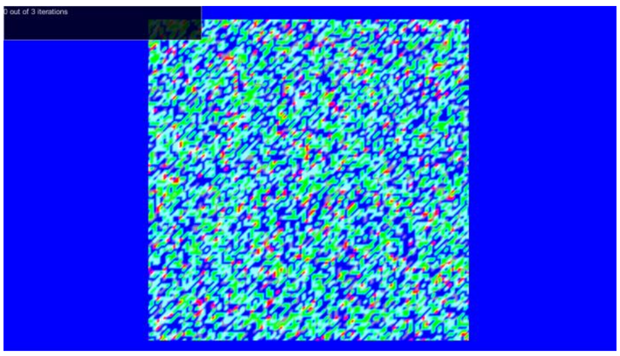
Vulcano Island Generation

Simon Karman
www.simonkarman.nl

For Procedual Content Generation I created a Vulcano Island Generator. Here I will explain how I created the application and why I have made certain decisions:

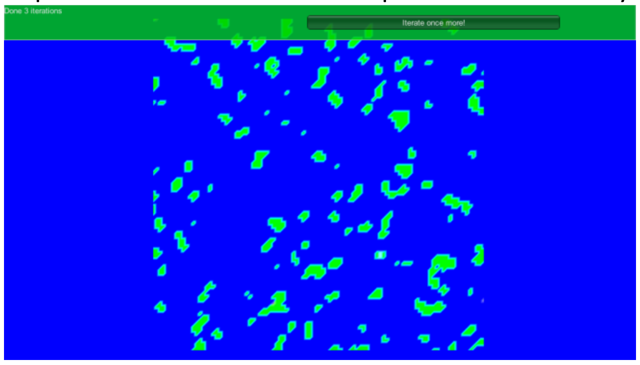
Initial setup

We start with generating a random map of 5 different types of ground.
The groundtypes: Water, Dirt, Grass, Stone and Lava
Each groundtype has it's own spawnchange. Water has the largest and lava the smallest spawnchange.



After generating the random map a cellular automata will be applied to the map. The cellular automata consists of only one simple rule. The rule will be executed on each vertex of the vulcano island.
Rule: The vertex groundtype will change to the most common groundtype in the 3x3 square vertices around the vertex.

This gives a really cool effect, but what it comes down to is that every vertex will turn into water and all other groundtypes dissapear. This result was to be expected since we only use one simple rule.



Next step is to define 'smart' rules to use in the cellular automata. The following rules are used this time:
- Lava check: Every water tile that is located next to lava will turn into stone
- Growing Grass: Every dirt next to grass will turn into grass
- Water Flow: Every tile located next to 2 or more water will turn into water
- All other tiles will still change to the most common groundtype in the 3x3 square vertices around the vertex.

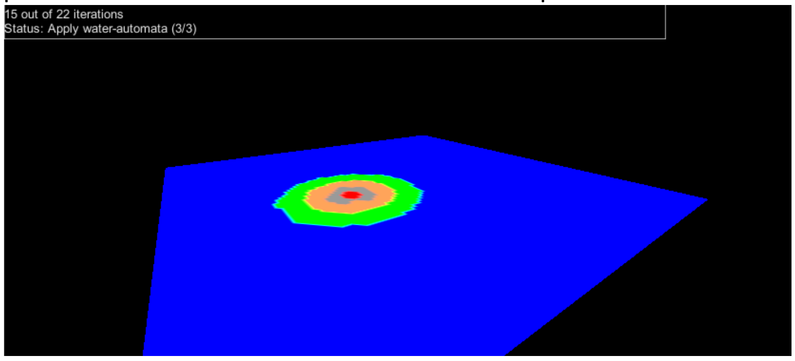
After playing around with the rules for some more time, but without any concrete results or visible improvements, I had to conclude that using a cellular automata to create an desired result from a fully random start is not a good approach. Using a cellular automata gives way to less control over the desired output. There must be smarter approaches for creating an vulcano island using cellular automata.

A different approach

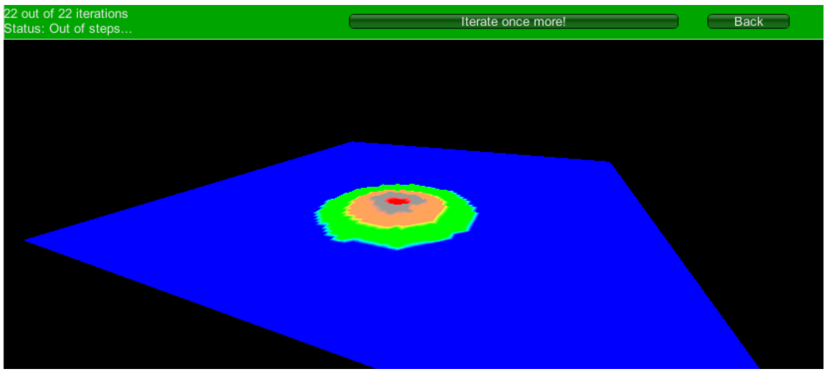
I started working on a system that uses an cellular automata after each step in the progress. In each step we add a new groundtype around the previous groundtype.

