Eurographics Symposium on Rendering (2005), pp. 1–6
Kavita Bala, Philip Dutré (Editors)

Pre-processing Environment Maps for Dynamic Hardware Shadows

Adam Arbree, Bruce Walter and Kavita Bala

From left to right, the original environment map, the approximate area lights, our results, and a reference image.

Abstract

Environment maps are a popular method of reproducing complex natural lighting. However, current methods for hardware environment map shadows depend on significant pre-computation and cannot support dynamic objects. This work presents a pre-process that decomposes an environment map into two components: a set of area lights and an ambient map. Once the map is split into these components, each is rendered with an appropriate mechanism. The area lights are rendered using an existing hardware-accelerated soft-shadow algorithm; for our implementation we use penumbra wedges [AMA02]. The ambient region is rendered using pre-integrated irradiance mapping. Using an NVidia 6800 on a standard desktop, we demonstrate high-quality environment map shadows for dynamic scenes at interactive rates.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Introduction

High dynamic range (HDR) environment maps and image-based lighting have quickly become valuable, if not essential, rendering tools. Fast and simple to acquire, environment maps can add rich, complex, natural lighting to synthetic scenes. However, computing the shading, and especially the shadows, from environment maps is computationally expensive. Our goal is to enable HDR environment map lighting with shadows in interactive applications with fully dynamic geometry on modern graphics hardware. In this paper, we show how to approximate environment maps as a set of area lights plus a low dynamic range remainder. These components can then be rendered interactively using existing techniques.

Previous algorithms for environment map lighting generally fall into two categories, point light sampling and pre-computed transfer methods. However, neither is well-suited for interactive applications with dynamic objects. Point sampling produces objectionable discontinuity artifacts unless hundreds of samples are used. Precomputed transfer methods either require static scenes or severely limit the allowed degrees of freedom. However there has been considerable recent progress in methods for interactively computing approximate soft shadows from area lights in fully dynamic scenes. Our work leverages these soft-shadow advances via a novel decomposition of environment maps. This approach could have significant impact on performance critical applications, such as games, where interactivity and visual quality are more important than strict accuracy.

We can roughly divide a map into relatively small regions that are much brighter than their surroundings and cast distinct shadows and relatively large regions which cast diffuse shadows. In many maps, especially for indoor scenes, the effects from the smaller regions dominate the shading and shadowing. Our method automatically finds these regions and extracts them as area lights. We then use the penumbra wedges technique of [AMA02] to quickly render their shadows. Due to the lack of good existing solutions for dynamic shadows from extremely large area lights, the rest of the environment map is rendered without shadowing. As demonstrated by our results, this approximation works very well for environment maps where shadows from the smaller regions dominate.

In the next section we describe related work and describe our environment map decomposition strategy in Section 3. Results are presented in Section 4 and discussed in Section 5.

2. Related Work

2.1. Soft-Shadows

A detailed overview of hardware soft-shadow methods is beyond the scope of this work; refer to Hasenfratz, et. al. [HLHS03] for a survey of recent techniques. We now briefly review the most relevant methods.

Heckbert and Herf [HH97] create soft shadows by sampling the light source and blending the hard shadows created for each sampled point light. While this technique can produce very accurate shadows, many samples are required for accurate shadows from large area lights. Recent techniques that intelligently point sample environment maps [ARBJ03, KK03, ODJ04] can be used with Heckbert and Herf's soft shadow algorithm to produce accurate environment lighting. However, the number of point samples required to approximate most maps is typically unsuitable for interactive applications.

Several researchers have suggested fast approximations to compute soft shadows for area lights [SS98, ARHM00]. Here we briefly describe papers that approximate soft shadows on GPUs. These techniques can be categorized based on whether their underlying implementation builds on shadow maps [Hai01, WH03, CD03], or shadow volumes [AAM03].

The shadow map based algorithms [Hai01, WH03, CD03] compute the umbra using a shadow map from the center of the area light. They differ in the techniques used to produce the gradient that approximates the penumbra. All these methods suffer from a common drawback; they create unnaturally large umbrae since they only project a penumbra outwards from a hard shadow edge. Given the size of the approximate area lights present in environment maps, this over-stated umbra could result in significantly incorrect (dark) images. For this reason, we did not use these approaches to approximate shadows for our area lights.

We choose to use penumbra wedges, [AMA02, AAM03, ADMAM03], since they are fast, do not over-state the umbra, and their performance is likely to scale well with future hardware improvements. The algorithm extrudes wedges from silhouette edges to conservatively enclose the entire penumbra volume. Using a shadow volume approach [Cro77], they determine points within the penumbra and calculate the fractional coverage of the light at each point by projecting the silhouette edges onto the light source.

