Vision, Modeling, and Visualization (2012)
M. Goesele, T. Grosch, B. Preim, H. Theisel, and K. Toennies (Eds.)

Fast Accurate Soft Shadows with Adaptive Light Source Sampling

Michael Schwärzler[1], Oliver Mattausch[2], Daniel Scherzer[3] and Michael Wimmer[4]

[1]VRVis Research Center, Austria, [2]University of Zurich, Switzerland,
[3]Max-Planck-Institut für Informatik, Germany, [4]Vienna University of Technology, Austria

Figure 1: *Our proposed method is capable of selecting and rendering a significantly reduced amount of shadow maps needed for a physically correct soft shadow solution using an adaptive light source subdivision.* Left: *Scene rendered from far with 289 fixed samples (10 FPS). Middle Left: The same view point rendered with our method with only 25 samples (67 FPS).* Middle Right: *The same scene, rendered from a closer view point with 289 fixed samples (10 FPS).* Right: *Our method reduces the number of needed samples to 105 (18 FPS).*

Abstract

Physically accurate soft shadows in 3D applications can be simulated by taking multiple samples from all over the area light source and accumulating them. Due to the unpredictability of the size of the penumbra regions, the required sampling density has to be high in order to guarantee smooth shadow transitions in all cases. Hence, several hundreds of shadow maps have to be evaluated in any scene configuration, making the process computationally expensive. Thus, we suggest an adaptive light source subdivision approach to select the sampling points adaptively. The main idea is to start with a few samples on the area light, evaluating there differences using hardware occlusion queries, and adding more sampling points if necessary. Our method is capable of selecting and rendering only the samples which contribute to an improved shadow quality, and hence generate shadows of comparable quality and accuracy. Even though additional calculation time is needed for the comparison step, this method saves valuable rendering time and achieves interactive to real-time frame rates in many cases where a brute force sampling method does not.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction

Algorithms for hard shadow rendering are widely used in today's games and applications. In contrast, the fast and correct calculation of soft shadows is a complex task and still an area of active research. Soft shadows are, in contrast to hard shadows, not cast by point lights without extents, but by area light sources. They do therefore consist of umbra (areas where the light source is completely blocked) and penumbra (areas where the light source is partly visible) regions, and despite the increased computational costs, using them is worth the effort: Nearly every shadow in reality has soft boundaries, so using soft shadows in rendering applications significantly increases the realism of the generated images (see Figure 2). Moreover, inherent shadow map artifacts like

aliasing at the shadow boarders are often hidden through the low frequency soft shadows.

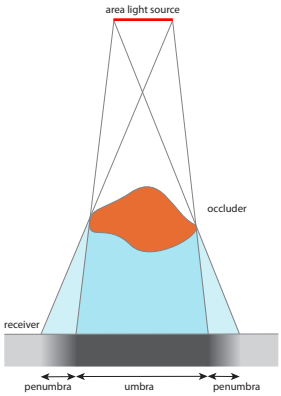

Figure 2: *An area light source leads to a soft shadow, which consists of umbra and penumbra regions.*

In this work, we suggest a novel approach (presented in Section 3) based on the idea of sampling the light source several times in order to obtain *physically correct* soft shadows: We optimize the number of sampling points needed for satisfying results by starting with only very few sampling points and adaptively adding more and more of them, depending on whether the sampling density is already high enough or not. This decision is made by projecting the shadow maps from four sample points forming a quad on the area light to the view point of the camera, and by comparing there how much they differ using hardware occlusion queries. Only if the space between the individual shadow boundaries is too large (i.e. banding artifacts are visible), the quad on the light source is subdivided, and new sampling points are added on the next level(s).

After creating n shadow maps by applying our adaptive sampling strategy and the corresponding weights, we discuss how they can be used to render physically accurate soft shadows at interactive or even real-time frame rates using both *deferred rendering* as well as *texture arrays* in our rendering framework (see Section 3.4).

2. Related Work

A vast amount of real-time soft shadow algorithms have been published during the last few years, most of them based on extensions to the shadow mapping algorithm (see Section 3.1) or the shadow volumes algorithm introduced by [Cro77]. We will therefore focus on the most relevant publications for our work (see [ESAW11] for an extensive overview).

