Procedural generation of phased river and terrain geometry for use in a Text-to-Scene conversion system

Bruce Alcock*
Rhodes University

Abstract

This paper uses methods of terrain representation, creation and realism described in literature. In particular a combination of terrain synthesis using Fractional Brownian Motion, and erosion procedures is examined, as well as the procedural formation of rivers via squig curves. Other techniques are examined, such as Triangulated Irregular Networks and ridges generation are discarded in terms of this project for reasons detailed. We find that using a combination of Fractional Brownian Motion, procedural formation of rivers via squig curves to form initial terrain, and hydraulic erosion for post processing, we have full control over the style of terrain: from jagged, mountainous regions to flat regions; and the phase of river from tightly rock controlled to flood plain regions.

1 Introduction

1.1 Problem Statement

In an effort to increase the richness of virtual worlds, the goal of this project is to create realistic, natural terrain with minimal input, but yet to be as configurable as possible so as to allow for a multitude of scenes to be created. The Text-To-Scene system (see Section 1.2) requires that the terrain generator can be given parameters to create different regions of terrain based on what the scene is composed of. Combining techniques and parameter tweaking we intend to produce terrains with different terrain and river phases on the same terrain.

1.2 Background

The creation of virtual worlds from natural language descriptions, specifically fiction text, requires not only the ability to process and derive information from English descriptions, but also the ability create three-dimensional geometry automatically from the derived descriptions. The quantity and quality of such descriptions may vary, depending on the style and genre of the fiction text being processed, and as such the geometry creation procedure must be capable of functioning with very little input.

There are many different aspects to the geometry-creation module of a Text-to-Scene (TTS) conversion system, including the selection or creation of object models to represent avatars and object that interact in the described scenes. In addition, while often not explicitly mentioned in fiction text, such avatars and objects must occur within a specific setting. We differentiate between two classes of setting, namely terrain and city-style settings. In particular, this research examines the automatic generation of terrain geometry for use in a TTS conversion system. In particular, this project aims to address the needs of the TTS converter for terrain, by being able to configure areas for city placement and action areas while being able to fill in the rest.

*Supervised by Kevin Glass and Shaun Bangay

Procedural methods have been chosen for use with this project to fit with the minimal input paradigm, which means that no pre-existing data needs to exist for the terrain generation. Procedural methods also afford a great deal of configurability: since the data has to be created, the implementer can determine how exactly this is done.

This paper uses the approach of synthesizing terrain and then applying realism techniques. Synthesis is the process of creating an initial dataset to describe a terrain, but we find that no proposed technique for doing this creates realistic terrain, and as such we then apply techniques for increasing the realism.

1.3 Overview

This paper is structured as follows: in Section 2 we discuss related work done in the field; in Section 3 we discuss the methods we choose to use, and Section 4 we discuss our implementation of them. Section 5 discusses our results with some renders and conclusions are reached in Section 6.

2 Related Work

Automatic terrain generation exists in two forms, the generation of terrain geometry from real world data [Zhou et al. 2007] and the synthesis of geometry using procedural techniques. While the former is guaranteed to produce realistic looking geometry, we choose to use procedural methods to fit with the minimal input philosophy and not rely on pre-existing data. Procedural methods have advantages of: providing a layer of abstraction, hiding the specific implementation from the user, parameterizing methods to encourage re-usability and variation, independance from the base model, random number usage with seeds to be reproducible, database amplification which is constructing a large amount of data from a small set of inputs, and the ability to replicate naturally occuring processes, which leads to the creation of data structures which reflect the real world better [Morkel and Bangay 2005].

Many different types of procedural techniques exist suitable for synthesising terrain geometry, and can be categorised according to representation, geometry synthesis, and techniques for improving the realism of the generated geometry. Representation refers to the way in which the data is stored and can be heightmaps [Benes and Forsbach 2001a; Benes and Forsbach 2002; Musgrave et al. 1989], voxels [Benes and Forsbach 2001a; Parker and Udeshi 2003], digital elevation models [Zhou et al. 2007] or triangulated irregular meshes [Duchaineau et al. 1997; Fowler and Little 1979]. We use heightmaps due to the regular interval spacing, which makes erosion calculations quicker for example, because the distances are constant; as well as the easy storage into two dimensional arrays.