2.2. Pre-computed Radiance Transfer (PRT)

Several recent papers have discussed using pre-computed, radiance transfer functions for real-time lighting and shadowing from environment maps, [KSS02, SKS02, RH02, SHHS03, NRH04, NRH03]. By expanding both the transfer function and the map's lighting function over some basis (either spherical harmonics or wavelets), the lighting calculation per point is reduced to the dot product of two coefficient vectors. After pre-computation, these methods trivially map to hardware and real-time frame rates can be achieved. However, their considerable geometric pre-computation either prevents dynamic objects or permits only small bounded deformation after additional pre-computation [JF03]. [KLA04] supports dynamics with PRT by computing visibility maps on the fly using coarse (lower level-of-detail) object meshes. However since these maps must be represented only by a few low order spherical harmonics, only very diffuse shadows are produced using this approach.

2.3. Ambient Occlusion

Ambient occlusion [AMH02] samples the space of directions around each vertex of an object and approximates the partial occlusion of ambient illumination as the fraction of these samples that hit surfaces. By integrating samples from the environment map for the unoccluded rays, this method can approximate environment map illumination. However, it requires significant pre-computation that prevents dynamics.

3. Environment Map Decomposition

Our environment map decomposition is designed to find compact bright regions in the map and convert them to area lights. Although even a uniform environment map will cast shadows, these "ambient" shadows tend to be very soft and indistinct. The dominant shadows in many HDR maps are caused by relatively small and bright regions within the map that typically correspond to features such as windows and light sources. By automatically identifying such regions and extracting them as area lights, we can then apply existing area light soft-shadow techniques to rapidly compute high quality shadows.

First we need to find appropriate compact regions in the map that are much brighter than their surrounding texels. We

do this using a simple center-surround linear filter similar to those used in edge detection. These filters have a center region with a positive weight surrounded by a region with a negative weight. We convolve the environment map with a series of filters of varying size. Then we choose the pixel and filter size with the maximum response and use them to select the appropriate position and size for the area light. The light's intensity is computed by integrating the environment map's intensity over the filter's center region.

Additional area lights are found using an iterative technique. After each light is found, we subtract the area light's intensity from the environment map. Iteratively subtracting each region helps prevent selecting overlapping lights and avoids counting illumination in more than one source. We then repeat the process to find the next candidate area light. This continues until the next candidate's total intensity falls below a threshold.

We use the existing penumbra wedges method and implementation to compute the shadows from our area lights. Because our area lights are typically much larger than the ones used in [AAM03] and its examples, we found that we could not use all of their features. It was impossible to use textured sources because aliasing problems related to the discrete coverage textures and the considerably better performance of the algorithm for spherical lights rather than rectangular sources was enough to justify preferentially choosing only circular regions.

By placing the spherical lights relatively far from the scene, they closely approximate circular area sources in the direction space of an environment map. To find such regions we use circular filters as shown in Figure 3. The inner region's radius is 75% of the outer radius and the weights of the inner and outer regions are +1 and -10 respectively. The disparity in the inner and outer weights assures that the filter responds positively only when the center covers a much brighter region than the surround. The filter weights are divided by the angular extent of the filter, roughly the magnitude of the gradient in the potential light's penumbra region. In our results we used twenty filter sizes ranging from 4.5 to 45 degrees in angular extent.

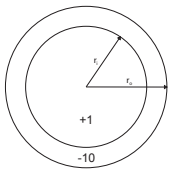

Figure 1: Our light finding filter, weight of inner circle is +1 and the outer circle is -10. The radii satisfy: $r_i = 0.75 \cdot r_o$.

Of course, standard filtering techniques are defined for flat images and an environment map is a non-Euclidean direction map. While fast convolution methods are still possible (e.g., using spherical harmonics), we implemented a much simpler though slower method. We compute the filter response centered at a particular environment map texel using a camera with the appropriate orientation and field of view to reduce the environment map locally to a 2D image. Then it is straightforward to apply the filter to this image. To reduce the search cost we only consider the brightest 10% of the pixels as potential filter locations.

Map Name	Num. Lights	FPS (3 boxes)	FPS (column)
Galileo	8	12.8	7.5
Grace	10	6.7	4.2
Kitchen	4	21.2	5.6
St. Peters	17	5.1	4.1

Table 1: Frame rates for each scene for each map at 512 x 512 pixels. The 3 boxes scene contains 38 polygons and the column scene 440 polygons.

