MIDPOINT DISPLACEMENT IN MULTIFRACTAL TERRAIN GENERATION

Rami Ramstedt and Jouni Smed
Department of Information Technology
University of Turku
E-mail: Rami.Ramstedt@gmail.com

KEYWORDS
Multifractal, Terrain, Procedural, Fractal, Generation

ABSTRACT

Midpoint displacement methods (MPDM) – originally introduced as stochastic parametric surfaces – have been used to create approximations of fractional Brownian motion in the past. With multifractals it is possible to create more realistic looking fractal terrain than with fractional Brownian motion, and also with more parameters to affect the generated terrains. This paper studies whether it is possible to generate multifractal terrains using the MPDMs. As a result, we introduce different methods that are significantly faster than using the spectral synthesis algorithm.

INTRODUCTION

The research on multifractals was initially started from the seminal work of Mandelbrot (1969, 1974), although they were named as multifractals later by Frisch and Parisi (1983) describing hierarchies consisting of sets, which have different Hausdorff-Besicovitch dimensions (Mandelbrot 1982). Pietronero and Siebesma (1986) proved that multifractals can be created with random multiplicative processes, for example by using multiplicative random walk.

These approaches using partially different terminologies originating from studying multifractals, turbulent cascades and multiplicative processes were tied together by Meneveau and Sreenivasan (1987) to produce the idea that multifractals can be created by using multiplicative cascades, which is the terminology used in the context of multifractal terrain generation.

Multifractal terrains have been previously been created with multiplicative cascade in spectral synthesis (Ebert et al. 2002), and by using Lévy noise in the Fourier synthesis method (Pecknold et al. 1993). These are based on the additive methods, that are used create fractional Brownian motion.

The simplest MPDM used in this paper is the wireframe MPDM (Musgrave 1993) also known as the triangular edge subdivision (Miller 1986) or Carpenter Mountains (Smith 1984). The second more advanced method is the "Diamond-Square"-scheme by Fournier et al. (1982). The third scheme we use is the "square-square" unnested subdivision by Miller (1986).

We begin by introducing the MPMSs and height calculation used in our study. This is followed by a comparison of the methods based on different metrics and final conclusions based on the comparison.

METHODS

The multifractal terrain generation algorithms in this paper consist of two parts. Their implementation is designed so that these two parts can be used interchangeably between all variants of each part.

The first part is one of the different MPDMs that make the three nested loops of the algorithms and the type of interpolation used to calculate the initial value for the new points. The three loops determine the order in which the points are used as a point of reference for the interpolations. The interpolations then use that point to find the absolute coordinates of the other points, and their corresponding height values, based on the relative locations of the other points used in the interpolations. The initial value for the new point is then calculated based on either the average or the weighted sum of the values of the points used in the interpolation. (Musgrave 1993; Ramstedt 2008)

The second part tells how the resulting value for the new point is calculated. In the case of a normal monofractals the calculation is entirely additive; in the case multifractals, the calculation is multiplicative. Heterogenic multifractals contain both multiplicative and additive operations, and by setting the parameter that controls the influence of the multiplicative operations to zero, they become entirely additive and, thus, produce a monofractal terrain having a single Hausdorff-Besicovitch dimension. (Ebert et al. 2002)

In the implementation, the random values for the point calculations are based on Gaussian distribution centered at zero. This entails that small height deviations are more likely than extreme deviations.

Midpoint Displacement Methods

Wireframe Midpoint Displacement Method

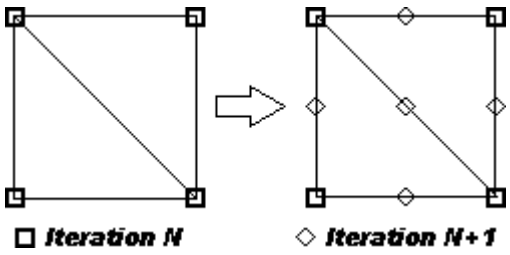

Figure 1: Triangle edge subdivision

The first step in calculating wireframe MPDM is to calculate the lengths of the sides of the height-map, which are equally long. The length of the sides is calculated with Equation (1), where l is the length of the side in number of points and n is the number of iterations. After that, an area in memory is reserved for the height-map, which will contain the square with sides of that length.

$$l_n = 2^n + 1 \qquad (1)$$

The second step in initialization is generation of random values for each four corners of the height-map.

After the initialization the actual iteration starts that consists of three nested loops. On each of the iterations in the most outer loop the step-size is halved. Step-size tells how far apart the points of calculation are from each other in the height-map. The first step-size, used in the initialization, is the size of the total height-map. Next, the inner loops are executed, and, finally, in the iteration new scale for height displacement is calculated. It is this rescaling of the height changes relative to the scale of the distances between the points, which creates the power law necessary to create the fractal terrains (Miller 1986). The iteration of the outmost loop stops when step-size 1 has been processed.

