A Precalculated Point Set for Caching Shading Information

J. Bikker[†1] and R. Reijerse[‡2]

[1]IGAD program, NHTV University of Applied Sciences, Breda, Netherlands
[2]Computer Graphics & CAD/CAM Group, TU Delft, Netherlands

Abstract

In this short paper, we present a caching scheme for low-frequent shading information. The scheme consists of a precalculated, carefully constructed sparse set of points, the density of which adapts to local shading requirements, which we estimate by calculating the ambient occlusion. This fixed set of points is then used to store shading information. This shading information is calculated once, or updated prior to rendering each frame. Finally, during rendering, the shading information stored in the point set is used to estimate shading for any point in the scene using interpolation. We apply this scheme to render soft shadows and indirect lighting in a real-time ray traced game environment.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism , I.3.6 [Computer Graphics]: Methodology and Techniques

1. Introduction

Recently, real-time ray tracing has reached a state where it has become an interesting option for rendering game graphics. For the field of games, ray tracing promises an intuitive approach to rendering, making many of the approximations used in rasterization unnecessary. At the same time, ray tracing enables correct visualization of shadows, reflections and refractions; these are hard to do well using rasterization.

Many features that are trivial to implement in a ray tracer, such as soft shadows and global illumination, come at a cost that is unfeasible in a real-time context, at least for now. The high cost of divergent rays and the almost linear dependency of rendering time on the total number of rays force us to restrict ourselves to the most basic features, despite the algorithmic simplicity of the more advanced ones.

We would rather not revert to the same approximations used in rasterization. Instead, we would like to use approximations that converge to the correct solution, so that we can scale up with future advances in hardware technology. One such approximation used for low-frequent shading is the irradiance cache [WRC88]. In this short paper, we propose a variation on this algorithm. We use a three-stage scheme, that decouples determination of the locations for point sampling the shading and shading itself, and that also decouples expensive shading calculations from actual rendering. In a preprocess step, we build a static point set. Then in a second step, we fill this set with shading information. Finally, during rendering, we query this shading information using interpolation.

2. Background and Previous Work

Precalculating and storing shading data is a common way to enable real-time rendering of scenes with complex lighting. Even when lighting for the scene is not static, storing illumination in a separate data structure is beneficial, as it allows undersampling of the (often low-frequent) shading information.

In the context of rasterization, precalculated shading information is usually directly linked to scene geometry. Shading data can be stored in vertices, or in shadowmaps [Wil78, RSC87]. Shading stored in vertices has the disadvantage that interpolation quality is directly related to mesh resolution. Shadowmaps and lightmaps on the other hand result in aliasing.

In the context of ray tracing, meshless schemes have been employed: Photon mapping [Jen96] uses the locations of

† bikker.j@nhtv.nl
‡ r.h.reijerse@student.tudelft.nl

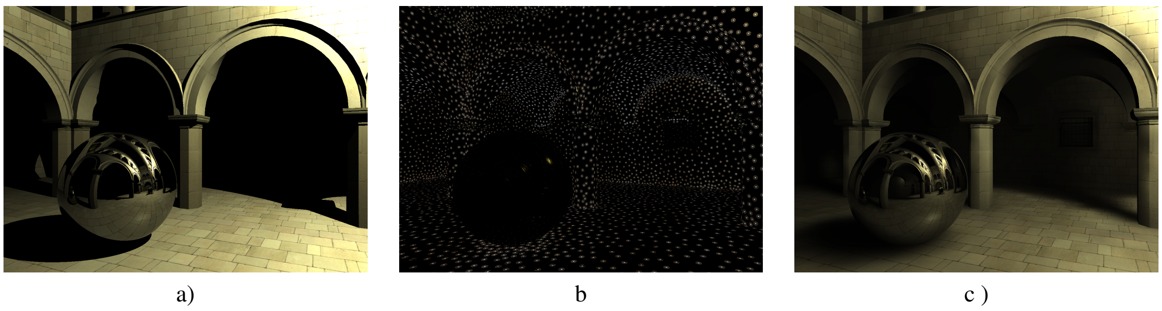


Figure 1: *Rendering using the point set. a) Direct illumination only; b) Visualization of the point set; c) Soft shadows. The size of the point set in these images is 112k, and was generated in less than 2 minutes.*