Synthesis of terrain geometry refers to the process of creating an initial dataset for the terrain and can be subdivision methods [Miller 1986; Prusinkiewicz and Hammel 1993], noise sampling [Musgrave et al. 1989] or feature generation [Belhadj and Audibert 2005; Zhou et al. 2007]. We use Fractional Brownian Motion (fBm)

which is a noise sampling technique because it is entirely reproducible, creates a reasonably believable base to start with and is very configurable due to parameterization.

The use of synthesis alone is not sufficient for the creation of convincing looking terrain geometry. As a result, various techniques exist for simulating real-world phenomena, including hydraulic and thermal erosion [Musgrave et al. 1989], river generation [Prusinkiewicz and Hammel 1993] and multi-layer heightmaps [Benes and Forsbach 2001b; Benes and Forsbach 2001a]. We use squig curve river generation to overlay procedural rivers onto the terrain since they model real world rivers reasonably well and provide control via recursive depth. We then combine this with hydraulic erosion because it simulates the real world process of erosion via water and sediment distribution.

The combination of these techniques mentioned (heightmaps, fBm, squig curves and hydraulic erosion) has not been attempted as yet and this project aims to combine them to produce realistic, configurable terrain. Using the configurability, another goal is to create multiple phases of terrain: with rivers and base terrain being able to go from tightly rock-controlled regions to flood plain regions.

No work has previously been done to specifically create terrain for a TTS system. TTS systems require that regions be configurable: especially for city placement or flattening action areas in the scene and we aim to address these concerns in this project [Glass et al. 2006].

3 Procedural Methods for Terrain Generation

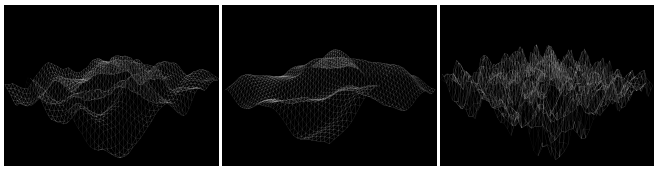
We intend to achieve our goals using heightmaps as a representation and creating a base terrain using fBm. In order to increase the realism, we will then overlay squig curves for rivers, flatten areas for cities or action areas and finally hydraulically erode the terrain in order to smoothly tie all the techniques together and give the terrain an aged look.

3.1 Representation

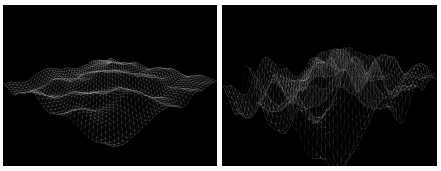
Heightmaps are a two-dimensional grid, each coordinate having an associated value describing the height of the terrain at that coordinate. This method of representation is used for this project due to the simplicity. However we have extended it to include the three erosion constants (see Section 3.3.3) for each position for enhanced configurability: it means each location can have its own set of erosion constants and therefore regions can be harder than others, more willing to take on sediment, and have the ability to hold more water at that location.

3.2 Geometry Synthesis

Geometry synthesis is needed to populate the heightmap with a set of values for an initial terrain. fBm is a synthesis method that takes a base noise function and samples it at multiple frequencies for each coordinate position. Perlin Noise is used as a base function to create pseudo-random values. Perlin noise works by calculating values for $[x,y,z]$ points by assigning pseudo-random, uncorrelated gradients $[a,b,c]$ to them and a value d which forms a linear equation for $[x,y,z]$ and can be retrieved using pre-calculated hash tables [Perlin 1985].



(a) Input/5, Height*5 (b) Input/8 (c) Input/2



(d) Height*2 (e) Height*8

Figure 1: Parameter modification of fBm

Fractional Brownian Motion (fBm) uses a base noise function and samples it at multiple frequencies. The base frequency is calculated using the equation:

$$a_0 = (N(p_0) + c_t)\, c_s + c_0$$

where a is the heightmap point, N is the noise generating function, p is the initial grid point, c_t is a constant translation transformation, c_s is a constant scaling transformation. Successive frequencies are calculated by:

$$a_i = a_{i-1} + a_{i-1}\,(N(p_i) + c_t)\, c_s w^i$$

where w^i is a frequency increment constant. Since the underlying noise function can be two dimensional, the generated points are continuous and produce a fractal terrain, which is useful for its endless level-of-detail and reproducibility, but even with the same seed for the noise function, the end terrain can appear very different due to the scaling and translation constants [Musgrave et al. 1989].

