Shadow Volume Reconstruction from Depth Maps

Michael D. McCool
University of Waterloo

Current graphics hardware can be used to generate shadows using either the shadow volume or shadow map technique. However, the shadow volume technique requires access to a representation of the scene as a polygonal model, and handling the near plane clip correctly and efficiently is difficult; conversely, accurate shadow maps require high-precision texture map data representations, but these are not widely supported.

We present a hybrid of the shadow map and shadow volume approaches which does not have these difficulties, and leverages high-performance polygon rendering. The scene is rendered from the point of view of the light source and a sampled depth map is recovered. Edge detection and a template-based reconstruction technique are used to generate a global shadow volume boundary surface, after which the pixels in shadow can be marked using only a one bit stencil buffer and a single-pass rendering of the shadow volume boundary polygons. The simple form of our template-based reconstruction scheme simplifies capping the shadow volume after the near plane clip.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*shadowing*; I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis—*range data*

General Terms: Algorithms, Human Factors, Performance

Additional Key Words and Phrases: shadows, hardware accelerated image synthesis, illumination, image processing.

1. INTRODUCTION

Shadows are a very important spatial cue. They help determine the relative positions of objects, particularly depth order and the height of objects above the ground plane. Since a shadow is basically a projection of the scene from an alternative viewpoint, cast shadows can also elucidate the shape of an object [Wanger 1992] and the positions of light sources.

Generating shadows is a classic computer graphics problem, and considerable research effort has been devoted to it [Crow 1977; Woo et al. 1990]. In this paper we

This research was sponsored by a grant from the National Science and Engineering Research Council of Canada.
Michael D. McCool, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1, mmccool@cgl.uwaterloo.ca; http://www.cgl.uwaterloo.ca/~mmccool/.
To appear in *ACM Transactions on Graphics*, 2000.

focus on the simplest case: hard-edged umbral shadows cast by point sources. Given a fast umbral shadow algorithm, soft shadows can be approximated by summing the contributions of several such sources. However, there are severe limitations in existing algorithms for interactive generation of even hard-edged shadows using the current generation of hardware.

1.1 Taxonomy and Tradeoffs

Existing hard shadow algorithms can be divided into four broad categories: ray casting, projection, shadow volumes, and shadow maps. Of these, only the last three are (currently) applicable to real-time rendering. There are also several hybrid algorithms that combine features from different categories.

A distinction can also be drawn between object-precision and image-precision algorithms. Object-precision algorithms result in more precise shadows, but require access to a polygonal representation of the scene.

Polygonal representations are not available in systems which use "alternative" modelling and rendering techniques such as selective raycasting of objects, run-based rendering of enumerated volumes, layered depth images, depth buffer implementation of CSG [Rossignac and Requicha 1986; Wiegand 1996], distance-volume primitives [Westermann et al. 1999], direct rasterization of biquadratic patches, depth shaders, adaptive isocontour tracing of parametric patches [Elber and Cohen 1996], etc.

Image-precision algorithms are less accurate but also place fewer constraints on the scene representation. In practice, even if the scene is represented with polygons, they can be easier to use and more flexible. For instance, many applications use polygonal primitives but may generate them on-the-fly from other representations, and do not maintain a polygonal database. Even if polygonal primitives are used and can be intercepted in a non-invasive way (for example, via OpenGL feedback [Kilgard 1997]), adjacency information (required for an important optimization in shadow volumes, for instance) may not be available. Fortunately, for many applications shadows do not have to be very accurate to be useful, and image-precision algorithms can be used.

In the following sections the three classes of shadowing algorithms applicable to real-time rendering are surveyed and compared. Our new technique is a hybrid of the shadow volume technique and the shadow map technique.

1.2 Projection Algorithms

Projection techniques project primitives away from the light source onto the surface of other primitives. Blinn's "fake shadows" algorithm [Blinn 1988] squashes all polygons in the object casting a shadow down onto the plane of another polygon, where they can be drawn in black or composited over previously rendered pixels to approximate a shadow of the object onto that plane. This technique is simple and is suitable for hardware acceleration, but does not scale or generalize well.

More general object-precision projection algorithms clip polygons into shadowed and unshadowed parts [Atherton et al. 1978; Weiler and Atherton 1977] in a prepass. Since they use polygon clipping, these algorithms require access to a polygonal representation of the scene.

Data structures such as BSP trees may be used to accelerate the polygon clipping

process [Chin and Feiner 1989] and manage the selective reclipping of polygons when objects are moved [Chrysanthou and Slater 1995]. The splitting planes of the BSP trees in these algorithms are given by the scene polygons and the shadow volume boundary, so these techniques are in fact hybrids of the shadow volume and projection approaches.

The projection approach can be applied to volumetric primitives by projecting semitransparent slices onto each other recursively [Behrens and Ratering 1998]. Projection techniques also exist for area light sources and soft shadows, in both exact object-precision [Dretakkis and Fiume 1994; Stewart and Ghali 1994] and approximate image-precision [Soler and Sillion 1998] forms.

1.3 Shadow Volumes

The classical shadow volume algorithm [Bergeron 1985; Bergeron 1986; Crow 1977] requires a representation of the scene as a polygonal database, and represents the boundary between illuminated and shadowed space with object-precision polygons. The intersection between the boundary of the shadow volume and the scene can be computed at image precision using per-pixel depth comparison and counting operations.

A *shadow volume boundary polygon*, or *shadow polygon* for short, is constructed for each edge of every object polygon in the scene. Each shadow polygon is a semi-infinite quadrilateral with two finite vertices corresponding to each pair of edge endpoints and two infinite vertices. The infinite vertices are placed at the limit of rays emanating from the light source and passing through each finite vertex. A clipper operating with homogeneous coordinates will reduce this semi-infinite quadrilateral to a finite size when it is clipped against the viewing frustum.

The shadow polygons together with their generating object polygon enclose a semi-infinite volume of space which is shadowed by that particular object polygon. The union of all the shadow volumes generated by all object polygons is the shadow volume for the scene.

Computing the union of the per-polygon shadow volumes can be done by counting "in" and "out" events along rays from the eye. Orientation of shadow polygons must be maintained so that the front faces of the shadow polygons point towards illuminated space. This is necessary so "in" events can be distinguished from "out" events. If the eye is in fully illuminated space, a ray from the eye to a point on a surface will pierce an equal number of front facing and back facing shadow polygons if and only if the surface point at the ray terminus is not inside the union of the shadow volumes, i.e. is illuminated.

In an important optimization, shadow polygons originating in non-silhouette shared object polygon edges can be removed, with the silhouette defined relative to each light source. A flatland example of a shadow volume after such editing is given in Figure 1A.

Shared-edge shadow polygons can be removed because the orientations of such polygons generated from shared edges between adjacent non-silhouette polygons will be opposed and will cancel. If both open and closed (non-manifold) objects can appear in the scene, weights must be added to the shadow volume boundary polygons. Nonmanifold, unshared edges should always generate shadow polygons with a weight of 1. Shadow polygons generated from shared silhouette edges should