SIGGRAPH 2010 Course: Global Illumination Across Industries

Point-Based Global Illumination for Movie Production

Per H. Christensen

Pixar Animation Studios

July 2010

Abstract

This course note describes a fast point-based method for computing diffuse and glossy global illumination, area light illumination and soft shadows, HDRI environment map illumination, multiple diffuse reflection bounces, final gathering for photon mapping, ambient occlusion, reflection occlusion, and volume scattering. The results are free of noise and the run-time does not increase due to displacement maps on surfaces, complex shaders, or dozens of complex light sources. These properties make the method suitable for movie production.

The first step generates a point cloud (surfel) representation of the directly illuminated geometry in the scene. The surfels in the point cloud are organized in an octree, and the power from the surfels in each octree node is approximated either as a single large surfel or using spherical harmonics. To compute the indirect illumination at a receiving point, we rasterize the light from all surfels using three degrees of accuracy: ray tracing, disk approximation, and clusters. Huge point clouds are handled by reading the octree nodes and surfels on demand and caching them.

The method is integrated into Pixar's RenderMan renderer (PRMan) and has been used in the production of more than 35 feature films to date.

1 Introduction

This course note starts with a brief discussion of how widely global illumination is now used in movie production. This is a fairly new development, made possible by improved algorithms and more powerful computers. The main part of the course note is a description of our point-based global illumination method: what inspired it, how it works, where it has been used, and several useful extensions and variations.

Our method consists of three steps. In the first step, a point cloud representing directly illuminated surfaces is generated. In the second step, the points (surfels) are organized into an octree, and the illumination from each octree node is represented as a single point (surfel) or using spherical harmonics. Rendering is the third step, with the approximate global illumination computed at each shading point using rasterization onto a raster cube.

The advantages of our point-based global illumination method are: fast computation time; noise-free results; global illumination is as fast as ambient occlusion; it handles complex geometry (including dense polygon and subdivision meshes, hair, leaves, and displacement mapping); the original geometric primitives do not have to be kept in memory; no ray-tracing acceleration data structure is needed; only a small fraction of the octree nodes and surfels need to be in memory at any given time; complex light source and surface shaders do not slow it down; environment illumination does not take any additional time.

The disadvantages are mainly that it is a multi-pass approach, and that the results are not guaranteed to be as precise as ray tracing. Our goal is not absolute numerical accuracy, just visually acceptable, consistent, and noise-free results. (However, the method does converge to the correct result as the surfels get smaller and denser and the rasterization resolution is increased.) The method is not suitable for mirror reflection, sharp refraction or sharp shadows; ray tracing is simply better for that purpose.

When comparing render times with ray-traced diffuse and glossy global illumination, keep in mind that the best case for ray tracing is simple scenes with a few large spheres and polygons with smooth color, while the worst case for ray tracing is complex scenes with many trimmed NURBS patches, subdivision surfaces with intricate base meshes, displacement mapping, etc., and high-frequency textures and shadows. In contrast, the point-based approach is immune to complex geometry and textures.

2 Global illumination in movie production

The first algorithms for global illumination were unsuitable for the very complex geometry and shaders used in movie production. Movie rendering requires efficient depth-of-field and motion blur, and absolutely no noise, aliasing, or temporal artifacts. It is also worth keeping in mind that a few images might be sufficient in some applications (e.g. architectural visualization), but more than a hundred thousand high-resolution frames are needed for a feature-length film.

It is only fairly recently that global illumination has been used in feature films. This is a very exciting development! The motivation is that it is easier to get realistic-looking indirect lighting using global illumination algorithms than if the bounce light has to be "faked" with many direct light sources and other tricks. Even though this motivation has been compelling for many years, it is only recently that the algorithms have become sufficiently efficient (and the computers sufficiently powerful) that it is feasible to use global illumination algorithms for movie production.

The honor of being the first feature film to use global illumination goes to "Shrek 2" [19]. Direct illumination was stored as 2D texture maps on the surfaces, and distribution ray tracing (against a coarsely tessellated version of the scene) was used to compute single-bounce global illumination. For efficiency, the distribution ray tracing results are interpolated using irradiance gradients. The same method, as well as a point-based approach similar to the method described in this note, has been used on several other Dreamworks movies.

The Arnold renderer is an unbiased path tracer. The beauty of the unbiased approach is that there are very few quality knobs to tune: the image quality is directly determined by the number of rays being shot. The disadvantage is slow convergence to a noise-free image. Arnold was originally developed by Marcos Fajardo and has been used at Sony on movies such as "Monster House" and "Cloudy With a Chance of Meatballs" [7].

