**Assessment submission 1: Dissertation Proposal**

|  |  |
| --- | --- |
| **Student name:** | Calum Lindsay |
| **Student number:** | 21010093 |
| **Project title:** | Where Have L-Systems Been Under-Utilized in 3D Games? |
| **Supervisor:** | Magnus Tullock |
| **Partner college:** | Shetland |

**Project summary/overview**

What is the product? What is your goal? What is the justification for your project? This should include a rough overview of the requirements, both functional and non-functional. There is no need for detailed scoping at this stage. [300 words]

The goal of the project is to explore the existing usage of L-Systems to generate content for 3D video games, propose and create solutions using L-Systems in areas where other techniques are traditionally used, and pioneer solutions using L-Systems in areas where little to no research has taken place.

Functional requirements:

A comprehensive review of the current literature must be performed to identify areas of greater and lesser exploration. Solutions to relatively unexplored problems must be proposed and developed. Solutions to well-known problems using other techniques must be developed using L-Systems. Experimentation must be performed to produce performance metrics of both the new solutions and existing counterparts. Created solutions must be critically analysed against existing solutions where appropriate. A 3D application must be created to demonstrate the created solutions.

Non-functional requirements:

Existing solutions using other techniques must be acquired or developed to perform performance measurements upon. An interface must be created to allow the tweaking of solution inputs for demonstration purposes. The performance of the created solutions must be reasonable for their intended use-case which may be real-time or offline depending on the specific solution.

Justification:

Lindenmayer Systems are self-rewriting string generation algorithms at heart and were first proposed as a way of modelling the development of plant like structures by Aristid Lindenmayer (<https://www.sciencedirect.com/science/article/pii/0022519368900799>) with later work expanding on this and importantly introducing the concept of a turtle renderer to produce rasterized images, culminating in a fascinating book: The Algorithmic Beauty of Plants (<http://algorithmicbotany.org/papers/abop/abop.pdf>).

Procedural generation in video games allows the fast creation of large amounts of content with smaller development teams which is important as over time “The most popular commercial games get larger, prettier, more atmospheric, and more detailed.” (<https://dl.acm.org/doi/pdf/10.1145/2422956.2422957?casa_token=_uJTWI93H4QAAAAA:EV86R8uCLHtdWp8Te0I1We9TYMyGBx4jWuD3f-2Zteqz6DTaaaYhRFQWUY0x4NP2-5O5d0fJR1am3A> : pg2) thus requiring more and more high quality content which is a challenge when “manual content production is already expensive and unscalable” (“).

Lindenmayer systems have seen limited use in games, mostly being used “for botanic modeling and generation” (<https://www.diva-portal.org/smash/get/diva2:832913/FULLTEXT01.pdf> : pg8) with famous examples that use SpeedTree being Battlefield 3, The Witcher 2, Gears of War 3(“), Elden Ring and Ghostwire: Tokyo (<https://en.wikipedia.org/wiki/SpeedTree>), however, their usage has been fairly limited in other areas such as terrain and scenario generation, and finding solutions in these unexplored areas could prove valuable if they improve current methods in an established area or if they open up an area that hasn’t been extensively investigated.

**Previous work**

Short summary and discussion of previous work known (not a full literature review yet). [200 words]

Extensive literature exists for solutions that use L-Systems to produce plants and trees beginning with the book: The Algorithmic Beauty of Plants (“). Further research has combined genetic algorithms to evolve plants (<https://link.springer.com/chapter/10.1007/978-3-319-90418-4_2>), implemented dynamic animation of the growth of plants using L-Systems to produce key frames within the animation and interpolating between these (<https://nccastaff.bournemouth.ac.uk/jmacey/OldWeb/MastersProjects/MSc09/Hampshire/thesis.pdf>), produced L-Systems that simulate the death of a plant (<https://www.niclab.mx/portal/sites/default/files/SemanticDeathInPlantSimulationUsingLindenmayerSystems_0.pdf>), and generation of 3D plants using L-Systems (<https://www.scitepress.org/PublishedPapers/2009/17853/pdf/index.html>).  
  
