Adaptive Representation of Specular Light Flux

Normand Brière Pierre Poulin

Département d'informatique et de recherche opérationnelle
Université de Montréal

Abstract

Caustics produce beautiful and intriguing illumination patterns. However, their complex behavior make them difficult to simulate accurately in all but the simplest configurations. To capture their appearance, we present an adaptive approach based upon light beams. The coherence between light rays forming a light beam greatly reduces the number of samples required for precise illumination reconstruction. The light beams characterize the distribution of light due to interactions with specular surfaces (specular light flux) in 3D space, thus allowing for the treatment of illumination within single-scattering participating media. The hierarchical structure enclosing the light beams possesses inherent properties to detect efficiently every light beam reaching any 3D point, to adapt itself according to illumination effects in the final image, and to reduce memory consumption via caching.

Keywords: caustics, global illumination, ray tracing, wavefront, beam, shaft.

1 Introduction

Illumination defines how surfaces appear in an image, and therefore is considered an important area of image synthesis. Unfortunately, simulating in global illumination the multiple interactions of light in a general context represents a complex task to solve accurately and efficiently. This explains why so much research has been devoted to illumination.

Global illumination algorithms can be loosely separated in two categories: radiosity and path tracing. Radiosity techniques [6, 26] capture the infinite interreflections in diffuse polygonal environments. With recent advances in discontinuity meshing and clustering, they can handle larger and larger scenes. However these techniques mainly target diffuse interreflections, and they typically become much more expensive when extended to directional (specular) interreflections [24, 4].

Monte Carlo Path Tracing [22, 23] and bi-directional [27, 17] techniques solve the light transport problem for any geometry and any surface property. These sampling-oriented approaches compute the radiance along rays in the environment. While they can produce stunning results, they often exhibit noise due to insufficient sampling. Specular interactions are better handled because rays are more narrowly directed; diffuse interactions on the other hand may require many more rays to reduce noise. Although several efficient variance reduction techniques have been proposed [28], the coherence between adjacent samples is not guaranteed, which makes the reconstruction of the illumination prone to errors that appear in the form of noise.

Two-pass techniques [29, 25, 3, 32] have been introduced to exploit the best of both worlds. However while noise can be reduced significantly, it is still present in singular situations, such as small light sources and mirrors.

In this paper, we concentrate our efforts on capturing light emitted from point light sources and distributed within a scene via perfectly specular surfaces (that is the distribution of specular light flux). These effects are often associated with caustics.

1.1 Coherent Light Beams

In order to accurately simulate the light specular distribution, we construct a set of light beams. The concept of coherent 3D beams is not new in image synthesis. Heckbert and Hanrahan [12] use them to represent *exact* visibility after perfect reflections off polygonal surfaces (refractions are treated approximately). Each beam is clipped by any polygonal blocker. They propose to trace beams from a point light source to deposit illumination from the unclipped beams as *shading polygons* on diffuse polygonal surfaces for efficient rendering of indirect specular illumination.

This approach has been successfully applied by Watt [31] on a fine triangular mesh defining a surface of water. The light reaching a specular triangle refracts and settles onto diffuse polygons as a shading triangle. The resulting illumination on the shading triangle is computed as the irradiance on the specular triangle weighted by the ratio of the area of both triangles. To simulate the effect due to the presence of uniform participating media, *light beams* may be defined by linking a pair of specular and shading triangles, treating the attenuation along each light

beam. Chuang and Cheng [5] can handle non-polygonal illuminated surfaces by searching the light beams within which any intersection point resides. Their light beams are enclosed in a hierarchy of bounding cones for more efficient point-in-beam detection. No participating media is treated. Instead of performing a scanconversion of the shading triangles, Nishita and Nakamae [19] also used light beams to treat uniform density participating media. Their triangular light beams originate from a premeshed ocean surface. They intersect all light beams with a plane for each scanline, and accumulate the contribution of each shading triangle. This results in an efficient rendering of underwater scenes provided that the camera lies under the ocean surface.

The light beams in the previous techniques are constructed from a set of pre-meshed surfaces. Their subdivisions must first be set to the desired accuracy. As caustics are highly localized effects, it is not obvious to estimate an appropriate level of detail. Also, these techniques can not in their current form handle caustics resulting from multiple interreflections and interrefractions.