Pixar's RenderMan renderer [20, 1] — also known as PRMan — has two main methods for computing global illumination: distribution ray tracing and a point-based method. (There is also an optional photon mapping pre-pass for multiple bounces, but this is rarely used.) Both methods have been used by our customer studios, but the trend seems to be toward the point-based method since it handles very complex scenes better. The point-based method has been used on more than 35 feature films, including the Pixar movies "Up" and "Toy Story 3".

3 Other related work

Our method is based on Bunnell's point-based GPU global illumination method [2]. His method also uses a hierarchy of colored disks (surfels) to represent the directly illuminated surfaces in the scene. The global illumination is computed by adding up the illumination from all the colored surfels (using aggregate surfels for distant geometry). His method was targeted at real-time rendering on a GPU, but it turns out that the point-based approach is also suitable for rendering much more complex scenes for movies.

Our method is also inspired by clustering radiosity methods [18, 17] and algorithms and data structures used for point-based subsurface scattering [10].

An early version of the method presented in this note was described in the book *Point-Based Graphics* [4]. A more detailed description can be found in the technical memo *Point-based approximate color bleeding* [5], which also has a more in-depth discussion of related work.

Recently, a GPU version of a method very similar to ours has been presented by Ritschel et al. [16]. The main difference is that their method rasterizes the surfels onto a single raster (instead of our cube of six rasters), with directions being transformed to raster pixels by an importance-warping function based on the BRDF. This is efficient and intuitive for glossy reflection, but we believe the single-raster approach is less advantageous for diffuse reflection (and isotropic volume scattering), and would be less efficient than our axis-aligned raster cube for environment map illumination.

4 Generating a direct illumination point cloud

The first step in the point-based method is to generate a point cloud of direct illumination. Each point consists of position, normal, radius, and color, and represents a small disk-shaped part of a surface — a surfel. The color is the direct illumination at that surface position. Figure 1 shows two examples of point clouds. The teapot point cloud on the left has around one thousand points, while the box point cloud on the right has more than half a million points.

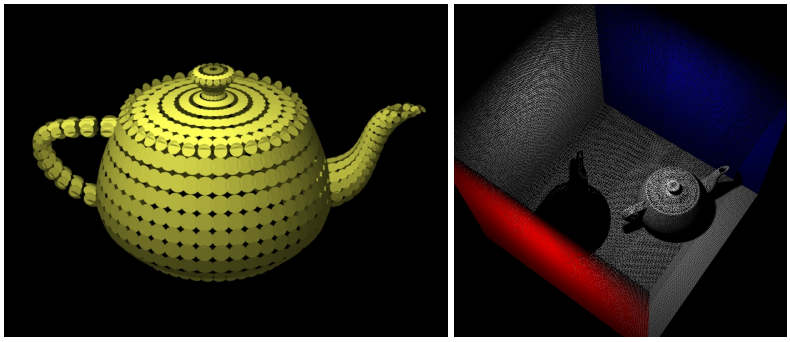

Figure 1: *Examples of point clouds.*

We use the terms "point", "disk", and "surfel" loosely and often interchangeably since each point in the point cloud represents a small disk-shaped surface region.

The point clouds could be generated in any number of ways, but in our implementation, we use PRMan to tessellate the surfaces into small micropolygons, compute the color of each micropolygon, and store the colored micropolygons as surfels. Computing the color of the micropolygons requires evaluation of the direct illumination (including shadows) from all light sources, running surface shaders (including texture map lookups, procedural shading, etc.), and possibly displacement shaders. This step is often referred to as "baking the direct illumination" in TD terminology.

5 Building an octree and computing spherical harmonics

In the second step, the surfels in the point cloud are organized into an octree. The octree nodes are split recursively until each node contains less than e.g. 16 surfels. We then do a bottom-up computation of aggregate power and area in the octree nodes.

If all surfels in a node have similar normals, we compute a single surfel with the average position and normal, and the sum of their power and area [2]. If the surfels have too varying normals, we compute a spherical harmonic representation of the power emitted from (as well as the projected area of) the surfels in that octree node [5]. Like Ramamoorthi and Hanrahan [15] we found that using just the first 9 spherical harmonics gives sufficient accuracy. The spherical harmonic coefficients for a leaf node are the sums of the coefficients for the surfels in that node. For a non-leaf node, the coefficients are simply the sums of the coefficients of its child nodes.

This step is usually by far the fastest of the three.

6 Rendering global illumination using the point cloud

The third step is rendering. In this section we focus on diffuse and glossy global illumination, but many variations and extensions are presented in section 8.

