Disney Bonus Section

The following are two bonus sections that we thought it would be useful to have included. These are from research done by Brent Burley here at Walt Disney Feature Animation.

The first is an article on work Brent done on trying to get a "single map soft shadow" technique working. Along the way he came across this "bias cone" technique.

The second is more of an abstract as it is basically some pictures and math showing an issue with PRMan derivatives and how to correct for this issue.

Shadow Map Bias Cone and Improved Soft Shadows

Brent Burley, Walt Disney Feature Animation

Abstract. A new shadow map sampling method using a "bias cone" is described that has fewer artifacts than sampling using a constant bias and it is shown how to apply this sampling to the generation of perceptually accurate soft shadows from a single shadow map.

1. Introduction

The use of depth-map based shadows (i.e. "shadow maps") typically requires a bias, or offset, which is added to the depth of elements in the scene before being compared with the depth value in the shadow map in order to prevent self-shadowing [Williams 78]. If the bias is too large, light leaks can occur where shadow occlusions are missed, and if the bias is too small, self-shadowing will darken elements in the scene that aren't in shadow. Midpoint depth maps, which are produced by averaging the nearest two depth samples at each point in the depth render [Woo 92], can help significantly but are not immune to bias problems. Surfaces known to not cast shadows can also be excluded from the shadow pass and thus be free from self-shadowing, but this can't solve every case since there are certainly surfaces that cast and also receive shadows.

Filtering by averaging the result of multiple depth tests in the neighborhood of the surface point (known as "percentage-closer filtering") is often used to soften or blur shadow boundaries [Reeves et al. 87] and implementations typically use a constant bias. A large filter radius is sometimes used to simulate shadow penumbras but at the expense of increased shadow bias artifacts. The constant-size of the penumbras also reduces the sense of contact expected when an occluding object touches the occluded surface (a table leg on a floor, for example).

2. Bias Cone

Simply increasing the bias linearly as the filter samples get further away from the shading point can significantly reduce bias artifacts. This is essentially sampling the surface of a cone instead of a disk (See Image 75). The slope or steepness of the cone could be exposed as a user parameter, but our current implementation just uses a fixed value of 2.0; it's steep enough to avoid most self-shadowing artifacts, and surfaces steep enough to penetrate the cone are already significantly darkened from the diffuse falloff so additional self-shadowing is not very noticeable. The base bias (the value of the bias at the tip of the cone) should be controllable by the user, but can usually be zero or near zero and produce very little self-shadowing even with a large filter size.

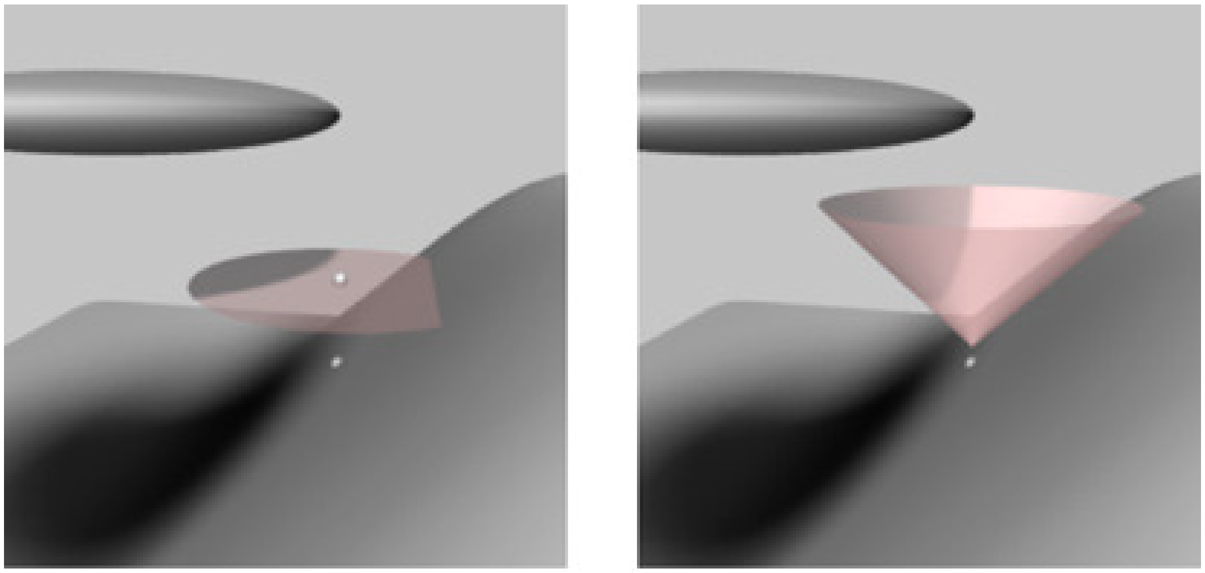
Image 75: Sampling a shadow around a shading point within the penumbra region. Left: even a large constant bias can result in self-shadowing (sampling disc penetrates surface). Right: sampling a cone instead of a disc avoids self-shadowing while permitting a much smaller base bias (the bias at the tip of the cone).