In summary for each iteration, we:

1. Choose the pixel p and the filter f with the highest normalized response. Let Ω_p be the solid angle defined by the inner region of f when pointed in the direction of p
2. Identify $\mathcal{P}$ as the set of pixels in the current map lying in Ω_p. Let the integrated intensity of these pixels be I
3. Set all pixels in $\mathcal{P}$ to black.
4. Create a new area light that covers $\mathcal{P}$ and has intensity I.

We terminate iteration when the intensity of the last light found is less than 2% of the total intensity of all previously found lights. This cutoff threshold is motivated by Weber's Law that claims that the smallest visible intensity difference is approximately 2%. By using this stopping criteria, we prevent lights from being created that would not cast discernible shadows. At the end of the whole process, we produce a map with all the area source regions removed. This map can then be preconvolved with the object's BRDFs to compute fast directionally-variant ambient illumination.

4. Results

All of our results were produced using a dual Xeon 3.06Ghz desktop with an Nvidia GeForce 6800 graphics card. To render the area lights, we use an implementation of penumbra wedges provided by the authors and described in [ADMAM03]. We added only a final pass to add our preconvolved ambient illumination. Depending on the number of lights found, decomposing each map took between 0.5-1.5 hours.

Table 1 and Figures 2 and 3 summarize our results. Figure 2 shows the original map and the area lights found by our

Figure 2: Top row: the original map; bottom row: area lights found by our method. Maps from left to right: Galileo's Tomb, Grace Cathedral, Kitchen and St. Peter's Basilica. All maps courtesy of Paul Debevec.

decomposition. Figure 3 shows side-by-side comparisons of our method to reference images computed using the method from [HH97] for 1024 sample points selected using the environment map sampling method from [ARBJ03]. When comparing frame rates in Table 1, it is important to realize that the time for the penumbra wedge algorithm is proporational both to the size of the lights and the area of the penumbra. For instance, this is why the frame rates for the St. Peter's Basilica map are similar to those for Galileo's Tomb map even though the former has nearly twice as many lights as the latter. For all maps and both scenes, the reference method ran at 0.1fps. We also refer the reader to our supplemental video which shows several short animations produced using our methods. We note that the capture software was imperfect and introduced some choppiness into the animations.

5. Discussion

Our results demonstrate the advantages our method. By separating the maps into two components, we isolate a few regions that cast primary shadows and focus the work of the rendering algorithms on reproducing only these effects. In many cases, the shadows produced are nearly identical to the reference shadows and, even when slightly different, our methods still qualitatively capture the rich natural feel of the original environment's lighting. For many applications, it is much more important that a method capture this richness with good performance than be pixel-to-pixel accurate.

However, it is also important to define the points where our images differ from the references. We note two small differences. First our approximate shadows tend to be slightly softer than the reference shadows. The primary cause is the loss of intensity variation across the solid angle of regions covered by the area lights. We attempted to address this issue using the textured lights provided by penumbra wedges, but we found that the pre-computed coverage textures needed to calculate the penumbra attenuation for textured lights cannot be made large enough to avoid aliasing in large penumbra. We chose to use flat light sources rather than have aliasing, and using another algorithm that better supports large textured sources could reduce this difference. Additionally, our circular filter also tends to find lights that circumscribe the outer boundaries of smaller regions and our area lights tend to be slightly larger than the areas they represent. Our choice to use circular filters was solely motived by penumbra wedges better performance for spherical lights and using different filter shapes could produce tighter fitting lights. The second difference we note is a slight change in tone between our images and the references. This results because penumbra wedges assume that all light from an area source arrives from the direction of its center while the reference method correctly integrates the illumination over the solid angle of the source.