In the two inner loops all the points of references are iterated through and used for the interpolation. The points of reference are the current step-size apart from each other. The points in the bottom and right edges of the height-map are never used as the point of reference; however they are used in the height calculations in the interpolations.

In the interpolation three new points are generated, these new points can be seen on the right-side in Figure 1, which illustrates that five new points are generated, but the points in the top and left side are only generated for the points in the top and left edges of the entire height-map. The point of reference in each interpolation is the point in the top-left corner, the location of each point used in the interpolation is either 0, half, or full step-size to the right or down from it. The initial height-value of the new point is the average of the two points that are the end points of the line, where the new point is in the middle of. All these lines can be seen on the left side in Figure 1. This initial height-value is then used in the height calculation of the new point.

Diamond-Square Midpoint Displacement Method

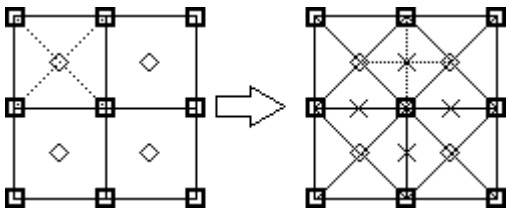

Figure 2: Diamond-square subdivision

Diamond-square MPDM is otherwise same as the wireframe MPDM, except that on each step-size the points of reference are looped through twice, each time using a different interpolation. Here, those interpolations are named after the shape of the interpolation; they can be also named based on the shapes they create.

The first interpolation is the square interpolation, shown in Figure 2 on the left side. In the square interpolation, simply one new point is generated in the center of the four points connected by the dotted cross-section in Figure 2. Height is then calculated for this new point using the average of the four points as the initial value.

The second interpolation is the Diamond interpolation. In the Diamond interpolation, the point of reference is not actually used in the height calculation, it is just used to determine the locations of the points in the interpolation. Two new points are generated in every interpolation, marked as x on the right-side in Figure 2. The points on the left and top are only generated in the left and top edges of the entire height-map. The new points get their height value as the average of the four surrounding points, marked by the dotted cross-section on the right side in Figure 2. For the new points at the edges of the height-map, the average of only three points is used. As we can see from Figure 2, the two points on the right and left that are used to calculate the average are the just generated points from the square interpolations. The two points that are top and bottom are from the previous iteration of the scale.

Unnested Subdivision Method

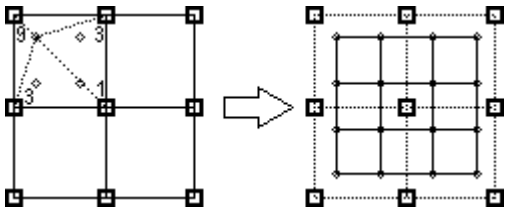

Figure 3: Unnested subdivision method

The unnested subdivision is more different from the wireframe MPDM and the diamond-square subdivision, than they are from each other.

The first step in the unnested subdivision is to create a memory area of size three-by-three filled with nine random values. These points are the square-shaped points in Figure 3.

After the initialization the actual iteration starts. The outmost loop in unnested subdivision is different from the one in Wireframe MPDM: instead of halving the step-size in every iteration, the step size is incremented by one at every step. The iteration of the outmost loop stops after the selected amount of iterations has been completed.

Inside the outmost loop a new height-map is generated on every iteration. The first step is to set the height-map generated in the previous iteration as the reference height-map for the iteration currently being calculated. Then, the length of the sides of the new height-map is calculated with Equation 2, where l is the length of the side as amount of the vertexes and n is the iteration. After that, the total number of the vertexes in the new height-map is calculated by using

Equation 3, where *a* is the total area as amount of vertexes and *n* is the iteration. The total number of points is then used to reserve the memory area for the new height-map.

$$l_n = 2 \cdot (l_{n-1} - 1) \quad (2)$$
$$a_n = (2^{(n+1)} + 2)^2 \quad (3)$$

Next, all other points in the old height-map are iterated through except the ones on the bottom and right edge of the old height-map. The points being iterated are used as the points of reference for the square-square interpolation. After the points have been iterated through in the two inner loops, new maximal height displacement is calculated, and the size of the height-map for the next iteration is set as the size of the newly created height-map.

In the square-square interpolation, four new points are always created. The initial height values for the new points are calculated as weighted sum in 9:3:3:1 ratio based on the relative location of the new point to the points in the old height-map. How the interpolation is done can be seen in Figure 3. The values of the new points are stored in the new height-map. Of the four points from the old height-map used in the interpolation, the point of reference is the point in top-left corner.
With the same amount of iterations the height-maps created with the unnested subdivision are slightly smaller than the height-maps generated using the other two MPDMs.

