Vol. 13, No. 1: 45–56

Subtractive Shadows: A Flexible Framework for Shadow Level of Detail

Christopher DeCoro and Szymon Rusinkiewicz
Princeton University

Abstract. We explore the implications of reversing the process of shadow computation for real-time applications that model complex reflectance and 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 approximations, 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 downsampled shadows 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 speed. With subtractive shadows, we are able to interactively manipulate and render arbitrary BRDFs and environment maps applied to complex, dynamic scenes with shadows, achieving in real time effects that previously required offline computation or preprocessing.

1. Introduction

While rendering methods that compute the global illumination of a scene inherently determine which regions are in shadow, most interactive rendering is performed using rasterization methods that instead utilize local illumination

1086-7651/07 $0.50 per page

calculations. These methods do not inherently consider occlusion, so additional techniques have been developed 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 77] and shadow-mapping [Williams 78] 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, 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 complex illumination from natural lighting environments and realistic reflectance models, applied in an interactive setting that allows animation and editing. While illumination from small numbers of discrete point lights can be computed individually, real-world illumination and reflectance are defined by continuous functions over the entire visible hemisphere, for which per-light methods are ineffective. To accurately render such lighting environments (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 nondiffuse materials. *Fast local illumination* methods allow the complete local lighting to be computed in constant time, yet they are ignorant of nonlocal geometry and so are unable to represent cast shadows. *Precomputed radiance transfer* methods do consider geometry and visibility, but they do so only at the cost of significant 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 trade-off between shadow quality and speed—if a higher frame rate is needed, an interactive system can lower the level of detail present in the shadows until the target is reached. This is inspired by existing work on geometric simplification, in which a particular level of detail (LOD) can be selected that approximates the original geometry yet is faster 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 the availability of algorithms for fast local illumination, shadow determination, and environment map sampling (we will discuss this further when presenting examples in Section 3). Our technique is as follows and is illustrated graphically 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.

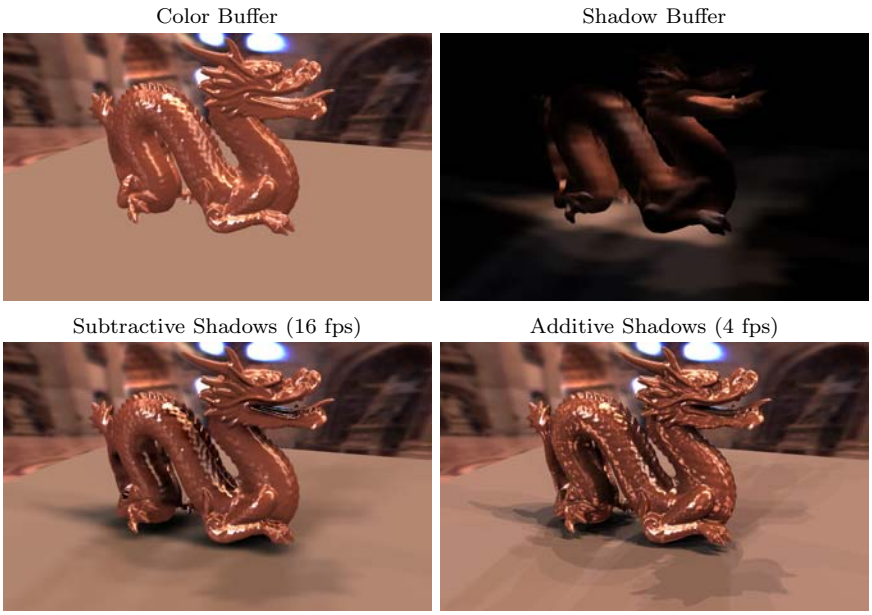

Figure 1. Color and shadow buffers. Even with low lighting detail (25 lights), subtractive shadows preserve high accuracy in the direct illumination along with soft shadowing. By contrast, additive shadowing produces an inaccurate speckled effect on the dragon due to undersampling, while the shadows are unpleasantly hard. In addition, by decreasing the shadow sampling detail to 1/16, subtractive shadowing preserves real-time frame rates for this high-resolution (1280×800) rendering.

