Subtractive Shadows: A Flexible Framework for Shadow Level-of-Detail

Christopher DeCoro and Szymon Rusinkiewicz
Princeton University, Department of Computer Science, Technical Report TR-781-07

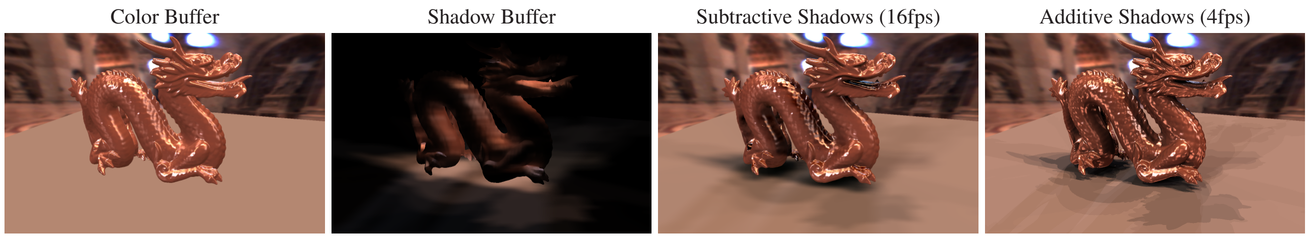

Figure 1: Color and Shadow Buffers. *Even with low lighting detail (25 lights), subtractive shadows preserve accuracy in the direct illumination along with soft shadowing. By contrast, additive shadowing produces inaccurate speckled effects on the dragon due to undersampling, while the shadows are unappealingly hard. In addition, by decreasing the shadow sampling detail to* $1/16$*, subtractive shadowing enables real-time frame rates at high resolution (1280x800).*

Abstract

We explore the implications of reversing the process of shadow computation for real-time applications that include complex lighting (such as that specified by an environment map). Instead of adding illumination contributions at each pixel across various lights, we compute the complete, unshadowed local illumination at each pixel using fast approximation algorithms, then subtract the lighting contribution from each light for which the pixel is in shadow. This provides flexible level-of-detail for shadow computation in ways that standard additive shadows do not, such as permitting the use of fast methods for accurate direct illumination combined with a small number of shadow-casting lights, and allowing for down-sampled shadow buffers to reduce fill cost. This technique preserves that portion of the scene with the greatest visual importance – the direct illumination – and allows shadows to be presented with lower fidelity in exchange for improvements in rendering time. With subtractive shadows, we can render complex scenes with arbitrary BRDFs, environment map illumination, and shadows, achieving in real time effects that previously required offline preprocessing.

1 Introduction

While rendering methods that compute the global illumination of a scene inherently determine which regions are in shadow, most real-time rendering is performed using rasterization methods that instead utilize local illumination calculations. These methods require additional techniques to simulate shadowing. The standard approach renders the scene illuminated by each light in turn, while limiting the effect to pixels visible from that light, which are identified as such by a *visibility determination* method (such as the well-known shadow volume [Crow 1977] and shadow mapping [Williams 1978] algorithms). The results from each light are accumulated to produce the final image, and so we will refer to this as "additive" shadowing. We explore, however, the effect of doing the opposite: computing the complete, unoccluded local illumination of the scene, then, for each light, subtracting from the scene the energy occluded by cast shadows.

Our technique is intended to address illumination from complex natural lighting environments. While illumination from small numbers of discrete point lights can be computed individually, real-world illumination is defined by a continuous function over the entire visible hemisphere, for which per-light methods are ineffective. To accurately render such lighting (commonly represented as tabulated spherical functions, or *environment maps*) researchers have developed several sets of techniques. *Environment sampling* reduces the continuous function to a set of important directional lights; while the result is simple to render, it cannot account for the complex continuous distribution of incoming light – especially noticeable for non-diffuse materials. *Fast local illumination* methods compute the complete local lighting in constant time, yet are ignorant of non-local geometry, and so are unable to represent cast shadows. *Precomputed radiance transfer* methods do consider geometry and visibility, but only at the cost of offline precomputation, and so are unable to support dynamic geometry or materials.

