Visibility Algorithms III: Occluders

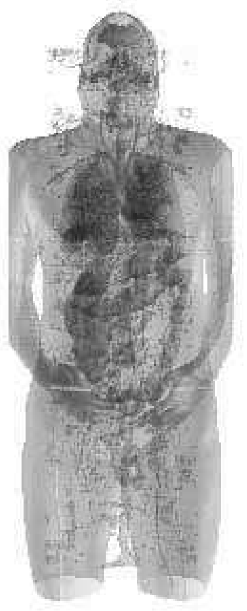
How to determine which surfaces are visible to the viewer and which are hidden and by how much

Input: Collection of isosurfaces

Output: Visible isosurface rendering

(Show Animation)

Isosurfaces



- Zero contours or level Sets of Functions
- Search and extraction of triangulated isosurfaces

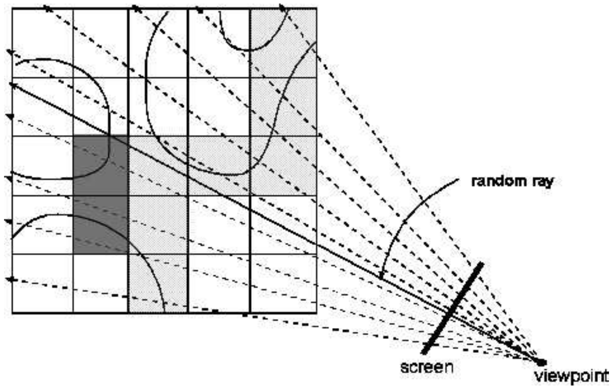
Outline: Visible IsoSurface Rendering using Occluders

A subset of polygons are usually chosen as occluders for a polygonal scene. However, polygons of an isosurface do not exist before the isosurface is extracted. For isocontouring, datasets are statically partitioned into blocks and therefore we consider choosing occluding blocks, which contain polygons to be rendered to create the occlusion map.

Occluder Selection: In the first stage of the algorithm, a number of rays are cast from the viewpoint to the volume in order to obtain an approximation of visible blocks. These blocks are treated as *occluding blocks*. Isosurfaces inside these occluding blocks are extracted and then rendered to create an occlusion map for the second stage of the algorithm.

Visibility Culling: In the second stage, the remaining blocks are traversed and culled against the occlusion map. Only isosurfaces in the visible blocks are extracted and rendered.

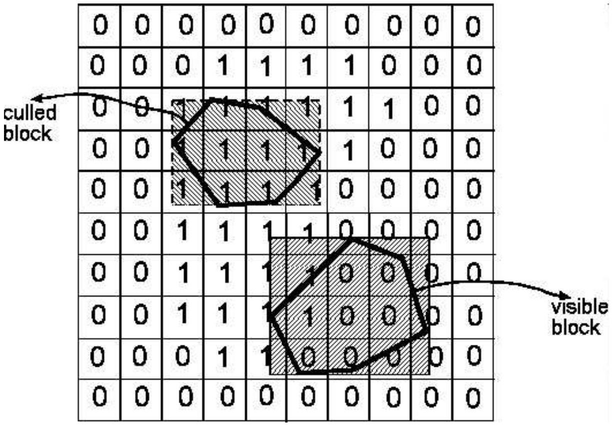
Occluder Selection



We use ray-casting to choose the preliminary set of occluding blocks by shooting a number of rays from the viewpoint into the volume. Any block chosen this way must be a visible block since from the viewpoint to this block there exists at least one ray that is not blocked by any other polygon in front of it. We use a combination of deterministic and randomly oriented rays. Neighboring blocks to the ray sampled blocks can also be added to fill ”overhanging” holes.

In the second stage, we first extract isosurfaces from the occluding blocks and render them to create an *occlusion map*. The remaining blocks are traversed and culled against the occlusion map. A hierarchical occlusion map may be constructed in order to decrease the time of visibility tests and occlusion map updates.

