**Investigating a real-time Hydraulic Erosion Simulation to be used for terrain generation in games**

1 Introduction

This portfolio project will contain an investigation into real-time hydraulic erosion when creating realistic terrain in video games and a program that will use techniques found in the investigation to create a real-time hydraulic erosion simulation. Terrain is one of the most important aspects of emulating a realistic virtual experience whether it’s in a computer game that requires a large-scale terrain for an open world experience, a movie that requires a fantasy style location that might be too dangerous to film in or a training simulation that requires the most realistic environment to help train people.

1.1 Aims and Goals

What needs to be achieved

* Aims – quite broad
* Objectives – very specific

2 Literature Review

General tips:

* Use diagrams, but be aware that for important stuff for your project you really need to explain in your own words

Extra sections here:

* Real world stuff
  + Include diagrams, pictures etc (all fully referenced)
* How have scientists modelled this?

2.1 Terrain

(INTRO CAN BE FINISHED LATER)

Generating realistic terrain for games is a long and time-consuming process. To solve this problem developers have found quicker and more effective ways for generating realistic terrain. When talking about terrain generation the first step is how that data is represented. Terrain data can be represented with two main models. These being a volumetric model and an elevation model.

2.1.1 Terrain Representation

The elevation model can be described with either an elevation function or a discrete heightfield. The most commonly used method is the discrete heightfield. The discrete heightfield uses a two-dimensional grid to represent the altitude of each position. This means that it is unable to recreate suspended materials like overhangs, caves, and arches. But it is a lot less data heavy which allows a simulation to be ran a lot faster with more area to work with. Elevation functions use a formula that can generate a point of altitude at any point in terrain. This allows for a program to generate multiple points to create a terrain of any size required.

The volumetric model is similar to the elevation model but instead of using two-dimensional space it uses voxels which allow for a three-dimensional space where each cell represents a material at a particular position. This allows for the use of features like overhangs, caves, and arches. Volumetric models are very data heavy which makes them slower to run. There are ways to optimize the data structure by using compression techniques like Sparse Voxel Octrees to reduce the memory cost. This is done using an octree. An octree is where a three-dimensional space gets recursively divided into subspaces of children nodes until each voxel only contains one point or multiple point of similar data. This technique can be used in open world games which require real time interaction.

2.1.2 Terrain Generation

* Make a bullet point list of the main topics that go into this section, e.g.:
* Data sets
  + Geographical
  + artist generated data
* Procedurally generated data
  + Fractal
  + Noise
  + etc

When creating terrain there are two main ways for acquiring the terrain data. These being data sets and procedural generation. Data sets can be created by artists modelling the terrain in a 3D modelling software or by collecting data from the real world. There are also ways for users to input a small amount of detail into a program which then outputs a model based on the users input. Procedural generation is a method where programs generate their own data with no user input to be used for terrain. (start explaining about each type \-(:/)-/ )

Fractals are defined as sets that exhibit self-similarity under scaling. This means when you zoom in on a fractal pattern, the same pattern will repeat itself over and over again. There are many different fractal algorithms that have been discovered each with their own unique properties and characteristics. The first fractal algorithm to be discovered was the Mandelbrot set which was discovered by a mathematician named Benoit Mandelbrot. He categorizes the self-similarity of fractals by a fractal dimension which can be described with the equation:

D = log(N) / log(1 / r)

Where D is the fractal dimension, N is the amount an object can be divided and r being the ratio how much the object has been divided. For example, the von Koch snowflake curve being known as an “early mathematical monster” where a simple line segment is divided into thirds and the middle third is replaced by two segments forming part of an equilateral triangle. At the next stage in the construction each of these 4 segments is replaced by 4 new segments with length 1/3 of their parent according to the original pattern. This then get repeated over and over again yielding the von Koch curve. So with the amount the object gets divided being 4 and the where each section is being scaled down by 1/3. The equation for the von Koch snowflake curve would be D = log(4) / log(3). This means that the fractal dimension equals 1.26.

2.2 Water/Fluid Simulation

(INTRO TO WATER NEEDED). 🡨 make sure to set the context, fluid sim is a vast topic. Write this intro once you’ve done the rest

There are two main methods for simulating water. These are grid based and particle based. Both grid-based and particle-based methods have their own advantages and disadvantages that make them better suited for different scenarios.

What are your main subheadings:

* Grid based
  + Full volume vs shallow water (height maps)
  + Identify the main topics, then think about how to organise them
* Particle based

The most efficient way for simulating water with a grid-based method would be by using a height field which uses a two-dimensional grid with each cell holding a height rather than using a three-dimensional grid with multiple data point to consider for each simulation step. Some methods that use height fields are the shallow water equations which only simulate a thin layer on the surface of the water and the pipe model which uses pipes that connect each cell. (TALK ABOUT SHALLOW WATER). (TALK ABOUT PIPE MODEL). (TALK ABOUT METHODS THAT USE 3D GRIDS).

The most used particle method is the Smooth Particle Hydrodynamics which is based on the Navier Stokes Equations. This method tracks each particle’s location instead of using a fixed grid. Each particle represents a quantity and mass. The quantity is calculated by a weighted sum of the neighbouring particles.

Having a particle-based simulation allows you to only simulate where there is water precent, while having a grid-based simulation requires the entire world to be simulated on each step. This means that particle-based simulating works better with small bodies of water and grid based would work more efficient on simulating larger bodies of water. This has led to hybrid methods that utilize the advantages of both grid based and particle based. A paper made by Nuttapong Chentanez and Matthias Müller investigates creating one such method. They do this by first simulating water on a two-dimensional height field using the shallow water equations. Then they detect when the height field struggles to simulate and generate particles at that point. The particles then later getter removed when they’re enter a point on the height field that can simulate again.

2.3 Erosion Algorithms

(INTRO TO EROSION)

There are multiple different types of erosion in the real world that can be simulated. These include weathering and denudation. Weathering is a small-scale erosion that usually effects stones and rocks. Denudation is an erosion process that mostly results in shaping the surface of an area causing rivers, valleys, or canyons.

Weathering can be split up into two sections. Chemical and thermal. Chemical weathering is the process when moisture makes contact with the surface of a material causing a chemical change. Thermal weathering is the process at which the change in temperature causes material to expand and retracted allowing cracks to form.

Denudation can be split into three sections. Gravity conditioned movement, splash erosion and fluvial erosion. Gravity conditioned movement is when loose material such as soil, gravel or sand moves when it reaches an equilibrium between the gravity and its inner tension. Splash erosion can be described as when raindrops make impact with a material like sand causing an indent in its place. And finally fluvial erosion is the process at which flowing water interacts with the landscape it is flowing down causing material to be remove from the surface being forced down stream with the water.

**Heading: games implemented**

* Talk about specific games and how they implement their methods (where known)

3 Design

o