3.3 Realism

Synthesis alone is not enough to create realistic terrain: it does not cater for creating rivers, flattening areas or time-based creation. Often it leaves the created terrain looking jagged and unrealistic. Therefore the need for post-processing of the initial terrain synthesis arises.

3.3.1 Squig Curves

The chosen method of river creation is via squig curves, which are a type of L-system and a subdivision scheme (see Figure 4) [Prusinkiewicz and Hammel 1993]. The technique works by randomly assigning edges (entry, exit and neutral) to an initial triangle. When subdivided the original edges are split into two (entry edges becoming entry and neutral, exit becoming exit and neutral, and neutral becoming two neutrals), and the three new edges between them being assigned accordingly. This process is repeated recursively. By tracing from the midpoint of entry to exit edges, it gives a fairly reasonable representation of a winding river (see Figure 5), and this is then plotted onto the main terrain heightmap

Algorithm 1 Water Change

if $\Delta w \leq 0$
  $a^v_{t+1} = a^v_t + K_d s^v_t$
  $s^v_{t+1} = (1 - K_d) s^v_t$
else
  $w^v_{t+1} = w^v_t - \Delta w$
  $w^u_{t+1} = w^u_t - \Delta w$
  $c_s = K_c \Delta w$

and vertices simply lowered to a base level where the rivers occur [Prusinkiewicz and Hammel 1993]. Squig curves have the feature that they will never intersect themselves and can achieve quickly or slowly winding rivers by varying the recursive depth.

3.3.2 Flattening

In order to integrate with the Text-To-Scene and city creation project, areas need to be flattened for city placement and action areas in the scene (presumably activities in the script will not be performed on hugely jagged terrain). If a point is within a specified radius, a cosine function is used to create a flattened area:

$$value = 1 - cos\left(\frac{d}{D} \times \frac{\pi}{2}\right)$$

where d is the distance from the center and D is the desired radius of the flattened area. The values produced are between zero and one, which can then be used to smoothly interpolate flat areas with the rest of the terrain.

3.3.3 Erosion

Musgrave *et al.* [Musgrave et al. 1989] make use of an erosion simulation over synthesised terrain to make it appear closer to real world terrain formations. Hydraulic erosion works by assigning each vertex v at time t an altitude a^v_t , a volume of water $w^v_t$and an amount of sediment s^v_t . Iterating through time each vertex v passes excess water and sediment to each neighbouring vertex u where the amount of water passed is defined as:

$$\Delta w = min\left(w^v_t, (w^v_t + a^v_t) - (w^u_t + a^u_t)\right)$$

Based on the amount of water passed, it is determined which direction the water is flowing (see Algorithm 1). If water is being passed to neighbours, the water at the current and neighbour vertices are calculated as well as the sediment capacity c_s to calculate sediment transfers with (see Algorithm 1), then sediment movement is calculated (see Algorithm 2). K_c is the sediment capacity which specifies the max sediment that may be suspended in a unit of water, K_d is the deposition constant which specifies the rate at which sediment settles out of a unit of water and is added to a vertex, and K_s is the soil softness constant, which specifies the softness of the soil and is used to control the rate at which soil is converted to sediment. What this does is to erode the terrain to an approximation of what happens in reality when rain falls on ground and redistributes soil [Musgrave et al. 1989].

Algorithm 2 Sediment Movement

if $s^v_t \geq c_s$
  $s^u_{t+1} = s^u_t + c_s$
  $a^v_{t+1} = a^v_t + K_d (s^v_t - c_s)$
  $s^v_{t+1} = (1 - K_d) (s^v_t - c_s)$
else
  $s^u_{t+1} = s^u_t + s^v_t + K_s (c_s - s^v_t)$
  $a^v_{t+1} = a^v_t - K_s (c_s - s^v_t)$
  $s^v_{t+1} = 0$

