ISSN:0975-9646

Matrix Based Method for Terrain Generation

Sujay Raj[#1], Prakhar Rastogi[#2], Dr. Badam Singh Kushvah[#3]

[#]*Department of Applied Mathematics, Indian School of Mines*
Dhanbad, Jharkhand, India

***Abstract*— Terrain, or land relief, is the vertical and horizontal dimension of land surface. Terrains when digitally generated can be used for geological studies as well as in computer simulations and games. In this paper, we discuss the various methods of generation of models of terrains as well as propose an algorithm for generating such terrains efficiently using numerical computation softwares which use matrices as their base, such as Matlab or Octave. Such an algorithm can not only be easily understood, but also provide better and faster terrain generation.**

***Keywords*— visualization, Terrain generation, Digital evelation models, Heightmaps**

I. Introduction

Terrain, or land relief, is the vertical and horizontal dimension of land surface. Terrain is used as a general term in physical geography, referring to the lay of the land. This is usually expressed in terms of the elevation, slope, and orientation of terrain features. Generating terrains digitally has not only been used by professional geologists and geophysicists [1] but they have been lot of attempts at generating terrains for other purposes like computer graphics and computer games [2] . The primary intent being the realistic description of features of planets that we live on, not just for an aesthetic perspective but also for entertainment.

The work in this paper is primarily divided into three parts. 1) Introduction to current terrain generation methods. 2) Proposal of the matrix method for terrain generation and its comparision with other methods. 3) Addition of additional effects that add an aesthetic appeal to the generated terrrains using statistical methods, so that they can be directly used in a variety of scenarios.

Paper is organized as follows. Section II describes the diamond square algorithm and the midpoint displacement algorithm. Section III postulates the algorithm and provides an overview of how the algorithm is similar to other algorithms but is optimized for numerical softwares that use matrices as their basic unit of computation. Section IV presents some statistical techniques that can be used presents experimental results showing results of images tested. Finally, Section V presents conclusion.

II. Diamond Square Algorithm

A. Heightmaps

In computer graphics, a heightmap or heightfield is a raster image used to store values, such as surface elevation data, for display in 3D computer graphics. A heightmap can be used in bump mapping to calculate where this 3D data would create shadow in a material, in displacement mapping to displace the actual geometric position of points over the textured surface, or for terrain where the heightmap is converted into a 3D mesh.

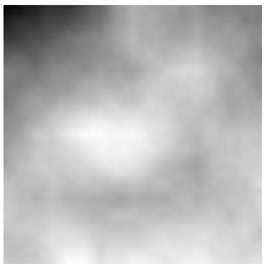


Fig. 1 A heightmap generated for this project. The whiter the area, the higher is its altitude. The darker areas signify low altitude.

A heightmap contains one channel interpreted as a distance of displacement or “height” from the “floor” of a surface and sometimes visualized as luma of a grayscale image, with black representing minimum height and white representing maximum height. [Figure 1]. When the map is rendered, the designer can specify the amount of displacement for each unit of the height channel, which corresponds to the “contrast” of the image. Heightmaps can be stored by themselves in existing grayscale image formats, with or without specialized metadata, or in specialized file formats.

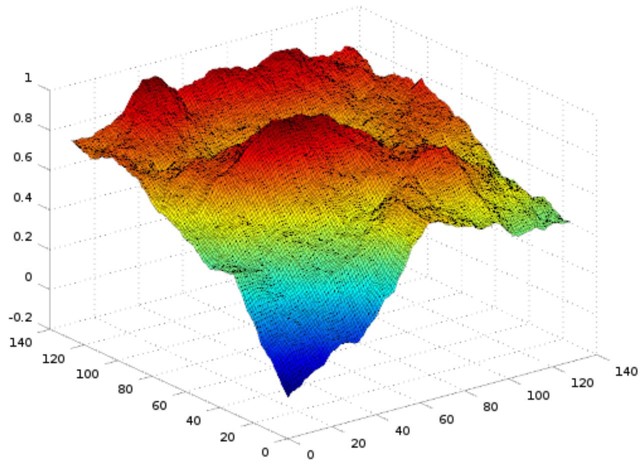


Fig. 2 A visualization generated for the above heightmap. The mountains and valleys can be seen clearly.

B. The Diamond Square algorithm

This algorithm is also known as the random midpoint displacement fractal, the cloud fractal or the plasma fractal, because of the plasma effect produced when applied. The idea was first introduced by Fournier, Fussell and Carpenter at SIGGRAPH 1982.[3]

The diamond-square algorithm starts with a 2D grid then randomly generates terrain height from four seed values arranged in a grid of points so that the entire plane is covered in squares.