Instead, we present a technique designed to support lighting and shadowing from realistic environment maps without significant preprocessing. It does so by leveraging the strengths of fast local illumination methods for direct lighting, with environment sampling for "subtractive" shadowing. It is based on the observation that the direct illumination of the scene is of primary visual importance, and that the shadowing, while providing essential visual cues, is secondary. Therefore, our technique preserves direct illumination in full detail, yet allows the rendering system to perform a tradeoff between shadow quality and rendering speed – if a higher frame rate is needed, the rendering system lowers the level of detail present in the shadows until the target is reached. This is motivated by existing work on geometric simplification, in which a particular level-of-detail (LOD) can be selected that approximates the original geometry, yet is simpler to render – incurring rendering error for increase in speed. Analogously, in many cases, sacrificing accuracy of shadow computation in exchange for improved rendering speed is an acceptable trade-off, so long as the visual fidelity of the direct illumination is maintained. The technique of subtractive shadows does just that, as we will demonstrate.

2 Algorithm Description

We assume availability of algorithms for fast local illumination, shadow determination, and environment sampling (see Section 3). Our technique is as follows, and is illustrated in Figures 1 and 2.

1. Sample environment map to create an approximation E
2. For each (shadow-casting) light $L \in E$:
   (a) Render radiance cache, if used for fast local illumination
   (b) Determine shadows of scene with respect to L
   (c) For each *shadowed* pixel, compute contribution of L
   (d) Composite into shadow buffer, S, which may have lower resolution than color buffer
3. Render unshadowed direct illumination into color buffer, C, using fast local illumination algorithm
4. Resample S to produce S', equal in size to C
5. Subtract S' from C, producing final, shadowed, output

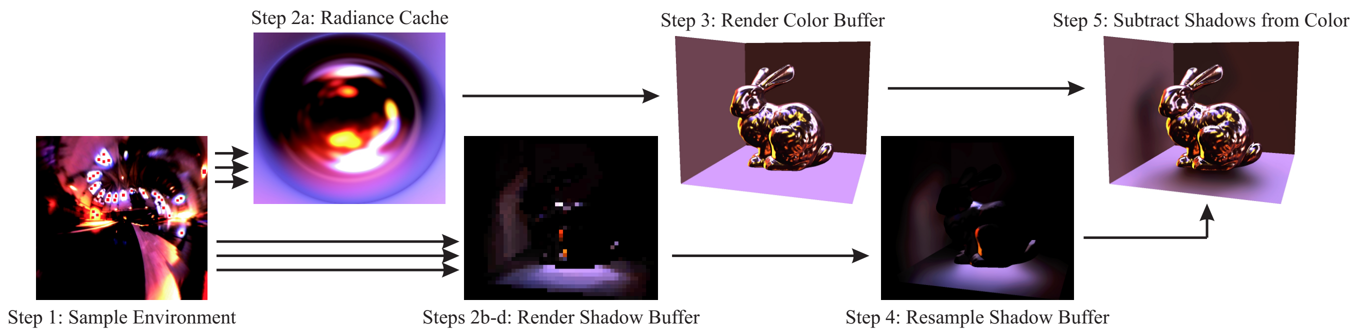
Step 2a: Radiance Cache
Step 3: Render Color Buffer
Step 5: Subtract Shadows from Color
Step 1: Sample Environment
Steps 2b-d: Render Shadow Buffer
Step 4: Resample Shadow Buffer

Figure 2: Subtractive Shadows Overview. *An illustration of the various buffers used in the subtractive shadows technique. Multiple arrows indicate that multiple passes are required. The creation of the radiance cache is optional, and can be replaced with another fast local illumination algorithm.*

The benefit of this procedure is that it allows for two time-quality tradeoffs, parameters that we refer to as the *illumination detail* and the *sampling detail*. A renderer can dynamically reduce both of the levels of detail as needed to reach a target frame rate.

Illumination Detail. The bottleneck is the loop over the shadow-casting lights L in Step 2. However, regardless of the size of L, the direct illumination C computed in Step 3 is maintained correctly. This is due to the use of the fast local illumination algorithms, such as the prefiltering approach of [Ramamoorthi and Hanrahan 2001] and [Heidrich and Seidel 1999], which will be of higher quality than achievable with sampling (for example, for highly peaked BRDFs) because it can represent the continuous illumination from the environment, which sampling cannot. Therefore, we can decouple the shadow-casting lights from the illumination used to compute reflectance, allowing us to use fewer shadow-casting lights as necessary to improve the frame rate. The resulting error is limited to shadow "hardening," and even this can be ameliorated by resampling (see next section). Fewer lights decreases both vertex transformation overhead and fill overhead for either class of shadow determination algorithms (shadow volumes or shadow maps). We refer to the number of lights in L as the *illumination detail*.