Since the calculation of physically correct soft shadows is generally considered too costly for real-time application, most soft shadow approaches for interactive or real-time applications estimate the complex area visibility (i.e. the amount of the area light source that is visible from a point on a surface) by calculating a single hard shadow from the center of the area light source, and simulate the penumbra using approximative heuristics. The simplifications used in these so-called *single sample approaches* will in general not result in physically correct soft shadows.

In [WH03], not only a shadow map, but also a so-called *Penumbra Map* is generated by analyzing the objects silhouettes from the position of the light source, allowing a penumbra region to be estimated in the illumination pass. [Fer05] suggests using a technique called *Percentage Closer Soft Shadows (PCSS)*, where *Percentage Closer Filtering (PCF)* by [RSC87] is applied and combined with a blocker search: PCF softens hard shadow boundaries by not only comparing the current depth to a single value in the shadow map, but by doing so with the neighboring pixels in the shadow map as well. The percentage of successful shadow tests specifies the shadow intensity. It helps to reduce aliasing artifacts at the softened shadow boundaries, but the penumbra is far from being accurate, as it always has the same size. PCSS therefore uses an additional blocker search in the shadow map, so the filter kernel can be adjusted according to the relation between light, blocker and receiver. To avoid the vast number of shadow map lookups for PCF several pre-filtering methods have been proposed [DL06, AMB*07] that allow real-time frame-rates.

Several papers [GBP06, GBP07, AHL*06, ASK06, SS07] have recently been published, which propose variants of a technique called *backprojection*. The idea is to use a single shadow map not only for depth comparison, but to employ it as a discretized representation of the scene. In order to calculate the visibility factor v for a screen-space pixel p, the shadow map texels are backprojected from p onto the light source, where the amount of occlusion is estimated. These approaches can produce more accurate results than PCSS and variations thereof, but are prone to artifacts (e.g. in cases when occluders overlap, when the light source is too close, or when the penumbra is extremely large) and one may have to backproject a huge number of shadow map texels, which is costly.

The most intuitive, but also slowest approach to generate *physically correct* soft shadows is to generate hard shadows from several sampling points on the area light source and accumulate this information (see Section 3.2). In order to minimize computation time, [HH97] suggest using only a few regularly distributed samples for the calculation. For each shadow receiver, a so-called *attenuation map* is computed by summing up the individual shadows, which is then used to modify the illumination of the object. So, for n sampling points and m receivers, $m \times n$ shadow maps are required. An improvement of this idea has been suggested by [ARHM00]: Instead of calculating and using an attenuation map for each receiver, a *single layered attenuation map* for

the whole scene is created, which allows interactive frame rates on modern graphics hardware. In the method proposed by [SAPP05], the visibility information of many shadow maps is combined into a precomputed compressed 3d visibility structure, which is then used for rendering. Employing CUDA support for irregular data structures, [SEA08] compute accurate soft shadows by evaluating the shadow solution for each visible pixel in screen. [SSMW09] sample the light source over multiple frames exploiting temporal coherence. Although they show cases where they converge to the physical correct result, they have problems with quickly moving objects and can therefore not guarantee correct results in all scene configurations.

Real-Time soft shadows can also be simulated with modified versions of the shadow volumes algorithm, in particular methods based on the Penumbra Wedges algorithm [AAM03, FBP06].

Our algorithm is based on the approaches which use multiple shadow maps per light, but we propose a novel adaptive sampling strategy in order to minimize both the number of shadow maps needed to obtain high quality soft shadows and the rendering time per frame.

3. The Algorithm

In this Section, we introduce our adaptive refinement strategy for the sampling of area light sources, and most importantly, our GPU-based subdivision evaluation criterion. Additionally, we discuss possible ways to render the (potentially) large amounts of generated shadow maps.

3.1. The Shadow Mapping Algorithm

Shadow mapping is an image-based algorithm first introduced by [Wil78]. Its basic idea is to view the scene from the position of the light source in a first pass, and store the depth values of the fragments in a texture (called the *shadow map*). The shadow map therefore contains the distances to all sampled surface points which are illuminated by the light source.

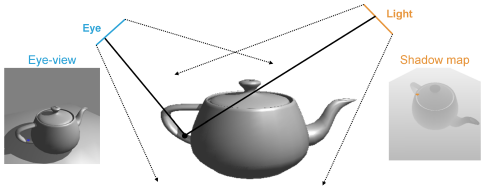

Figure 3: *The shadow mapping algorithm: The depth values as seen from the light source are stored in a shadow map, and are then used in a second pass to generate shadows on the objects.*

In the second pass, the scene is rendered from the camera's point of view. Every fragment is transformed into light space, where its distance to the light source is compared to the corresponding value in the shadow map. If the distance to the current fragment is larger than the shadow map value, it lies in shadow; otherwise it has to be illuminated by this light source. Figure 3 illustrates the basics of the algorithm.

