VII.1

THE SHADOW DEPTH MAP REVISITED

Andrew Woo
Alias Research
Toronto, Ontario

Introduction

The shadow depth map has proven to be an effective shadow determination approach in image synthesis (Williams, 1978; Hourcade, 1985; Reeves *et al.*, 1987). Not only does it handle any class of geometric primitives with equal ease, it is also the only shadow approach that requires a storage complexity independent of the number of objects in the scene (Woo *et al.*, 1990): an advantage when it comes to complex scenes. However, the depth map is prone to aliasing problems, of which some have been improved (Hourcade, 1985; Reeves *et al.*, 1987) using some filtering techniques. In this Gem, we attempt to reexamine the Moiré Pattern aliasing problem, and offer a superior solution.

The Shadow Depth Map

The basic shadow depth map approach :is very simple. Perform a Z-buffer scan-conversion and visibility determination of the scene from the perspective of a light source instead of the eye. However, the visibility is simplified in that only the depth information is kept in 2-D grid or map, with no information about the visible objects at all. During rendering of each pixel, the point or region to be shaded is projected onto the appropriate cell(s) in the depth map. A *SampleShadow routine* is called to compare the depth value for each applicable depth map cell with that for the current intersection point. If the intersection point is further away from the light source than the distance stored in the depth map for that cell, then the point is in shadow; otherwise, it is not.

The Moiré Pattern Problem

In Fig. 1, when the closest surface to the left of the point marked (*) is to be shaded, it will improperly result in self-shadow because the depth map value is further away from the light source than the closest surface. Reeves *et al*. (1987) attempt to solve this problem by introducing a *bias* that is added to the depth value on the fly so the same surface will be in front of the depth map value. Note that if *bias* is too large to avoid the moiré pattern problems, then objects that should be shadowed might be fully lit. If the *bias* is not large enough, then improper self-shadowing occurs much more often—this usually results in the appearance of noisy surfaces or moiré patterns (noise from the randomness attached to the *bias* value)—see Fig. 2. Thus, no matter how carefully *bias* is chosen, the shadow results are usually unsatisfactory, and are especially noticeable for low depth map resolutions < 512 × 512.

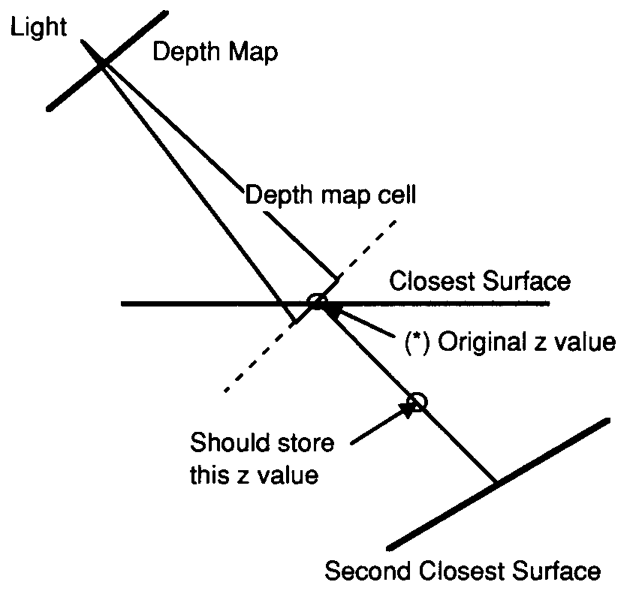

Figure 1.