Sampling Detail. Another flexible level-of-detail can be achieved in Step 4. Observe that shadows (especially the soft shadows that result from continuous lighting environments) are low-frequency phenomena, compared to the potentially very high-frequency direct illumination (consider perfect specularity, for example). We therefore can reduce fill overhead with minor loss of quality by reducing the shadows' *sampling detail*: reducing the resolution of the shadow buffer S computed in Step 2. The fill cost required by S can be the dominant component of the total fill requirements of the application (especially for shadow volumes, which may rasterize large portions of the screen for each light); we allow this overhead to be decreased in exchange for minor quality degradation. S is then resampled in Step 4 to match the size of C.

We have seen that this resampling, in addition to improving fill rate, also provides a better quality shadow by performing an effective (although not physically correct) approximation of soft shadows through blurring S, allowing fewer shadow-casting lights to be used for comparable quality. Also, as the low-frequency shadowing term is distinct from the high-frequency direct illumination, we are able to filter S only once, after all lights have been composited, and to do so without the loss of high-frequency detail in the direct illumination that would result from blurring the final rendered result of additive shadows. The analogous technique for additive shadowing would instead require the per-light visibility to be blurred separately for *each* light, which is then modulated with the unoccluded light to produce the shadowed result.

The resampling must respect normal and depth variation. Our resampling filter uses full-resolution positions and normals; for each output pixel in S' we identify the corresponding neighboring pixels in the downsampled shadow buffer S and penalize differences in position and normal, using the formula Gaussian$(||S'_p - S_p||)(S'_n \cdot S_n)$, where p and n represent position and normal in their corresponding buffers. This simple approach produces smooth results from low-resolution samples (see Figure 2). We have shown the range of effects achievable by resampling shadows, both physically plausible and stylized, in a previous work [DeCoro et al. 2007], to which we refer the reader for additional analysis and details.

3 Examples

Implementation details. Our technique is not specific to any particular algorithms for fast local illumination, visibility determination, or environment sampling. However, for the examples shown here, we chose to use stenciled shadow volumes [Heidmann 1991] for visibility, spherical harmonic irradiance maps [Ramamoorthi and Hanrahan 2001] for fast local diffuse lighting, prefiltered environment maps [Heidrich and Seidel 1999] for fast local specular lighting, and structured importance sampling [Agarwal et al. 2003] of the environment map. More generally, for a fixed viewpoint, an arbitrary BRDF with distant lighting can be tabulated per-frame and cached as a texture, known as a radiance cache, and this texture can be used to perform fast local illumination [Miller and Hoffman 1984]. We refer the reader to those papers for additional detail. Finally, we also use deferred shading [Molnar et al. 1992; Hargreaves 2004], which renders geometry once per frame, and uses image-space rendering passes to composite additional lights. All examples are rendered at 1280x800 using 32-bit floating point buffers, on a 3GHz Intel Core2 CPU with GeForce 8800 GPU. The bunny model (35K vertices), triceratops (5K vertices) and horse (50K vertices) in Figure 5 are shown lit with the Grace Cathedral, St. Peter's Basilica, and Eucalyptus grove datasets, respectively.

Variable shadow-casting lights. As seen in Figures 5 and 4, decreasing the number of lights (the illumination detail) has smaller visual impact for subtractive shadows (shadows become "harder") than for additive shadows (direct illumination is also affected). As a result, the subtractive algorithm has a higher quality at a given frame rate. Further, the decrease in quality is limited to the shadows. The figures also demonstrate an important conclusion about our method, in that it shows the largest improvements in quality for BRDFs with large peaks, such as high specularity. This is logical, as the sampled representations are derived from the environment map, not the BRDF, so we would expect these to perform best on diffuse surfaces, in which environment map variation is most significant.