3.2. Estimating Soft Shadows with Light Source Sampling

An area light source can be approximated by n different point light source samples. A shadow map allows us to evaluate for every screen space fragment if it is illuminated by its associated point light.

$$\tau_i(x,y) = \begin{cases} 0 & \text{lit from point light } i \\ 1 & \text{in shadow of point light } i \end{cases} \quad (1)$$

$\tau_i(x,y)$ is the result of the hard shadow test for shadow map i for the screen space fragment at position (x,y). Under the assumption that the point sampling on the area light source is dense enough (i.e n is high enough), the soft shadowing result ψ (i.e., the fractional light source area occluded from the fragment) can be estimated by the proportion $\hat{\psi}_n$ of shadowed samples

$$\hat{\psi}_n(x,y) = \frac{1}{n}\sum_{i=1}^{n}\tau_i(x,y). \quad (2)$$

3.3. Adaptive Refinement of the Sampling Density

Generating soft shadows with multiple shadow maps per light is computationally expensive due to the high sampling density which is required to render smooth, visually appealing penumbra regions. If the density is too low, banding artifacts are likely to appear, and the human visual system does not perceive a soft shadow anymore, but several hard shadows (see Figure 8).

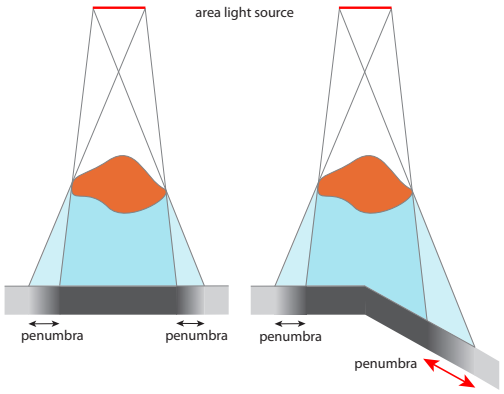

Figure 4: *A slight change in the receiver geometry can cause a significant increase of the penumbra size.*

The larger a penumbra is, the more samples are necessary to create a smooth transitions between the individual

hard shadows. The minimum required sampling density is not easy to predict, though: It depends on the relation between light, blocker and occluder. As can be seen in Figure 4, a slight rotation of the receiver geometry leads to a drastic increase of the penumbra size, making more samples necessary. Due to the perspective projection, the camera's point of view plays a major role here as well, as it determines the size of the penumbra in screen space: if the camera is very close to the shadow, the penumbra region can be as large as the whole frame buffer.

To avoid redundant shadow map computations caused by using a constant high sampling density only required in some worst cases, we suggest to select the sampling points adaptively whenever the scene configuration or the camera position changes.

3.3.1. Generating Shadow Maps

The first step in our algorithm is to create the initial shadow maps at the corners of the area light source. These are the only shadow maps which are always generated; all the others are only computed if necessary (see Section 3.3.3). We assume a square area light source for an easier explanation in this work, but similar subdivision strategies can be found for other kinds of light sources, as our splitting criterion is independent of the actual subdivisions performed. The shadow maps are generated from the sampling points using standard uniform shadow mapping with a perspective projection.

3.3.2. Reprojection

After the creation of the initial sampling points, we project the shadow maps into the same space in order to compare them. It is important that the refinement is dependent on the observer's position and the view: For example, it makes no sense to refine a soft shadow which is far away and hardly visible, while for shadows very close to the camera, it is important to have more samples in order to obtain a smooth penumbra. We therefore project the shadow maps into camera space, where a comparison makes such a view-dependent refinement possible.

The reprojection step is done similar to the second step in the regular shadow mapping algorithm, but instead of using the shadow values from the shadow map for illumination, they are directly used for comparisons as described in Section 3.3.3. In order to generate the correct subdivision level needed for the current screen buffer size, the comparison render target extents must have the same dimensions. If a smaller amount of shadow maps is desired (at the cost of physical accuracy, leading to banding artifacts), the resolution of the comparison render targets can be lower (see Section 3.4.3).

