Mesh to Voxel Transformations for Optimised Physics-Based Interactions

T. Lefley F. Ponjou-Tasse

Progress Report, 2015

Motivation

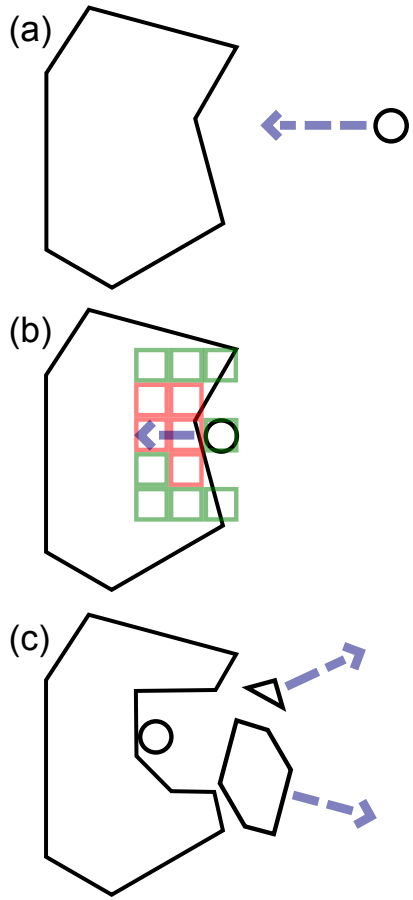

Figure: A simple collision and its volumetric resolution.

- Meshes only store surface information
- Reasoning on interior difficult
- Most destruction algorithms limited
  - Only work on convex shapes
  - Or have to split concave shapes first
- By precomputing volume data we have an O(1) check for inside/outside a shape

Voxelisation

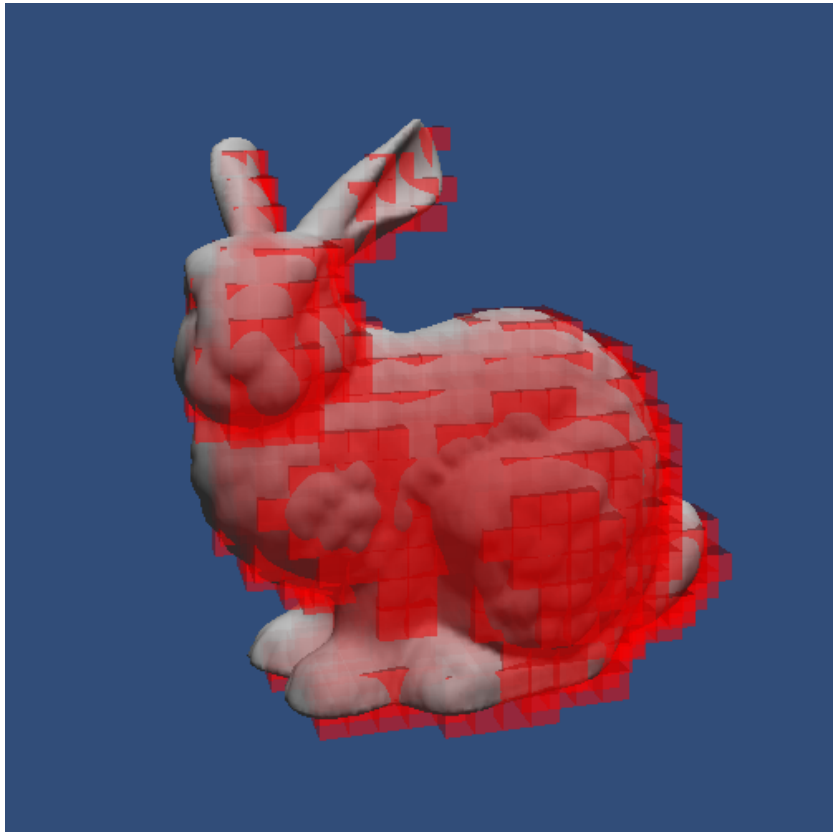

Figure: Voxelisation of the 'Stanford Bunny' model, composed of 69,666 triangles.