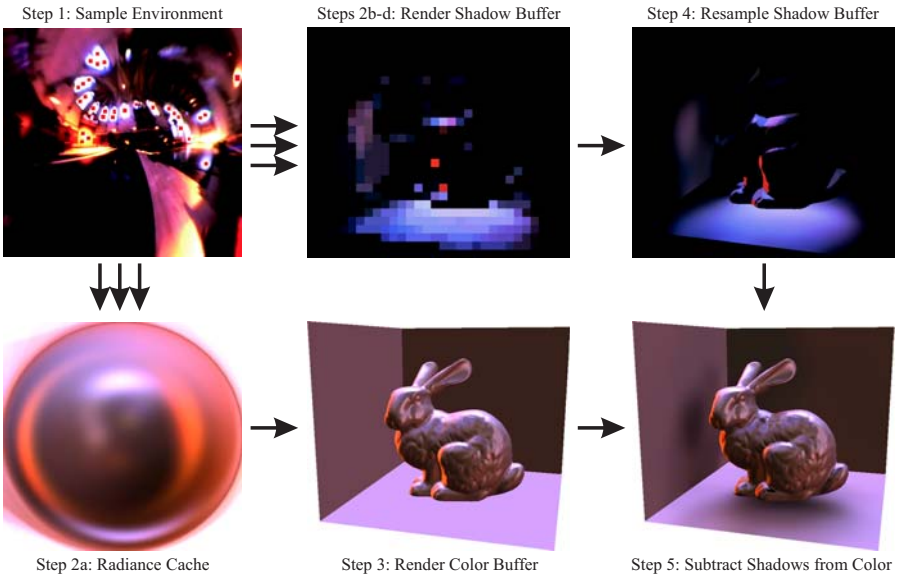

Figure 2. Subtractive shadows overview. An illustration of the various buffers used in the subtractive shadows technique. Multiple arrows indicate that multiple passes are required. The creation of the radiance cache is optional and can be replaced with another fast local illumination algorithm.

   (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.

2.1. Level-of-Detail Parameters

The benefit of this procedure is that it allows for two time-quality trade-offs, 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.

2.2. Illumination Detail

The bottleneck in this algorithm is the loop over 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 due to the use of fast local illumination algorithms, such as prefiltered environment maps [Heidrich and Seidel 99, Ramamoorthi and Hanrahan 01]. These display higher quality than that achievable by sampling (for example, for highly peaked BRDFs) as they represent continuous illumination from the environment, which sampling cannot. Therefore, we decouple shadow-casting lights from the illumination used to compute reflectance, allowing us to use fewer shadow-casting lights as necessary to improve frame rate. The resulting error is limited to shadow "hardening," and even this can be ameliorated by resampling (see Section 3.3). Using fewer lights decreases both vertex and fill overhead for either class of shadow algorithms (shadow volumes or shadow maps). We refer to the number of lights in L as the *illumination detail*.

2.3. 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 the full-resolution positions and normals; for a given output pixel in S', we identify corresponding neighboring pixels in the downsampled shadow buffer S and penalize differences in position and normal, using the formula $\text{Gaussian}(||S'_p - S_p||)(S'_n \cdot S_n)$, 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. 07], to which we refer for additional analysis.

3. Examples

3.1. 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 91] for visibility, spherical harmonic irradiance maps [Ramamoorthi and Hanrahan 01] for fast local diffuse lighting, prefiltered environment maps [Heidrich and Seidel 99] for fast local specular lighting, and structured importance sampling [Agarwal et al. 03] 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*, which can be used to perform fast local illumination [Miller and Hoffman 84]. We refer the reader to those papers for additional detail. Finally, we also use the technique of deferred shading [Molnar et al. 92, Hargreaves 04], which renders geometry once per frame and uses image-space rendering passes to composite additional lights. All examples are rendered at 1280×800 using 32-bit floating-point buffers, on a 3 GHz Intel Core2 CPU with a GeForce 8800 GPU. The bunny model (35K vertices), triceratops (5K vertices), and horse (50K vertices) in Figure 3 are shown lit with the Grace Cathedral, St. Peter's Basilica, and eucalyptus grove datasets, respectively.