particles that hit scene geometry. The density of the photon map depends on the amount of incident light; areas that receive little light also receive few photons. The irradiance cache [WRC88] uses a scheme that leads to a point set with a density that adapts to local scene complexity, as well as details in the lighting: Few points are created for areas with low variance in illumination, while areas with high frequent details receive many samples. These points are created on the fly however. In a multithreaded environment, this leads to excessive synchronization.

A recent paper [LZT*08] describes a meshless approach using a precalculated point set. Their scheme is designed for GPUs, and relies on a hierarchical point set with a high density, rather than adaptive density. Furthermore, their scheme targets precomputed light transport (PRT, [SKS02]) rather than direct storage of irradiance. Furthermore, the scheme is tightly coupled to the shading calculations.

In this short paper, we present a scheme that maintains the benefits of meshless schemes as well as a sampling density that adapts itself to local requirements. In contrast to irradiance caching, we do not create this point set on the fly; instead, the point set is created once, during a preprocess that estimates optimal positions for point sampling low-frequent shading information. We use a variation of the Poisson Disk process for this: The density of the points is determined using ambient occlusion [ZIK98], which provides a good estimate of local scene complexity. This static point set is then used to store shading information. During rendering, the shading data stored in the point set is interpolated to obtain the final shading for any point in the scene. Calculating shading information is thus decoupled from actual rendering.

3. Constructing the Point Set

A point set generated using a Poisson Disk process has various desirable properties. Samples distributed this way prevent aliasing artifacts, and capture the continuous function that they represent well. The Poisson Disk distribution is the result of a random sampling process with a minimum-distance rejection criterion. It is this distribution that is the basis for our point set, and the minimum distance between two points is a key parameter for the algorithm. We create this set on the surfaces of the scene. Furthermore, we link the minimum distance between individual points to local scene complexity. Contrary to the irradiance cache, we estimate local complexity using ambient occlusion.

3.1. Dart Sources

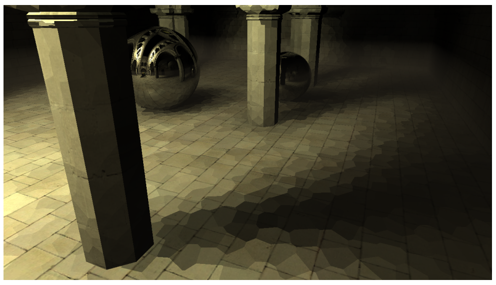
A first step prepares the dart throwing process that is used to obtain the Poisson Disk distribution. Darts will be thrown from several points in the scene. The locations of these points are determined by shooting particles from the light sources in the scene, a process similar to photon shooting. The particles make several diffuse bounces, and at each vertex of their path, a hemispherical 'dart source' is created. This ensures that darts are only thrown at locations of the scene that are reachable from the scene lights; the result is essentially a flood fill. Points are thus never generated on the inside of solid objects or on the outside of the scene. This is achieved without any explicit knowledge about solid volumes in the scene.

3.2. Throwing Darts

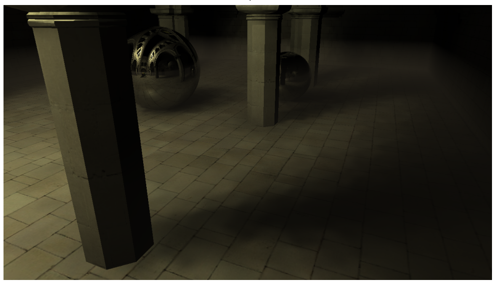
Once the dart sources are generated, darts are thrown by these sources. Where a dart hits scene geometry, the ambient occlusion is calculated. Based on this value, a search radius is determined, which lies between a minimum and a maximum radius. If an existing dart is found within the search radius, the new dart is discarded. This process leads to a point set with a density that is greater in occluded areas, and very sparse on large planar surfaces (see Figure 2).