3.3.3. Subdivision Evaluation

The comparison of four neighboring shadow maps in camera space is done in a pixel shader by applying a 2-pass strategy: In the first pass, the reprojected depth values of the four shadow maps are evaluated as in the original shadow map algorithm: For each screen space fragment, the 4 corresponding shadow values are calculated and summed up (i.e. each fragment obtains an integer value between 0 and 4), and stored in the comparison render target.

In the second pass, the stored accumulated shadow values are used to identify potential regions that produce banding artifacts: banding artifacts appear whenever the distances between the hard shadow borders are too large, so that the shadow is perceived as multiple hard shadows instead of a single soft shadow. We therefore investigate the 1-ring neighborhood of each penumbra texel (indicated by a texel with a value between 1 and 3) in the comparison render target texture, and check if there is at least one neighboring texel that has a different value. If this simple condition is fulfilled, the subdivision level is assumed to be sufficient for this texel; otherwise, the area light source has to be subdivided further.

In order to quickly evaluate the need for a subdivision, we exploit the functionality of *hardware occlusion queries* [BMH98, Ope07], which are usually used to evaluate visibility by counting the number of pixels drawn on the screen. By discarding all fragments for which the subdivision level is sufficient, the remaining fragments can efficiently be counted. If at least one pixel is output, the area light source needs further refinement in this frame. Note that similar to lowering the resolution of the comparison render target as explained in Section 3.3.2, increasing this threshold and tolerate a few fragments causing banding artifacts can also help to reduce the number of shadow maps.

3.3.4. Generating Additional Sampling Points

If the subdivision evaluation suggests creating a further refinement level on the area light source, new sampling points (and the corresponding shadow maps) have to be created. In the case of a two-dimensional rectangular area light source, we suggest using a quadtree-like structure: If the comparison steps makes a subdivision necessary, the rectangle is split into 4 sub-quads, and new shadow maps are generated on all new corners (See Figure 5).

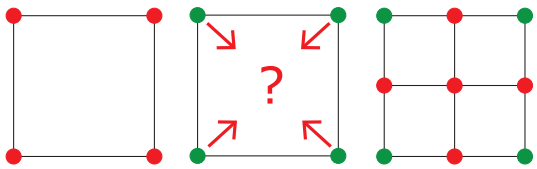

Figure 5: *Subdividing a rectangular area light source:* Left: *Generate sampling points at the quad corners.* Middle: *Compare corresponding shadow maps in a common projection center (camera space).* Right: *If necessary, subdivide the quad into 4 sub-quads, repeat steps for each sub-quad.*

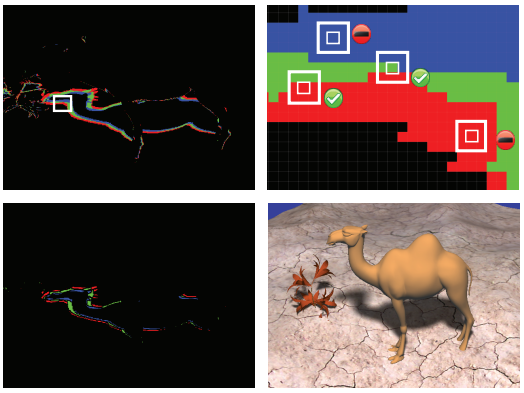

Figure 6: *Test for further subdivision.* Top Left*: The shadow values of 4 shadow maps in a quad are projected to camera space and accumulated. For visualization purposes, the amount of received shadow has been color-coded: red = 1, green = 2, blue = 3, black = 0 or 4.* Top Right: *Close-up view of the marked region in the left image. For each fragment with a value from 1-3, the 1-ring neighborhood is tested for different values (green tick). If no different value is found, the distance between the shadow maps is too large, and the test fails (red symbol).* Bottom Left:*: The fragments that failed the test are drawn in the second pass, and counted using a* hardware occlusion query. *If at least one pixel is drawn, the light source needs to be subdivided.* Bottom Right:*: Final result after subdivision.*

For the new subdivision level, the whole procedure is repeated again: Shadow maps are generated from the new sampling points' positions, and are compared to their quad neighbors. This refinement process is repeated until either the sampling density is high enough in all areas to fulfill the condition defined in Section 3.3.3, or a predefined maximum number of shadow maps has been created.

