Procedural Generation Using Noise

Michael Li

DRAFT 1

Dr.Dianne Hansford	Director
Dr.Yoshihiro Kobayashi	Second Committee Member

Ira A. Fulton Schools of Engineering
School of Computing, Informatics, and Decision Systems Engineering
Spring 2021

Abstract

Procedural Content Generation is a method of creating data algorithmically, often using stochastic models. These methods can be used to generate complex environments as opposed to manually creating environments by hand or by using photogrammetric techniques. Procedural generation can use a variety of techniques to achieve a stochastic or partially stochastic goal, including methods such as fractals, noise, deep learning as examples.

Contents

1. Introduction

This paper surveys various methods of procedural generation and their applications in generating geological formations. While procedural generated content can vary from terrain to creatures and stories, the focus of this paper is primarily on terrain and more specifically geological formations. One of the most famous cases of this procedural generation for geological formations is in Minecraft, which implements a modified version of Perlin noise in order to generate all of its worlds. Other examples can include games such as the Elder Scrolls II: Daggerfall, which employed various forms of procedural generation to determine the location of non-player characters, the layout of dungeons, as well as the terrain itself. In more complex cases, procedural generation can be used to create fake histories, with the more well known example of Dwarf Fortress. However, procedural generation's applications are not only limited to games. In the creation of the Lord of the Rings movies, procedural generation was applied to help create the many different forms of orcs in the movie, as well as ensuring that each orc was able to be animated. While procedural generation is often described as being stochastic, in reality it is not entirely so. The majority of the techniques used for procedural generation include some level of user control, as well as a specified input to use as a starting point for the equation. This input, known as a seed is transformed through code to create the output, in this case geological formations.

The main topic that this paper will cover is the mapping of geological formations. While this can be done through height maps, which are black and white, or colored maps where the colors or intensity of the black/white indicate the height of each point. This acts as a two dimensional representation of the terrain, although it has difficulty in representing structures that have undersides.

1.1 Overview and Definitions

high level overview of how random numbers in computers work - stochastic

define noise high level, use perlin nosie as example

define fractal generation, high level

define cellular automata

define branching/ recursion?

define if statements - semi hard coded

these are ways of representing the maps that are created
define polygons, high level
define voxels, high level, contrast with polygons

2. Mapping

Before the development of more advanced noise generation techniques, some basic procedural generation techniques were used as early as the 1980s. One of the earliest usages of procedural generation was in Rogue which used a three by three grid in order to generate the layout of the level, with hallways randomly connecting the rooms. Some of these techniques were developed in order to bypass the computational restraints of the time.

Perlin noise was developed for use in the movie industry, although it later became a foundation for many other procedural generation algorithms as it provided a lot of control as well as randomness to be used. It was developed in 1983 for use in the sci-fi movie Tron, to map textures onto computer generated surfaces for visual effects. Perlin noise has been used for many different visual elements, ranging from the texture creation it was created for to particle effects such as fire, smoke and clouds, as well as landscapes and geological features. It has a variety of uses due to its ability to create a naturalistic appearance.

3. Height Maps

As mentioned previously, Perlin noise can be used to generate height maps in order to pseudo-randomly create geological formations. While height-maps have the advantage of being two-dimensional and less complicated to run, they have difficulty in rendering a variety of geological formations. Due to the two-dimensional nature of a height map, they are unable to render more complex features such as alcoves, arches, and any other three-dimensional feature. In terms of randomness, this can create a sort of uniformity in geometry, where the only features are hills and valleys. This can be mitigated to some extent by the use of layering Perlin noise.

Implementation and how perlin noise works
While Perlin noise saw great success, it was succeeded by algorithms such as Simplex noise, designed to alleviate some of the problems with Perlin noise. This included the computational complexity and the artifacting in the noise created.
Some of the other uses of height maps include the voxel space rendering system, using voxel raster graphics to display three-dimensional geometry with low memory and processing requirements. This was developed in the early 90's, involving a height and color map to position the pixels on the screen. While this technique was not historically used with noise generating algorithms, the rendering system fits the requirements for the use of techniques such as two-dimensional Perlin noise. At the time, displaying complex height-maps in three-dimensions was difficult computationally, and the voxel space technique allowed this to happen.
Talk about usage of height-maps, weaknesses
some weakness include - only 2 dimensional geological formations, so archways and caves are not possible
might be mitigatable by layering multiple different height maps, but increases computation time
example of practical application in rendering or pre generating voxel space heightmaps

4. Polygons

4.1 Rendering Differences

Constructing polygons from height maps, similar problem