3. Improved Soft Shadows

A two-pass filter can be used to produce perceptually accurate soft shadows where the size of the penumbra varies based on the distance to the occluding object [Hoffman 2002]. Shadows can be very sharp at the point of contact and then soften gradually as the occluding object moves away giving the impression of the shadow produced by an area light. The first pass finds the average occlusion depth within the filter region. This value is then used to scale the filter size in the second pass, which is just the traditional percentage-closer filter. Fernando also developed this technique independently [Fernando 2005]. However, the method presented here has a number of advantages – the bias cone improves the accuracy at the contact point and reduces the overall bias artifacts, and the depth filter described below does a better job at detecting small, nearby occluding objects and has fewer artifacts in regions of overlapping occluding objects than the simple depth average presented by Fernando. An example is shown in Image 76.

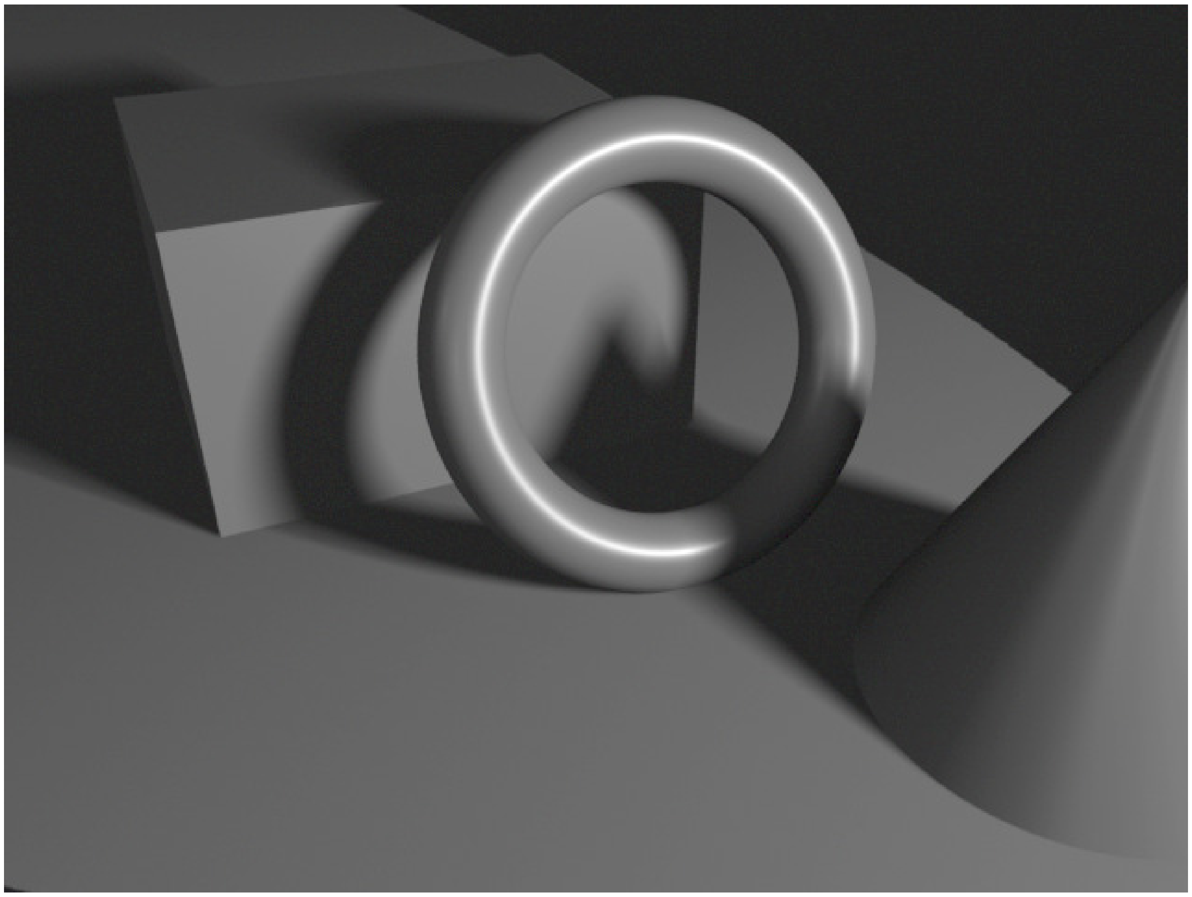
Image 76: Single map soft shadow – size of penumbrae vary with the distance to the occluding object. Shadows are very sharp at point of contact w/ no bias artifacts.

