Single-pass Shadow Volumes for Arbitrary Meshes (sap_0247)

Morgan McGuire*
Williams College

Figure 1. Left: An unfinished townhouse model during a CAD editing session. Red edges are boundaries and red faces have incorrect winding direction. Center: A fast stenciled shadow method [EK02] fails on this mesh; [AW04] is correct but makes 2 passes. Right: We generate correct results in 1 pass.

1 Introduction
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.

A boundary is an edge adjacent to only one face in a mesh. The preferred GPU stencil-buffer implementation of the shadow volume algorithm [EK02] requires boundary-free meshes with no exposed backfaces. This precludes many models, such as: unfinished data during editing (figure 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 [AW04] lift the watertight requirement by incrementing the stencil buffer by $\pm k$, where $k = 2$ for most geometry. GPUs only support $\pm 1$ increments, so multiple passes reduce shadow rendering performance by at least 2x. Our approach retains the original performance while operating on arbitrary meshes.

2 Algorithm
We group shadow polygons by their associated increment $k \in \{+1, -1, +2, -2\}$. We entirely forgo the stencil buffer and instead accumulate each of those increments in a separate channel of an off-screen color buffer. 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. We 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. The resulting buffer (figure 2 bottom) has the property that at each pixel,

$$s = (R - G) + 2(B - A)$$  

is non-zero if and only if that pixel is in shadow. When rendering the visible scene (figure 2 top), we therefore modulate lighting by $(s = 0)$ to produce shadows.

Roettger et al. [RI02] 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.

A further benefit of our approach is that it builds signed accumulation on unsigned data formats and adders. That is important because fast 8-bit fixed-point buffers and most deployed consoles do not support signed accumulation. Although some new high-end hardware has signed floating-point accumulation, tricks for avoiding it may remain important for at least another decade; a number of U.S. patents—e.g., 7034849, 6650327—are not being licensed and prevent many vendors from implementing of signed blending on their GPUs.

Figure 2. Top: Mobius strip with boundary and exposed backfaces; and correct shadow. Bottom: Color count buffers.


* e-mail: morgan@cs.williams.edu