 2 of those days is work on the project for at least 40% of that day, so a 4 week task is about a full week worth of work.

A screenshot of a project

Description automatically generated

## Project control

To manage this project properly I will be following a the Gantt chart above and working on the tasks in the allotted time frames, day-to-day I will be keeping up to date with the current task and tracking my progress on it using the chart. The project’s performance will be monitored by marking progress towards tasks on the Gantt chart and by marking tasks as completed. The project will be successful if all the tasks are completed and the requirements are met by the implementation.

## Project evaluation

This project’s artefacts, the procedural 3D cave generation program, will be evaluated against the project’s primary objectives, secondary objectives and the task list. If the program meets all the objectives and all tasks have been completed then the project is a success. Other project deliverables such as the literature review and final report will be evaluated to see if they provide good research into the project topic and give a good explanation of the project as a whole.

Seeing as this project is focused on creating a piece of technical software that isn’t focused on users there won’t be a need for user evaluation. However, the software will still be evaluated, in more of a measurable way rather than an opinionated way. The aim of the evaluation will be to look at the primary objectives of the project, as well as the secondary objectives, and compare the final deliverables to those objectives. A good way of evaluating the objectives will be a comparative method, for example one of the objectives mentions the layout of the cave system being realistic, a map of a cave generated from this project can be evaluated/compared against a real-world map of a cave.

This sort of evaluation will be used for each of the primary objectives in order to measure how well they were accomplished and whether they were done to a standard that means the project has achieved its goal.

Additionally, in order to test if the final deliverable actually works/runs properly, I plan to implement some basic unit tests and other tests on the program itself, this will involve writing testing code to make sure the program does what it is supposed to do. For example, a test about the procedural generation algorithm might involve testing to see if the output is within certain constraints, such as paths generated from each branch etc.

# References

De Castro, L.N. (2006) Fundamentals of natural computing: basic concepts, algorithms, and applications. Boca Raton: Chapman & Hall/CRC (Chapman & Hall/CRC computer and information science series).

Prusinkiewicz, P. and Lindenmayer, A. (1990) The algorithmic beauty of plants. New York: Springer-Verlag (The Virtual laboratory).