3.2. Variable Shadow-Casting Lights

As we see in Figure 3 (left and middle columns) and Figure 5, decreasing the number of lights (the illumination detail) has a smaller visual impact for subtractive shadows (the 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

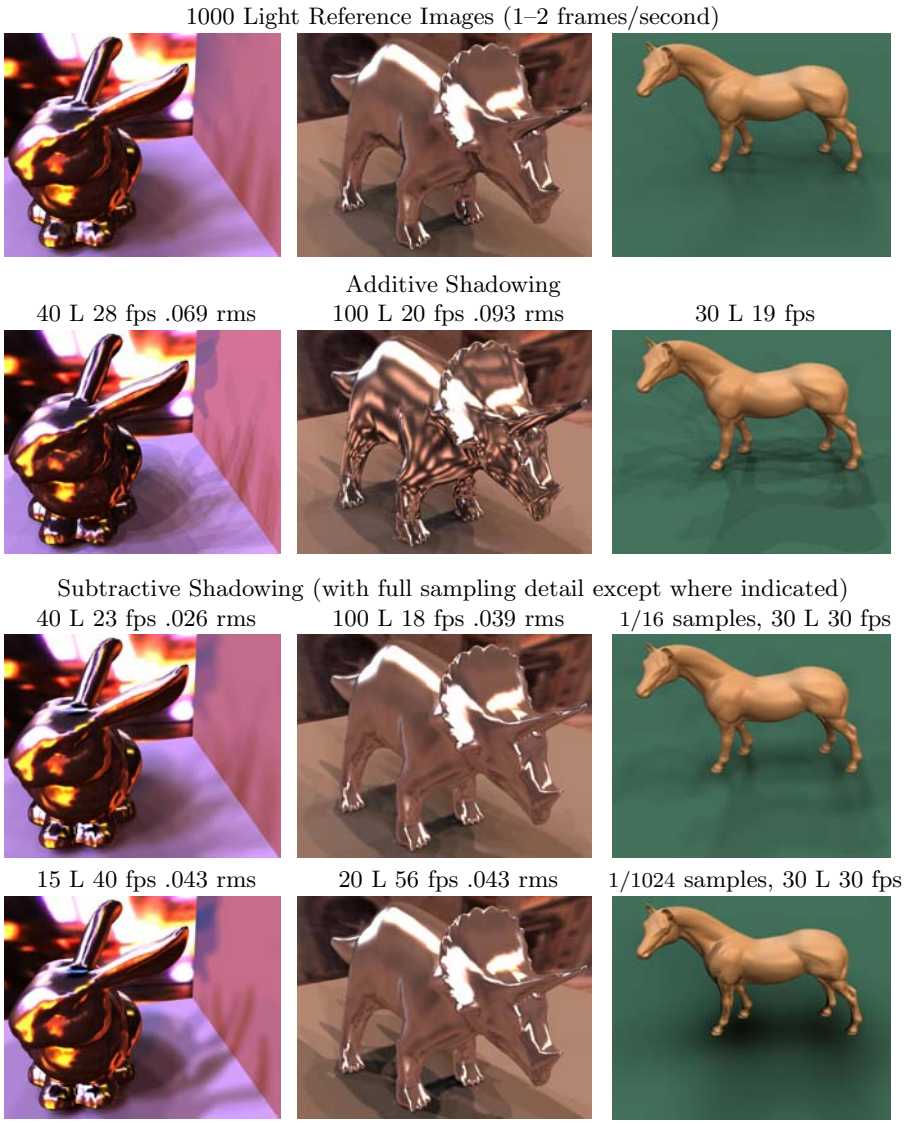

Figure 3. Varying parameters. Compared to a reference, the bunny (shininess 40) shows degradation in rendering quality for the additive algorithm as the number of lights decreases (note the area below the eye, and the root-mean-squared error relative to the reference). This is more apparent with the specular (shininess 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 reduce the fill cost typically associated with shadowing, while continuing to render plausible soft shadows.

quality is limited to the shadows. The figures also demonstrate an important conclusion about our method, which is that it shows the largest improvements in quality for BRDFs with large peaks, such as high specularity. This is logical, as sampled representations are derived from the environment, not the BRDF, so we would expect these to perform best on diffuse surfaces, in which environment variation is most significant.

