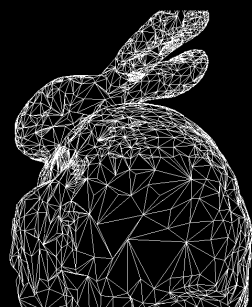
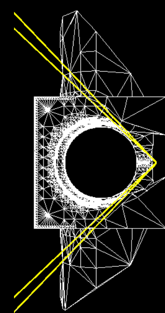
Dynamic Meshing Using Adaptively Sampled Distance Fields





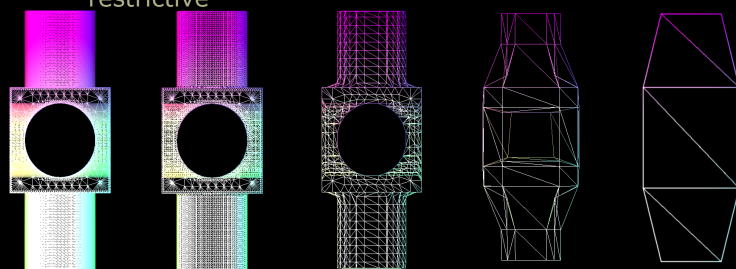
Jackson Pope, Sarah F. Frisken and Ronald N. Perry
MERL – Mitsubishi Electric Research Laboratory

Level of Detail Meshing

- Models often feature more detail than required
  - laser range scanners generate over-triangulated meshes
  - distant objects can be less detailed with no loss of visual quality
  - real-time applications have restricted frame time available for rendering objects
- Render objects differently according to viewing parameters $\Rightarrow$ Level of detail meshing:
  - pre-generate one or more *static* meshes
  - adaptively refine and decimate a *dynamic* mesh

Static Level of Detail Meshes

- Use different models according to visual importance
  - + simple, usually distance based
  - – visual artifacts when switching models
  - – restrictive



Five static LOD models of the same object with 202264, 45544, 3672, 232 and 28 triangles respectively

Dynamic Level of Detail Meshes

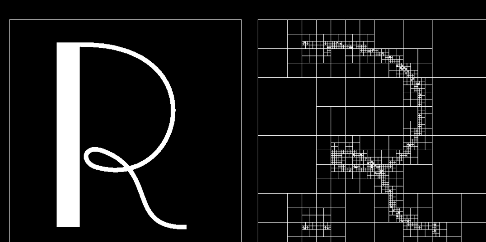
- Single mesh created, elements added and removed according to viewing parameters
  - + optimal mesh for every frame
  - – requires per frame calculation
- Examples: View Dependent Progressive Mesh (Hoppe 97), Hierarchical Dynamic Simplification (Luebke and Erikson 97)
  - fast, but limited by input geometry
  - hard to integrate with other requirements, e.g. collision detection

Adaptively Sampled Distance Fields

- Distance fields represent the distance to the object surface
  - can be signed to denote inside/outside
  - can be minimal distance
  - gradient of the field corresponds to surface normal (at the surface) or direction to the surface
- Adaptively Sampled Distance Fields (ADFs)
  - sample the distance in a spatial domain
  - hierarchical, to enable adaptive, detail-directed sampling
  - the dynamic meshing implementation is octree based

Adaptively Sampled Distance Fields

- Distance values stored in octree cell corners
  - three types of cells: inside, outside and surface
  - all distance values for inside cells are inside
  - all distance values for outside cells are outside
  - surface cells contain both inside and outside values

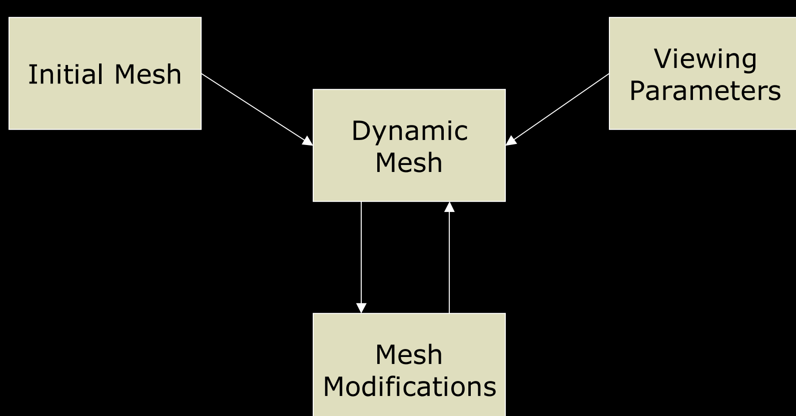


An object and its 2D ADF showing adaptive cell subdivision

Triangulating ADFs

- Generate a mesh vertex in each surface cell
  - different to edge based techniques (e.g. Marching Cubes)
- Connect vertices to those in neighbouring cells to form triangles
- Mesh vertices are *relaxed* onto the surface by following the distance field to the surface
- Algorithm details available in:
  - Frisken and Perry, “Kizamu: A System for Sculpting Digital Characters”, SIGGRAPH 2001

Dynamic Meshing Using ADFs

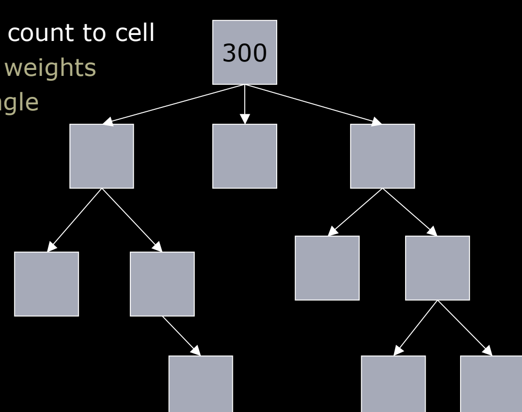


Initial Mesh Creation

- Data Preparation in surface cells
  - store a normal cone in each surface cell to enable fast back-face culling and silhouette detection
  - normal cone in a cell encompasses those of its children
  - create a neighbour look up table
  - position a mesh vertex on the surface in each cell

