



White Paper

Intel Software Solutions
Group

Jeffrey M. Freeman

Procedural Terrain Generation With Fractional Brownian Motion

Information for Developers and ISVs

This white paper investigates techniques for leveraging procedurally generated terrain with fractional Brownian Motion on the CPU and multi-texture blending on the GPU.

December 2007

Intel Corporation



Contents

1	Introduction	3
2	Previous Work	5
3	Implementation	7
	3.1 Multitextured Pixel Shading	8
4	Future Work	9
5	References	11
6	About the Author	13
A	Additional Images	15

Figures

3-1	Fractal Terrain, Simple fBm	7
-----	-----------------------------------	---

§



1 Introduction

The computer graphics industry has a long history of attempting to model real world terrain. These efforts try to capture the seemingly limitless complexity of natural terrain through modeling and rendering techniques. As early as the late 1960's Dr. Benoit Mandelbrot linked natural forms that maintain a level of self-similarity such as coastlines to mathematical constructs [1]. Notable achievements in this field since that time have utilized fractals to achieve approximations of terrain patches using stochastic processes such as fractional Brownian motion. In this article, we demonstrate several techniques of generating terrain patches as proposed by Dr. F Kenton Musgrave [2] along with texture blending and Shader Model 3.0 to create a synthetic scene on integrated graphics solutions such as the Intel® 965 Express Chipset and Mobile Intel® 965 Express Chipset family.

First, we describe a list of previous work in this field followed by the approach utilized by our implementation, which leverages both the CPU and GPU to render the scene. Source code is provided with the demonstration to be used in your terrain rendering extensions and implementation.

§





2 Previous Work

A number of researchers have investigated terrain generation using fractals to perturb surfaces in 2D and 3D space. B. Mandelbrot provided some of the earliest representations of terrain generation with fractals by comparing the self-similarity of mountainous terrain to Brownian motion, resulting in realistic skylines when charting a 2D random walk. Later work by Mandelbrot and Musgrave later showed increasingly compelling approximations of terrain utilizing fractional Brownian motion in 3D space with Perlin noise and the concept of multifractals both represented in [2].

Noise-based systems for generating fractal terrains as proposed by [2] and [4] are not exclusive to creating good approximations to real landscapes as several other calculations have been used to create aesthetically pleasing approximations to real terrain. Of interest in this group, include mid-point displacement calculations including the diamond-square and triangle-edge subdivision algorithms, Poisson faulting, and Fast Fourier Filtering.

While some of these systems can and do produce realistic looking scenes, the noise synthesis method proposed in [2] is utilized in this work as these calculations provide an interesting set of controls to the resulting terrain from a mathematical model. While these properties do not necessarily provide a mechanism to definitively control the shape of the rendered scene as indicated in [5] to constrain terrain to realistic properties, they do provide many interesting real and imaginative results. We present a CPU based set of algorithms demonstrating these controls balanced with smooth stepped texture blending in the pixel shader on the GPU using Microsoft DirectX 9 and Shader Model 3.0.

