Single-pass Shadow Volumes for Arbitrary Meshes

We introduce a new method for rendering shadow volumes of arbitrary meshes as fast as those of specially prepared models, i.e., with a single pass over the shadow geometry. We use only unsigned 8- or 16-bit formats, which are supported on all major video-game console and workstation GPUs.

Algorithm

Group shadow polygons by their associated $k \in \{+1, -1, +2, -2\}$ increment (see [AW] for details). Accumulate each of those increments in a separate channel of an off-screen color buffer (no stencil buffer is used). A $k = +1$ shadow polygon is colored with RGBA=(1,0,0,0), one with $k = -2$ is colored (0,0,0,1), and so on.

Render all faces in a single pass with additive blending. The key idea is counting instances of each kind of polygon rather than the sum of their $k$ values, so that all intermediate values fit in positive integer values. The resulting buffer (fig. 2) has the property that at each pixel, $s = (R - G) + 2(B - A)$ (1) is zero if and only if that pixel is in shadow even for hard models like this Mobius ring.

To render the visible scene (fig. 3), modulate lighting by $(s == 0)$, as if using a shadow map, to produce shadows.

Related Work

A boundary ("crack") is an edge adjacent to only one face in a mesh. The preferred GPU stencil-buffer implementation of the shadow volume algorithm [EK] requires boundary-free meshes with no exposed backfaces. This precludes many models, such as: unfinished data during editing (fig. 1), cutaway views, pre-existing or stock art assets, and measured scientific data sets. In each case, adjusting the mesh to eliminate boundaries is either impractical or inappropriate. Aldridge and Woods lift the watertight requirement by incrementing the stencil buffer by $\pm k$, where $k = 2$ for most geometry. GPUs only support $\pm 1$ stencil increments, so multiple passes reduce shadow rendering performance by at least 2x. Our approach retains the original performance while operating on arbitrary meshes. Roettger et al. previously used alpha multiplication by $\frac{1}{2}$ and 2 for shadow counting; our method achieves orders of magnitude more range before saturation and accommodates $|k| = 2$ increments at no additional cost.


Fig. 4. a) [EK]’s stencil counts b) Our color counts. c) Both render the same visible scene for this model.

Fig. 5. Classic graphics models with and without boundaries. 20fps on GeForceGo 7900 with 415k shadow polygons.

Fig. 6. 20x20x20 grid of quads casting thousands of correctly overlapping shadows.

Fig. 7. Game example: 26 characters, 500k visible polygons, tone-mapping, and full self-shadowing at 30fps. Ogres have boundaries and knights are crack-free.