Resampled shadow buffer. By rendering S at a lower resolution than C and resampling, we achieve an increase in speed, as shown in the right column of Figure 5. Additionally, smoothing the shadows provides a better approximation of the shadows in the reference image than additive shadowing, even if not physically correct (see also Figures 1 and 3). The $1/16$ sampling detail image is qualitatively comparable to the reference, and while no longer accurate,

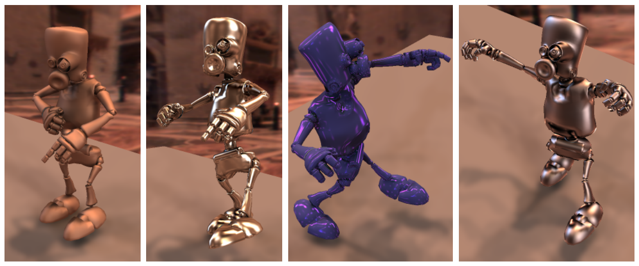

*Figure 3: **Dynamic Scene and BRDF:** We show stills from a scene with dynamic geometry, camera, and reflectance, captured at 30-40 FPS, with 50 lights and 1/16 shadow samples. We are able to edit BRDF parameters in real time with shadowing, while maintaining high quality rendering, by using subtractive shadowing with radiance caching for local illumination. The figures show our animated robot (8500 vertices) with a moderately glossy Phong material, a highly specular Phong material, a Lafortune fit of measured metallic blue paint, and a Torrance-Sparrow BRDF. A similar animated scene is shown in the video on the accompanying website.*

the $1/1024$ sampling detail image provides a plausible soft shadow, with only 1000 pixel samples of S for a 1280x800 rendering. This process reduces or eliminates the fill bottleneck typically associated with shadow volumes (note that the frame rate does not change from $1/16$ to $1/1024$, indicating that below $1/16$ detail in this example, fill rate is not a limiting factor).

Dynamic Scenes. We show in Figure 3 several frames (taken from the accompanying video) from a scene with a moving camera, non-rigid deformations and complex dynamic BRDFs. Because our method requires a minimal amount of precomputation per frame (shadow volume determination and radiance caching), no more than is commonly performed in interactive applications such as video games, we are able to render such scenes at real-time rates. This example uses a 128^2 radiance cache, which we have found acceptable for most situations, and renders at 30-40 FPS. The use of a radiance cache causes a negligible overhead; by itself the cache in this example renders at over 500 FPS.

4 Discussion

Our technique for rendering shadows under complex lighting environment readily invites comparison to the class of precomputed radiance transfer algorithms [Sloan et al. 2002], which provide a similar functionality. These methods precompute a transfer function mapping incoming to outgoing radiance for known geometry and reflectance. While these generalize directly to a much wider range of indirect illumination effects, such as interreflection and subsurface scattering, their inherent precomputation limits their use in many applications. For example, this technique was developed in the context of an interactive material editing system; changes in reflectance and their effect on the (dynamic) scene are necessarily required to be visualized immediately. We demonstrate dynamic materials and animated geometry in the accompanying video, from which stills are shown in Figure 3; these effects would not be possible with precomputed radiance transfer.

Certain limitations of the implementation that we have chosen to demonstrate the subtractive shadows concept may restrict its use in a production context. A key example is the lack of physically-correct soft shadowing. Additionally, our system as implemented focuses specifically on environment map illumination, though local lighting can be directly integrated into our renderer by additively compositing the local lights along with the direct environment map illumination in Step 4. However, our goal was to focus the comparisons between additive and subtractive systems of equivalent implementation complexity, and so this additional functionality, while clearly important for many applications, was not implemented in this demonstration system. We anticipate that the gains of the system we demonstrate would also apply to more real-world systems. Also, our system shows less significant improvement on scenes with mostly diffuse materials. While there exist methods to suitably render diffuse scenes and discrete local lights, a main goal of ours was to demonstrate how to incorporate more complex materials and natural environment lighting in a shadowing-aware real-time system, without the sort of precomputation complexity – and therefore limits on the dynamic nature of the scene – required by precomputed radiance transfer.