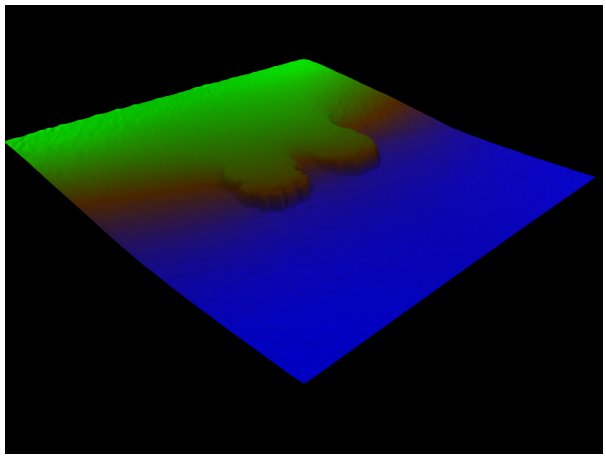


Figure 2: Natural phenomena replication

4 Implementation in Blender

The techniques are implemented as Python scripts for Blender: which is an open source 3D modelling package, used for other purposes in the Text-to-Scene conversion system.

4.1 Representation

The heightmap is represented as a two dimensional array containing four values for each element: the height, and the three erosion constants (see Section 3.3.3). Having each location have its own set of erosion constants means that areas may be harder than others for instance, which affords the possibility to model very tough rocks in the middle of a hill and erode this for instance (see Figure 2).

4.2 Geometry Synthesis

Blender provides a pre-implemented fBm generation method, with parameters for the vertex position, the fractal increment, the lacunarity or gap between successive frequencies, the number of octaves, and the base noise function. We find that New Perlin noise is the most appropriate type of noise for the base function. Two additional parameters for modification are: the input vertex position, which can be scaled to stretch the terrain, effectively creating more rapidly or slowly changing terrain (see Figure 1a,b,c); and the scaling on the output value, which can make for higher, more jagged terrain with higher values (see Figure 1d,e). By altering parameters, we can control the formation of the terrain, which means sections of the terrain can be made to vary from others: having tightly rock

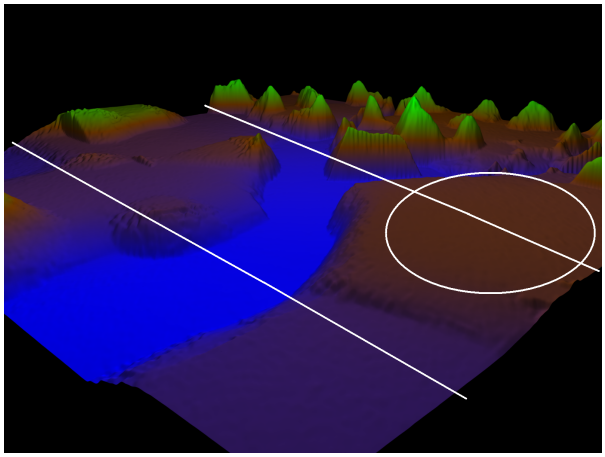
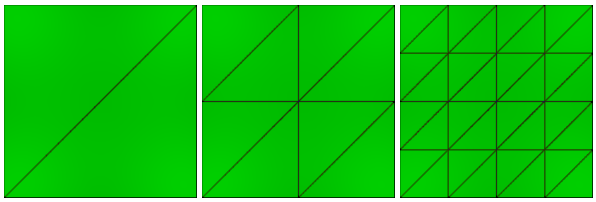


Figure 3: Terrain formation control



(a) n=0 (b) n=1 (c) n=2

Figure 4: Squig Curves: Grid after *n* levels of subdivision

controlled, jagged areas to flood plains, as can be seen in Figure 3: the far area being tightly rock controlled, the middle area a medium region and the close area a low flood plain.

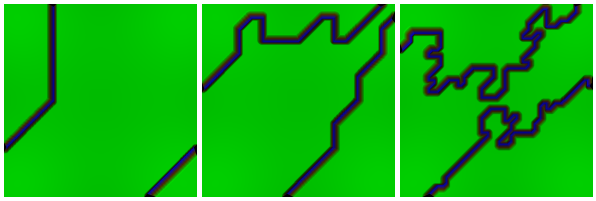
4.3 Realism

4.3.1 Squig Curves

As an observation on the method proposed, edges can be shared rather than creating two separate edges and assigning one to be an entrance and the other an exit. Also it is unnecessary to assign a label of entrance and exit, rather we use a system where an edge can be neutral, marked or unassigned (during subdivision only) [Prusinkiewicz and Hammel 1993].