6.1 Rasterization

To compute the global illumination at a point we recursively traverse the octree and rasterize the illumination from each surfel or cluster of surfels. The solid angle (projected area divided by the square distance) of each cluster is used to determine whether it or its children should be used. The main time-vs-accuracy knob of the method is a parameter that specifies the maximum solid angle that is acceptable for using a cluster in place of its children.

In all but the simplest scenes, there will be too much indirect illumination if the contributions from all surfels and clusters are simply added together. Instead, we must ensure that surfels covered by other surfels do not contribute. To do this, we compute a coarse rasterization of the scene as seen from each receiving point. One way to think of this is as a low-resolution fish-eye image of the scene as seen from each receiving point. For efficiency, we use an axis-aligned cube raster since that avoids trigonometric functions and means that we can use the same environment map rasterization for all surface orientations. Figure 2 shows a Cornell box scene with direct illumination (including ray-traced shadows), along with three examples of the rasterization of direct illumination onto the six faces of a raster cube. The top raster cube is for a point on the ceiling of the box while the middle raster cube is for a point on the lid of the right teapot. Gray raster pixels indicate directions below the horizon at that point. The bottom raster

cube is for glossy reflection at a point on the left teapot. A higher raster resolution is required for narrow glossy reflection; in this example we used 12×12 pixels on each cube face for diffuse reflection and 30×30 pixels on each face for glossy reflection.

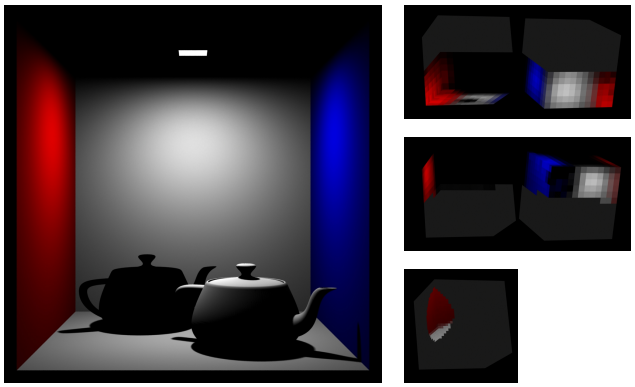

Figure 2: *Rasterization onto six raster cube faces.*

We use three different strategies to rasterize the surfel disks; the appropriate rasterization strategy depends on the distance between the receiving point and the surfel:

- For surfels that are far away from the receiving point, we can just use the octree node representing a cluster of surfels (either a single aggregate surfel or evaluating the spherical harmonics in the direction from the cluster to the receiving point).
- For surfels that are too close to be reasonably approximated by a cluster, but not extremely close, we rasterize each surfel individually.
- For surfels that are very close to the receiving point, we use short-range ray casting. This is much faster than general ray tracing since we only trace against a few disks and their colors are already known.

6.2 Convolution with the BRDF

In order to compute the final global illumination value at the receiving point, we need to convolve the rasterized incident illumination with the BRDF. We do this by looping over all raster pixels and multiplying the rasterized colors with the BRDF for the direction corresponding to that raster pixel, and with the solid angle corresponding to the raster pixel. For diffuse reflection we can disregard all raster pixels below the horizon, and pixels above the horizon are multiplied by the cosine of the pixel direction and the surface normal. For glossy reflection we can disregard all raster pixels where the BRDF is zero, and the remaining pixels are multiplied by the glossy BRDF.

Figure 3 shows an example of point-based global illumination computed with this method. The image shows direct illumination (including ray-traced shadows) plus

glossy global illumination on the left teapot plus diffuse global illumination on all other surfaces. The image was rendered at 1K resolution and took 21 seconds to render on a dual quad-core Mac. (Generating the point cloud took 3 seconds and building the octree also took 3 seconds.) The direct illumination image in figure 2 (left) took 9 seconds to render on the same computer. In this case, the global illumination adds less than 1.5 times the direct illumination render time. If there were more light sources and more complex shaders, the direct illumination would take much longer to compute, while the global illumination would still only add 12 seconds or so.

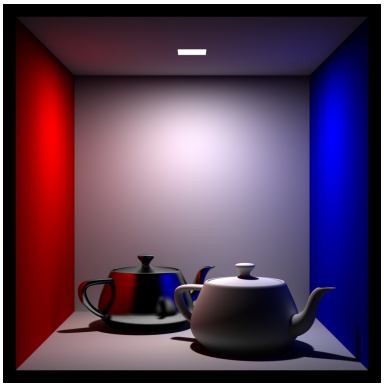

Figure 3: *Point-based diffuse and glossy global illumination.*

6.3 Caching

For point clouds containing a few million surfels, the entire point cloud and octree can be stored in memory. However, for more complex scenes it is necessary to read octree nodes and surfels on demand and cache them. We use two caches: one for groups of surfels, and one for small groups of octree nodes. Both are read from file on demand, cached in memory, and replaced in the cache using a least-recently-used policy.

