Crafter Engine

Real-Time Volumetric Terrain Rendering and Data Processing
By
Lukasz Bator
Introduction

• Problem

• Terrain Rendering Approaches
  – Height Map only approaches
  – Object Substitution
  – Multiple Height Maps
  – Procedural Generated Terrain.
  – Volumetric/Voxel Rendering

• Crafter Engine

• Results
Problem definition

Create Terrain that can be easily modified in real-time with:

- Cliffs
- Caves
- Tunnels
- Overhangs
Height Map only

Pros:
- Very simple to implement
- Fast to render
- Very Good on memory
- Very Good on CPU
- Easy terrain modification
- Fast Collision detection

Cons:
- No Caves
- No Overhangs
- No Cliffs
- Needs to be supplemented
Object Substitution

Pros:
- Simple to implement
- Height-Map Based
- Fast To Render
- No shape limitations
- Good on CPU
- Good on Memory

Cons:
- Needs models for all objects
- Terrain modification creates extra problems
Multiple Height Maps

Pros:

- Good on Memory
- Height-Map based
- Removes some shape limitations

Cons:

- Hard to implement
- Still some shape restrictions exist
- Slower than Height Maps
- Real-time terrain alteration very hard to implement.
**Procedural Generation**

**Pros:**
- Very fast with GPU
- Very efficient with memory
- No shape restrictions

**Cons:**
- Usually data is stored as a “seed”
- Easy alteration
- Requires newer GPU with Geometry Shaders
Volumetric/Voxel Terrain

Pros:
- No shape restrictions
- Very easy LOD
- Very easy real-time terrain changes
- Combination of Height-Maps and Procedural Generation

Cons:
- Very Large Memory requirements
- Hard to implement
Crafter Engine (on CPU)

- Collision Detection
- Basic Gravity
- Infinite Terrain
- Block Picker tool
- Simple Light system
Crafter Engine: Structure

Data Generation

- Generate Voxel type information (Rock, Sand etc.)
  - Height-Map
  - Perlin Noise
- Compute Occlusion of Voxels
- Compute Light partial information
Crafter Engine: Structure

Data Processing
- Encoding Voxel's data
- Decoding Voxel's data
- Data Streaming
- Saving Data to HDD
- Collision Detection
- Ray Casting for Block Picker
Crafter Engine Structure

Render
- Very simple Tessellation
- Light information computations
- Double Buffering
- Viewing frustum culling
Results

- 16x16 Chunk Area in memory
- 16x16x128 Blocks in Chunk
- 8,388,608 Total blocks with real-time operations.
- Rough estimate for Memory: 1.75GB without optimizations
Results

- All core utilization during heavy load (Terrain Generation)
400MB RAM usage by TestGame with Crafter Engine
Research References

http://cg.alexandra.dk/tag/ray-marching/


MA, USA, December 2009.

Denmark.


Questions?