- HLSL GPU implementation
- Triangle phase
  - Project triangle onto XZ plane
  - Find intersected voxel columns
  - Mark first voxel below triangle
- Propagation phase
  - Moves up Y plane
  - XORs with voxels below

Destruction

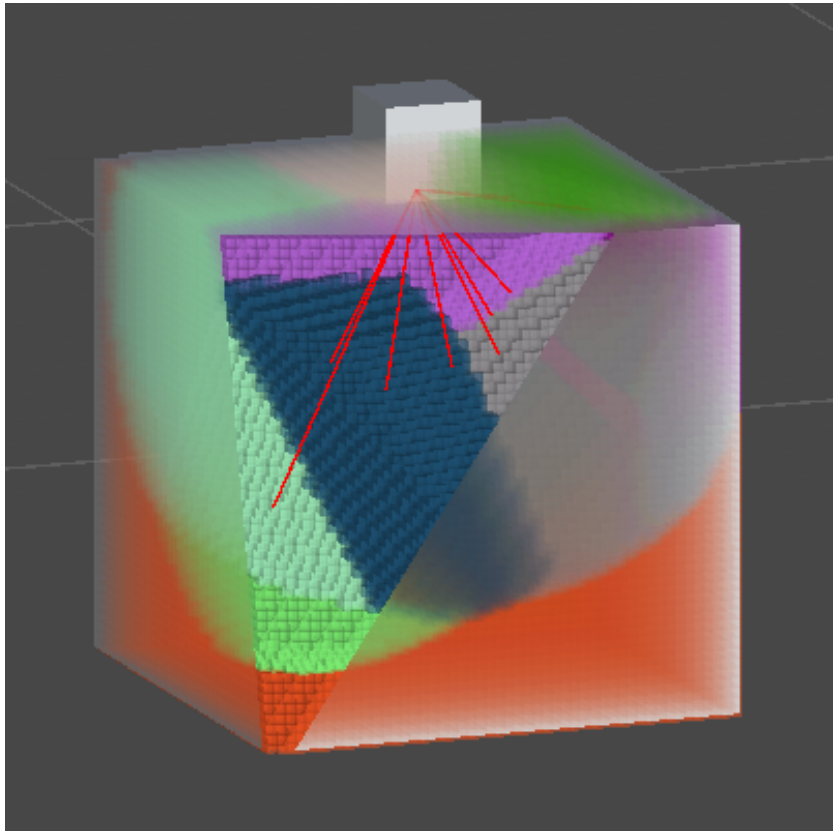

Figure: Labelling of voxels to their correct fragments.

- Process collision information
  - Find collision point
  - Calculate force
  - Generate fragment points
- Construct 3D voronoi diagram
  - Label each voxel by nearest point
  - Within radius
- Find islands
  - Flood fill