3.1 Depth Filtering Details

Simple depth averaging over uniformly spaced samples is not sufficient to produce good results. There are two issues in particular that must be considered. Uniform sampling over a large filter area will either require a prohibitive number of samples or will likely miss close contact points and small occluding objects. Simple mean value averaging of the depth values also will tend to miss close contact points as the occlusion distance will be pulled away by other occluding objects within the filter radius that are not as close.

A more robust way to produce the depth estimate is to concentrate samples near the center of the filter. This addresses both issues since close contact can only occur near the center of the filter and the extra samples will help ensure that the close contact point is seen, and the concentration of samples also gives objects near the center of the filter more weight. Close contact points can be further strengthened by weighting the samples as a function of the distance from the center of the filter. A simple linear falloff for the sample weights is sufficient, but any monotonic falloff function may be used.

3.2 Pseudocode

```
function sampleDepth(p, searchradius):
    // given a point p in shadow space and a search radius,
    // determine average occluder distance
    totalWeight = 0
    sz = 0
    for each sample location near p within search radius:
           z = depth value for sample from shadow map
           if z < sample.z:
           // compute sample weight based on linear falloff
           weight = 1 - 0.9 * (sample.radius / searchradius)
           sz += z * weight
           totalWeight += weight
    if totalWeight > 0:
             sz /= totalWeight
    result = sz

function sampleBlur(p, filterradius):
    // given a point p in shadow space and a filter radius,
    // perform percentage-closer filtering to blur the shadow
    occ = 0
    for each sample location near p within filter radius:
           z = depth value for sample from shadow map
           if z < sample.z:
             occ += 1
    result = occ / nsamples

function softShadow(p, minradius, maxradius):
    // given a point p in shadow space and a min and max filter radius,
    // blur the shadow with a radius based on the avg occlusion depth
    sz = sampleDepth(p, maxradius)
    if sz > 0:
           radius = minradius + (maxradius - minradius) * (1 - sz/p.z)
           result = sampleBlur(radius)
    else:
        result = 0
```

Notes:

1. The sample.z value is computed as p.z + baseBias + biasSlope * sample.radius. The addition of sample.radius to the base bias value produces sample points on the bias cone. The biasSlope value can be controllable or can just be set to 1.0.
2. The Hammersley Point Set [MathWorld] has been found to produce a good radial sampling pattern where the x coord is taken as the angle (normalized from 0 to 2 pi) and the y coord is taken as the radius.
3. For the sampleBlur function, the sample points should have uniform density in order to properly compute the percentage-closer filtering. The Hammersley Point Set is evenly spaced in x and y but is no longer evenly spaced when taken as polar coordinates. However, the uniform density of the points can be easily restored by setting the sample radius to sqrt(y).
4. For the sampleDepth function, taking the y coordinate of the Hammersley Point Set directly as the radius naturally concentrates samples near the center of the filter region and produces a good result.

3.3 Additional Implementation Features

In addition to varying the size of the penumbrae, the occlusion depth can also be used to vary the shadow opacity as shown in Image 77. Additional controls that may be useful are controls that allow shaping the shadow and opacity falloff. We've also had success with using this technique with deep shadow maps [Lokovic and Veach 2000] that enable motion blurred shadows among other things. Deep shadow maps store multiple depths at each pixel location. For the depth estimate, we use the depth nearest the light source as the representative depth for each pixel. Once the filter size is determined for the shading point, filtering proceeds in the usual way.

Image 77: Production still using single map soft shadow technique. In this image, the opacity of the shadow is decreased as a function of distance to add to the effect. © Disney

References

[Williams 78] Lance Williams. Casting curved shadows on curved surfaces. In Computer Graphics (SIGGRAPH '78 Proceedings), pages 270–274, August 1978.

[Woo 92] Andrew Woo. Graphics Gems III, The Shadow Depth Map Revisited, pages 338–342. Academic Press, 1992.

[Reeves et al. 87] William T. Reeves, David H. Salesin, and Robert L. Cook. Rendering antialiased shadows with depth maps. In Computer Graphics (SIGGRAPH '87 Proceedings), pages 283–291, July 1987.

[Lokovic and Veach 2000] Tom Lokovic, and Eric Veach. Deep Shadow Maps. In Computer Graphics (SIGGRAPH 2000 Proceedings), pages 385-392, July 2000.

[Hoffman 2002] Craig Hoffman, Walt Disney Feature Animation. In-house shader code.

[Fernando 2005] Randima Fernando. Percentage-Closer Soft Shadows. Sketch presented at SIGGRAPH

2005.

[MathWorld] Eric W. Weisstein. "Hammersley Point Set." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/HammersleyPointSet.html