7 Point-based global illumination in movies

Although the point-based method described in the previous sections is relatively new, it has already been used in the production of quite a few movies.

7.1 Pioneering uses at Sony and ILM

Rene Limberger at Sony Imageworks did the first testing of our prototype implementation of point-based ambient occlusion in 2005. (Ambient occlusion can be computed

with a simplified version of the point-based global illumination method; this is discussed in section 8.5.) Figure 4 shows one of his first test scenes, a pre-production version of an island from the CG movie "Surf's Up".

Figure 4: *Point-based ambient occlusion test from "Surf's Up" © Sony Pictures Imageworks.*

Shortly thereafter, Christophe Hery tested and integrated the point-based global illumination method into the rendering pipeline at ILM. They had found that ray-traced ambient occlusion and global illumination from very dense displaced geometry was too slow, and put the point-based approach to immediate use in the special effects work on the movie "Pirates of the Caribbean: Dead Man's Chest" [8]. The point-based global illumination method was used on Davy Jones and his crew of pirates; figure 5 shows an example.

Figure 5: *Point-based global illumination in a final frame from "Pirates of the Caribbean: Dead Man's Chest" © Disney Enterprises and Jerry Bruckheimer, image courtesy of Industrial Light & Magic.*

7.2 Disney and Pixar

Dale Mayeda at Disney used point-based illumination for special effects such as fire, explosions, and fireworks on the movie "Bolt" [14]. The upper left image in figure 6 shows an early test, with the colored spheres illuminating the ground and characters. The other three images are final movie frames where the point positions are generated by a dynamic particle system.

Figure 6: *Point-based illumination in "Bolt" © Disney Enterprises, images courtesy of Dale Mayeda.*

Pixar's Max Planck used point-based ambient occlusion for some shots on "Wall-E". Jean-Claude Kalache, Peter Sumanaseni, Stefan Gronsky, Michael Sparber and others used point-based global illumination on the majority of the shots on "Up" and "Toy Story 3". Figure 7 shows two point clouds used in the production of "Up"; they are from the living room in Carl's house. The point cloud on the upper left is for the living room illuminated by the key light, and the point cloud on the upper right is for the living room illuminated by fill lights. (The global illumination from each was computed separately in order to provide more flexibility in compositing the final images.) The image on the lower left has no global illumination, while the image on the lower right is the corresponding final movie frame with global illumination.

Figure 8 shows four additional images from "Up"; the images on the left have no global illumination, while the images on the right are final movie frames with global illumination. Notice in particular the sunlight streaming in through the door opening and being reflected off of the floor and onto the wall in the upper right image.

Figure 7: *Point clouds and point-based global illumination from the movie "Up" © Disney and Pixar, images courtesy of Peter Sumanaseni.*

Figure 8: *More point-based global illumination from "Up" © Disney and Pixar, images courtesy of Peter Sumanaseni.*

Figure 9 shows three pairs of images from "Toy Story 3". The upper right image shows indirect illumination, most prominently from the red phone onto Woody's sleeve and hands. The middle right image shows a warm orange glow from the sunlit floor onto the bottom of the blue bean bag. The rail and red box also receive reflected sunlight. In the lower right image, purple color bleeds from Buzz's head onto the inner helmet behind his head, and the bright sun on the baby's forearm is reflected onto its upper arm and sleeve.

Figure 9: *Point-based global illumination from "Toy Story 3" © Disney and Pixar, images courtesy of Stefan Gronsky.*

7.3 Other movies

PRMan's point-based global illumination method is currently integrated into the lighting workflow at ILM, Disney, Pixar, Double Negative, MPC, and several other studios. So far, our point-based global illumination has been used in the production of the following movies: Pirates of the Caribbean: Dead Man's Chest, Eragon, Surf's Up, Spiderman 3, Pirates of the Caribbean: At World's End, Harry Potter and the Order of the Phoenix, The Chronicles of Narnia: Prince Caspian, Fred Claus, Beowulf, The Spiderwick Chronicles, Iron Man, Indiana Jones and the Kingdom of the Crystal Skull, 10,000 BC, Batman: The Dark Knight, Quantum of Solace, Cloverfield, Doomsday, Hellboy 2: The Golden Army, Inkheart, Wall-E, Bolt, Star Trek (2009), Terminator 4, The Boat that Rocked, Fast & Furious 4, Angels and Demons, Night at the Museum 2, Up, Transformers 2, Harry Potter and the Half-Blood Prince, 2012, Sherlock Holmes, Percy Jackson & the Olympians: The Lightning Thief, The Green Zone, Iron Man 2, Prince of Persia: The Sands of Time, and Toy Story 3. (A few of these movies only used point-based ambient occlusion.)