The diamond-square algorithm begins with a 2D array of size 2n + 1. The four corner points of the array must firstly be set to initial values. The diamond and square steps are then performed alternately until all array values have been set.

(i) The diamond step. For each square in the array, set the midpoint of that square to be the average of the four corner points plus a random value.

(ii) The square step. For each diamond in the array, set the midpoint of that diamond to be the average of the four corner points plus a random value.

At each iteration, the magnitude of the random value should be reduced. (consider the figure towards the right of the page for a demonstration.)

III. Matrix Diamond Square algorithm

Given a 2x2 matrix Terr, with floating point values. Following is the code in MATLAB for the algorithm.

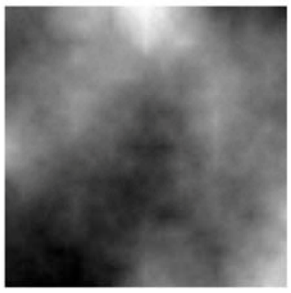
++

```
numIter = 7;
Terr = ones(2,2)*1000;
height = 1000;
for i = 1:numIter
        Terr = interp2(Terr,1);
        Terr = Terr + ( height (2 *rand(size(Terr)) –1));
        height = height/2;
end
Terr = Terr + min(min( Terr ) );
Terr = Terr / max(max( Terr) );
figure
imshow( Terr );
figure
mesh( Terr );
figure
surf( Terr );
imwrite( Terr, 'image.png');
```

++

IV. Results

Each time the loop is run, a new image containing the heightmap of the image generated is stored in the file called image.png . There are three representation of the terrain.
First is the grayscale heightmap generated, which looks like a plasma cloud. The white portion represent the heighest point of the terrain.



` Fig 2. Greyscale representation of the terrain

The second representation is in the form of a mesh, which takes the greyscale value as the height, and plots them as x, y and z coordinates giving it a 3-Dimensional mesh. This is where our terrain gets a 3D feel to it.

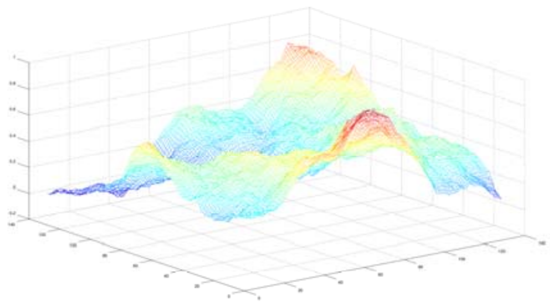
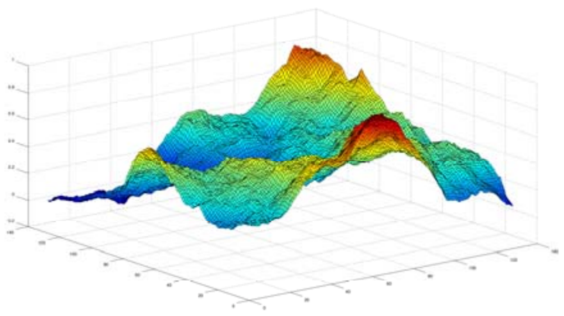


Fig 3. Mesh representation of the terrain

The third representation of the image is in the form of a surface, which plots a surface on the generated mesh. This is where our terrain actually looks like a real life mountaneous region, with mountains and valleys.

Fig 3. Mesh representation of the terrain



V. Conclusions

A matrix algorithm version of the diamond square algorithm was successfully implemented and tested in this paper. Different terrains can be generated by changing the initial value of the matrix instead of just '1'. Also, each time a random terrain would be generated. This can be controlled by seeding the random number generator to a particular value and resetting it each time the random matrix is generated. Also, the best performance of this algorithm will be when the size of Terr array is of the form 2N + 1 because there won't be any kind of memory collision.

.

Acknowledgments

We are thankful for the guidance and support of our mentor Dr. Badam Singh Kushvah, Assistant Professor, Department of Applied Mathematics, Indian School of Mines, Dhanbad.

References

[1] Hirt, C. "Digital Terrain Models.". Encyclopedia of Geodesy: 1–6. doi:10.1007/978-3-319-02370-0_31-1. ISBN 978-3-319-01868-3.
[2] Miller, Gavin S. P. (August 1986). "The definition and rendering of terrain maps". ACM SIGGRAPH Computer Graphics 20 (4): 39–48. doi:10.1145/15886.15890.
[3] Fournier, Alain; Fussell, Don; Carpenter, Loren (June 1982). "Computer rendering of stochastic models". Communications of the ACM 25 (6): 371–384. doi:10.1145/358523.358553