Starting with a full water map we execute the following steps.
Step1: Choose a random point on the map. This point will be used as the center of the island.
Step2: Add lava around the island center
Step3: Add stone around the lava
Step4: Add grass around the stone
Step5: Add dirt around the grass

When adding a groundtype around another groundtype we replace all the water with the new groundtype within a certain inner-radius. Between this inner-radius and a given outer-radius water tiles will also be randomly replaced by the new groundtype.
Remember that after adding each groundtype a cellular automata will run that changes each tile to the new groundtype if it occurs more than 4 times in the 3x3 square vertices around the vertex.


15 out of 22 iterations
Status: Apply water-automata (3/3)

After defining the groundtypes of all the vertices the height of the vertices will be randomly. The height is based on the groundtype. After applying a random height the vertices will be equalized. And the water and lava will get liquified. When liquifying all connected vertices of the same groundtype will get the lowest height of the surrounding vertices. Aftewards an other equalize will be preformed on the whole terrain.

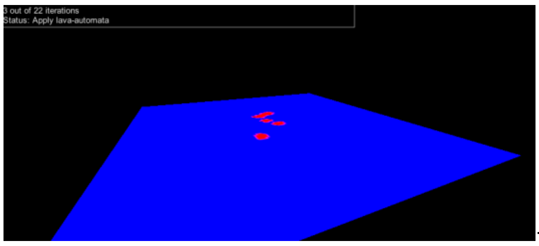

22 out of 22 iterations
Status: Out of steps...

This already looks way more like a Vulcano Island. The problem with the islands that are generated is that they all look like the same and the shapes of the islands are quite boring aswell. From this point on I started adding small improvements and extensions to the generation progress. But I will keep using the above generation sceme as a standard.

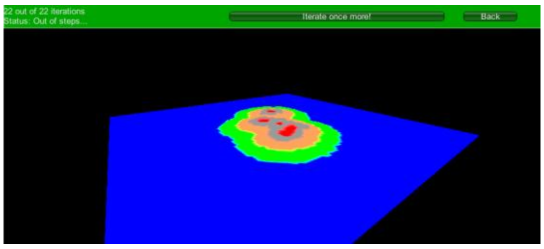
Extensions

Extension 1: Multiple Vulcano Centers

We don't chose one vulcano center, but we will chose more randomly located centerpoints. This way the shape of the island is already randomized a bit.



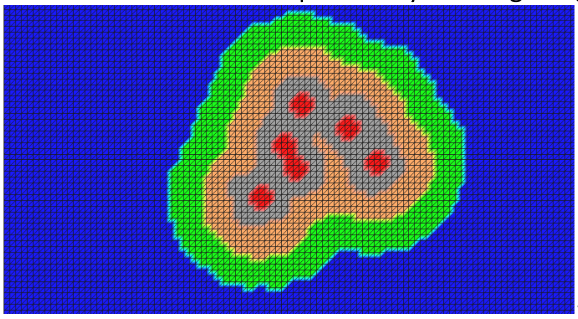
->



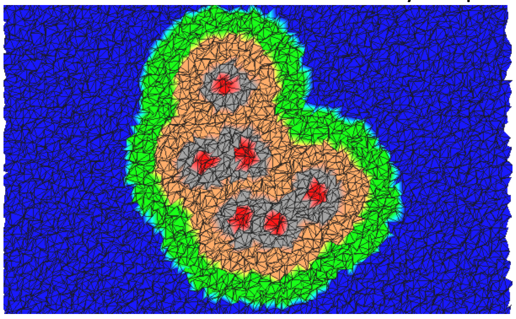
Still noticable is that the layers of groundtypes are of equal size on every side of the island. This is why the island still looks a bit boring and the same everytime we generate it.

Extension 2: Vertex Offseting

Every vertex will be 'shaked' a bit. This means that the vertices will get a random offset in the XZ-direction. As a result the previously axis-aligned grid is now shaked to a more naturaly shaped grid.