An attractive approach samples the illumination by tracing photons from the light sources, and deposits them onto surfaces and/or in volumes. If only energy is associated to each photon [1, 11, 15, 30], reconstructing the illumination corresponds to solving a density estimation problem. By adding wavefront information to each photon [21, 7, 8], irradiance at any point along its path can be computed. Reconstructing the illumination then requires less samples in the filtering process. Another lower sampling solution is possible when considering adjacency information between the origins of the samples. Collins [7] exploited this adjacency information in order to determine the resolution of illumination maps associated with diffuse surfaces. Adjacency was used to estimate the wavefront on the intersected surface, and the kernel size and shape to distribute energy onto the elements of the illumination map on the surface. Only planar diffuse surfaces and no participating media were assumed.

Photon tracing techniques have proven to be both general and efficient, while capable of capturing many of the most complex illumination patterns. However, it is difficult to determine when the distribution of photons is sufficient for accurate illumination evaluation, how many of these photons to include in the filter kernel, and which photons really contribute within this kernel. This leads to possible illumination artifacts such as noise if the kernel is too small, or blur otherwise.

As suggested by researchers [14, 15, 16], one can gain in efficiency by replacing some of its generality to capture specific phenomena. In this paper, we focus on one such phenomenon: caustics.

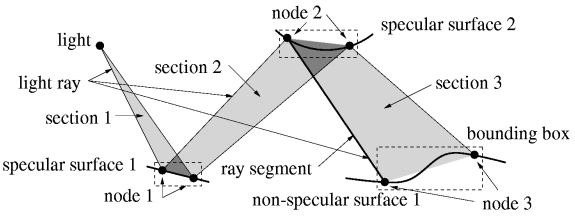

Figure 1: A 2D representation of a light beam.

Mitchell and Hanrahan [18] describe an accurate solution for solving this particular situation. Instead of sampling the illumination from the light source, they determine all light rays (Fermat critical paths) reaching any desired point in a scene. However this process requires a priori knowledge of interacting specular surfaces to avoid considering exponentially growing potential combinations.

To bypass this problem, we propose to calculate an approximation of the illumination produced by the specular flux rather than an exact characterization. We present in Section 2 an adaptive subdivision scheme to construct light beams issued from a point light source as samples of its specular distribution. It can handle any type of geometry (conics, bicubic patches, etc.) and generates coherent light beams from an arbitrary number of reflections and refractions. In Section 3, we explain how the flux density can be extracted from the light beams, and introduce extensions to participating media. Section 4 presents some techniques developed to reduce both memory and computing requirements. Finally extensions are discussed in Section 5, and we conclude by revisiting the achievements of our approach.

2 Light Beams

A light ray is traced from a point light source. It reflects or refracts onto various perfectly specular surfaces until it reaches a non-specular surface. The flux density at any point along this ray is a function of the propagated wavefront [21, 18, 7]. Infinitely many such rays would characterize the entire specular distribution of light in the environment.

In order to sample the light flux, we encode adjacency information in the form of *light beams*. A light beam is issued from a point light source and goes through a solid angle corresponding to three points on the *light image* (image as seen from the light source, illustrated in Fig. 10 (center)). The three *supporting light rays* are traced through the scene onto specular reflective or refractive surfaces until they reach a non-specular surface. At each interaction with a surface, the three intersection points are linked into a *node*, and this node is connected

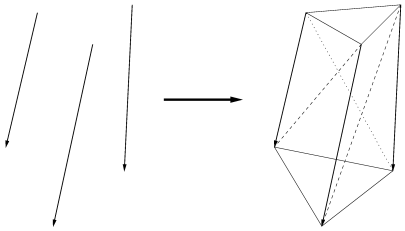

Figure 2: Supporting ray segments and corresponding light beam section.