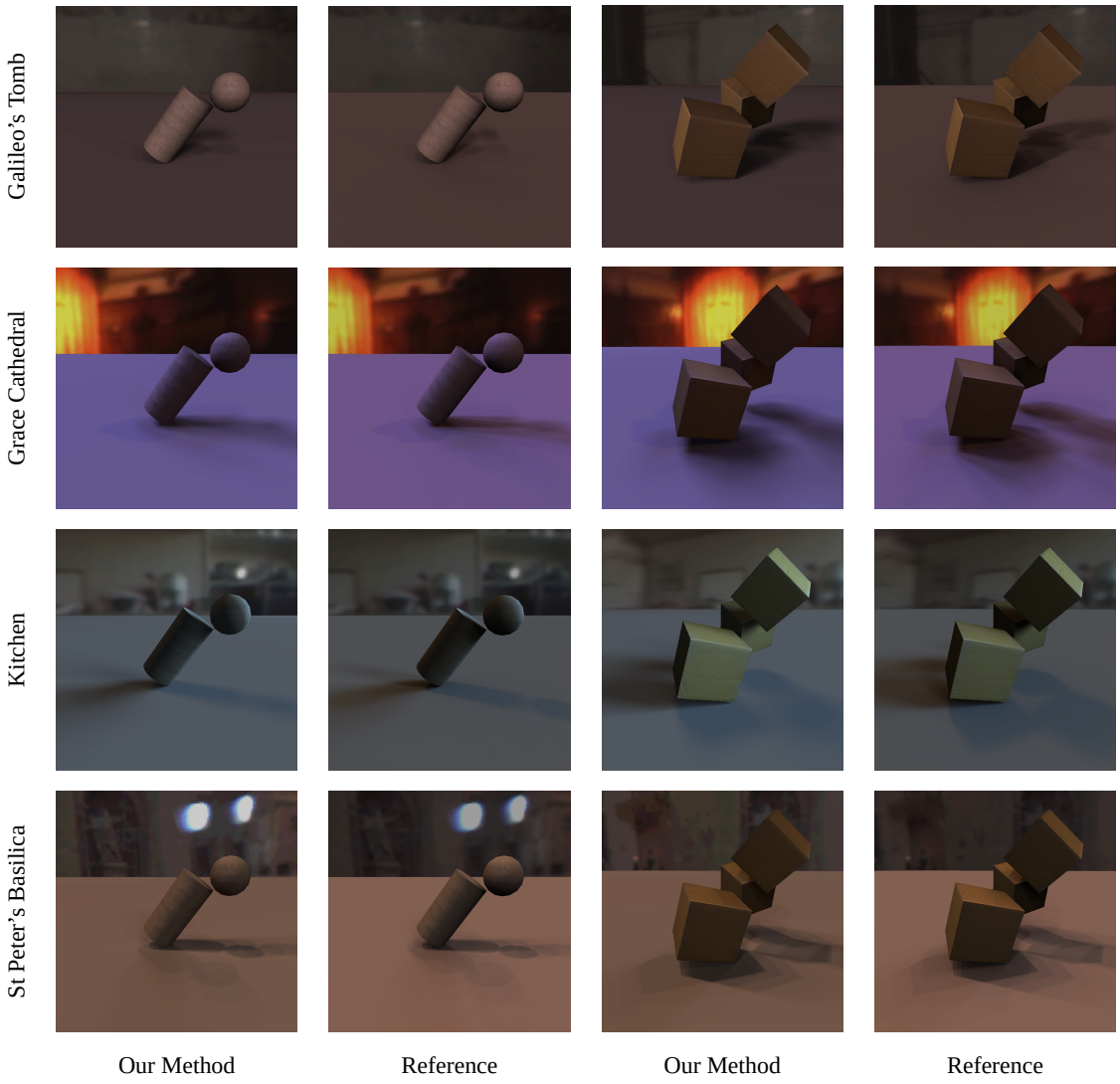

Figure 3: Results for two test scenes. The 3 boxes scene contains 38 polygons and the column scene 440 polygons.

We also note that our performance is completely determined by the performance of penumbra wedges. Since penumbra wedges perform shading calculations for each point in each wedge, their performance is proportional to the the average number of points in a wedge times the number of wedges. Since larger lights have larger penumbra, the performance of penumbra wedges depends directly on the size of the light. For the lights found by our method, this dependence can be a considerable factor in performance. However, performance could be traded for accuracy by shrinking the lights after decomposition.

6. Conclusions

This paper presents a technique to decompose environment maps into area lights and ambient illumination. Unlike any previous methods, this approach captures the most important qualitative features of environment map shadows while still maintaining interactive performance for fully dynamic scenes. By approximating HDR environment maps using area lights, we are able to abstract out their most important features for efficient rendering. This approach can easily enable high-quality environment map shadowing in interactive performance-critical applications such as games.

Another strength of our decomposition method is its generality. Thus in the future, we can not only take advantage of improved graphics hardware, but also improved soft shadow

algorithms if they become available. But our reliance on existing hardware shadowing algorithms can also be a drawback. Our current system works best for environment maps where the lighting is dominated by small bright regions. Relatively uniform maps or maps with very large area lights, such as the open sky, will not work as well because of the lack of good interactive techniques for computing ambient-style shadows. If such methods are developed in the future, they should easily integrate into our novel HDR environment map decomposition approach.