8 Variations and extensions

This section describes several useful variations and extensions to the method: area light sources and soft shadows, environment illumination, multiple diffuse bounces, final gathering for photon mapping, ambient occlusion, reflection occlusion, and global illumination in volumes. We also mention additional result types that can be generated with the method.

8.1 Area light sources and soft shadows

As Bunnell [2] pointed out, diffuse area light sources can be handled by simply treating them as brightly illuminated surfaces. With this approach, the area lights can have arbitrary shapes and color variations across their surface — just as regular surfaces can. The only requirement is that a point cloud must be generated for the light sources and shadow casters.

Figure 10 shows a scene with three area light sources: a teapot light source with a procedural checkerboard texture, a sphere light source with a displacement map and a texture map, and a dragon light source that is also texture-mapped. The area lights illuminate three objects: a dragon, a sphere, and the ground plane. Note the soft shadows on the ground plane. (To avoid over-saturation that makes the textures hard to see, the light source objects are rendered dimmer here than the intensity used for generating their emission point cloud.)

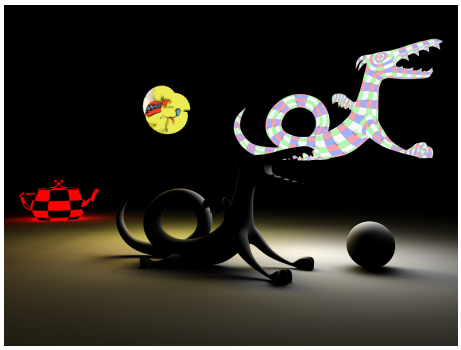

Figure 10: *Point-based area lights.*

As mentioned in section 7, figure 6 shows more examples of point-based area light sources.

8.2 Environment illumination

Environment illumination from a standard or high-dynamic range image is easily added to the method. First a filtered environment map lookup is done for each direction and solid angle corresponding to a raster pixel. With the axis-aligned cube rasterization approach, this can be done once and for all for each environment map, so comes at negligible run-time cost. Then, during the rasterization for each receiving point, the rasterization simply renders disk colors "over" the environment map colors. The result is that the environment illuminates the receiving point from the directions that are unblocked by geometry.

Figure 11 shows environment illumination on a dragon. The upper left image shows an HDRI environment map with four areas of bright illumination: two white, one blue, and one green. The lower left image shows a raster cube with environment map colors as seen from a point on the dragon's back. The right image shows the dragon scene illuminated by the environment map; this image took 11 seconds to render at 1K resolution.

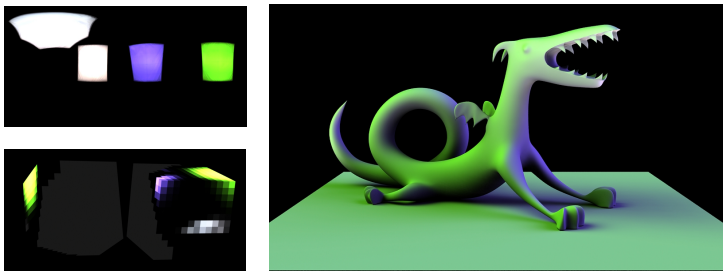

Figure 11: *Point-based environment illumination: HDRI environment map (dimmed for display), rasterized environment map, and dragon illuminated by the environment map.*

Note that this method will blur high frequencies in the environment map. The raster resolution can be increased if finer illumination details are desired.

8.3 Multiple diffuse bounces

A simple extension allows the method to compute multiple bounces of diffuse reflection. For this variation, it is necessary to store surface reflection coefficients (diffuse color) along with the direct illumination and area of each surfel. Then we can simply iterate the algorithm, updating the radiosity of the surfels in the point cloud in each iteration. It is often sufficient to do these multibounce computations on a point cloud with significantly fewer surfels than what is used for the final rendering.

Figure 12 shows a comparison of zero, one, and two bounces in a Cornell box scene. As expected, the global illumination gets brighter with more bounces.