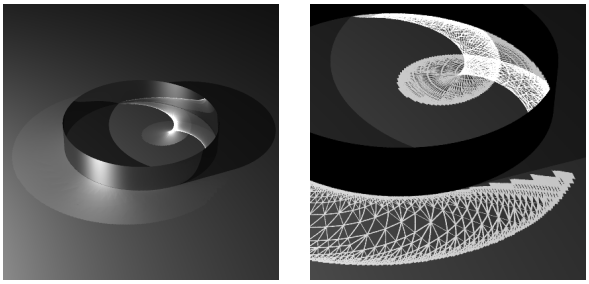

Figure 3: Left: caustic produced by a ring; Right: the corresponding mesh formed by the intersections of the reflected light beams with the floor.

to the previous node by ray segments forming a *section* of the light beam (Fig. 2). A section is built as a polyhedron with eight triangles. A light beam therefore consists of a linked list of triangular based sections. Fig. 1 illustrates (in 2D) elements of such a structure. Fig. 10 (left) illustrates it in an actual 3D scene. Fig. 3 shows the familiar caustic (cardioid) within a ring and a dimmed reflection outside the ring. The mesh is obtained by intersecting the light beams with the floor.

At this point, our light propagation process corresponds to the grid-pencil tracing described by Shinya *et al.* [20]. We therefore inherit advantages of their approach. To avoid high memory requirements, Shinya *et al.* deposit onto the polygonal surfaces the illuminance polygons carried by the light beams. Fortunately, by enclosing the light beams within a hierarchical structure, we will be able to use all information they provide thanks to a caching scheme. A volumetric representation has interesting properties and benefits such as the ability to treat participating media but also, to handle rough or bumpy illuminated surfaces.

2.1 Adaptive Refinements

We would like to keep the light distribution within a light beam as *coherent* as possible in order to approximate the light flux within a light beam only from the supporting rays. To ensure an appropriate distribution of light

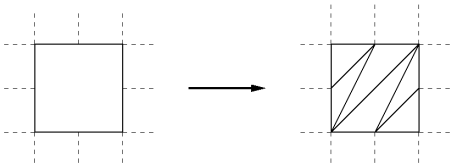

Figure 4: Quadtree element with four T-vertices and corresponding triangularization.

beams, one can use a high basic subdivision for the light image, resulting in finer beams, and rely afterwards on re-unification of the light beams considered well enough approximated at a coarser level. This scheme based on uniform subdivision and adaptive re-unification provides a beam distribution which misses fewer features than purely adaptive subdivision for the same maximum subdivision level.

An adaptive subdivision of the light image is however more efficient. It can be performed according to various criteria such as the ordered list of objects encountered along the light rays defining a light beam (object paths), their directions, and the variation of the flux density (wavefront) along the light rays. By forcing a maximum difference of one subdivision level between neighbor light beams (balanced quadtree), it is easier to construct well behaved triangular light beams from quadtree elements since they are defined with at most four T-vertices (Fig. 4).

The center image of Fig. 10 shows the resulting balanced subdivision (started at a resolution of 8×8) when applying the criteria of object paths. In this case, no features have been missed and the resulting light beams represent our approximation of the specular light flux.

Adaptive subdivision schemes unfortunately do not guarantee that the light flux within a light beam is fully characterized by its supporting rays. Indeed for non-constant curvature surfaces, light may reflect outside of its beam. If these situations are detected, they should force a refinement. As a light beam becomes thinner, it better characterizes its light flux since its supporting rays usually converge to a single light ray.

2.2 Light Specular Distribution

At the end of this preprocessing step, we have a volumetric representation of the light flux via specular transfers in the scene. The light beams are themselves polyhedral, but because the supporting rays are defined by ray tracing (actually pencil tracing [21]), they can handle arbitrary reflection/refraction combinations for any type of primitive without requiring any surface meshing. This representation can be adapted according to discontinuities

within the light flux, the light distribution within the light beams, and the participating media traversed.

Each point light within the scene can require up to six light images, defining a cube around the source. For a point light source distant with respect to the scene, the user can orient the light image according to the specular surfaces. The various parameters controling adaptive subdivision can be specified by the user.

3 Flux Density Computing

With a distribution of light beams, one can estimate the light flux at any 3D point P in the scene. Suppose P lies within a section of a light beam, and the light flux passing through the light beam is coherent. The flux density at P can be approximated from the information provided by the three supporting rays. The construction is illustrated in 2D in Fig. 5. We build a plane intersecting P and oriented along the average of the three supporting ray segments. The wavefronts are evaluated at the three intersection points P_i of the rays with this plane, and a barycentric interpolation of these wavefronts is assigned as the wavefront at P.