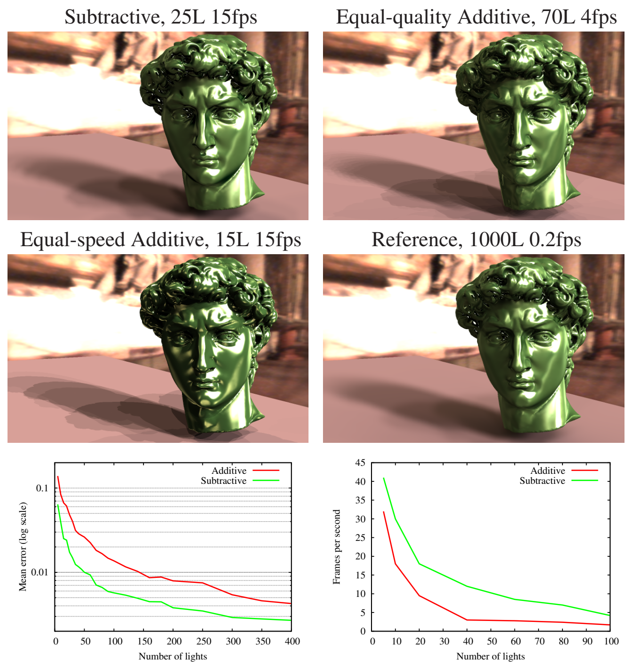

Figure 4: We compare the result of our subtractive shadows method with 25 lights and 1/4 sampling detail (top-left) to a high-quality reference rendering computed using 1000 lights (bottom-right). For roughly equivalent quality additive shadows (top-right), our method is significantly faster; while maintaining significantly higher quality compared to an equivalent speed additive rendering (lower-left). The major artifacts of our method, compared to the reference, are in the shadowed regions. We consider shadows cast by all objects in the scene, note for example the shadowed regions of the hair. However, the more noticeable direct illumination is well-preserved, which is not the case for the 15-light additive example. As shown in the graphs, subtractive shadows exhibits consistently superior quality and frame rate across varying levels of lighting detail. The model consists of approximately 125K vertices, rendered at 1280x800 in Galileo's Tomb.

As the results show, the subtractive shadows technique allows for simple yet flexible variable level of detail for shadow rendering. The technique generalizes to surfaces of arbitrary reflectance, and allows the developer to achieve high-quality natural illumination, while preserving the ability to render shadows at arbitrary speed with easily adjustable parameters. Through this technique, we give the lighting designer the same flexibility that geometric LOD algorithms have provided to modelers for years.

References

Agarwal, S., Ramamoorthi, R., Belongie, S., and Jensen, H. 2003. Structured importance sampling of environment maps. *ACM Transactions on Graphics 22, 2003*.

Crow, F. C. 1977. Shadow algorithms for computer graphics. In *(Computer Graphics) Proceedings of SIGGRAPH*.

DeCoro, C., Cole, F., Finkelstein, A., and Rusinkiewicz, S. 2007. Stylized shadows. In *International Symposium on Non-Photorealistic Animation and Rendering (NPAR)*.