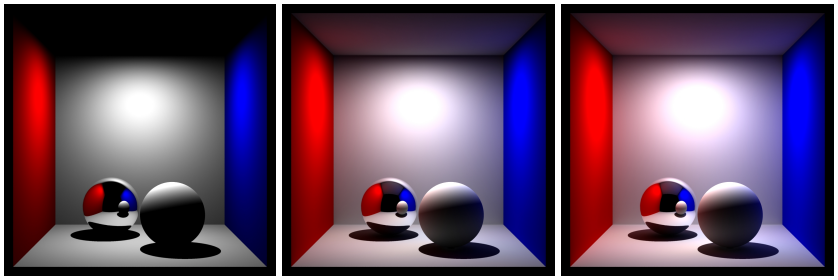

Figure 12: *Cornell box with spheres: direct illumination, 1 bounce, and 2 bounces.*

8.4 Final gathering for photon mapping

Another application of this method is for photon mapping [9] with precomputed radiance and area estimates [3]. The most time-consuming part of photon mapping is the ray-traced final gathering step needed to compute high-quality images. The point-based global illumination method presented here can be used directly to speed up the final gathering step. In this application, the points in the point cloud represent coarse global illumination computed with the photon mapping method (instead of direct illumination). Figure 13 shows a box with two teapots (one specular and one displacement-

Figure 13: *Textured box with displacement-mapped teapot: direct illumination only, photon map, radiance estimates, and point-based global illumination.*

mapped diffuse), three light sources, texture maps, and procedural displacement. The upper left image shows the direct illumination alone. It took 8 seconds to render this image at 1K resolution. The upper right image shows the photon map for this scene, with the photon powers shown. The photon map contains 2.6 million photons and took 38 seconds to generate (on a single core). The lower left image shows the radiance estimates at the photon positions. The estimates were computed from 200 photons each, and took 12 seconds to compute. The lower right shows the final image, where point-based global illumination was used instead of the usual ray-traced final gathering. (Shadows and the reflections in the chrome teapot were computed with ray tracing.) This image took 65 seconds to render at 1K resolution.

8.5 Ambient occlusion and reflection occlusion

Ambient occlusion [22, 13] is a representation of how much of the hemisphere above each point is covered by geometry. Ambient occlusion is widely used in movie production since it is relatively fast to compute and gives an intuitive indication of curvature and spatial relationships. It is often computed with ray tracing: a number of rays are shot from each shading point, sampling the "coverage" above that point. But we can compute ambient occlusion with a point-based approach instead.

Point-based ambient occlusion is a simplified version of the point-based global illumination algorithm. There is no need to store direct illumination colors in the point cloud; only the surfel areas are needed. Only the alpha channel is used in the rasterization phase. Also, there is no need to sort according to distance since near-vs-far does not matter for occlusion. Figure 14 (left) shows point-based ambient occlusion on a NURBS car. The point cloud used for the computation contains 4.7 million surfels.

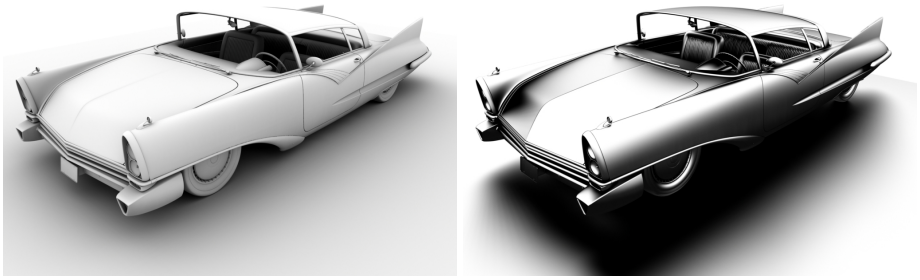

Figure 14: *Point-based ambient occlusion and reflection occlusion.*

As mentioned earlier, figure 4 shows another example of point-based ambient occlusion.

Reflection occlusion [13] is similar to ambient occlusion but for a narrow cone of directions. In ray-traced rendering it is sometimes used as a cheap alternative to glossy

reflection. (For the point-based method there is no significant difference in the time to compute glossy reflection or the corresponding reflection occlusion.) Figure 14 (right) shows an example of point-based reflection occlusion.

8.6 Global illumination in volumes

Points in volume point clouds do not have normals; each point represents a sphere in 3D space rather than a disk on a 2D surface. When computing volume scattering, the light comes from all directions so all six raster cube faces can receive light.

If the volume is inhomogeneous (i.e. has spatially varying extinction), the spheres and clusters need to be sorted (front-to-back or back-to-front) before rasterization in order to calculate the correct extinction.