3.4. Evaluating the Shadow Map Information

After the computation of the shadow maps, their contribution must be evaluated in an illumination render pass. This step is basically similar to the second render pass in the standard shadow mapping algorithm. Still, difficulties can arise due to to differing subdivision depths (Section 3.4.1) and due to the large amount of depth textures which have to be sampled (Section 3.4.2).

3.4.1. Assigning Shadow Map Contribution Weights

If all shadow maps generated with our refinement strategy contribute to the final soft shadow solution with equal weight, the darkness of the penumbra can sometimes vary slightly from the exact solution, if the distribution of the adaptively selected sampling points varies significantly. We therefore apply weights on the sampling points: In areas of many subdivision, the individual samples are assigned a smaller weight, and will not contribute as much to the darkness of the penumbra as the ones with a large weight.

In case of a 2D area light source that is subdivided as proposed in Section 3.3.3, the weight ω_i assigned to the i^{th} shadow map is calculated with

$$\omega_i = \frac{1}{(2^d+1)^2}, \tag{3}$$

where d is the subdivision depth. The sum of all weights is 1 if all samples reach the same subdivision depth d. Otherwise, the weights have to be normalized to make sure the final accumulated shadow values lie between 0 (fully lit) and 1 (fully shadowed).

3.4.2. Soft Shadow Visualization using n Shadow Maps

For the calculation of soft shadows, the information from all generated shadow maps has to be checked for each screen space pixel. The hard shadow test values (0 or 1) from all shadow maps i are multiplied with their weight ω_i and summed up, resulting in an estimate for the percentage of occlusion. If the number of shadow maps is high, this can lead to problems due to the limited amount of textures that can be sampled in a single rendering pass.

A way to solve this is to make use of a *deferred rendering* system introduced by [DWS*88] as well as a so-called *accumulation buffer*, which is a screen-space buffer with a single data channel. For each shadow map, we render the scene in a separate render pass. Instead of using the obtained hard shadow value of a screen space fragment $f(x,y)$ directly for illumination, we multiply it with its weight and add it to the accumulation buffer at the position $f_{acc}(x,y)$. A preliminary depth pass helps to ensure that only shadow values from "valid" (i.e. visible) fragments contribute to the accumulation buffer.

After n render passes, all shadow maps have been evaluated, and the accumulation buffer is filled. Now, in a final rendering pass, the scene is illuminated: For each screen space fragment $f(x,y)$, the corresponding accumulation buffer value $f_{acc}(x,y)$ is sampled and used as the occlusion percentage. Note: Since current graphics hardware does not support read and write operations on render targets at the same time, two instances of the accumulation buffer have to be created and swapped each rendered frame, resulting in an additional need for memory on the GPU.

Alternatively, the introduction of so-called *Texture Arrays* in newer graphics APIs makes it possible to send up to 512 textures with the same size and format to the shader, where they can be sampled arbitrarily. This functionality is perfectly suited for our purposes, as it allows us to sample many shadow maps from within the same pixel shader instance. The current fragment's occlusion value can therefore be obtained without the need for additional passes, saving n

read/write operations as well as the memory previously consumed by the accumulation buffer.

3.4.3. Filtering

As already stated in Section 3.3.2, the resolution of the comparison render target can be defined to be smaller than the frame buffer resolution in order to trade physical accuracy for a lower number of sampling points (and therefore higher performance). Since fewer shadow maps are generated, banding artifacts are more likely to become visible. Similar problems occur if the camera is extremely close to a penumbra, so that the maximum number of shadow maps is not sufficient to generate an appealing penumbra region, or if a few pixels with banding artifacts are allowed during the GPU-based splitting evaluation (See Section 3.3.3).

In order to improve the smoothness of the transitions between the individual shadow maps, we therefore suggest sampling them using a small PCF kernel in such situations. PCF filtering softens the shadow boundaries, and a version with a 2x2 kernel can be used on modern graphics hardware without performance hit.

4. Results and Evaluation

All tests and images in this paper were calculated with a comparison render target buffer size of 1024×768p, and a shadow map size of 512^2. The system on which we were testing our approach consisted of an Intel Core i7-920 Processor with 4 Cores, 6GB RAM, and a NVidia Geforce 580GTX with 1.5 GB Memory.