Rebuilding the Mesh

Marching Tetrahedrons

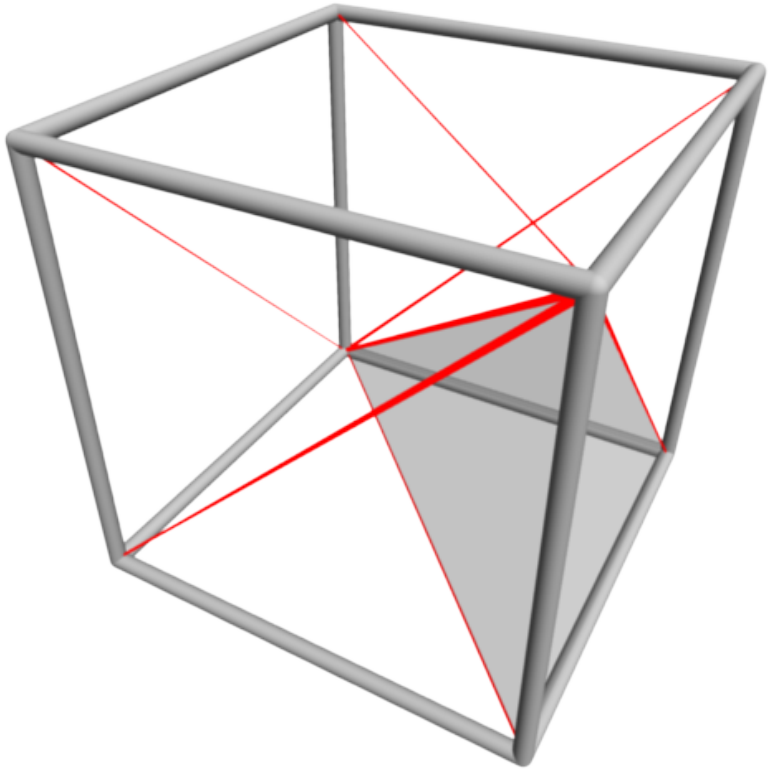

Figure: A cube divided into six tetrahedra, with one tetrahedron shaded.

- Form triangles
  - For each cube of 8 voxels
  - Look up correct triangle

Rebuilding the Mesh

Marching Tetrahedrons

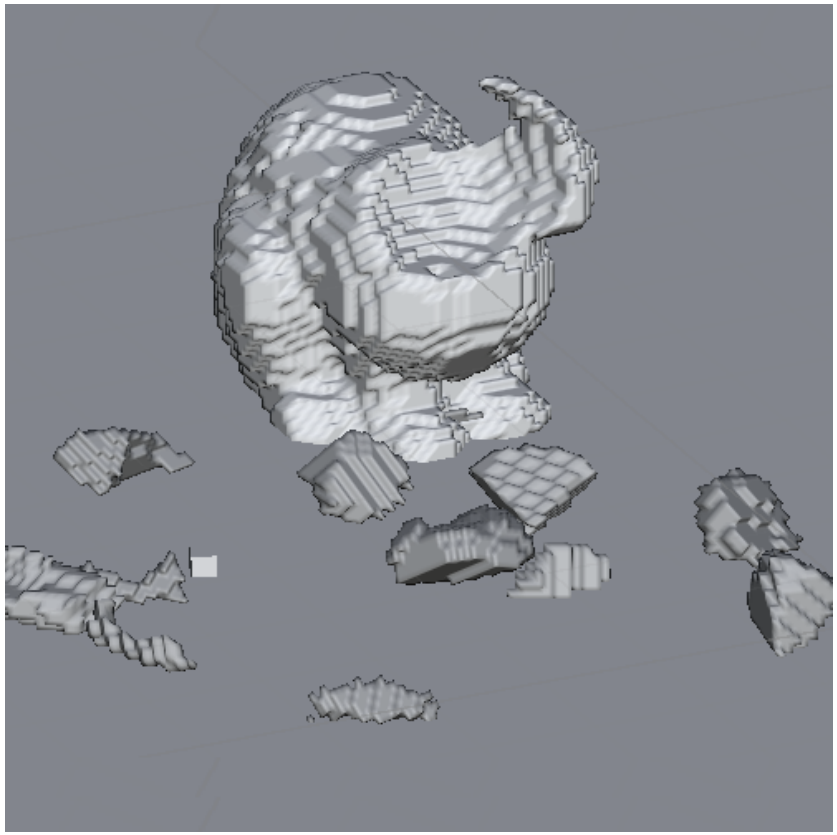

Figure: Fractured Stanford Bunny remeshed using the Marching Tetrahedrons algorithm.

- Form triangles
  - For each cube of 8 voxels
  - Look up correct triangle
- Problems
  - Too slow (implementation?)
  - Wasted complexity on interior voxels
  - Detail loss
- Working on new solution

The Next Steps

- Implement new remeshing algorithm
- Optimise for speed
  - Multithreading
  - Remove unnecessary complexity
- Post destruction calculations
  - Fragment mass
- Evaluation
  - Framerate
  - Memory usage
  - Physical accuracy
- Write-up