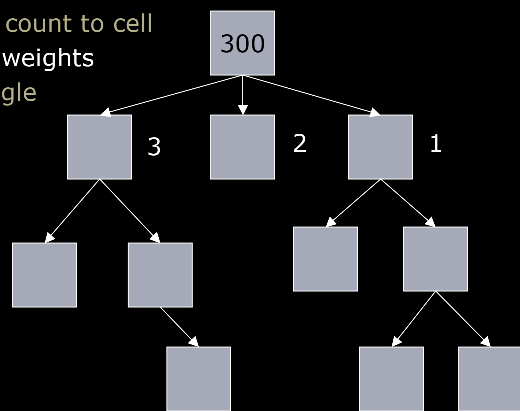
Initial Mesh Creation

- View-independent mesh creation
  - Assign triangle count to cell
  - Calculate child weights
  - Distribute triangle count between children



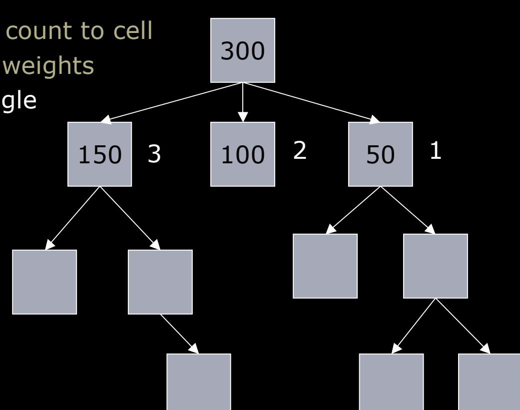
Initial Mesh Creation

- View-independent mesh creation
  - Assign triangle count to cell
  - Calculate child weights
  - Distribute triangle count between children



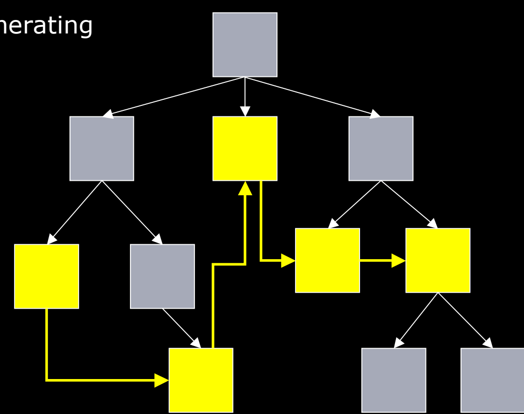
Initial Mesh Creation

- View-independent mesh creation
  - Assign triangle count to cell
  - Calculate child weights
  - Distribute triangle count between children



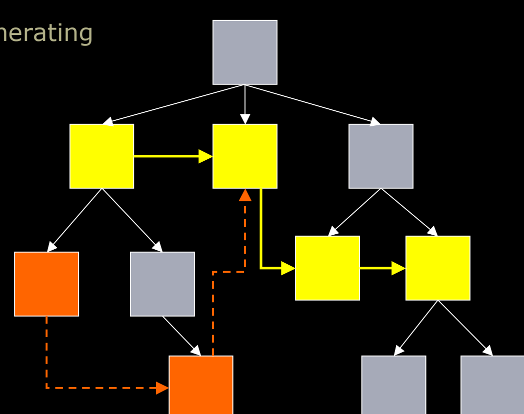
Dynamic Meshes

- Updating the active list
  - List of cells generating triangles
  - Ascend tree to reduce detail
  - Descend tree to increase detail



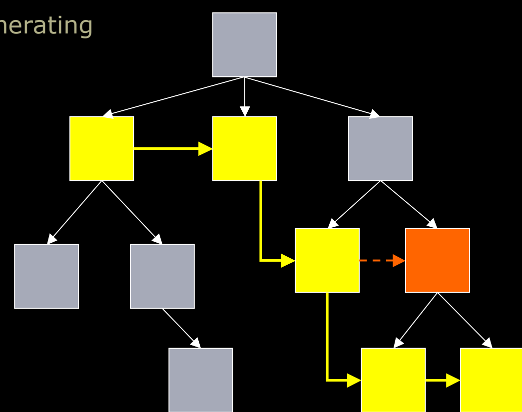
Dynamic Meshes

- Updating the active list
  - List of cells generating triangles
  - Ascend tree to reduce detail
  - Descend tree to increase detail



Dynamic Meshes

- Updating the active list
  - List of cells generating triangles
  - Ascend tree to reduce detail
  - Descend tree to increase detail



Viewing Parameters

- Cells are weighted according to viewing parameters and weighting functions
- We currently use:
  - on silhouette, back-facing, in view frustum, surface roughness
- Can also use:
  - distance to viewpoint, screen-space projected error, specular highlight,...

Conclusions

- ADFs can be used to generate dynamic triangle meshes
  - triangle meshes generated from ADFs are detail-directed, low triangle counts remain in areas of geometrical simplicity
  - octree structure provides framework for fast view frustum and back face culling
  - resulting dynamic mesh has low triangle counts in smooth and visually unimportant regions
- In addition, ADFs provide useful information for collision detection and real-time physics

Demo!