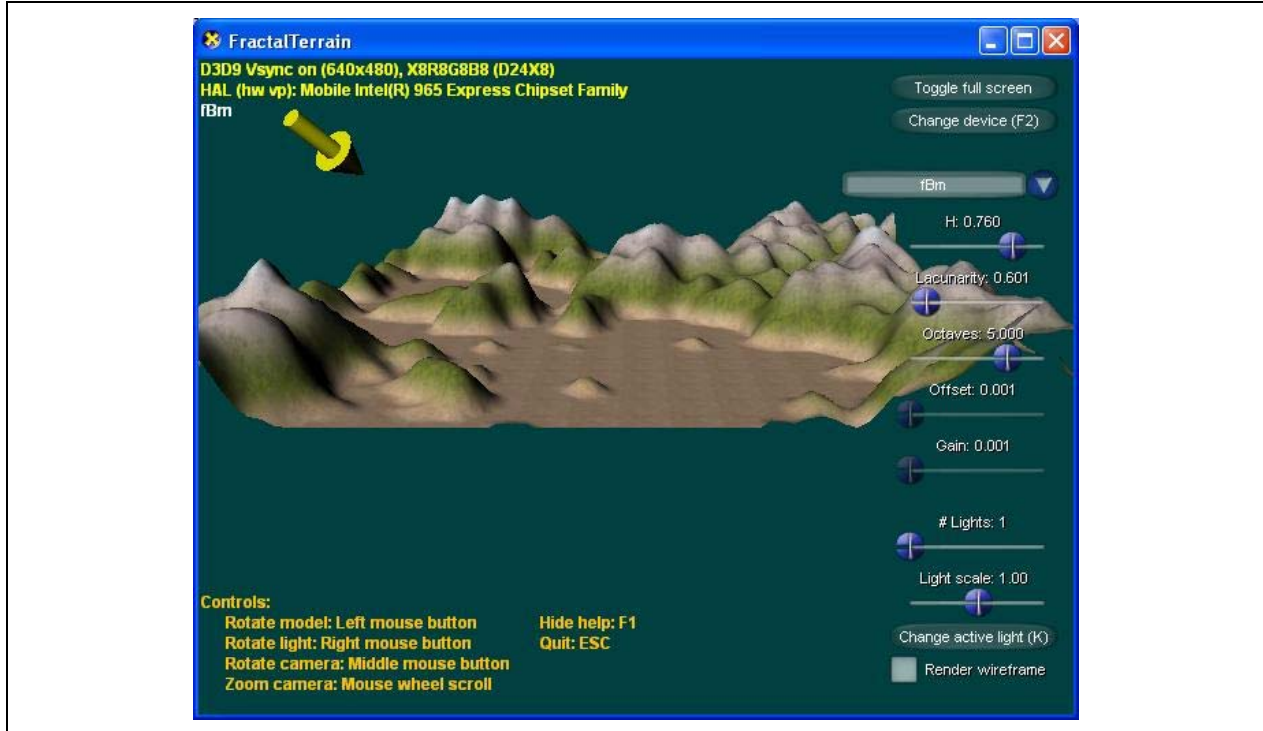
§



3 Implementation

Our implementation was inspired by Musgrave's work in [2], showcasing three methods from that text: simple fBm, hybrid fBm, and the ridged multifractal algorithm, each based on Perlin's noise algorithm. The output from these methods is used to perturb the Z direction of a fixed size polygon mesh.

Figure 3-1. Fractal Terrain, Simple fBm



In Figure 3-1, we present our implementation. On the right hand side, one can see the controls used to adjust properties of each fBm algorithm as selected from the combo box. Our demo is adapted from the BasicHLSL demo from [7] with default algorithm parameters adjust to demonstrate interesting terrain properties.

Method Parameters:

- (H) Hurst index – In mathematical literature, classifies the fBm and dictates fractal dimension.
- (Lacunarity) – Dictates the gap between successive frequencies.
- (Octaves) – Dictates the number of frequencies and scales Level of Detail in the scene.
- (Offset) – Offset from the lowest elevation and determines “multifractality” [2].
- (Gain) – Controls the amplitude of the frequency.



3.1 Multitextured Pixel Shading

Pixels are sampled from texture images and Hermite spline interpolated $\{0..1\}$ over regions of existence based on elevation as passed to the pixel shader function. The multi-texturing blend operation, proposed by [3] was chosen over older fixed-function texture blending operations and implemented without a blend map. While a blend map would produce a finer and more realistic blend of each region's texture in multi-textured surfaces as demonstrated in [3], the chosen method provides reasonable blending for higher altitude scenes which necessarily lack a finer level of detail due to the height of the viewpoint as may be experienced in a flight simulator. Each elevation zone is separated by constants indicating gradual shifts between regions of sand, grass, rock, and snow to create a blended effect over the range of elevations with regions nearing the next texture region taking on pixel hues of samples of that area, a method suggested by [9] in the shader effect file.

