Real-time Procedural Planet with OpenGL

Final presentation

Axel Angel

EPFL

June 2013

Table of Contents

Table of Contents

"Real-time Procedural Planet with OpenGL"?

Real-time : interactivity constraint: ~16ms per frame

Procedural : algorithmically generated

Planet : very wide terrain: 476×10^6 km^2

OpenGL : multi-platform rendering API

- Many applications:
  - Cinema
  - Simulations
  - Video games

  but yet few studies.
- Challenging:
  - Reactivity constraint
  - Realism
  - Creativity

Table of Contents

Planetary Mesh

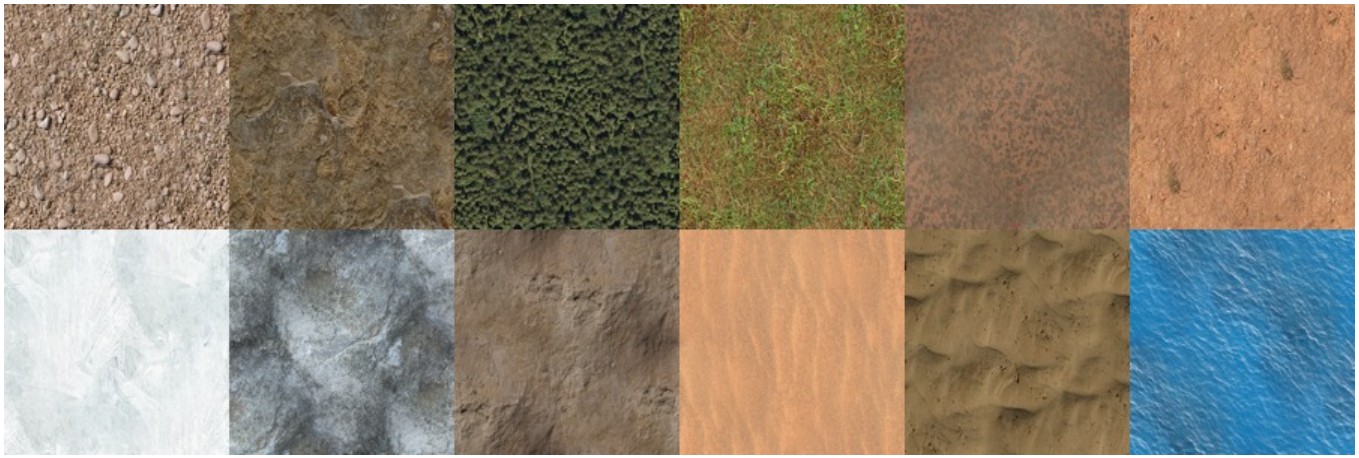
How to generate heightfield procedurally?

- Start with a “grid”
- For each point, generate height with noise
- Spherical? Not that easy

When to generate (new) points? SRAM split.

Land texturing

- Real photo: which set to choose from?
- Coordinates displacement
- Textures Blending: which textures and how?



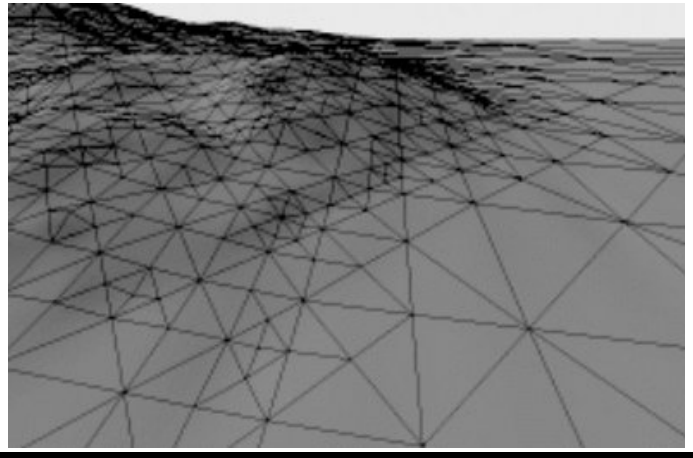
Sky and water texturing

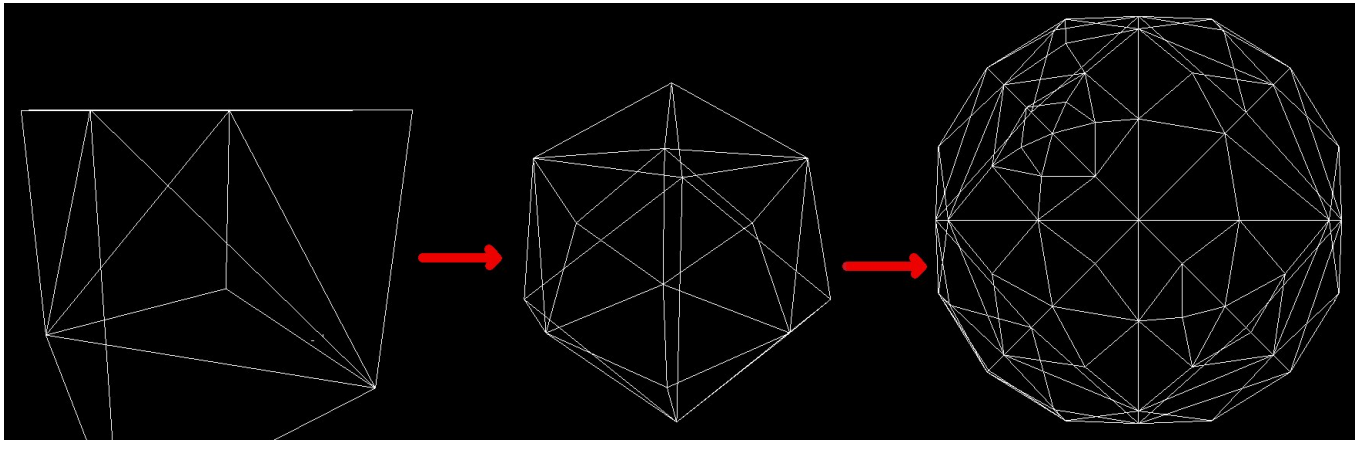
- Both based on fBm Perlin noise
- Different usage of the noise
- Animated and in real-time (shaders)

Table of Contents

Spherical **R**eal-time **A**dapting **M**esh (SRAM)

- Derived from Real-time Optimally Adapting Mesh (ROAM)
- Spherical shape
- Not optimal





Encountered difficulties with:

- Shaders: difficult to debug (visual feedback)
- Buffers: hard to guess which better
- Optimisations
  - Partial update
  - Quadrant zones
  - Activation distance
  - SRAM deadline
  - ...

Table of Contents

Now it's a great time to show them my beautiful demo

Table of Contents

- Interesting results
- Valuable experiences
- Performance
- Goal reached?

Would have added lots more but time runs out so fast.