The global illumination can be between surface points and volume points or between volume points. Figure 15 (left) shows illumination from surfaces to a volume, and figure 15 (right) shows illumination from a volume to another volume.

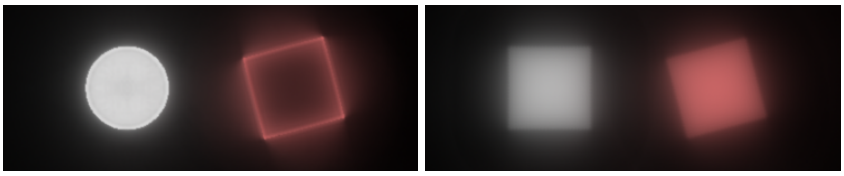

Figure 15: *Point-based global illumination in a volume.*

8.7 Other result types

Given the raster cube of occlusion and illumination colors for a point, it is really fast to compute other quantities as well. For example:

- The average unoccluded direction, also known as "bent normal". This is often used for an environment map lookup to give a coarse approximation of the environment illumination. (It should be pointed out that the raster-based environment illumination described in section 8.2 is more accurate and just as fast.)
- The average illumination direction. This is a weighted average of the unoccluded directions, with the weights being the brightness of an environment map. This can be useful for fast, approximate relighting.
- Spherical harmonics coefficients that represent the directional visibility (unocclusion). These coefficients can quickly be multiplied with spherical harmonic coefficients for an environment map, giving the approximate environment illumination at a surface point. Such image-based lighting can be used for fast illumination calculations [15].

9 Irradiance caching, gradients, and interpolation

Irradiance caching with irradiance gradients [21, 19, 12] is a very useful technique to speed up ray-traced and point-based global illumination calculations. It places more samples in regions with large variation in indirect illumination, and few samples where the indirect illumination is smooth.

Our approach is a bit different from standard irradiance caching [21] since we use the Reyes rendering algorithm [6] instead of tracing rays from the camera. The Reyes algorithm divides surfaces into patches, tessellates each patch into a grid of micropolygons, and then computes the colors at the micropolygon vertices (shading points). We first compute (or look up) the indirect illumination and gradients at the four corners of the grid. If the differences between the four values are consistent with their gradients, we can simply do a biquadratic interpolation across the grid. If a difference is too large, then we split the patch into two subpatches, compute the indirect illumination and gradients at the two split points, and recurse. (The same technique also works for triangular grids.) Figure 16 shows a tessellated surface patch and a case where the irradiance can be safely interpolated as well as case where it cannot.

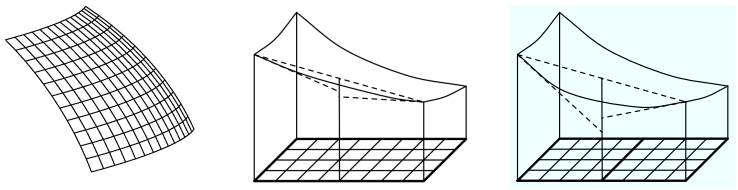

Figure 16: *Tessellated surface patch and interpolation of global illumination.*

We have implemented this technique within our ray-traced and point-based global illumination algorithms. Since the point-based irradiance computations are so fast, the relative speed-ups are more modest for point-based than for ray-traced computations. We usually see about 2 times speed-ups in typical scenes.

We compute the point-based irradiance gradients in a rather crude manner: the raster cube of incident radiances is resampled into a hemispherical raster which is then used for Ward and Heckbert's irradiance gradient formulas [21]. (We have empirically found that it is sufficient to use a hemisphere raster resolution that is the same resolution as a single raster cube face.) It would probably be more efficient to compute the gradients directly (without the intermediate hemispherical raster) using the irradiance gradient formulas of Gautron [11].

10 Conclusion

Point-based global illumination is fast and robust and can handle complex movie production scenes. The technique also works for area lights with soft shadows, environment map illumination, multiple diffuse bounces, final gathering for photon mapping, ambient occlusion, reflection occlusion, and volumes. The technique is implemented in Pixar's RenderMan renderer and is widely used in movie production.

As far as I know, no movie has used the multi-bounce global illumination or volume capabilities, so that remains an interesting area of future exploration.

Acknowledgements

First of all, I would like to thank Dana Batali and my colleagues in Pixar's RenderMan Products group for their help and support in this project.

Thanks to Michael Bunnell for developing and publishing the original GPU-based method. Without his chapter in *GPU Gems 2* I would not have gotten started on the work described in this course note.