Height Calculation

The height calculations used are mostly based on the ones invented by Musgrave (Ebert et al. 2002). The height calculation is what determines if the result is a monofractal or a multifractal.

The parameters for height calculation are the roughness exponent, initial frequency, monofractality quontient, maximal height displacement and the type of the height calculation. These are set in the initialization of the height calculation. Also in the initialization the initial value of scaling factor is calculated.

Additive

The additive height calculation can only be used to generate monofractals. This is the normal method of height calculation originally used in the MPDMs.

In the additive height calculation the initial value from the interpolation is simply added to a randomly generated value from a Gaussian distribution centered on 0.

The change of scale in height is done on every iteration of the most outer loop. The scale is changed by multiplying the previous value of variance of the Gaussian distribution with scaling factor *s*. For additive height calculation *s* is calculated by using Equation 4, where *r* is the roughness exponent. For additive height calculation it is necessary to calculate the value of s only once at the initialization. The accurate connection between roughness exponent *r*, Hurst exponent and fractal dimension was not studied in this paper. However, larger *r* results in higher fractal dimensions.

$$s = 2^{-r} \quad (4)$$

Simple Multiplicative

The first experiment in this research was attempt to simply change the addition to a multiplication in the height displacement. As a result the height values escaped quickly to extreme changes in height, creating a very jagged terrain.

This problem was improved with a different way of scaling the variance of height changes in scale iterations. The new way of change in variance is calculated with Equations 5, 6 and 7, where *f* is frequency, *n* is the iteration, *s* is scaling factor, *r* is the roughness exponent, σ^2 is variance. The initial frequency is 0,5.

$$f_{n+1} = f_n \cdot 2 \quad (5)$$
$$s_{n+1} = {f_{n+1}}^{-r} \quad (6)$$
$${\sigma^2}_{n+1} = {\sigma^2}_n \cdot s_{n+1} \quad (7)$$

Heterogenic Multifractals

Musgrave's Multiplicative Calculated at a Point, Statistics by Altitude and Ridged height calculations were implemented for MPDMs in this study (Ebert et al. 2002). The Bounce-Back height calculation was created in this study as a new form of multifractal height calculation. The goal in its creation was to remove the unrealistically common pits from the terrain, and make all changes in the terrain to raise the terrain. The results heights h_r are calculated repectively with Equations 8, 9, 10 and 11. Where ξ is a random value from Gaussian normal distribution, s is the scaling factor, n is the iteration, h_i is the initial height from the interpolation. The monofractality quontient *m* is used to control how close the multifractal will be to the additive monofractal.

$$h_r = (|\xi| + m) \cdot s_{n+1} \cdot h_i \quad (8)$$
$$h_r = h_i + ((\xi + m) \cdot s_{n+1} \cdot h_i) \quad (9)$$
$$h_r = h_i + s_{n+1}(m - |\xi|)^2 \quad (10)$$
$$h_r = h_i + ((|\xi| + m) \cdot s_{n+1} \cdot h_i) \quad (11)$$

COMPARISION AND CONCLUSION

The multiplicative spectral synthesis methods are commonly used to generate multifractal terrains. Therefore the computation speeds of the multifractal generation methods presented in this paper were compared against the corresponding spectral synthesis versions by Ebert et al. (2002). The difference compared to those spectral synthesis implementations, was that instead of using Perlin noise Simplex noise was used, which is more advanced and faster version of Perlin noise. (Gustavson 2005)

The calculations were done using the same computer under the same conditions. The implementations were done using C++ language. The accuracy of the implementations is 64-bit floating point numbers and signed integers, except for the simplex noise which was using 32-bit floating point numbers. Each method was run ten times and the averages of these results are in Table 1.

The iterations were run for 15 iterations when using the MPDMs, generating height maps the size of 32769×32769 vertexes, the result height-map of the unnested method was slightly larger: 32770×32770 vertexes. The spectral synthesis methods were made to generate 32769×32769 size height-maps, and as result they needed to be iterated only for the maximum of 14 iterations of octaves, iterations beyond that would have exceeded the Nyquist limit (Ebert et al. 2002).

