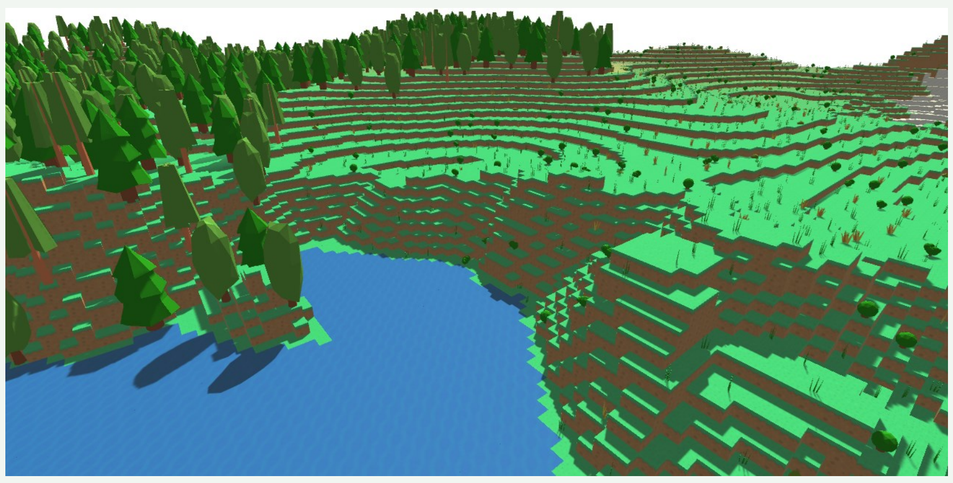
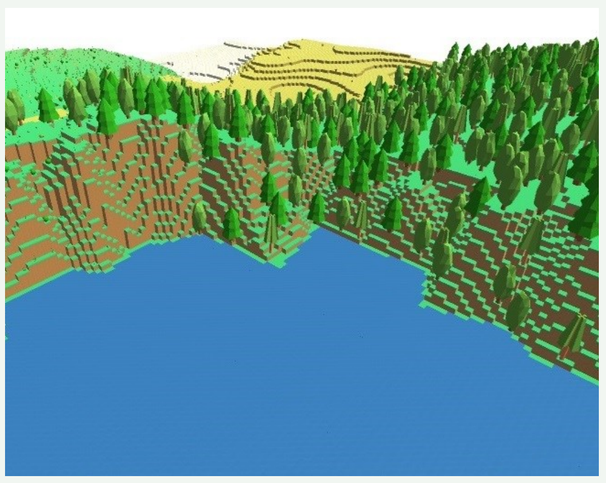
Procedural Infinite Terrain Generation with Noise Algorithms

https://courses.cs.ut.ee/2016/cg-project/spring/Main/ProjectProceduralLandGeneration Demo: https://youtu.be/1FV8VTWAH-g

Manually designing and building virtual landscapes is a dauntingly long and expensive process. To ease such workloads in simulations and video game development, a wide range of procedural algorithms have been developed. This project proposes one such algorithm that is able to theoretically generate an **unlimited** number of distinct **infinite** 3D landscapes. Each terrain is determined by an integer **seed**, which means that the same seed will always generate the exact same landscape. The algorithm's result is visualized with voxels which each represent 1 cubic meter of terrain. The project was developed for a Bachelor's thesis and for the Computer Graphics Project course.





Procedural Terrain Generation

Finite

- Terrain with a **fixed size** is generated.
- Does not have to be generated in real time.
- Allows **easier** implementation of algorithms to generate common natural sights. For example:
  - Rivers: Multiple points on the terrain can be easily selected and a river constructed between them.
  - Erosion: It's easy to check if an area is affected by a nearby mountain range or a large body of water and then the terrain can be shaped accordingly.