Thanks to Rene Limberger at Sony Imageworks who prompted me to start developing this method and tested early prototypes, and to Christophe Hery at ILM who was brave enough to push this technique into movie production while the code was still in development.

Also thanks to Max Planck, Peter Sumanaseni, Jean-Claude Kalache, Stefan Gronsky and Guido Quaroni at Pixar, Dale Mayeda at Disney, Philippe Leprince at Double Negative, Anders Langlands at MPC, and all our other "power users" for putting this technology to good use.

References

[1] Anthony A. Apodaca and Larry Gritz. *Advanced RenderMan: Creating CGI for Motion Pictures*. Morgan Kaufmann Publishers, 2000.

[2] Michael Bunnell. Dynamic ambient occlusion and indirect lighting. In Matt Pharr, editor, *GPU Gems 2*, pages 223–233. Addison-Wesley Publishers, 2005.

[3] Per H. Christensen. Faster photon map global illumination. *Journal of Graphics Tools*, 4(3):1–10, 1999.

[4] Per H. Christensen. Point clouds and brick maps for movie production. In Markus Gross and Hanspeter Pfister, editors, *Point-Based Graphics*, chapter 8.4. Morgan Kaufmann Publishers, 2007.

[5] Per H. Christensen. Point-based approximate color bleeding. Technical Report #08-01, Pixar Animation Studios, 2008.

[6] Robert L. Cook, Loren Carpenter, and Edwin Catmull. The Reyes image rendering architecture. *Computer Graphics (Proc. SIGGRAPH 87)*, 21(4):95–102, 1987.

[7] Larry Gritz. Production perspectives on high performance graphics. *High Performance Graphics Conference*, 2009. Keynote talk, available on-line at www.larrygritz.com/docs/HPG2009-Gritz-Keynote-clean.pdf.

[8] Christophe Hery. Shades of Davy Jones. Interview available on-line at features.-cgsociety.org/story_custom.php?story_id=3889, 2006.

[9] Henrik Wann Jensen. Global illumination using photon maps. In *Rendering Techniques '96 (Proc. 7th Eurographics Workshop on Rendering)*, pages 21–30, 1996.

[10] Henrik Wann Jensen and Juan Buhler. A rapid hierarchical rendering technique for translucent materials. *ACM Transactions on Graphics (Proc. SIGGRAPH 2002)*, 21(3):576–581, 2002.

[11] Jaroslav Křivánek and Pascal Gautron. *Practical Global Illumination with Irradiance Caching*. Morgan & Claypool Publishers, 2009.

[12] Jaroslav Křivánek, Pascal Gautron, Greg Ward, Henrik Wann Jensen, Eric Tabellion, and Per H. Christensen. *SIGGRAPH 2008 Course Note #16: Practical Global Illumination with Irradiance Caching*, 2008.

[13] Hayden Landis. Production-ready global illumination. In *SIGGRAPH 2002 Course Note #16: RenderMan in Production*, pages 87–102, 2002.

[14] Dale Mayeda. Interactive lighting of effects using point clouds in 'Bolt'. *SIGGRAPH 2009 talk*, 2009.

[15] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance maps. *Computer Graphics (Proc. SIGGRAPH 2001)*, 35:497–500, 2001.

[16] Tobias Ritschel, Thomas Engelhardt, Thorsten Grosch, Hans-Peter Seidel, Jan Kautz, and Carsten Dachsbacher. Micro-rendering for scalable, parallel final gathering. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2009)*, 28(5):132:1–132:8, 2009.

[17] François X. Sillion. A unified hierarchical algorithm for global illumination with scattering volumes and object clusters. *IEEE Transactions on Visualization and Computer Graphics*, 1(3):240–254, 1995.

[18] Brian Smits, James Arvo, and Donald Greenberg. A clustering algorithm for radiosity in complex environments. *Computer Graphics (Proc. SIGGRAPH 94)*, 28:435–442, 1994.

[19] Eric Tabellion and Arnauld Lamorlette. An approximate global illumination system for computer generated films. *ACM Transactions on Graphics (Proc. SIGGRAPH 2004)*, 23(3):469–476, 2004.

[20] Steve Upstill. *The RenderMan Companion*. Addison-Wesley Publishers, 1990.

[21] Gregory J. Ward and Paul S. Heckbert. Irradiance gradients. In *Proc. 3rd Eurographics Workshop on Rendering*, pages 85–98, 1992.

[22] Sergei Zhukov, Andrei Iones, and Gregorij Kronin. An ambient light illumination model. In *Rendering Techniques '98 (Proc. 9th Eurographics Workshop on Rendering)*, pages 45–55, 1998.