For some surfaces such as those modeled with a bump map, the wavefronts are not exact because a bump mapped surface does not correspond to a real surface. However they can provide a good approximation when the bump map has a low C^2 amplitude. Igehy [13] proposes to use the more general ray differentials to deal with such surfaces.

Another approximation of the flux density at P [31] is $E_P = I\Delta\omega/\Delta A$, where I corresponds to the intensity of the point light, $\Delta\omega$ to the solid angle at the light subtended by the light beam, and ΔA to the area of the triangle defined by the plane and the three ray segments. If P lies on a surface, the flux density E_P is multiplied by $\cos\theta$, where θ is the angle between the interpolated ray and the surface normal at P. The wavefront and triangle area approximations converge to the same result as the light beams are subdivided more finely. The triangle approximation can be used in all situations, but results in a more *blocky* appearance because no illumination is shared between adjacent illumination triangles. The wavefront approximation usually converges faster for it provides a continuous (C^0, but not C^1) illumination reconstruction.

Fig. 6 shows the main caustic produced by a glass sphere. The caustic in the image on the left is too small to observe the variations within the caustic region. The image on the right focuses on the caustic illumination. Colors have been scaled down in order to show the details in this small region. We used triangle areas to approximate the flux density. Fig. 7 shows the illumination pattern created by a point light source placed under a reflective bump mapped sphere. The direct illumination has not been not computed to display only the reflected illumination on the floor. The image on the left uses the triangle area approximation; the image on the right uses the wavefront barycentric interpolation.

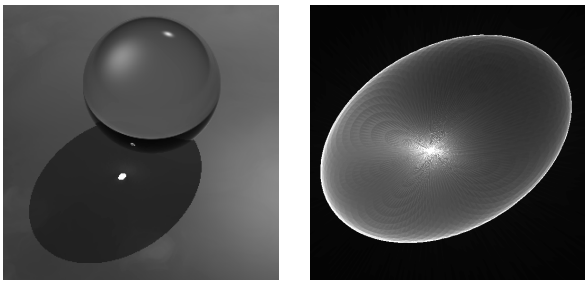

Figure 6: Caustic under a glass sphere and a zoom on the illumination.

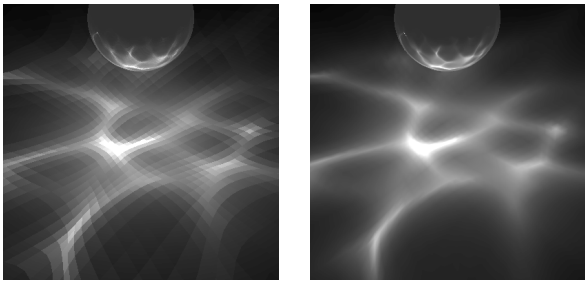

Figure 7: Reflection under a bump mapped sphere. Left: triangle area; Right: interpolated wavefront.

3.1 Shadowing

Both techniques provide an approximation of the flux density at P *if* this light actually reaches P. However smaller objects may lie between the supporting rays without intersecting them. For more accurate shadowing during the rendering phase, the Fermat critical path is approximated by simply interpolating a light ray from the supporting light rays (Fig. 5). The shadow ray segment within a section uses the same barycentric weights at both adjacent nodes as we did for the interpolated wavefront. The resulting shadow ray is traced back to the light source through the specular interactions, and tested for occlusion with the entire scene.

On a concave shape, light beams might not reach the interior of its surface. To reduce illumination problems on concave shapes, each section of each light beam is extended in length to enclose as much as possible the intersected surface(s) (Fig. 8). The longer the extension, the less efficient the structure becomes because a point can then fall in more light beams. By tracing a shadow ray, we also make sure the illumination is properly reaching

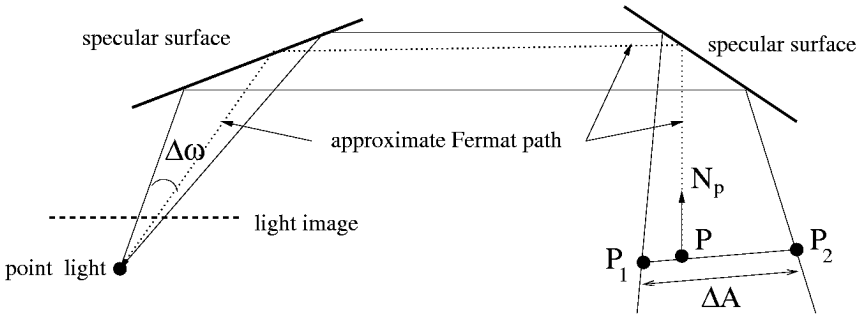

Figure 5: Computing the flux density at P.

