Eunho Kim

Controlled Procedural Terrain Generation Using Software Agents

**Abstract: This project will attempt to create terrain that satisfy the user’s needs. Generating terrain helps a lot of different causes from entertainment (games) to flight simulators. While giving the user control over the main themes of maps, there will still be an element of novelty. This project will also aim to create larger structures such that the map wouldn’t just be flat. These elements of randomness are included by using software agents. These agents will use a controlled random number generator to help create the map while having constraints.**

I. INTRODUCTION

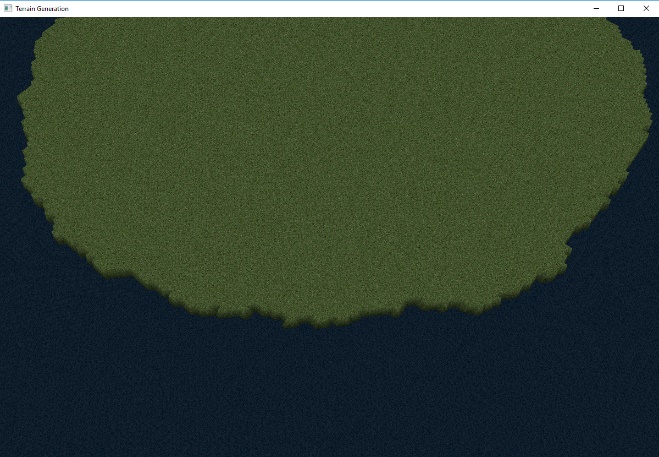
Procedural terrain generation is becoming more and more useful as there is more need for randomly seeded maps. People seek out maps that can be controlled to some extent using user-defined variables. Using software agents in terrain generation is an interesting idea because software agents are random but not completely random. This means that they are random in the sense that in the global scale the points selected are random. However, these agents follow a procedure that can result to less local randomness.

The model is generated on a 256 x 256 vertex height map using the software agents. To view the model, you can use the keys W, A, S, D to move around the model and your mouse left click to change views.

The original papers did not specify on any of the graphics contexts. For rendering I used triangle meshes and for textures I bound free seamless water/grass bmp pictures onto the vectors.

II. METHOD

II.I COASTLINE AGENTS



A coastline is what defines the terrain. While there is not always a defined island on the map (instead the land might be sometimes stuck onto one of the sides of the map) it is a possible formation. This software agent allows for other software agents to create different types of land points such as mountains, beaches and rivers.

Instead of thinking about coastline agents as agents that cut off mass of land into the sea, think of them as agents that bring the land up from the bottom of the sea. This is done by getting a random point in the map covered by water and creating more software agents to perform the landmass adding task. The task is that they each receive two random points on the map, an attractor and a repulsor. These points include the randomness of the land generation that software agents bring. The landmass prefers to be in the middle of the map also. These factors are shown by a scoring system of the land mass. The formula for this scoring system is

where

This formula keeps the landmass from just being an ordinary circle and includes local randomness. It also keeps it a continuous stretch of land instead of multiple different islands.

In my code, there is an option to input the MAX\_SIZE and MIN\_SIZE of the land mass inside the data.txt file located inside the Data folder. Changing these values will increase or decrease landmass prevalence on the map.

The algorithm for coastline generation is given:

1 **for** **each** software agent

2 **do**

3 point = random border point

4 **for each** point p adjacent to point

5 **do**

6 score p

7 generate land mass in the point that has the highest score

II.II BEACH AGENTS



Beach agents are agents that create a smooth transition from the sea to the landmass. While there aren’t beaches on all sides of the coast, it is included in some stretches of coastlines.

In my code, there is an option to input the BEACH\_WIDTH for how far the beach should stretch into the landmass. Changing these values will increase or decrease beach prevalence on the map.

The algorithm for beach generation is given:

1 **for each** software agent

2  **do**

3 **if** height > = landmass height

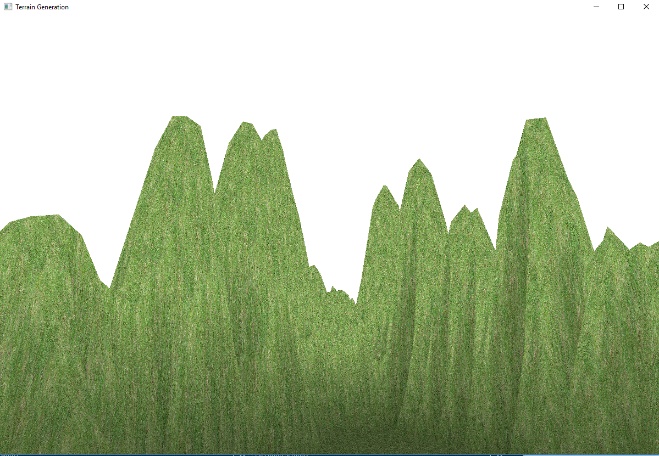
4 **then**

5 location = random shoreline point

6 flatten area around location

7 smooth area around location

II.III MOUNTAIN AGENTS



Mountain agents introduce different elevation to our previously flat landmass. They don’t represent one mountain; they represent one mountain range. Therefore, we must create multiple mountains. We start by finding a location and a direction. Every time we elevate the mountain until it hits the MIN\_ALTITUDE defined inside the data file. The software agents elevate it by elevating a random slope and combining it with a wedge. This combined with the direction change will create a mountain range.