3.3. Implementation Details

Calculating ambient occlusion is a relatively expensive part of the scheme. To prevent many ambient occlusion queries for darts that are rejected anyway, we first check within a



a)



b)

Figure 2: *Direct visualization of the shading information results in a Voronoi diagram. a) The Voronoi cells; b) The smoothed result.*

maximum radius for existing darts. If none are found, the dart is trivially accepted. Likewise, we use the minimal radius for a trivial reject.

4. Storing and Updating the Shading Information

Using the point set to store shading information rather than calculating this shading per pixel has several benefits. First of all, the size of the point set is typically much smaller than the amount of pixels. Besides that, calculating the shading information can now be decoupled from the rendering process. Shading can e.g. be updated incrementally, or in a view dependent manner. Another option is to store several shading values in a single point of the set, so that soft shadows can be recalculated for every frame, while diffuse indirect lighting is updated for only a few sample points per frame.

Depending on the density of the point set, it is suitable for storing diffuse indirect lighting, ambient occlusion and approximate soft shadows. We noticed that for soft shadows, the discontinuities near occluders cannot be captured correctly unless a dense set of points is used. For diffuse indirect illumination however, even a very small set of points looks plausible.

5. Point Set Queries

Before the point set can be used for rendering, a structure is assembled for efficiently querying the point set. This data structure is a regular 3D array, which holds arrays of pointers to the points in the set. Each point is inserted as an oriented disc, and added to every grid cell it intersects.

We also experimented with an octree structure instead of a 3D array. However, the octree is an order of magnitude slower than the naive grid. Even using the grid, query times are significant, due to the large amount of cache misses. Using the same shading information for 2x2 rays hitting the same primitive is an approximation that is visually indistinguishable from the original approach, but about twice as fast. In practice, querying the grid for shading information takes approximately 50% of the time needed to send shadow rays to a single light source.

Visualization of the shading information stored in the point set by searching for the nearest point yields an irregular approximation of the original signal (see Figure 2). A more visually pleasing result is obtained by adding a random number to the calculated distances to the points. If enough samples are taken, this results in a smoothed Voronoi diagram. Alternatively, the smoothed solution can be calculated directly:

$$\begin{aligned} C_p &= searchradius - distance_p, \\ shade &= \sum_{p=1}^{N} (shade_p C_p) / \sum_{p=1}^{N} C_p, \end{aligned}$$

where N is the number of points within the search radius, C_p is the contribution of a single point, $shade_p$ is the shading value stored in that point, and shade is the final interpolated result.

This will cause lone points to have a linear gradient from the point itself towards the edge of the search radius. Instead of a linear gradient, a quadratic or cubic gradient can be used, to make the shape follow the Voronoi edges more closely.

Note that this algorithm implements part of the Shepard approximation [She68]. The full algorithm would be too costly in the context of a real-time ray tracer.

6. Results

We implemented this caching scheme in the Arauna real-time ray tracer [Bik07]. The point set and the grid have been successfully applied to store indirect lighting and soft shadows. We tested the algorithm on indirect illumination of the well-known Sponza Atrium scene (See Figure 3). The shadows in this scene are ray traced. For this rendering a point set of 60k points was used. This set captures the indirect illumination well; for soft shadows, a larger set of 112k points was used to capture the higher frequency details near occluders with acceptable fidelity. Query times for 60k or 110k points are comparable, but obviously, updating the larger set for dynamic lights takes longer.