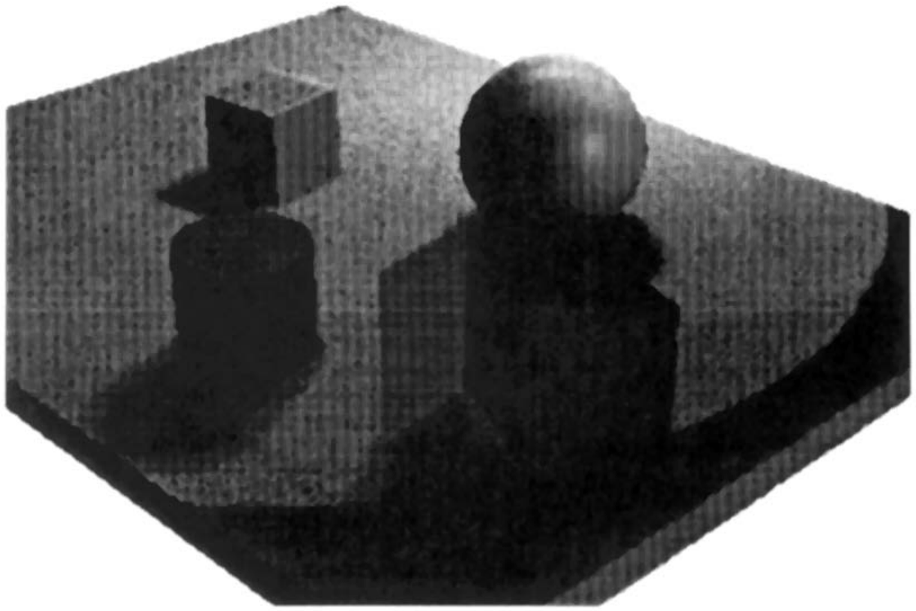

Figure 2. The old algorithm.

Another solution instead of *bias* is proposed here. If the depth map cell only encounters one surface, then the depth map value is set to the same value as if there were no surfaces in that cell, i.e., the *z* value is set to some large value. Such a surface will always be fully lit where it intersects that cell.

If there is more than one surface that hits the depth map cell, then the depth map cell value is assigned to be an average (halving the sum of the values is a good rule of thumb) between the first two visible surfaces with respect to the light source; see Fig. 1. In this way, the closest surface should always be lit and the second surface should always be in shadow. Thus, the *bias* value is not needed at all, and the results generated are far superior.

A Boundary Case

Special care is required at the outer limits of a spotlight's cone of illumination. When sampling occurs outside this cone, make sure that an *in shadow* result is returned. Otherwise, spurious brightly lit lines may appear at the juncture between lit and shadowed regions. Reeves *et al.* (1987) provided source code that had this error.

Some Optimizations

The changes proposed above not only produce better quality shadows, but should also be faster computationally. For example, there is no need to compute bias values on the fly for each shadow sample (in the *SampleShadow* routine). Furthermore, because halving the depth value between visible surfaces results in large value differences during depth comparisons, depth values can be compared in integer first, then done in floating-point only if the integer comparison is inconclusive. This tends to help out on platforms where integer evaluations are much faster than floating-point. Available is some pseudo-code showing Reeves *et al.'s* *SampleShadow* routine vs. the new approach.

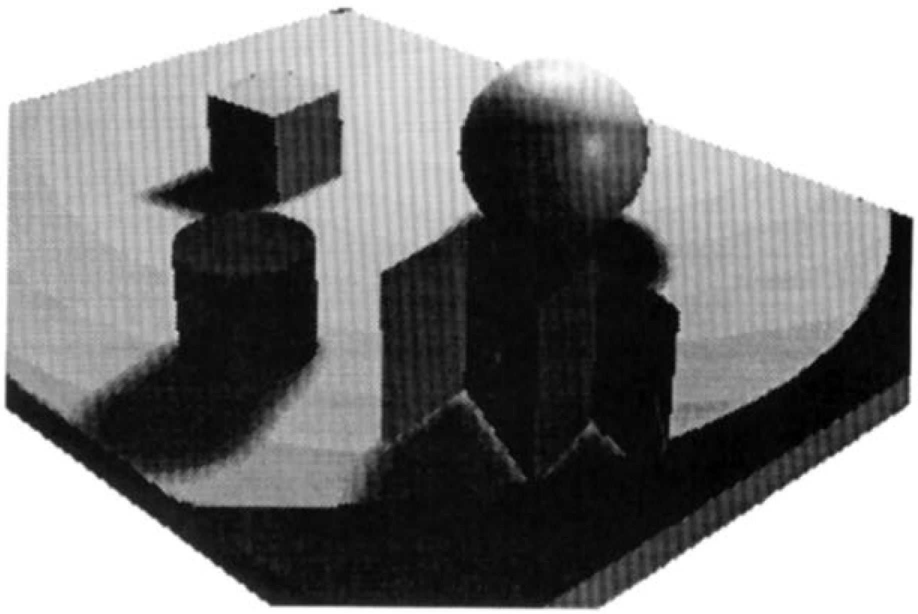

Figure 3. The new algorithm.

Conclusions

The changes made to the shadow depth map approach seem to work quite well, even at a low resolution of 256 × 256. This can be seen in Figs. 2 and 3, where a 256 × 256 depth map is used to render the scene: Notice the difference in the noise level of the surfaces. The spotlight frustum is 75 degrees, and the minimum and maximum bias values are 0.2 and 0.4, respectively. Thus, a 1,024 × 1,024 resolution depth map is not always necessary for realistic shadows. With this approach, a very reasonable job can be achieved at much lower resolutions and thus use up a lot less computation and memory.

See also G2, 273; G2, 275.