In my code, there is an option to input the MAX\_ALTITUDE, MIN\_ALTITUDE, NUM\_MOUNTAINS, and MOUNTAIN\_WIDTH. for how far the beach should stretch into the landmass. Changing these values will increase or decrease beach prevalence on the map.

The algorithm for mountain generation is given:

1 location = random point on land

2 direction = random direction

3 **for each** software agent

4 **do**

5 raise wedge perpendicular to direction

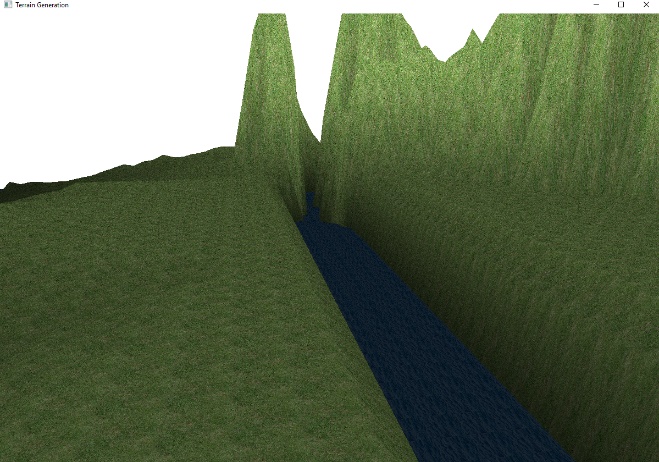
6 smooth area around location

7 location = next point in direction

8 **randomly do**

9 direction = original direction +- 45

II.IV RIVER AGENTS



Rivers are generated between the coastline and a mountain base. Each software agent creates one river from these two points. To create a river, one must lower a wedge of terrain until it reaches the mountain point.

In my code, there is an option to input the NUM\_RIVERS to limit the number of rivers on the map.

The algorithm for river generation is given:

1 cpoint = random point on coastline

2 mpoint = random point on the mountain

3 **while** cpoint **not** near mpoint

4 **do**

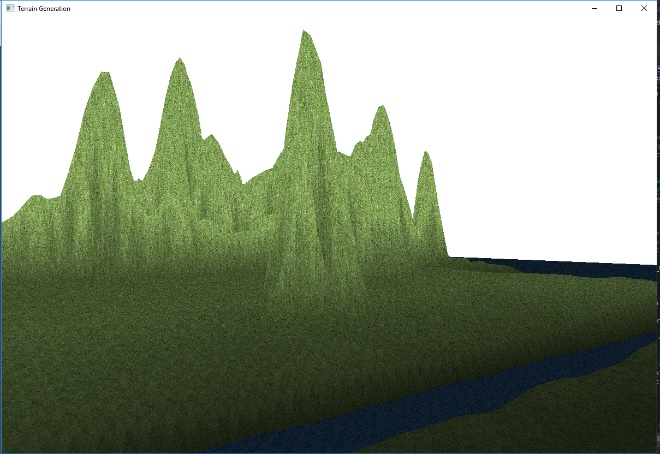
5 flatten wedge downhill

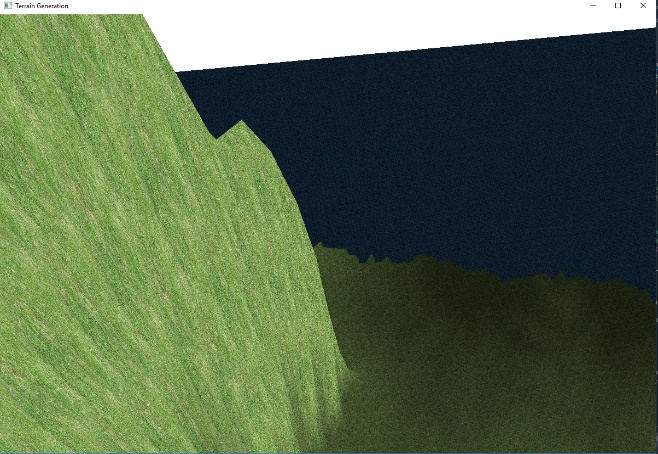
6 smooth area around cpoint

7 cpoint = next point closer to mpoint

III. RESULT

The result of the controlled procedural terrain generation using software agents is a semi-success. There are multiple places where I was not satisfied with. First of all, the algorithms that were given were all very vague and up to interpretation. This resulted in my mountains/rivers to stay in scope of the paper, but also looked bad. I also wanted to use more textures than the grass and water texture.





IV. REFERENCES

[1] Parberry, Ian, and Jonathon Doran. “Controlled Procedural Terrain Generation Using Software Agents - IEEE Journals & Magazine.” *Design and Implementation of Autonomous Vehicle Valet Parking System - IEEE Conference Publication*, ieeexplore.ieee.org/document/5454273/citations?tabFilter=papers.