4.1. Implementation

We implemented both rendering methods described in Section 3.4.2 in a DirectX 10 rendering framework application using a two-dimensional rectangular light source. For the shadow maps, we use 32Bit floating point textures with a size of 512^2, and store the depth linearly. The maximum allowed number of shadow maps generated by our subdivision strategy is 289, representing a subdivision depth of 4 levels. For the deferred rendering implementation, we use 32Bit floating point textures with the same dimensions as the frame buffer for both the accumulation buffer as well as for the needed depth buffer.

The implementation using texture arrays to evaluate the shadow illumination performs slightly better (approximately 10% faster) than the deferred rendering solution, since the shadow map evaluation can be done in a single pass, and no additional read/write operations on the accumulation buffer are needed. Still, the deferred rendering solution seems to be an acceptable alternative for the application of our method in rendering systems using older APIs.

4.2. Visual Comparison

As can be seen in Figure 1, Figure 7, Figure 9 and Figure 10, the achievable visual quality of our proposed solution with only a few shadow maps is nearly identical to images with a significantly higher (fixed) amount of sampling points. In Figure 7, we also show the shadow solution computed by the PCSS method (with 64 samples for the blocker search and 64 samples for the filtering step). Since there the visibility is calculated using only a single shadow map from the center of the light source, the resulting shadow differs significantly from our solution computed with correct visibility.

In Figure 8, we show the illumination results generated with a smaller comparison render target, leading to banding artifacts due to the lower number of shadow maps. These artifacts can easily be hidden by applying a simple PCF filter, but the physical accuracy is of course negatively affected by this approximation. Figure 9 and Figure 10 demonstrate the achievable speed-up that can be gained by reducing the comparison render target resolution as well as the introduced error.

4.3. Performance

The goal of our algorithm is to improve the generation of physically correct soft shadows by adaptively selecting only the light source samples which do really contribute to the visual quality of the penumbrae. The reduced number of needed shadow maps increases the overall rendering performance, but the subdivision evaluation produces an overhead of approximately 30% of the rendering time per frame. In scene configurations where the penumbra regions are comparatively small, and a significant reduction of shadow map samples is possible, even real-time performance can be achieved with our approach. Of course, whenever a penumbra is extremely large and fills a wide area of the frame buffer, and the system maximum number of samples has to be used, the method performs worse than sampling the light source with this fixed maximum number.

4.4. Limitations

Since the size of the penumbra regions can change drastically within a short time, the number of needed samples can vary widely as well, making our approach not suitable for applications where a guaranteed constant frame rate is necessary (like for example in real-time 3D games). We therefore see the use of this method in modeling and design scenarios (e.g. for light design purposes), where a fast real-time preview of a physically correct shadowing solution is necessary. In the worst case, when using this method in systems with a maximum number of usable shadow maps in combination with scenes in which large penumbras are prevalent leads to the situation that no performance gain can be achieved (see Section 4.3).

Figure 7: *Visual Comparison, from left to right: (1) Regular sampling with 289 shadow maps, acting as ground truth for our comparisons (8 FPS). (2): Our method, 93 shadow maps (17 FPS). (3): Difference image between ground truth and our approach with 93 shadow maps, scaled by factor 5 for visualization purposes. (4) PCSS soft shadow solution with visibility calculated from only a single shadow map (64/64 samples for blocker search/filtering step, 370 FPS). (5) Difference image between ground truth and PCSS, scaled by factor 5 for visualization purposes.*

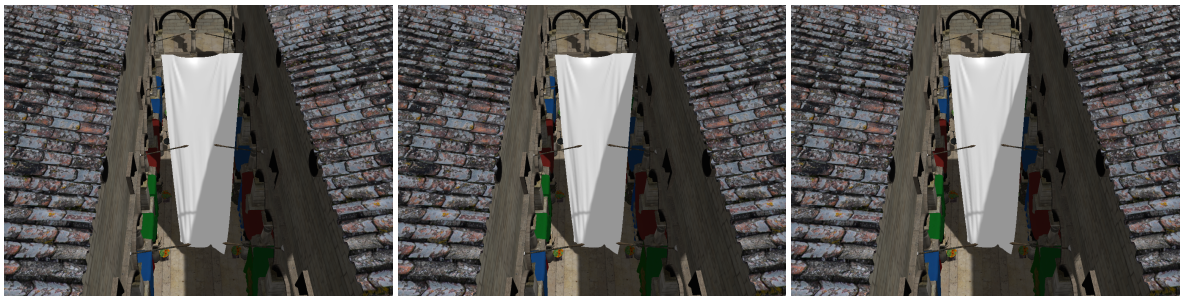

Figure 9: *Visual Comparison using the complex* Sponza Atrium *scene:* Left: *Regular sampling with 289 shadow maps, acting as ground truth for our comparisons (2.5 FPS).* Middle: *Our method, 163 shadow maps (5 FPS).* Right: *Our method with only half the comparison render target size and a 3x3 PCF filtering requires only 14 shadow maps and is rendered at 40 FPS. See Figure 10 for difference images.*