Other papers have used L-Systems for level generation (<https://www.jait.us/uploadfile/2024/JAIT-V15N2-276.pdf>), generation of cities (<https://web.archive.org/web/20060114082225/http://www.vision.ee.ethz.ch/~pmueller/documents/procedural_modeling_of_cities__siggraph2001.pdf>) (there’s another..), volumetric spaceship generation (<https://dl.acm.org/doi/pdf/10.1145/3520304.3528775?casa_token=7MOs98nNai0AAAAA:uSQgCc8x3ttin0TiPVI1fvd0pSMtNW2dfx58NTSt3GDJ9bWYb59Fse3slkaIeJgihCmeyD9tb-hsBQ>), and generation of river deltas (<https://core.ac.uk/download/pdf/322445609.pdf#page=132>).

**Technologies**

Proposed technologies to be used (with student’s level of experience/expertise with each) and justification (including brief discussion of alternatives). [100 words]

Using a game engine will significantly reduce the workload compared to writing all the rendering code from scratch which will allow more focus to be given to the main task of researching and developing L-System based algorithms.

Unreal engine 5 has been chosen due to its extensive use in the industry with Epic Games’ own titles such as Fortnite, Rocket League and Fall Guys as well as big hitters like Kingdom Hearts III (<https://www.unrealengine.com/en-US/spotlights/unreal-engine-spotlight-kingdom-hearts-iii>) and Hogwarts Legacy (<https://www.unrealengine.com/en-US/developer-interviews/why-avalanche-worked-to-deliver-a-hogwarts-game-with-soul>).

There are other options such as Unity and Godot. In this case it would not make much difference which of these options was used due to the student having no experience with any of them, however, they do have extensive experience with C++ which is what Unreal Engine is written in and a pre-existing desire to learn Unreal Engine.

C++ will be used to create the procedural generation code as it easily integrates with Unreal Engine and it is a high-performance low-level language well-suited to producing code that may run on a critical path such as during game environment initialization.

**Development methodology**

Proposed development methodology and justification (including brief discussion of alternatives). [200 words]

As this project is quite large, does not have a clearly defined destination, and the timescales of individual tasks are not reasonably predictable the simple waterfall method where timelines and tasks are planned then plotted out ahead of time doesn’t work particularly well. Instead an iterative and incremental development methodology will be used as these provide more flexibility and are “associated with many successful large projects, and recommended by standards boards” ( [https://www.craiglarman.com/wiki/downloads/misc/history-of-iterative-larman-and-basili-ieee-computer.pdf pg 10](https://www.craiglarman.com/wiki/downloads/misc/history-of-iterative-larman-and-basili-ieee-computer.pdf%20pg%2010)).

A specific named methodology will not be used as these are typically designed for teams creating commercial software instead of lone developers without involved clients. The chosen methodology will use a Kanban board to track what tasks are pending, in progress and complete, and Scrum inspired sprints will be used where tasks are chosen each week based on a ranking of their estimated value. The pending column of the Kanban board will serve as this ranking and will be updated each week reconsidering the value of all the tasks in the list as well as uncompleted tasks from the previous sprint which may be moved to pending if their estimated value drops below other tasks. New tasks will be added to the pending column as they are formulated.

**Project plan/timeline**

Proposed timeline with major milestones. (No need for a formal Gantt chart, although that could be good. Whether it’s text or a diagram, this needs to be clear and reasonably detailed.) [100 words or equivalent in diagram]

Oct – Nov – Literature review

Oct – Nov – Familiarization with technologies to be used

Oct – Dec – Initial development of demonstration application.

Nov – Jan – Exploration and prototyping phase.

Jan – Apr – Development of suite of final demonstration algorithms and integration with demonstration application.

Apr – May – Comparison of results to previous techniques, compilation of dissertation outcomes and final write up of dissertation.

**Supervisor meeting arrangements**

How often? When? How? (Phone? Skype? Face to face?) We need a commitment to a weekly meeting.

(Every / Every Second) (Wednesday / Thursday) Afternoon for (30 mins / 1 hour) using Microsoft Teams.

**Ethics**

Have you read the UHI Research Ethics Framework? Have you completed a REC1 or REC1-D Student Application for Ethical Approval form and submitted it along with this proposal? Are there any ethical risks in your project?

I have submitted a student application for ethical approval through the online portal. There are no ethical risks associated with my project.

**Reference list**

List all references cited above, following the UHI Student Referencing Standard.