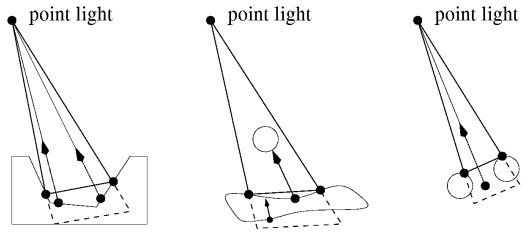

Figure 8: Extending a light beam section and tracing back shadow rays.

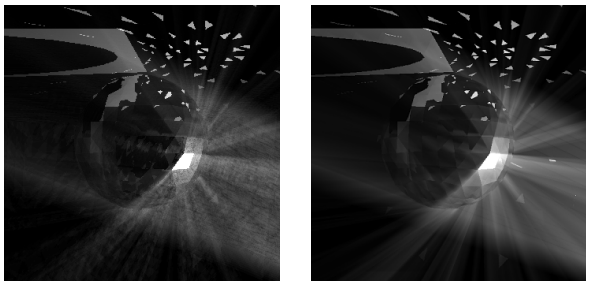

Figure 9: Disco era memories: sampled and analytical.

P within the extended sections. Fortunately, these tests can be neglected in many situations (flat and convex surfaces) when the light beams are more finely subdivided than the size of objects in the scene.

3.2 Shading

The flux density at P and its interpolated incident direction are inserted in the local reflection model to determine the light reaching the virtual camera. Therefore non-specular surfaces can have any reflection model. The total light contribution from P issued from all specular interactions is computed by summing up the contributions from all the light beams P lies within. Fig. 12 shows illumination produced by caustics reflected on a galleon and refracted within a pool of sharks.

The 3D nature of the light beams can take into account participating media attenuation and shading under the assumption of single scattering. Accumulated attenuation is stored at each intersection point of supporting light rays. One therefore needs only to compute the attenuation in the current section rather than along all the sections back to the point light source. The shadow ray is traced only in the current section, and its attenuation at the beginning of this section is interpolated from the attenuations of the supporting light rays. The storage required by the light beams is lower than the volumetric photons of Jensen and Christensen [16], although their technique allows for multiple scattering in participating media.

Fig. 9 shows a disco sphere in uniform density participating media. The image of the left shows a rendering using 100 sample points along eye rays. In this case, noise is still noticeable. However, if we compute the entry and exit points of eye rays with light beam sections, the shading equations resulting from uniform density media simplify, leading to a smoother rendering, as illustrated at the right. The direct illumination in Fig. 11 (center) is not included to display the fine light beams reflected in the uniform density media. Note that these light beams are also visible via reflection on the polygonized disco sphere. Fig. 11 (left) shows an image where the attenuation of the light due to non-uniform participating media is approximated by 10 samples along shadow ray segments.

4 Efficiency

Light beams offer various advantages to render specular indirect illumination. However it is critical to determine efficiently the light beams a point P lies into. We organize the light beams into a hierarchical structure to provide an efficient point-in-beam determination. Also, this hierarchical structure allows efficient caching of its ele-

ments. Therefore only a fraction of the total structure needs to be kept in memory at any given time. We detail in the following sections the advantages obtained with this hierarchical structure.

4.1 Hierarchical Point-in-beam

A typical scene with specular surfaces can result in thousands to millions of light beams per point light source. This requires an efficient hierarchical structure to identify the light beams enclosing a point P, which we refer to as *point-in-beam query*.

The intersection points of supporting rays forming the nodes of adjacent light beams are joined by a bounding box oriented along a global coordinate system. Two bounding boxes of consecutive nodes form a shaft [10] which corresponds to their convex hull. Adjacent light beams are enclosed by those shafts, and shafts are themselves organized into a hierarchy of shafts. The bounding boxes form as well an equivalent hierarchy. Similar structures have been used with success in previous applications [2, 9].