§



4 Future Work

There are a number of interesting problems remaining to be tackled with respect to fractal terrain generation. Applications of automatic landscape generation face decisions associated with conflicting game-play elements in the storyline or unrealistic features that present themselves from both fBm and other fractional models of terrain.

In addition, most terrain generation methods are calculation intensive and are not real-time. While some fractal algorithms lend themselves easily to multi-threading, the result is still time consuming as is the case with the Mandelbrot set [10] and may not apply well without significantly reducing the size of the perturbed surface greatly or reducing the number of iterations inspected. An interesting and difficult problem remains open for multi-threading a fractional Brownian motion implementation since it is a global operation, taking into account different domains of the variable. One could split processing by row or column, spanning a surface's Z values or even by quadrants for processing with results that skew the controls of the algorithm's parameters and render more noticeable regions of disjoint terrain.

It is this interdependence of previous values from the calculation that introduces the very properties we are seeking from fBm introduced by way of the Hurst index, Lacunarity, and Octave. As such, it becomes difficult to split calculations amongst physical processing units. Investigation of this concept with Dr. Mandelbrot [8] shows that there may be a solution to the threading problem through truncation of fBm. However, some level of introduced errors may be present in the calculation as a result. If the desired scene produced by the rendering process is aesthetically pleasing, some inaccuracy in the operation may be acceptable. We would like to determine what effect a multi-threaded fBm algorithm would have on a rendered terrain as well as analyze its performance for multi-core processors with optimized code.

§





5 References

- [1] **[Mandelbrot67]** B. Mandelbrot. How Long Is the Coast of Britain? Statistical Self-Similarity and Fractional Dimension. *Science*. New Series, Vol. 156, No. 3775. May 5, 1967). pp. 636-638.
- [2] **[EMPPW94]** Ebert, Musgrave, Peachey, Perlin, Worley. *Texturing and Modeling: A Procedural Approach*. Academic Press Inc. 1994. Musgrave Chapters 7-9, Perlin Chapter 6.
- [3] **[Luna06]** Frank Luna. *Introduction to 3D Game Programming with DirectX 9.0c: A Shader Approach*. Wordware Publishing Inc. Chapter 11.
- [4] **[MKM89]** Musgrave, Kolb, Mace. The Synthesis and Rendering of Eroded Fractal Terrains. SIGGRAPH 1989. *ACM Computer Graphics*, Volume 23, Number 3, July 1989.
- [5] **[SS05]** Stachniak, Stuerzlinger. An Algorithm for Automated Fractal Terrain Deformation. *Computer Graphics and Artificial Intelligence*. Ed. D Plemenos. ISBN 291425607-8, 64-76, May 2005.
- [6] **[HR06]** Hardy, Roberts. Blend Maps: Enhanced Terrain Texturing. *Proceedings of SAICSIT 2006*. pp 61-70.
- [7] **[MSSDK07]** Microsoft Corporation DirectX SDK (November 2007). <http://www.microsoft.com/downloads/details.aspx?familyid=4b78a58a-e672-4b83-a28e-72b5e93bd60a&displaylang=en>.
- [8] **[Mandelbrot07]** B. Mandelbrot. Personal Email Communication. November 2007.
- [9] **[Hayes07]** Hayes, Jeremy. jeremy.hayes@intel.com. December 2007.
- [10] **[Intel05]** Using SSE3 Technology in Algorithms with Complex Arithmetic. <http://softwarecommunity.intel.com/articles/eng/3426.htm>. February 23, 2005.





6 About the Author

Jeff Freeman is a Software Engineer in the Software and Solutions Group, where he supports Intel integrated graphics solutions in the Client Scale Enabling team. He holds a B.S. in Computer Science from Rensselaer Polytechnic Institute. He can be reached at jeffrey.m.freeman@intel.com.