Infinite

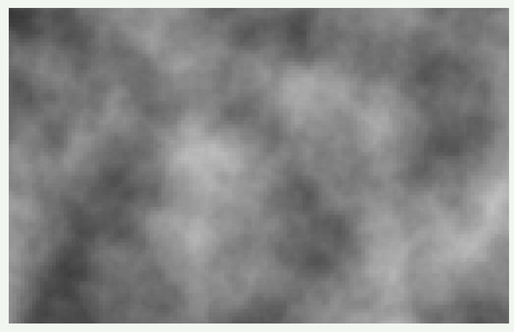
- Terrain is divided into an **infinite** number of chunks and each chunk is generated completely **independently** from other chunks.
- The algorithm must be fast enough to keep up with the user's movement on the terrain in **real time**.
- Implementing common natural sights is a lot harder.
- The main problem: **Very limited knowledge** about rest of the terrain while generating any chunk.
  - While generating a chunk, no checks about neighbouring chunks can be made since they might not exist.

Noise Algorithms

A **noise algorithm** is a function that maps every *n*-dimensional vector to a deterministic, but seemingly random real number. To generate terrain, one such algorithm called **Simplex noise** was used.

Simplex noise has **2 useful properties** for infinite terrain generation:

- The value at any point can be calculated without knowing any other values.
- The values for neighbouring points have a similar magnitude.

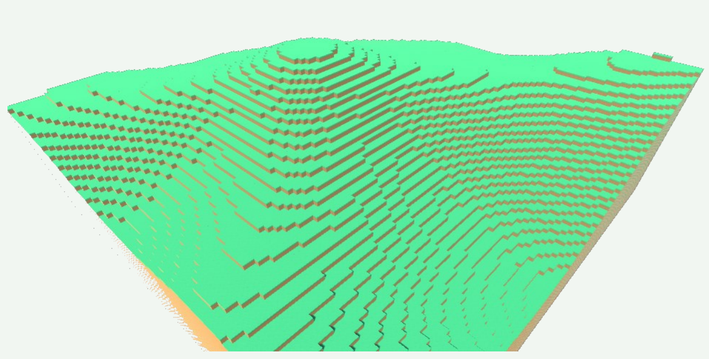


Simplex noise

The Solution

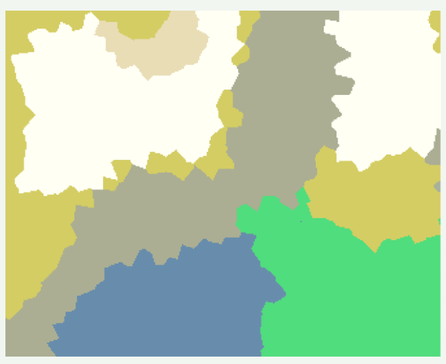
First, each chunk's coordinates are used as input for 2D Simplex noise. Then the noise values are used as a **heightmap** for a terrain. This means that each terrain column is lifted to a height according to the Simplex noise value at that point. Due to Simplex noise properties, this can be done for any point in the world and it is assured that neighbouring chunks fit together even when generated independently.

As terrain with just a heightmap looks quite bland, several other improvements were implemented. Firstly the world is divided into regions called **cells** using a customized Voronoi diagram based approach. Secondly each cell is assigned a **biome**, which determines the height and ground material type in that area. Lastly, suitable plant density for each biome is selected and a number of biome-specific **plants** are placed in each biome.

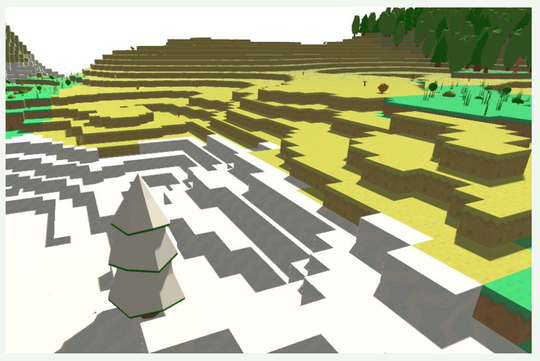


Step 1: Simple terrain with Simplex noise for a heightmap.

Step 2: Terrain segmentation using Voronoi cells.



Step 3: Biomes assignment to cells using Simplex noise.



Step 4: Terrain is further shaped based on biomes and plants are added.

University of Tartu
Faculty of Science and Technology
Institute of Computer Science
Supervisor: Raimond-Hendrik Tunnel

Andreas Sepp
Computer Science
3rd year Bachelor's

UNIVERSITY OF TARTU
Institute of Computer Science
1632

Study IT in .ee
sponsored by Skype™