Sahil Dhanju
CSCE 641

Final Project Proposal - **Controlled Perlin Noise for Generating User Defined 3D Terrains**

Description

Perlin Noise is an extremely common method for producing naturally appearing “random” behaviors. One of its most common applications is for generating naturally occurring textures, such as wood or grass, but can also be used to generate animations, landscapes, volumetric objects (clouds), effects (fire) and deformations. Perlin noise can be easily generated for any kind of dimension making it such that applications of noise in one dimension can be similarly applied in another. However one major drawback to this method is an inability to control properties of the generated noise, making it difficult for an exterior agent, like a user, to force a certain pattern or form. The critical problem to be solved is developing a method in which a user can specify the general form of a 3D terrain created with Perlin noise such that the attributes like mountains, hills, plains, caves and valleys can be given by the user and the outputted terrain would follow the given specification while still being considered “natural looking”

Importance

Procedural generation allows the quick creation of an almost infinite amount of unique behaviors. Procedurally generated textures, meshes and animations play a crucial role in media such as games and animated movies. Noise is an important building block in the creation of procedurally generated entities. Controlling the behavior of noise which is going to be the seed for some kind of generation introduces a mechanism where an artist can control the procedurally created output while not having to tediously control every aspect of the generation. For example, having a mechanism to control 3D perlin noise would give a level designer working on a game tool to specify the general form of a procedurally created level while not having to architect the entire level.

Previous Work

The basis of this proposal is comes from the paper *Cheng WC., Lin WC., Huang YJ. (2014) Controllable and Real-Time Producible Perlin Noise*. The paper specifies a way to control the value of the perlin noise function following some kind of user specified through an optimization process. It seems to focus specifically on texture generation as the application. This paper sites *Yoon, J., Lee, I.: Stable and controllable noise. Graphical Models 70, 105–115 (2008)* which has a similar approach.

There are other papers such as *Doran, Jonathon, and Ian Parberry. Controlled Procedural Terrain Generation Using Software Agents.* which specifies a method for controlling randomly generated terrain but doesn’t utilize Perlin noise.

Proposal

Sahil Dhanju
CSCE 641

I would like to take these techniques and apply them to terrain generation which can either be created by treating a 2D noise output as a heightmap, or treat 3D noise as the terrain itself. The user should be able to specify features such as mountains, rivers, valleys, etc. and have the outputted terrain match.

Originality

The application of trying to control 2D/3D Perlin noise with the intent of creating terrain is the key distinction between this proposal and already existing work. This project would test how effective the existing methods are at creating terrain that is generated with specific features but still maintains an “organic” look. The intent is to see how much a user has to specify until they a terrain looks like the specification and if the terrain loses its natural feel, what could be changed in the method to make it adept at user controlled terrain generation.

Goals

1. Somehow acquire the papers mentioned and read up on them. (Unfortunately I can’t find any free way to get them) Also research how uncontrolled perlin noise has been applied to terrain.
2. Create a 2D and 3D Perlin Noise generator with no control. This will involve implementing the method stated in the *Improved Perlin Noise* paper.
3. Write a renderer to view the Perlin Noise by either interpreting it as a heightmap or the terrain itself. The renderer will use OpenGL and rasterization and should have a camera that can be used to pan around the generated terrain. If rasterization doesn’t go well ray tracing will be used instead.
4. Implement the method specified in the above papers to control the noise and try inputting certain images to see if the outputted noise matches what the papers specify.
5. Expand the control to not only take in images but also other formats such as coordinates or patterns to determine terrain.
6. Analyze generated terrain, is it acceptable? What parameters can be tuned? Are there other methods a user can specify control?
7. If all goes well, this method could be applied to other applications, such as controlled fire effects or controlled deformations.
8. Additionally a graphical user interface could be created to manipulate how the terrain should be generated by the user

Steps 1-3 should be completed by the update point and the rest should be completed to some extent by the final point.