References

[AAM03] ASSARSSON U., AKENINE-MOLLER T.: A geometry-based soft shadow volume algorithm using graphics hardware. *ACM TOG 22*, 3 (2003), 511–520.

[ADMAM03] ASSARSSON U., DOUGHERTY M., MOUNIER M., AKENINE-MOLLER T.: An optimized soft shadow volume algorithm with real-time performance. In *HWWS '03* (Aire-la-Ville, Switzerland, Switzerland, 2003), Eurographics Association, pp. 33–40.

[AMA02] AKENINE-MOLLER T., ASSARSSON U.: Approximate soft shadows on arbitrary surfaces using penumbra wedges. In *EGRW '02* (2002), pp. 297–306.

[AMH02] AKENINE-MOLLER T., HAINES E.: *Real-Time Rendering*. AK Peters, 2002.

[ARBJ03] AGARWAL S., RAMAMOORTHI R., BELONGIE S., JENSEN H. W.: Structured importance sampling of environment maps. *ACM TOG 22*, 3 (2003), 605–612.

[ARHM00] AGRAWALA M., RAMAMOORTHI R., HEIRICH A., MOLL L.: Efficient image-based methods for rendering soft shadows. In *SIGGRAPH '00* (2000), pp. 375–384.

[CD03] CHAN E., DURAND F.: Rendering fake soft shadows with smoothies. In *EGRW '03* (2003), pp. 208–218.

[Cro77] CROW F. C.: Shadow algorithms for computer graphics. *Computer Graphics (Proceedings of SIGGRAPH '77)* (1977), 242–248.

[Hai01] HAINES E.: Soft planar shadows using plateaus. *J. Graph. Tools 6*, 1 (2001), 19–27.

[HH97] HECKBERT P. S., HERF M.: *Simulating Soft Shadows with Graphics Hardware*. Tech. Rep. CMU-CS-97-104, CS Dept., Carnegie Mellon U., Jan. 1997.

[HLHS03] HASENFRATZ J.-M., LAPIERRE M., HOLZSCHUCH N., SILLION F.: A survey of real-time soft shadows algorithms. *Computer Graphics Forum 22*, 4 (Dec. 2003), 753–774. State-of-the-Art Reviews.

[JF03] JAMES D. L., FATAHALIAN K.: Precomputing interactive dynamic deformable scenes. *ACM TOG 22*, 3 (2003), 879–887.

[KK03] KOLLIG T., KELLER A.: Efficient illumination by high dynamic range images. In *EGRW '03* (2003), pp. 45–50.

[KLA04] KAUTZ J., LEHTINEN J., AILA T.: Hemispherical rasterization for self-shadowing of dynamic objects. In *EGSR '04* (2004), Eurographics Association, pp. 179–184.

[KSS02] KAUTZ J., SLOAN P.-P., SNYDER J.: Fast, arbitrary brdf shading for low-frequency lighting using spherical harmonics. In *EGRW '02* (2002), pp. 291–296.

[NRH03] NG R., RAMAMOORTHI R., HANRAHAN P.: All-frequency shadows using non-linear wavelet lighting approximation. *ACM TOG 22*, 3 (2003), 376–381.

[NRH04] NG R., RAMAMOORTHI R., HANRAHAN P.: Triple product wavelet integrals for all-frequency relighting. *ACM TOG 23*, 3 (2004), 477–487.

[ODJ04] OSTROMOUKHOV V., DONOHUE C., JODOIN P.-M.: Fast hierarchical importance sampling with blue noise properties. *ACM TOG 23*, 3 (2004), 488–495.

[RH02] RAMAMOORTHI R., HANRAHAN P.: Frequency space environment map rendering. In *SIGGRAPH '02* (2002), pp. 517–526.

[SHHS03] SLOAN P.-P., HALL J., HART J., SNYDER J.: Clustered principal components for precomputed radiance transfer. *ACM TOG 22*, 3 (2003), 382–391.

[SKS02] SLOAN P.-P., KAUTZ J., SNYDER J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH '02* (2002), pp. 527–536.

[SS98] SOLER C., SILLION F. X.: Fast calculation of soft shadow textures using convolution. In *SIGGRAPH '98* (1998), pp. 321–332.

[WH03] WYMAN C., HANSEN C.: Penumbra maps: approximate soft shadows in real-time. In *EGRW '03* (2003), pp. 202–207.