To thicken the river paths, as well as provide a gradual fade from river to normal base terrain, we use a radius of influence calculation, assigning heightmap values based on how far they are from a river path, and this radius can be altered which can assist in interpolating with the terrain to create jagged areas to flood plain areas as can be seen in Figure 3 where the far area has narrow rivers and the close area wider ones.

Another configurable option would be to select the recursive depth of subdivision for each triangle so as to provide meandering sections of river and swiftly changing ones. This is still to be implemented.



(a) n=1 (b) n=3 (c) n=5

Figure 5: Squig Curves: Path after *n* levels of subdivision

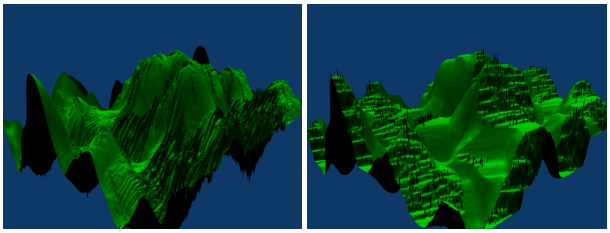


Figure 6: Wave erosion

4.3.2 Flattening

This is implemented as per the theory, using a cosine graph of configurable radius and can be seen in Figure 3 shown as the circled area. The method takes as input the center of the circle and the radius so as to be of use to the Text-To-Scene converter. If a more rectangular region is needed to be flattened this can be achieved by flattening a few smaller circles in a row.

4.3.3 Erosion

A first attempt to duplicate the results detailed in the paper failed, with wave-like erosion images being formed [Musgrave et al. 1989]. This problem was diagnosed to be unfair distribution of water and sediment: because the neighbours are checked in the same order every time, the first few being checked were getting preference. This problem is clearly visible in the two renders in Figure 6. The solution was to distribute the water proportionally looking at each neighbour as one of the eight [Benes and Forsbach 2002].

The authors recommend two heightmaps: one from a previous step and the currently worked on one which means calculations are being repeated to calculate each new neighbour [Musgrave et al. 1989]. We find that using a pre-calculation step to calculate the proportion each neighbour will receive, we can eliminate the need for a second array, replacing it with a variable length array for neighbours that will receive sediment and water and the proportions thereof. The erosion method also implements evaporation with an exponential method to prevent pools of stagnant water building up and creating jagged areas of terrain due to skewed erosion calculations [Benes and Forsbach 2002]. The method will continue to run after the amount of timesteps specified, until all the water has evaporated, as this gives the sediment contained in the water time to settle. The rain occurs every set number of timesteps and stops once the desired number of timesteps has passed.

Since the erosion algorithm is a global procedure the implemented method takes as input the desired number of timesteps to erode by, and if certain areas need to be eroded more than others to mimic

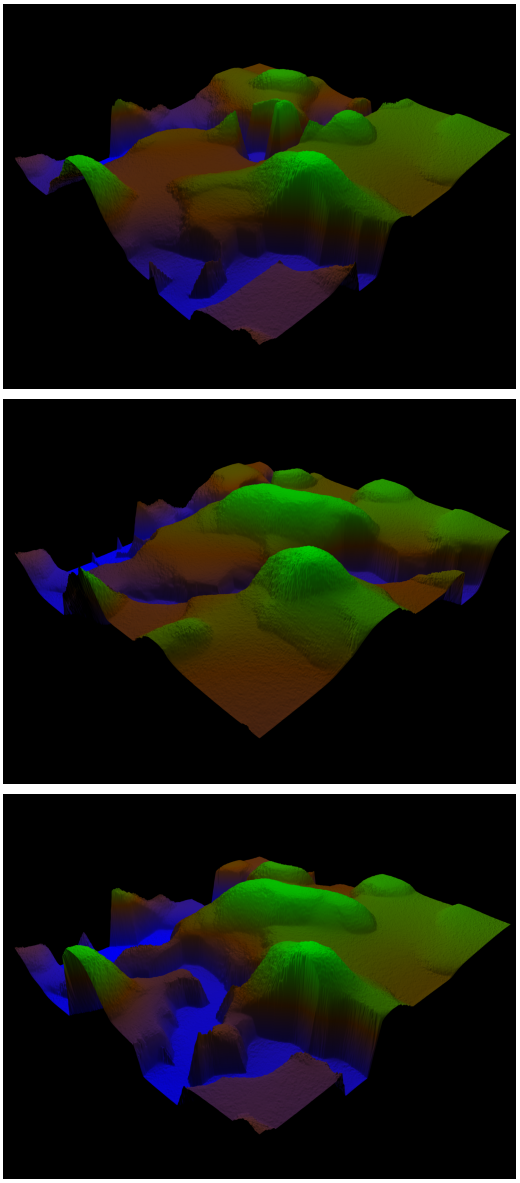


Figure 7: Results

softness or hardness of areas this can be achieved because each location on the heightmap has its own set of erosion constants.