3.3. Resampled Shadow Buffer

By rendering the shadow buffer S at a lower resolution than C and resampling, we can achieve a increase in frame rate, as in Figure 3 (right column). Additionally, through smoothing we can achieve a better approximation of the shadows in the reference than additive shadowing, even if not physically correct (see also Figures 1 and 4). The $4\times$ downsampled (1/16 sampling detail) image is qualitatively comparable to the reference image, and while no longer accurate, the $32\times$ downsampled (1/1024 sampling detail) image provides plausible soft shadows, with only 1000 pixel samples of S for a 1280×800 rendering. This process reduces or eliminates the fill bottleneck typically associated with shadowing algorithms, in particular with shadow volumes (note that the frame rate does not change from $4\times$ to $32\times$, indicating that above 1/16 detail in this example, fill rate is not a limiting factor). This is especially notable considering that fill overhead tends to be the bottleneck in many real-time applications (which generally use smaller models than those presented here).

3.4. Dynamic and Complex Scenes

We show in Figure 4 (left, middle) several frames from a scene with a moving camera, nonrigid 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 many interactive applications, 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. In Figure 4 (right) we demonstrate generalization of our method to complex shadow casters and (self-)shadowing receivers, in which a prefiltered environment is necessary to preserve the subtle lighting from atmospheric scattering (sampling would avoid the blue tint from the sky, instead concentrating around the sun).

Figure 4. Dynamic scene and BRDF; complex geometry. We show (left, middle) stills from a scene with dynamic geometry, camera, and reflectance, captured at 30–40 FPS, with 50 lights and 1/16 shadow samples for an 8500-vertex model. We enable editing of BRDF parameters (shown for glossy Phong and Torrance–Sparrow BRDFs) in real time with shadowing, while maintaining high quality through the use of subtractive shadowing with radiance caching for local illumination. The landscape model (66K vertices, 30 lights) demonstrates preservation of complex lighting by using a prefiltered diffuse environment (note the blue tint from atmospheric scattering, which would not be maintained with sampled lighting alone), while retaining important shadowing cues applied to a highly nonplanar shadow caster and receiver.

4. Discussion

Our technique for rendering shadows under complex lighting readily invites comparison to the class of precomputed radiance transfer algorithms [Sloan et al. 02], which provide a similar functionality. These methods precompute a transfer function that maps 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, our 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 support the dynamic materials and animated geometry shown in the video stills in Figure 4; 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

