CSE 167

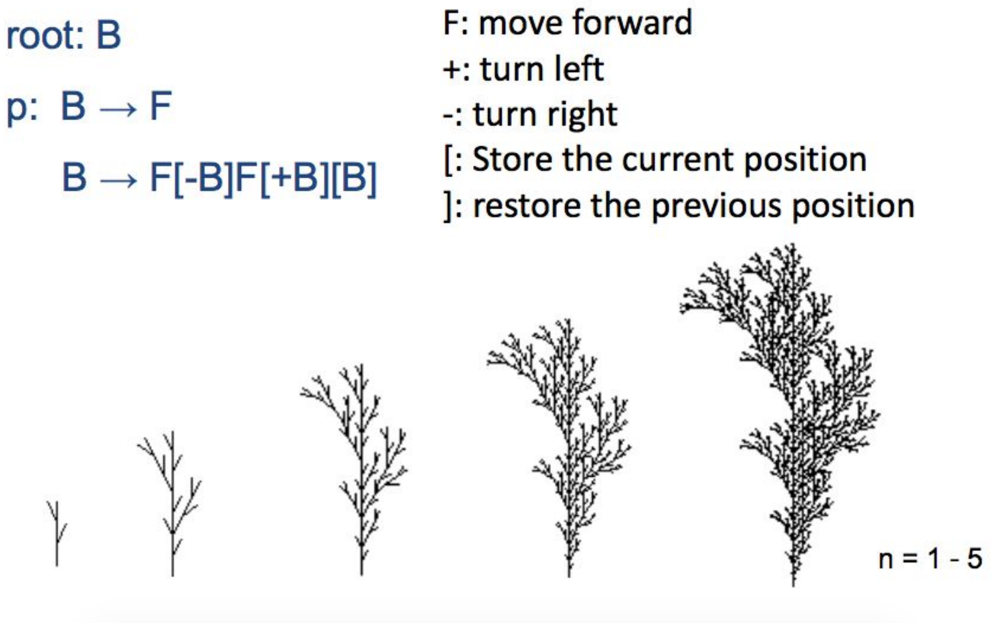
Discussion 09 ft. Jiasheng
12/04/2017

Contents

- Procedural plants with L-system
- Procedural terrain(credits to Kaiser)
- Bezier surface(credits to Yining)
- Water effects(credits to Jiasheng)

Procedural plants w/ L-system

- Grammar = {variables, constants, initial state, production rules}
- 2D example ref
-



root: B

p: B → F

B → F[-B]F[+B][B]

F: move forward

+: turn left

-: turn right

[: Store the current position

]: restore the previous position

n = 1 - 5

Procedural plants w/ L-system

- Requirements: at least 4 variables with parameters
- How to add parameters?
  - Treat each variable as a function
  - Now F[-B]F[+B][B] becomes e.g. F(0.5)[-(60)B]F(1.0)[+(90)B][B]
  - What does this mean?
  - You should be using some kind of regulated randomization for parameters
- How to extend this to 3D?
  - Add rotational variable(s): how much to rotate in xz-plane and how much to rotate w.r.t. y-axis
  - E.g. F(0.5)[&(90)^(30)B]

Procedural Terrain: Step 1, Make some Noise!

We are going to make a world/map with varying height coords for each (X, Y)

Create a noisy image (probably using Perlin Noise)