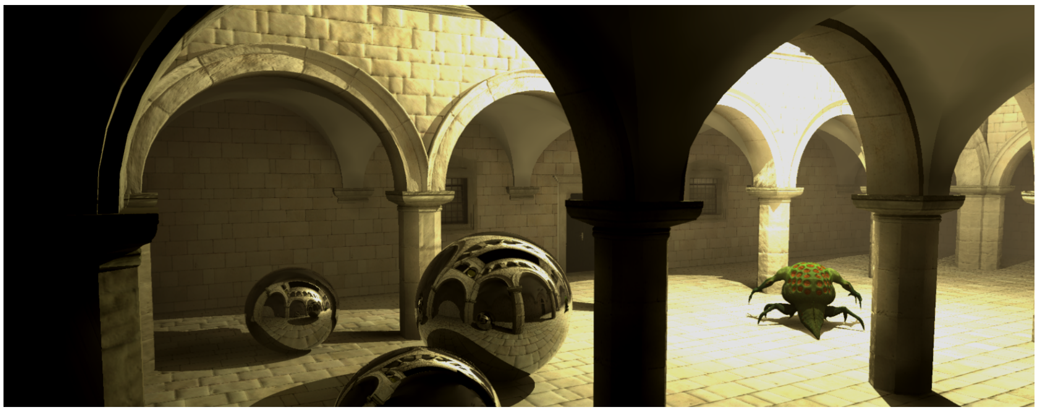


Figure 3: *The Sponza Atrium rendered at several frames per second, with diffuse indirect lighting.*

7. Conclusions and Future Work

In this short paper, we have shown that a sparse set of points can be used to cache low-frequent shading information. Such a set can be determined in a preprocess by using ambient occlusion to steer a Poisson Disk process. The set can then be used to decouple the calculation of shading information from actual rendering, and to undersample this shading information. Adaptive density of the point set guarantees optimal precision in areas of high variance.

The quality of the point set depends heavily on its size, and how well it adapts to local requirements. Currently, we use ambient occlusion for this; a better set could be obtained by using the first derivative of the ambient occlusion. We would like to explore this further.

Besides improving the point set itself, we would like to work on applications of the point set. So far, we have applied it to static shading. We would like to explore ways to update the shading dynamically. The presented algorithm also promises to be suitable for non-static scenery. Rigid animation and deformable geometry could be handled by transforming points along with the geometry they belong to. The aim is ultimately to use this algorithm in a real-time ray traced game.

8. Acknowledgments

The Sponza Atrium scene was modeled by Marko Dabrovic. The BugBackToad model was modelled by Son Kim. Erik Jansen provided useful suggestions for improving the paper.

References

[Bik07] BIKKER J.: Real-time ray tracing through the eyes of a game developer. In *Interactive Ray Tracing, 2007. RT '07. IEEE Symposium on* (Ulm, Germany, 2007), IEEE, pp. 1–10.

[Jen96] JENSEN H.: Global Illumination Using Photon Maps. In *Rendering Techniques '96 (Proc. 7th Eurographics Workshop on Rendering)* (1996), Springer, pp. 21–30.

[LZT*08] LEHTINEN J., ZWICKER M., TURQUIN E., KONTKANEN J., DURAND F., SILLION F. X., AILA T.: A meshless hierarchical representation for light transport. *ACM Trans. Graph. 27*, 3 (2008), 1–9.

[RSC87] REEVES W. T., SALESIN D. H., COOK R. L.: Rendering antialiased shadows with depth maps. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1987), ACM, pp. 283–291.

[She68] SHEPARD D.: A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference* (New York, NY, USA, 1968), ACM, pp. 517–524.

[SKS02] SLOAN P.-P., KAUTZ J., SNYDER J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM, pp. 527–536.

[Wil78] WILLIAMS L.: Casting curved shadows on curved surfaces. In *SIGGRAPH '78: Proceedings of the 5th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1978), ACM, pp. 270–274.

[WRC88] WARD G., RUBINSTEIN F., CLEAR R.: A Ray Tracing Solution for Diffuse Interreflection. In *Computer Graphics* (1988), pp. 85 – 90.

[ZIK98] ZHUKOV S., IONES A., KRONIN G.: An Ambient Light Illumination Model. In *Rendering Techniques '98 (Proceedings of the Eurographics Workshop on Rendering)* (1998), pp. 45–55.