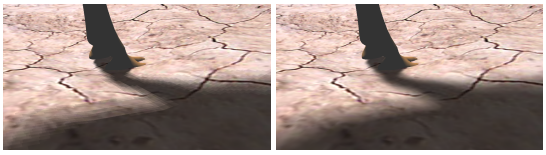

Figure 8: Left: *Reducing the resolution of the comparison render target leads to the use of fewer shadow maps and therefore to banding artifacts.* Right: *By Applying a simple* 3×3 *PCF filter, the artifacts can be significantly reduced – but physical correctness is not guaranteed anymore.*

Figure 10: *Visualized differences between the screenshots of the* Sponza Atrium *scene in Figure 9:* Left: *Difference image between ground truth (Figure 9, left) and our approach with 163 shadow maps (Figure 9, middle), scaled by factor 40 for visualization purposes.* Right*: Difference image between ground truth (Figure 9, left) and our method with reduced comparison render target size and PCF with 14 shadow maps (Figure 9, right) , scaled by factor 40 for visualization purposes.*

5. Conclusions and Future Work

We presented an algorithm which is able to render physically accurate soft shadows that in most cases outperforms the regular light sampling method with a fixed sampling rate, since only the samples which contribute to the visual quality are computed and evaluated. The decision whether another sampling point is needed in-between two neighboring ones is being reached by reprojecting the corresponding shadow maps to the camera's point of view and comparing them there using an occlusion query. The time needed for these checks is often more than compensated by the reduced number of shadow maps which have to be calculated.

In our test application, we were able to render soft shadows of a quality similar to the ones generated with 289 samples, but at interactive or even real-time frame rates. Performance can even be further increased by relaxing the subdivision criterion and using a simple PCF filter to hide potential banding artifacts.

As a future work, we want to reduce the computation time needed for the comparison step by finding better ways to handle the time-consuming occlusion queries and especially the corresponding GPU/CPU synchronization. This could for example be achieved by exploiting the temporal coherence between consecutive frames, so that the current subdivision state of the area light source is reused and only adapted when necessary in the next frame. Moreover, we plan to investigate the relation between the banding artifacts in case of a lower-resolution comparison render target and the necessary shadow filtering kernel sizes, so that self-regulating filtering mechanisms can be found. As a similar enhancement, filtering could be restricted to regions with banding artifacts only, further increasing the rendering performance. Further research effort could also be spent on finding techniques for a more randomized subdivision strategy, or on extending the algorithm to volumetric light sources.

6. Acknowledgments

The competence center VRVis is funded by BMVIT, BMWFJ, and City of Vienna (ZIT) within the scope of COMET - Competence Centers for Excellent Technologies. The program COMET is managed by FFG. This work was also supported by the EU FP7 People Programme (Marie Curie Actions) under REA Grant Agreement no. 290227.

References

[AAM03] ASSARSSON U., AKENINE-MÖLLER T.: A Geometry-based Soft Shadow Volume Algorithm using Graphics Hardware. *ACM Trans. Graph. 22*, 3 (2003), 511–520. 3

[AHL*06] ATTY L., HOLZSCHUCH N., LAPIERRE M., HASENFRATZ J.-M., HANSEN C., SILLION F.: Soft Shadow Maps: Efficient Sampling of Light Source Visibility. *Computer Graphics Forum 25*, 4 (dec 2006). 2

[AMB*07] ANNEN T., MERTENS T., BEKAERT P., SEIDEL H.-P., KAUTZ J.: Convolution Shadow Maps. In *Rendering Techniques 2007: Eurographics Symposium on Rendering* (Grenoble, France, June 2007), Kautz J., Pattanaik S., (Eds.), vol. 18, Eurographics, pp. 51–60. 2

[ARHM00] AGRAWALA M., RAMAMOORTHI R., HEIRICH A., MOLL L.: Efficient Image-based Methods for Rendering Soft Shadows. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (2000), pp. 375–384. 2