Credits:

- Steve Pitzel steve.pitzel@intel.com: Art/Textures
- Jeremy Hayes jeremy.hayes@intel.com: Guidance on terrain shading

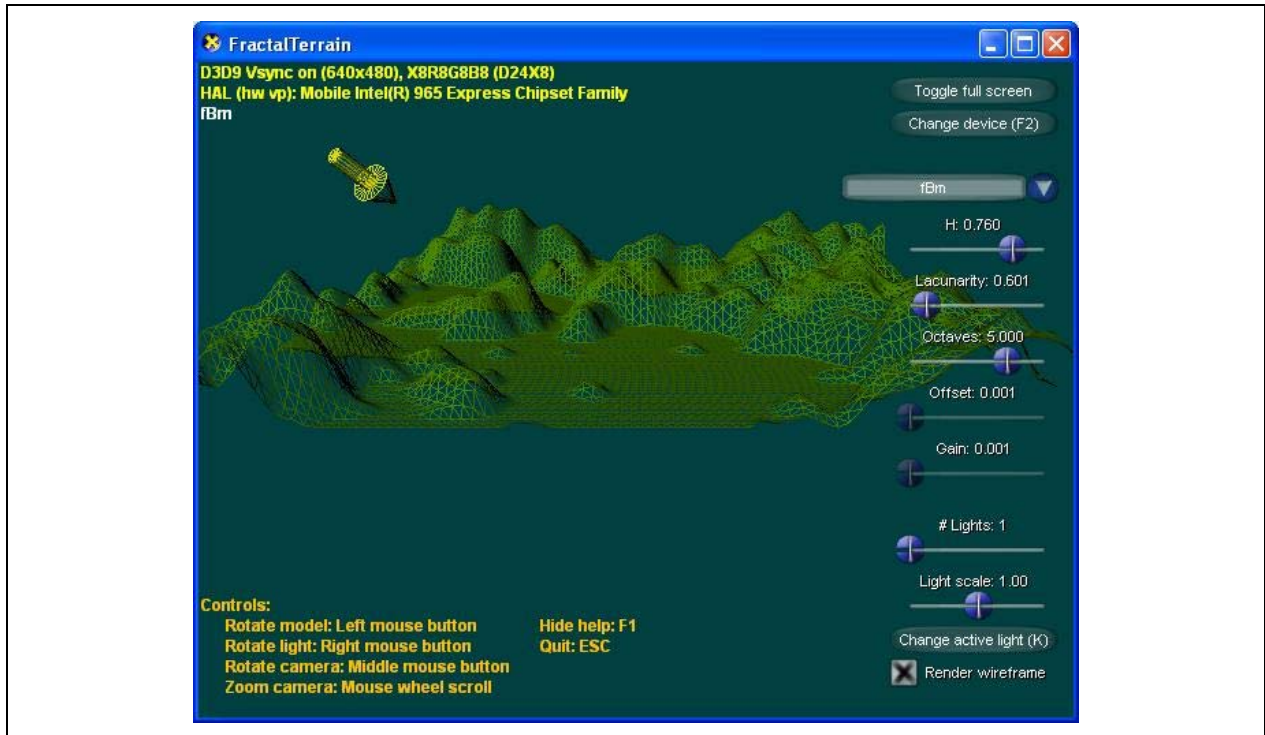
§





A Additional Images





§



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

This white paper, as well as the software described in it, is furnished under license and may only be used or copied in accordance with the terms of the license. The information in this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Intel Corporation. Intel Corporation assumes no responsibility or liability for any errors or inaccuracies that may appear in this document or any software that may be provided in association with this document.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See www.intel.com/products/processor_number for details.

The Intel processor/chipset families may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Copies of documents, which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or by visiting Intel's Web Site.

Intel and the Intel Logo are trademarks of Intel Corporation in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2007, Intel Corporation. All rights reserved.