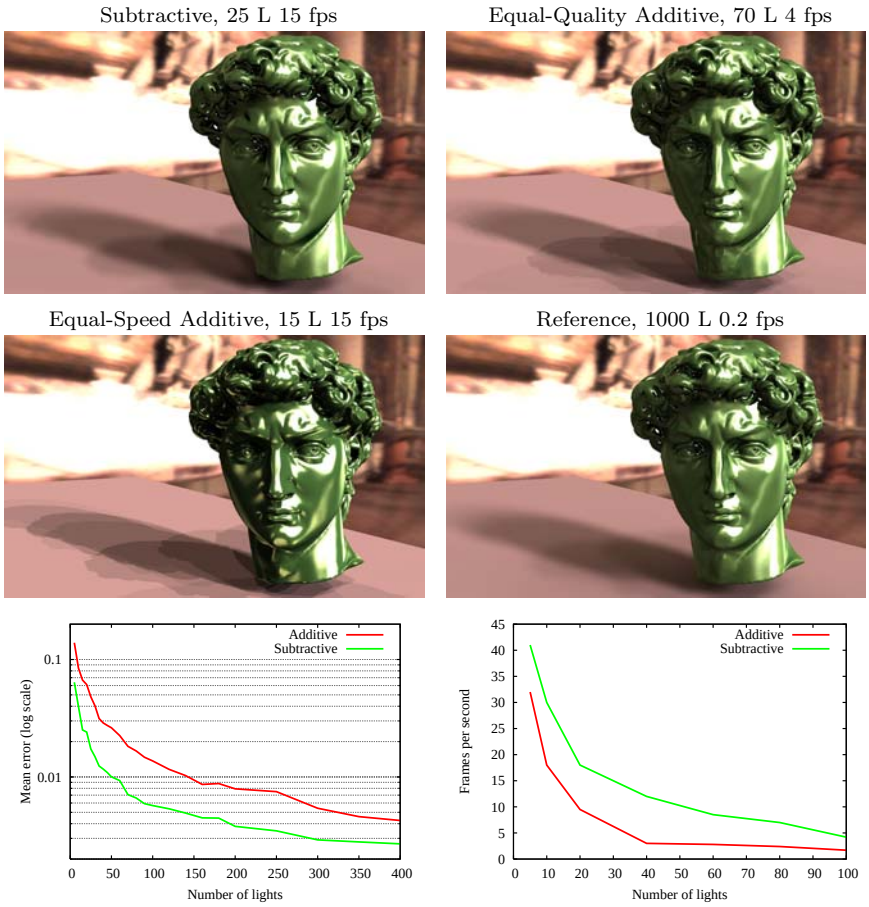

Figure 5. We compare the result of our subtractive shadows method with 25 lights and 1/4 sampling detail (top left) to a high-quality reference rendering computed using 1000 lights (bottom right). For roughly equivalent quality additive shadows (top right), our method is significantly faster, while maintaining 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 1280×800 with the Galileo's Tomb lightprobe.

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, although notable gains can be achieved when using diffuse material illuminated in highly nonlocalized environments, such as the sky in Figure 4 (right). 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.

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 et al. 03] Sameer Agarwal, Ravi Ramamoorthi, Serge Belongie, and Henrik Jensen. "Structured Importance Sampling of Environment Maps." *ACM Transactions on Graphics* 22:3 (2003), 605–612.

[Crow 77] Franklin C. Crow. "Shadow Algorithms for Computer Graphics." *Proc. SIGGRAPH '77, Computer Graphics* 11:2 (1977), 242–248.

[DeCoro et al. 07] Christopher DeCoro, Forrester Cole, Adam Finkelstein, and Szymon Rusinkiewicz. "Stylized Shadows." In *Proceedings of the 5th International Symposium on Non-Photorealistic Animation and Rendering*, pp. 77–83. New York: ACM Press, 2007.

[Hargreaves 04] Shaun Hargreaves. "Deferred Shading." Presentation, *GDC 2004*, 2004.

[Heidmann 91] Tim Heidmann. "Real Shadows, Real Time." *Iris Universe* 18 (1991), 28–31.

[Heidrich and Seidel 99] Wolfgang Heidrich and Hans-Peter Seidel. "Realistic, Hardware-Accelerated Shading and Lighting." In *Proceedings of SIGGRAPH 99, Computer Graphics Proceedings, Annual Conference Series*, edited by Alyn Rockwood, pp. 171–178. Readings, MA: Addison Wesley Longman, 1999.

[Miller and Hoffman 84] Gene S. Miller and C. Robert Hoffman. "Illumination and Reflection Maps: Simulated Objects in Simulated and Real Environments." In

Course Notes for Advanced Computer Graphics Animation, SIGGRAPH, New York: ACM Press, 1984.

[Molnar et al. 92] Steven Molnar, John Eyles, and John Poulton. "PixelFlow: High-Speed Rendering Using Image Composition." *Proc. SIGGRAPH '92, Computer Graphics* 26:2 (1992), 231–240.

[Ramamoorthi and Hanrahan 01] Ravi Ramamoorthi and Pat Hanrahan. "An Efficient Representation for Irradiance Environment Maps." In *Proceedings of SIGGRAPH 2001, Computer Graphics Proceedings, Annual Conference Series*, edited by E. Fiume, pp. 497–500. Reading, MA: Addison-Wesley, 2001.

[Sloan et al. 02] Peter-Pike Sloan, Jan Kautz, and John Snyder. "Precomputed Radiance Transfer for Real-Time Rendering in Dynamic Low-Frequency Lighting Environments." *Proc. SIGGRAPH '02, Transactions on Graphics* 21:3 (2002), 527–536.

[Williams 78] Lance Williams. "Casting Curved Shadows on Curved Surfaces." *Proc. SIGGRAPH '78, Computer Graphics* 12:3 (1978), 270–274.

Web Information:

http://jgt.akpeters.com/papers/DeCoroRusinkiewicz08/

Christopher DeCoro, Princeton University, 35 Olden Street, Princeton, NJ 08540 (cdecoro@cs.princeton.edu)

Szymon Rusinkiewicz, Princeton University, 35 Olden Street, Princeton, NJ 08540 (smr@cs.princeton.edu)

Received December 12, 2006; accepted in revised form March 6, 2008.