[ASK06] ASZÓDI B., SZIRMAY-KALOS L.: Real-time Soft Shadows with Shadow Accumulation. In *Eurographics 2006 Short Presentations* (2006), pp. 53–56. 2

[BMH98] BARTZ D., MEISSNER M., HÜTTNER T.: Extending Graphics Hardware for Occlusion Queries in OpenGL. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware* (New York, NY, USA, 1998), HWWS '98, ACM, pp. 97–ff. 4

[Cro77] CROW F. C.: Shadow Algorithms for Computer Graphics. In *Proceedings of the 4th annual conference on Computer graphics and interactive techniques* (July 1977), George J., (Ed.), vol. 11, ACM Press, pp. 242–248. 2

[DL06] DONNELLY W., LAURITZEN A.: Variance Shadow Maps. In *In SI3D 06: Proceedings of the 2006 symposium on Interactive 3D graphics and games, ACM* (2006), Press, pp. 161–165. 2

[DWS*88] DEERING M., WINNER S., SCHEDIWY B., DUFFY C., HUNT N.: The Triangle Processor and Normal Vector Shader: a VLSI System for High Performance Graphics. *SIGGRAPH Comput. Graph. 22*, 4 (1988), 21–30. 5

[ESAW11] EISEMANN E., SCHWARZ M., ASSARSSON U., WIMMER M.: *Real-Time Shadows*. A K Peters/CRC Press, Boca Raton, FL, USA, 2011. 2

[FBP06] FOREST V., BARTHE L., PAULIN M.: Realistic Soft Shadows by Penumbra-Wedges Blending. In *Graphics Hardware, Vienne, Autriche, 03/09/2006-04/09/2006* (http://www.eg.org/, 2006), Eurographics, pp. 39–48. 3

[Fer05] FERNANDO R.: Percentage-closer Soft Shadows. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Sketches* (New York, NY, USA, 2005), ACM, p. 35. 2

[GBP06] GUENNEBAUD G., BARTHE L., PAULIN M.: Real-time Soft Shadow Mapping by Backprojection. In *Eurographics Symposium on Rendering (EGSR 2006), Nicosia, Cyprus* (2006), Eurographics, pp. 227–234. 2

[GBP07] GUENNEBAUD G., BARTHE L., PAULIN M.: High-Quality Adaptive Soft Shadow Mapping. *Computer Graphics Forum, Eurographics 2007 proceedings 26*, 3 (septembre 2007), 525–534. 2

[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. 2

[Ope07] OPENGL WORKING GROUP: OpenGL Occlusion Query Extension. http://www.opengl.org/registry/specs/ARB/occlusion_query.txt, Apr. 2007. 4

[RSC87] REEVES W. T., SALESIN D. H., COOK R. L.: Rendering Antialiased Shadows with Depth Maps. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (1987), ACM Press, pp. 283–291. 2

[SAPP05] ST-AMOUR J.-F., PAQUETTE E., POULIN P.: Soft Shadows from Extended Light Sources with Penumbra Deep Shadow Maps. In *Graphics Interface 2005* (May 2005), pp. 105–112. 3

[SEA08] SINTORN E., EISEMANN E., ASSARSSON U.: Sample-based Visibility for Soft Shadows Using Alias-free Shadow Maps. *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering 2008) 27*, 4 (June 2008), 1285–1292. 3

[SS07] SCHWARZ M., STAMMINGER M.: Bitmask Soft Shadows. *Comput. Graph. Forum 26*, 3 (2007), 515–524. 2

[SSMW09] SCHERZER D., SCHWÄRZLER M., MATTAUSCH O., WIMMER M.: Real-Time Soft Shadows Using Temporal Coherence. In *Advances in Visual Computing: 5th International Symposium on Visual Computing (ISVC 2009)* (Dec. 2009), Lecture Notes in Computer Science, Springer. 3

[WH03] WYMAN C., HANSEN C.: Penumbra Maps: Approximate Soft Shadows in Real-Time. In *Proceedings of the 14th Eurographics workshop on Rendering* (2003), Eurographics Association, pp. 202–207. 2

[Wil78] WILLIAMS L.: Casting Curved Shadows on Curved Surfaces. *Computer Graphics (SIGGRAPH '78 Proceedings) 12*, 3 (Aug. 1978), 270–274. 3