Accelerating Ray Tracing using Constrained Tetrahedralizations

Ares Lagae & Philip Dutré – Katholieke Universiteit Leuven

Contribution

- Constrained tetrahedralizations, a new acceleration structure for ray tracing
- Exploratory work

Motivation

- Acceleration structures for ray tracing are studied in computer graphics for decades
- Yet, several problems remain, e.g. handling dynamic and deforming geometry
- Acceleration structures for ray tracing are also studied in computational geometry
  - BVH, kd-tree, grid
  - Mostly practical solutions
  - Computational geometry
  - Delaunay ray triangulation
  - Mostly theoretical solutions
- Apply acceleration structure from computational geometry in computer graphics

2D Constrained Triangulations

- 2D triangulation + constraints (edges)
  - 2D triangulation is built from a set of vertices
  - The constraints are edges that must appear in the triangulation

3D Constrained Tetrahedralizations

- 3D tetrahedralization + constraints (faces)
  - 3D tetrahedralization is built from a set of vertices
  - The constraints are faces that must appear in the tetrahedralization

Ray Traversal

1. Locate ray origin
2. Traverse tetrahedralization, one tetrahedron at a time
3. Stop at constrained face

- Locating the ray origin is potentially costly
- Accelerate
  - Grid, monotone subdivision
- Avoid by exploiting ray connectivity
- Rays start at camera position or where previous ray ended

- Traverse tetrahedralization
  - One tetrahedron at a time
  - Given entry face, determine exit face
  - Several methods

Examples

- Intersecting ray
- Non-intersecting ray

Construction

- Piecewise linear complex (PLC)
  - Very general geometry representation
  - Arbitrary polyagons, holes, non-manifold geometry, ...
  - Polyagons must properly intersect
  - Tetrahedralizations cannot have intersecting faces

1. Triangle soup → PLC
   - Eliminate self-intersections
2. PLC → constrained tetrahedralization
   - TetGen, CGAL

Ray Tracing Cost

- Comparison with kd-tree
  - Ray tracing cost = number of tetrahedra / nodes

- Teapot-in-a-stadium problem

Advantages

- Deforming and dynamic geometry
- Deforming → deform tetrahedralization
- Dynamic → efficient update
- Time complexity of ray traversal
- Constrained tetrahedralization
- Linear in local geometric complexity
- Hierarchical acceleration structure
  - Logarithmic at best in global geometric complexity
- No practical results yet
- Effort might only show up for large scenes

- Optimal constrained tetrahedralizations
  - Weight tetrahedralization = SAH for kd-trees
  - Unified data structure for global illumination
  - Associate data with vertices, edges, faces and tetrahedra
  - Level-of-detail
  - Meshes and triangulations use similar data structures

Disadvantages

- Constructing constrained tetrahedralizations
  - TetGen, CGAL
- Geometry preconditioning
  - Eliminating all self-intersections from triangle soup
- Absolute performance
  - Limited testing, limited optimization