5 Results

The three renders in Figure 7 show common results produced by this project, what can be emphasized here is that the only difference between the images is the squig river paths, other than that the images have the same base terrain and it is the erosion procedure that makes each image unique. The render in Figure 3 shows the configurability of the methods, with differing radius of influence for the river based on the section, as well as modified fBm output scaling to create the jagged area to floodplain transition, as well as the flattened region (refer to Section 4.2).

6 Conclusions

The project is capable of producing terrain with minimal input, and is very configurable yet very simple as has been mentioned in each relevant section. A multitude of scenes can be created by altering parameters, in particular offsetting the fBm would create completely different regions, but also the squig curve river generation which is completely random within its ruleset. With reference to the results in Figures 3, 7, the erosion procedures as well as configurable control of all parameters the project is capable of producing realistic, natural terrain, combined with flattening for cities or action areas for the TTS system, we have achieved the goals of the project.

References

BELHADJ, F., AND AUDIBERT, P. 2005. Modeling landscapes with ridges and rivers: bottom up approach. In *GRAPHITE '05: Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, ACM Press, New York, NY, USA, 447–450.

BENES, B., AND FORSBACH, R. 2001. Layered data representation for visual simulation of terrain erosion. In *SCCG '01: Proceedings of the 17th Spring conference on Computer graphics*, IEEE Computer Society, Washington, DC, USA, 80.

BENES, B., AND FORSBACH, R. 2001. Parallel implementation of terrain erosion applied to the surface of mars. In *AFRIGRAPH '01: Proceedings of the 1st international conference on Computer graphics, virtual reality and visualisation*, ACM Press, New York, NY, USA, 53–57.

BENES, B., AND FORSBACH, R. 2002. Visual simulation of hydraulic erosion. In *Journal of WSCG 2002: Proceedings of the 10th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2002*, WSCG, 79.

DUCHAINEAU, M., WOLINSKY, M., SIGETI, D. E., MILLER, M. C., ALDRICH, C., AND MINEEV-WEINSTEIN, M. B. 1997. Roaming terrain: real-time optimally adapting meshes. In *VIS '97: Proceedings of the 8th conference on Visualization '97*, IEEE Computer Society Press, Los Alamitos, CA, USA, 81–88.

FOWLER, R. J., AND LITTLE, J. J. 1979. Automatic extraction of irregular network digital terrain models. In *SIGGRAPH '79: Proceedings of the 6th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 199–207.

GLASS, K. R., MORKEL, C., AND BANGAY, S. D. 2006. Duplicating road patterns in south african informal settlements using procedural techniques. In *AFRIGRAPH 2006: Proceedings of the 4th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, ACM Press.

MILLER, G. S. P. 1986. The definition and rendering of terrain maps. In *SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 39–48.

MORKEL, C., AND BANGAY, S. D. 2005. *Non-interactive Modeling Tools and Support Environment for Procedural Geometry Generation*. Master's thesis, Rhodes University.

MUSGRAVE, F. K., KOLB, C. E., AND MACE, R. S. 1989. The synthesis and rendering of eroded fractal terrains. In *Proceed-*

ings of the 16th annual conference on Computer graphics and interactive techniques, ACM Press, 41–50.

PARKER, E., AND UDESHI, T. 2003. Exploiting self-similarity in geometry for voxel based solid modeling. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, ACM Press, New York, NY, USA, 157–166.

PERLIN, K. 1985. An image synthesizer. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 287–296.

PRUSINKIEWICZ, P., AND HAMMEL, M. 1993. A fractal model of mountains with rivers. In *Proceeding of Graphics Interface '93*, 174–180.

ZHOU, H., SUN, J., TURK, G., AND REHG, J. M. 2007. Terrain synthesis from digital elevation models. *IEEE Transactions on Visualization and Computer Graphics 13*, 4 (July/August), to appear.