Procedural generation and realtime rendering of planetary bodies

Martin Kahoun

MFF UK

18.04.2011

Table of contents

1. Introduction
2. Data representation
   - Level of detail
   - ROAM
   - Spherical ROAM
3. Terrain generation
   - Noise and fractals
   - Making noise
   - Implementation
4. Future work

Motivation

- Spare time of indie/hobby game developers spent on creating graphics content
- Eventually provide placeholder objects for demonstration purposes
- Not that much discussed topic
- Number of games already uses or could use procedurals:
    - *Spore*
    - *Elite*
    - *.kkrieger*
    - ...

Goal of the project

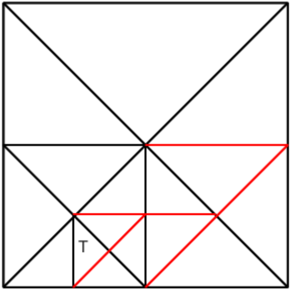
- Provide procedural planets for space based games
- Remove visual artifacts on poles
- Ensure reuse of generated content

LOD algorithms overview

- Various binary or quad tree approaches
- Realtime Optimally Adapting Meshes *(Duchaineau, '97)*
- Geometry clipmaps *(Lossaso, Hoppe, '04)*
- Spherical clipmaps *(Clasen, Hege, '06)*

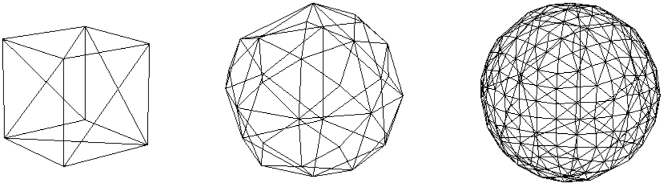
The ROAM algorithm

- Binary tree of right triangles:
  - Leaves are rendered
  - 2 meta objects (square, diamond)
  - Recursive splitting
- Removes T-junctions
- Ensures mesh watertightness
- Uses visible error metrics to split/merge triangles



Mapping ROAM to sphere

- Idea by O'Neil: take 12 ROAM triangles and form a cube, move every new vertex to the sphere surface
- Takes care of possible memory woes at outer edges
- Wouldn't be icosahedron better?
  - Not worth the effort



Fractal based methods overview

- Requirements:
    - No pre-generated data
    - Dynamic – yielding results on demand
    - Easily adopted for spherical landscape
- Studied approaches:
    - Fault lines
    - Plasma fractal
    - Noise & fractal Brownian motion

The fractal Brownian motion

Height as sum of noise samples

$$h = \sum_{i=n}^{k} w_i \cdot f(\mathbf{v} \cdot n_i)$$

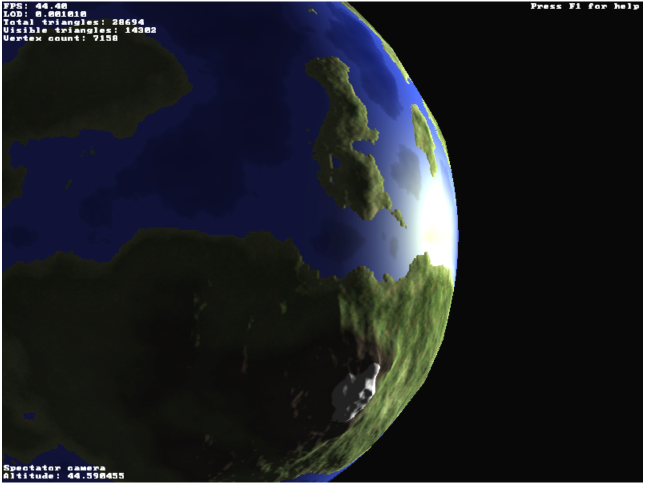
h	final height of the vertex $\mathbf{v}$
k	number of noise samples (also called octaves)
w_i	i-th octave weight
n_i	lacunarity (or exponent) of i-th octave
f	the noise function (3d Perlin noise in our case)

- Values of w_i and n_i either precalculated or sequentially changed between iterations

From geometry to shading

- Generate height lookup texture, normal map and "weather" map
- Create geometry (ROAMing)
- Render with GLSL shaders:
  1. Underwater vertices pushed to the sea level
  2. Height-based texturing
  3. Colour blending according to "weather" map
  4. Apply per pixel lighting (using normal map)

Produced image



```
FPS: 44.40
LOD: 0.001010
Total triangles: 28694
Visible triangles: 14302
Vertex count: 7158
```

```
Press F1 for help
```

```
Spectator camera
Altitude: 44.590455
```

Future work

- Better landscape generation algorithms:
  - Further study available approaches
  - Find a way to express weather zoning
  - Introduce rocks, etc.
- Improve world scaling (*Google Earth* effect)
- Remove lookup texture generation
- ...

Questions?

Thank you for your attention