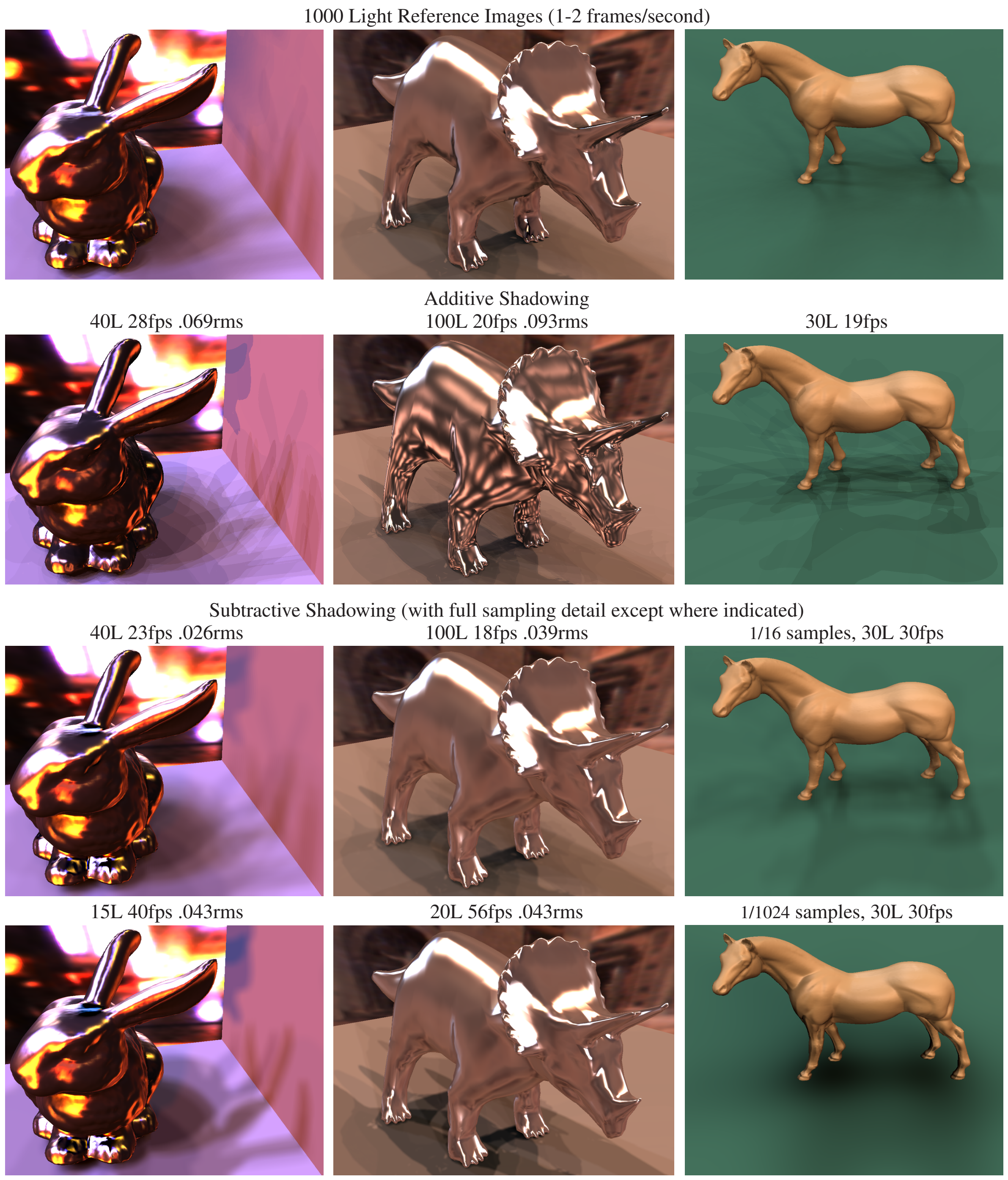

Figure 5: Varying Parameters. *Compared to a reference, the bunny (shininess 40) shows degradation in rendering quality for the additive algorithm as the number of lights decreases (note the area below the eye, and the root-mean-squared error relative to the reference). This is more apparent with the shinier (n=500) Triceratops. Subtractive shadows using prefiltered environment maps have higher quality and roughly equal speed at equal lighting detail, and maintain quality even at low level-of-detail; the only artifact is the hardening of the shadows. Further, as shown in the right column, by reducing sampling detail we can reduce the fill cost typically associated with shadowing, while continuing to render plausible soft shadows.*

HARGREAVES, S. 2004. Deferred shading. *GDC 2004*.

HEIDMANN, T. 1991. Real shadows, Real time. *Iris Universe, No. 18*.

HEIDRICH, W., AND SEIDEL, H.-P. 1999. Realistic, hardware-accelerated shading and lighting. In *Proceedings of SIGGRAPH*.

MILLER, G. S., AND HOFFMAN, C. R. 1984. Illumination and reflection maps: Simulated objects in simulated and real environments. In *Course Notes for Advanced Computer Graphics Animation, SIGGRAPH*.

MOLNAR, S., EYLES, J., AND POULTON, J. 1992. PixelFlow: High-speed rendering using image composition. *Computer Graphics 26, 1992*, 2.

RAMAMOORTHI, R., AND HANRAHAN, P. 2001. An efficient representation for irradiance environment maps. In *Proceedings of SIGGRAPH*.

SLOAN, P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic low-frequency lighting environments. *ACM Transactions on Graphics 21, 2002*.

WILLIAMS, L. 1978. Casting curved shadows on curved surfaces. In *(Computer Graphics) Proceedings of SIGGRAPH*.