Perlin is random-ish, the information is more uniform

Each X, Y has a color (1-255) which can be height

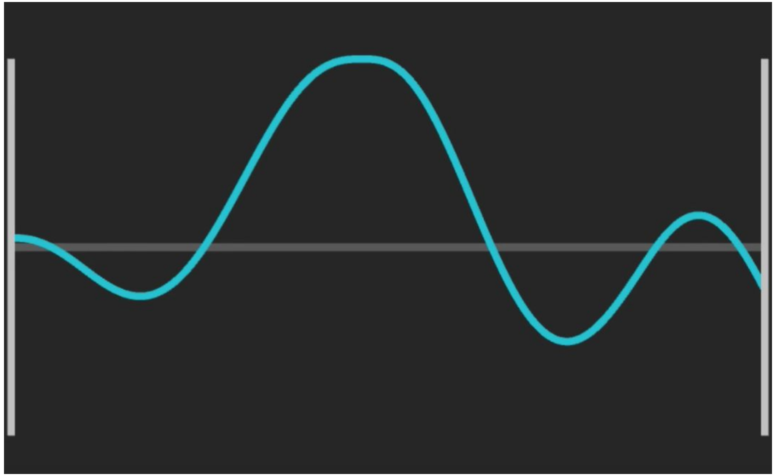
Procedural Terrain: Step 2, Interpret

Because the noise is continuous,

it resembles a wave!

Now we can assign sea-level, sand level,

mountains, etc.



Some suggestions for making it look nicer

- Dividing the terrain into ‘cells’
  - You can divide the terrain into small units like voronoi cells
- Grouping the cells into, well, ‘groups’
  - Using k-nn or some grouping/classification method
- Assigning ‘biomes’ to groups
  - Biome = [grassland, rocky areas, pond/lake/sea, sandy beach, etc]
  - Assign biome to each group using height, neighbor information
  - E.g. swamps can occur near ponds or lakes but not in sandy beach
  - E.g. sea is likely to occur at lower altitude and rocky areas in higher altitude

Bézier Surface

Interactive demo :http://kovacsv.github.io/JSModeler/documentation/examples/bezier.html

Bezier surface is **Bezier curves in 2D**. We had 4 control points for a cubic bezier curve, so we need 4*4=**16 control points** for a bi-cubic bezier surface.

Recall that when learning **parametric surfaces** in MATH20E, we used parameters u,v,s,t,etc. instead of x,y,z to represent a complex surface.

The equation for a point on a Bezier surface is (u,v are the parameters we chose along the two directions of the bezier curve, Bn,i() is the Bernstein polynomial equation, Pij are the control points):

$$P(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_i^n(u) B_j^m(v) P_{ij}$$

$$B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i}$$

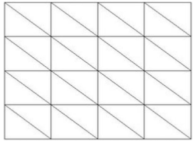
$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

Bézier Surface

For example, to draw one bi-cubic surface, you need to

1. Initialize 16 control points in a 4x4 2D array.
2. Given P(u,v), plug in u and v to the equations.
   a. You can also use matrix forms to simplify parametric equation.

3. After we have all the points, we have to connect them into a mesh. The way to do it is to connect 3 adjacent points to be a triangle.

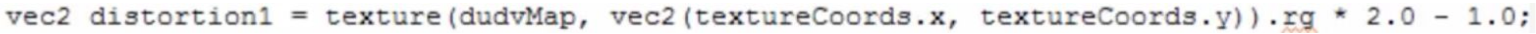


Finally, just like how we used 10 bezier curves to create a roller coaster trail, you can do something more interesting with bezier surfaces.

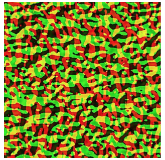
Water Effects

Water effects consist of both reflection and refraction. You need to distort both of them using a du/dv map and change the scene you can see at different angles using Fresnel effect.

Water is essentially a plane (mirror). You can add distortion to it to mimic the effect of waves by treating the dudv map as a texture and read off the pixel values



```
vec2 distortion1 = texture(dudvMap, vec2(textureCoords.x, textureCoords.y)).rg * 2.0 - 1.0;
```



Water Effects

Fresnel Effect takes in the angle from the camera to the point on the water. You should be able to see more refraction under the water if the angle (to the normal of water) is close to 0. Similarly, you should see more reflection that occurs above water if the angle is close to 90

for example, color = (1-projection) *reflectColor+ projection * refractColor

projection is the dot product of water normal and vector from pos to cam

You will need another framebuffer in order to include reflection other than environment mapping

Water Effects

Here is a full implementation for water effect

https://www.youtube.com/watch?v=HusvGeEDU_U&list=PLRIWtICgwaX23jiqVByUs0bqhnalNTNZh&index=1