Table 1: Comparision of Calculation Times in Milliseconds

Wireframe	Additive	36730.7
Wireframe	Multiplicative	37191.0
Wireframe	At Point (8)	38145.6
Wireframe	Statistics by Altitude (9)	38171.5
Wireframe	Bounce-Back (11)	38710.8
Wireframe	Ridged (10)	38198.6
Wireframe	Average	37858.0
Diamond-Square	Additive	40580.3
Diamond-Square	Multiplicative	40669.7
Diamond-Square	At Point (8)	45125.5
Diamond-Square	Statistics by Altitude (9)	44391.3
Diamond-Square	Bounce-Back (11)	44869.1
Diamond-Square	Ridged (10)	44492.7
Diamond-Square	Average	43354.8
Unnested	Additive	51220.8
Unnested	Multiplicative	51461.7
Unnested	At Point (8)	53281.0
Unnested	Statistics by Altitude (9)	54073.6
Unnested	Bounce-Back (11)	54676.8
Unnested	Ridged (10)	54437.0
Unnested	Average	53191.8
Spectral Synthesis	fBm (4)	1002687.1
Spectral Synthesis	Multiplicative	1051458.4
Spectral Synthesis	Ridged (10)	3082298.4

The differences in speed between the MPDMs are not very large. The average execution time of Diamond-square was 15% longer than that of the Wireframe MPDM, and the execution time of the unnested subdivision was 40% longer than that of the Wireframe MPDM. All of these methods were extremely faster than the spectral synthesis methods. When comparing the simple additive and multiplicative methods, spectral synthesis method calculated 28 times longer than the Wireframe MPDM. In the case of the more complex Ridged height calculation, spectral synthesis took 67 times longer to calculate than MPDMs on average. This difference is significant since at this level of resolution, the spectral synthesis methods are only capable of being used for preprocessed terrain generation, while the MPDMs are fast enough to be used in runtime terrain generation.

Table 2: Height Calculation Average Times in Milliseconds

Additive	42843.93
Multiplicative	43107.40
At Point	45517.37
Statistics by Altitude	45545.47
Bounce-Back	45709.43
Ridged	46085.57

From the average times of the height calculations in Table 2, it can be seen that all the different types of height calculations are of the same degree in speed. Fastest height calculation is the Additive, and The Simple Multiplicative being the close second. The slowest is The Ridged height calculation, but the difference is only 7.6% compared to the Additive. Thus when MPDMs are used the height calculation makes lesser difference than the choice of the MPDM.

The unnested subdivision creates useable terrains with all height calculations, although with many of the height calculation and parameter combinations create jagged and spiky terrains. For the more advanced multifractal height calculations the results are equal to those generated by the spectral synthesis methods, and free of artifacts. However in spectral synthesis it is possible to set the terrain to be of any size, and the amount of iterations is not dependent on the size and the lacunarity is not static.

The other two MPDMs, however, have the problem of having artifacts in the terrain (Musgrave 1993).

From these results we can conclude that it is possible to generate multifractal terrains using midpoint displacement methods. Also, the much faster computation speed of MPDMs makes runtime generation of multifractals possible on significantly larger scale than by using spectral synthesis.

REFERENCES

Ebert, D.; F. Musgrave; D. Peachey; K. Perlin; and S. Worley. 2002. *Texturing and Modeling A Procedural Approach, Third Edition.* San Francisco, CA: Morgan Kaufmann

Frisch, U.; and Parisi, G. 1983 "Turbulence and Predictability of Geophysical Flows and Climate Dynamics". Varenna Summer School LXXXVIII

Fournier, A.; D. Fusell; and L. Carpenter. 1982. "Computer Rendering of Stochastic Models." *Communications of the ACM* 25, No.6 (Jun), 371-384.

Gustavson, S. 2005. "Simplex noise demystified". Linköping University.

Mandelbrot B. 1969. "Turbulence of Fluids and Plasmas". Polytechnic Institute of Brooklyn. New York: Interscience.

Mandelbrot B. 1974. "Intermittent turbulence in self-similar cascades: divergence of high moments and dimension of the carrier" *Journal of Fluid Mechanics* 62, 331-358.

Mandelbrot B. 1982. *The Fractal Geometry of Nature.* New York, NY: W.H. Freeman and Company

Meneveau C.; and Sreenivasan K. 1987. "Simple Multifractal Cascade Model for Fully Developed Turbulence" *Physical Review Letters* 59, No.13 (September), 1424-1427.

Miller, G. 1986. "The definition and rendering of terrain maps" *ACM SIGGRAPH Computer Graphics* 20, No. 40, 39-48.

Musgrave, F. 1993. "Methods for Realistic Landscape Imaging". PhD Thesis. The faculty of the Graduate School, Yale University

Pecknold, S.; S. Lovejoy; D. Schetzer; C. Hooge; and J. Malouin. 1993. "The Simulation of Universal Multifractals". Department of Physics, McGill University.

Pietronero, L.; and Siebesma, A. 1986. "Self-Similarity of Fluctuations in Random Multiplicative Processes" *Physical Review Letters* 57, No.9 (September), 1098-1101.

Ramstedt, R. 2008. "Artificial Fractal Terrain Generation". Master's Thesis. Department of Information Technology, University of Turku.

Smith, A. 1984. "Plants, Fractals, and Formal Languages" *ACM SIGGRAPH Computer Graphics* 18, No. 3.