Procedural Content Engine

Technical Design Document

## Development Team

This procedural content engine was created by Will Pye and uses several assets from outside source. These assets are listed in the Assets subsection of the Game Design Document.

## Technical Summary

The Procedural Content Engine was developed over a three week period with only one person working on it. The only software necessary for the project is the Unity game engine. There is no production cost or revenue expected as I am just a mere student making a procedural world.

The game will be deployed for PC.

## Hardware

The only team member used both his desktop and laptop for this project, neither of which were acquired for the project itself and thus there is no cost for hardware.

## Software

The only software used for this project are the Unity game engine and git. The procedural engine was created in the Unity game engine and git was used for version control. All software involved in this project is free.

## Engine

The game engine utilized for this Procedural Content Engine is Unity as it is necessary for this class and has easily manipulatable classes for creating custom meshes. Unity also has a large store of assets that are used to populate the mesh.

## Complexity

The majority of the scripting for this project is found within CreateMesh.cs. This script creates a grid of 100 x 100, calculates a value for the y coordinate using Perlin noise, and then creates a mesh with this information. The script updates and changes the mesh when the player moves so that the map is constantly being built around the player. Once the grid is created and applied to a mesh, the script adds trees and shrubs to the map using the vertice coordinates previously calculated. The program runs smoothly with a 100 x 100 grid, but increasing the grid just slightly increases the computational load significantly. CreateMesh.cs runs its intensive functions whenever the player moves one meter in any direction. A new vertex grid and list of triangles are created and applied to the mesh with a complexity of O(n^2). On these updates, trees and shrubs are placed where applicable and are also checked to see if their coordinates fall within the square box around the player of length 100. This process is directly related to the number of trees and shrubs currently on the map, with a complexity of O(n). The assigning of plant positions happens during the creation of the grid, in which each coordinate is checked to see if it is suitable for a plant.