

# CSCI 4229 Assignment 3: Fractal Terrain

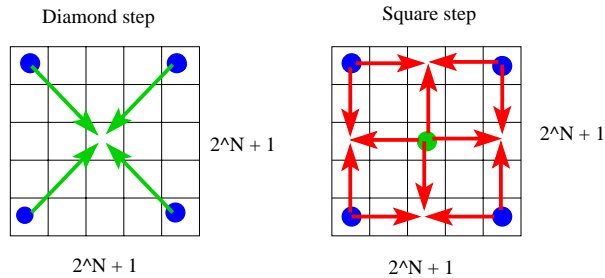
Due: Feb. 17.

Stochastic models provide a means of generating natural looking scenes, without the burden of explicitly modeling scene details. In this assignment you will write a GLUT/OpenGL program to generate and render a synthetic terrain map.

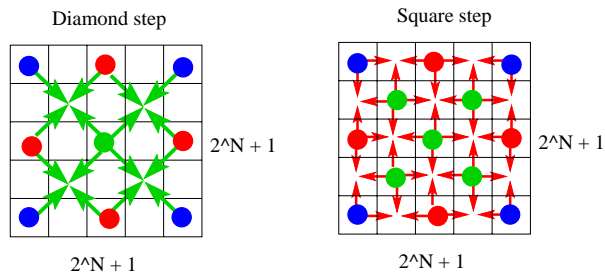
A terrain map describes the topography of an area by providing an altitude and color value for each  $(x, y)$  coordinate. A fractal terrain can be generated by starting with a coarse initial random map, then recursively adding self-similar detail at finer and finer scales. One such algorithm is the Diamond-Square algorithm of Fournier et al (CACM'82).

## Method:

1. Create a  $(2^N + 1)$  by  $(2^N + 1)$  matrix  $T$ , and assign a random altitude to the 4 corner locations.
2. **Diamond step:** assign the average of these 4 corners plus a perturbation to the midpoint of the grid.
3. **Square step:** assign the average of the 4 corners of each diamond plus a perturbation to the midpoint of the diamond.



4. At each following level of detail, the diamond step is applied to each square from the preceding level, then the square step is applied to each of these diamonds.



$N$  is the maximum level of detail. The perturbation  $p$  added at each point starts at level 1 with a coefficient  $0 < r < 1.0$ . At subsequent level  $n$ ,  $-r^n \leq p \leq r^n$ , which we can achieve by using a random number generator, for example:

```
p = 2*r*(double)rand()/(double)RAND_MAX - r;
```

where  $r$  is the current  $r^n$ . You should keep track of the min ( $T_{min}$ ) and max ( $T_{max}$ ) altitude for the entire map.

This description of fractal terrains is derived from <http://www.javaworld.com/javaworld/jw-08-1998/jw-08-step-p2.html>, which describes generating fractal terrain maps for Java 3D.

Using  $N = 5$  and  $r = .5$ , you will be able to generate the matrix  $T$  containing an altitude at each location. In order to render this terrain map, you will also require a color for each location. Generate a color matrix  $C$  containing an RGB vector for each map location. For  $T(i, j) < 0.7$ , generate a shade of green for the vertex based on the altitude and a perturbation  $-.1 < p < .1$ . For  $T(i, j) \geq 0.7$  generate a shade of gray based on altitude plus a perturbation  $-.1 < p < .1$ .

Render the terrain map using the color map and the `GL_TRIANGLE_STRIP` construct. Assume the altitude is the  $Y$  component, and that  $X$  and  $Z$  range from  $-.5$  to  $.5$ . Apply a translation  $t = [0, -.2, -3.0]$  and an  $X$  rotation of  $-15^\circ$  to the model view matrix. Set up your windows such that the entire terrain map is visible. Use GLUT's  $Z$  buffering.

Your program should allow the user to quit by striking 'q' or 'Q'. You should provide your own reshape function which scales the terrain with window size, maintaining the correct aspect ratio. You should define a popup menu for the left mouse button which allows the user to select from 'North', 'South', 'East' and 'West' views and 'Quit'. North views the terrain from the  $-Z$  axis toward the origin. South views the terrain from the  $Z$  axis toward the origin. East views the terrain from the  $X$  axis toward the origin. West views the terrain from the  $-X$  axis toward the origin.