While shafts can give some indication about the light distribution, it is really in the enclosed light beams, down at the leaves of the hierarchy, that the flux is encoded. We use the hierarchy mainly to cull efficiently irrelevant light beams in order to determine a point-in-beam query logarithmically in the number of light beams.

4.2 Efficient Memory Management

The hierarchical nature of the structure also allows to free much of its information that can be recomputed when required. By storing only the ordered list of intersected objects (object path) for each light ray, we can rebuild efficiently and exactly any portion of the hierarchy. Because many point-in-beam queries often occur within the same 3D region, the localized behavior of caching the more recently requested portions of the hierarchy allows for efficient memory management.

Table 1 shows the impact on rendering time of reducing the memory requirements relative to keeping all structures (100%) in memory. We used different image resolutions of the sharks scene (Fig. 12 (right)). All light object paths as well as the entire bounding box hierarchy of the nodes are kept in memory. For reference, in our implementation, the data structure of a bounding box is about 10 times smaller than the data structure of a shaft joining two bounding boxes because of the various plane equations defining the shaft itself. Recomputing a light shaft corresponds to building it back from the bounding box hierarchy. Recomputing a light beam corresponds to retracing three light rays from their object paths to intersect the proper objects.

At a resolution of 256×256, keeping 10% of all light shafts requires only 1.06 more time than keeping everything, while keeping 10% of the light beams requires 1.25 more time. One can observe from this table that for this typical scene, keeping as little as 1% of the structures for both light beams and light shafts is still reasonable with respect to the rendering time. Keeping none (0%) of the structures means that every light ray is recomputed from its object path and every light shaft from its bounding boxes. In this case, rendering times only slightly increase with the resolution. In all other cases, caching efficiency increases with image resolution as more localized and coherent calculations are performed. This process allows us to construct more detailed light beams of complex light interactions within the same amount of memory space.

4.3 Wavelength Dependency

To capture wavelength-dependent phenomena, the light beam structure is extended to wavelengths. When a light beam traverses space where no interactions depend on the wavelength of the light carried within, the light beam remains the same as previously described. However when a light beam interacts with a refractive surface for instance, we construct new sections for a number of sampled wavelengths according to the spectrum of the light in the beam. The reconstructed shading at a point is then dependent upon the wavelength of all light beams it lies in. Fig. 11 (right) shows a crystal pyramid and the colored illumination (due to refraction) rendered at three sampled wavelengths. The *dispersion image solid* of Shinya *et al.* [20] would be applicable for more precise results.

4.4 Results

Table 2 gives some statistics for rendering the images in this paper. All renderings were computed on an SGI R10000 Impact at 195 MHz with 128 MB. The light image preprocessing lists the minimum and maximum resolutions of the adaptive subdivision, the number of light beams created, and the complete preprocessing time in seconds. The final rendering times of the images are given in minutes in the rightmost column. All renderings to gather the statistics were computed at a 512×512 resolution, 1 sample per pixel. All structures were kept in memory for the rendering of the entire image.

An important general observation about the light image preprocessing is that tracing the light rays and building the entire hierarchical structure represent typically a small fraction of the image rendering time. This indicates that we could continue to improve on the light beams distribution and therefore on the illumination details without much impact on the total rendering time. This favors a more sophisticated design for adaptive subdivision criteria and re-unification strategies.

Figure : Description	Light Image Preprocessing Resolution Min	Resolution Max	Beams (×1,000)	Time (secs)	Rendering Time (mins)
3 left : Ring	16×16	1024×1024	18	8	4
6 left : Glass sphere	128×128	128×128	16	47	63
6 right : Glass sphere: zoom					67
7 left : Bumped sphere: triangles	64×64	64×64	4	17	19
7 right : Bumped sphere: wavefronts					24
9 left : Disco sphere: 100 samples	16×16	256×256	23	50	3,200
9 right : Disco sphere: analytical					1,380
10 right : Mirrors	8×8	512×512	23	23	16
11 left : Smoke on cube: 10 samples	16×16	256×256	7.5	4	312
11 right : Refraction from pyramid	16×16	256×256	45	540	90
12 left : Galleon	256×256	256×256	65	840	132
12 right : Sharks	256×256	1024×1024	144	1020	120

Table 2: Statistics on some images displayed in this paper.