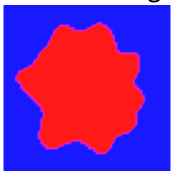
->



Extension 3: Perlin Noise in radius

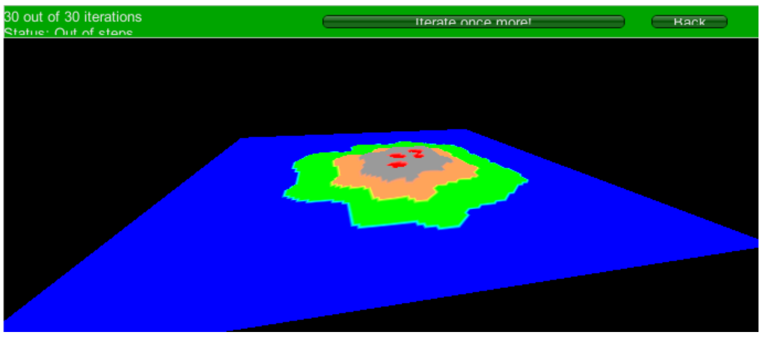
The layers of groundtypes around the island are still around the same size on every side of the island. To counter this even more I added perlin noise to the center radi. This makes the size differ from different angles.

The following images show a circle deformed by perlin noise:


Perlin noised radius with low resolution

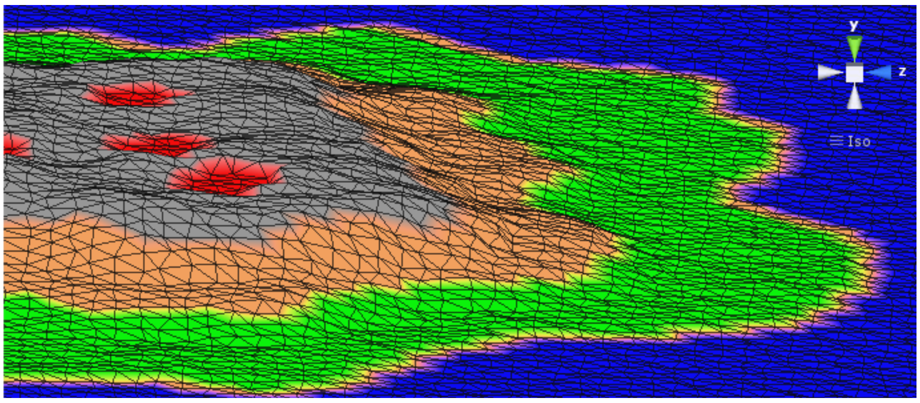
Perlin noised radius with high resolution

When added the above perlin noised radius to the island. We get lot's of inlets along the coastal line. This makes the island look a lot better.



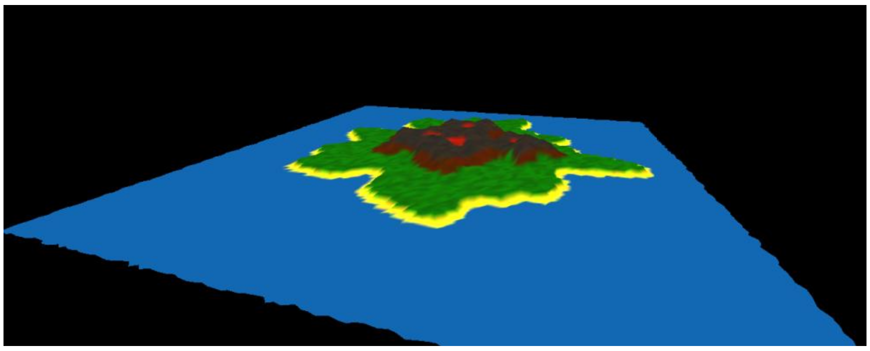
Extension 4: Sand Border

To increase the effect of it being an island I added a sand border around the whole island. This makes the transition from sand to water more clear.



Extension 5: Color of GroundTypes

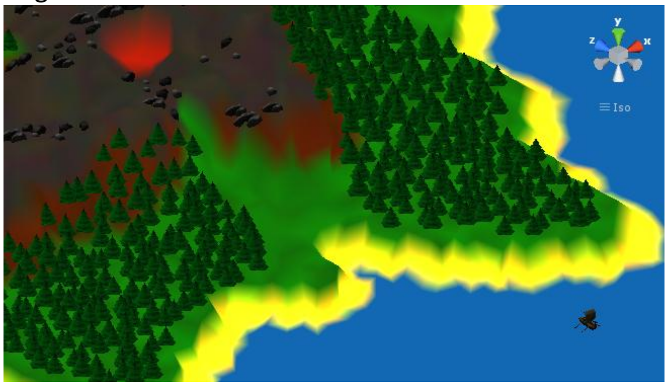
I tweaked the colors of the different GroundTypes. In the image below you can see the result of the new colorscheme.



What you might have also noticed in this picture is that the groundtype colors are not all the same fixed color. Grass isn't just green anymore. The color is randomized to make different kinds of green visible aswell. I didn't randomize the color of the sea to create a beautiful clear blue sea.

Extension 6: Adding Props

To give the island even more emersive value. I added 3 different types of props to the game.
I added trees, boats and lighthouses.



Download

You can create Vulcano Island yourself when you download the application. Using the same seed will regenerate the same vulcano island.
http://www.simonkarman.nl/projects/vulcanoisland