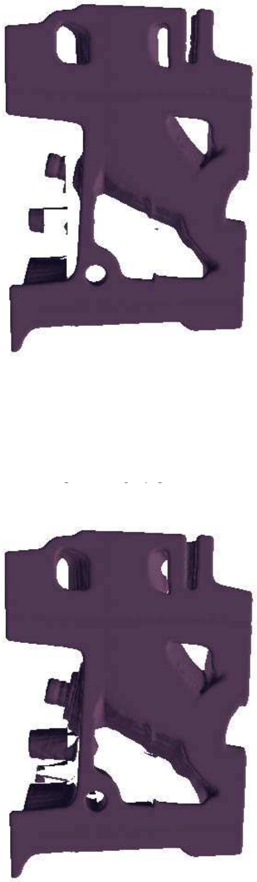
Visibility Culling



0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	0	0	0
0	0	1	1	1	1	0	0	0	0
0	0	1	1	1	1	0	0	0	0
0	0	1	1	1	1	0	0	0	0
0	0	1	1	1	1	0	0	0	0
0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

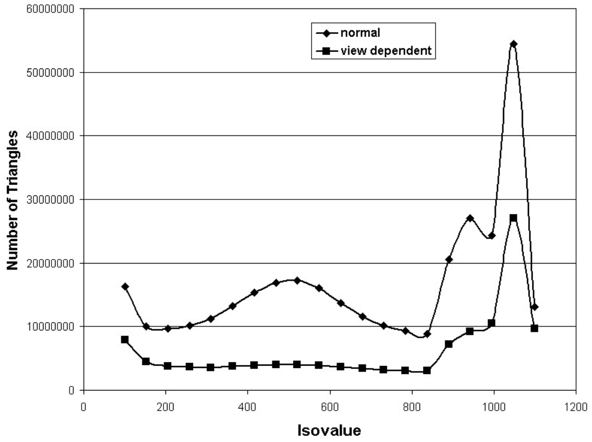
An occlusion map or mask is a $2D$ bitmap corresponding to the image rendered with polygons extracted from the blocks. An entry of the mask is set to 1 if the corresponding pixel in the rendered image is covered, otherwise it is set to 0. Visibility tests are applied to only those blocks whose ranges contain the isocontour. Eight vertices of each such block are projected onto the image plane. If there is any 0 entry of the occlusion map within the projected area, the block is considered as visible. This culling process also includes view frustum culling since blocks with projected bounding boxes outside the screen are automatically culled.

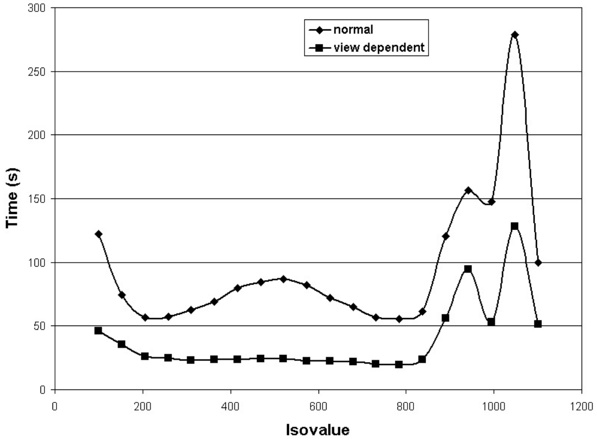
Visible Isosurface Rendering of an Engine Block



(55%)

Comparisons: Visible Isosurfaces vs All Isosurfaces





Reading Assignment and News

Topics of Midterm II are Illumination I - VI, and Visibility I - III, as well as Projects I - IV and the questions therein. The format shall be similar to Midterm I.

Please also track the News section of the Course Web Pages for the most recent Announcements related to this course.

(http://www.cs.utexas.edu/users/bajaj/graphics25/cs354/)