Image Resolution	Timings relative to full memory 0%	1%	10%
64×64	5.67 / 2.33	2.28 / 1.75	1.61 / 1.44
128×128	5.82 / 2.39	1.80 / 1.56	1.44 / 1.22
256×256	5.90 / 2.42	1.67 / 1.47	1.25 / 1.06

Table 1: Reduction of light beams / light shafts.

The second observation is that the *distribution* of the light beams in the scene has a major impact on the rendering time, often more than the scene complexity itself. For instance the sphere (Fig. 6) and the ring (Fig. 3) have about the same number of light beams, but the ones on the ring are well behaved and tightly distributed. The light beams refracted by the sphere are also narrowly distributed in the caustic region, but the ones reflecting on the sphere are spread all over the scene without contributing much to the illumination. This makes any hierarchical organization of the light beams fairly inefficient, as demonstrated in the rendering times about 17 times longer, even when zooming on the caustic region. We suggest in Section 5.2 that truncation of light beams carrying negligible flux density should bring much relief here.

As expected, attenuation in participating media requires also more computation. Again the nicely packed light beam distribution from the light to the cube of Fig. 11 (left) is rendered in far less time than the exploding light beams on the disco sphere of Fig. 9. In the latter image, as many as 100 samples along the eye ray still produce a noisy image while the analytical solution (entry and exit points of all intersected light beams with eye rays) is smooth and computed in less than half the time.

The relative slowness for rendering these two images is mainly due to the fact that we used a ray tracer to compute them. However, other rendering techniques, based upon scanconversion of light beam sections [19], might lead to more efficient rendering.

5 Extensions

In this section, we propose some ideas that can improve the quality of the rendering and reduce computation time. Although they have not been fully implemented, we feel confident these improvements will lead to better results.

5.1 Incremental Update of Light Beams

Our hierarchical structure inherits properties similar to those present in an interactive editor of a scene rendered via ray tracing [2]. For instance, when an object is modified interactively or in an animated sequence, the light beams that must be recomputed can be efficiently determined thanks to the hierarchy. They include changing surface reflection/refraction properties, geometry, and position/orientation.

Indeed if none of the light beams issued from a light image region are modified by a moving object, the indirect illumination carried by these beams will be identical at each frame and therefore does not need to be recomputed. The resulting images therefore exhibit no undesirable noise from its indirect illumination.

As shown in Table 2, building the light beam hierarchy is itself only a small portion of the total rendering time. However for a precise characterization of the light flux,

with re-unification of a high resolution light image, the gain of these incremental updates can become significant.

5.2 Handling Degenerated Light Beams

The thinner a light beam, the better the approximation of its light flux. However there will always be light beams that do not intersect the same surface (on object silhouettes), or that spread over a large solid angle after interacting with a surface. We call these light beams *degenerated* as their insertion within the hierarchical structure can significantly reduce its efficiency. When the supporting rays of a light beam do not share the same object path, we should neglect its contribution to the indirect illumination. Indeed in this case, the flux is not properly defined and extracting information from that beam may produce undesirable artifacts. When such an incoherence in a light beam is detected, it is automatically subdivided in the preprocessing step. If the maximum subdivision level is sufficient, the neglected light beams should therefore not be noticeable.

We can also cut any light beam at a distance for which the flux density falls below a given threshold. Unless the light flux is reconcentrated in a small region, which can be verified, it should speed up the rendering by avoiding computing negligible illumination.

Unfortunately, these solutions do not prevent a potential explosion of higher levels of the hierarchy. Since the top level always encloses every light beam, it is likely that any bounding volume would cover the entire 3D space unless all light rays share the same object path. Rather than computing and testing degenerated bounding volumes, we propose instead using an independent hierarchy for each object path. By doing so, point-in-beam queries stay logarithmic provided that the number of object paths is very small compared to the number of light beams.

5.3 Importance-driven Subdivision Criteria

Assume the basic light beam subdivision is sufficient to characterize most specular transfers. One can easily project onto the image plane the three intersection points forming the final node of a light beam on the receiving non-specular surface(s). Assuming this triangle is not blocked, reflected, or refracted when projected on the image, this indicates the area of the image covered by this illumination triangle. If the projected triangle is not small enough, the light beam is subdivided.

We can also proceed by first determining all the intersection points used to construct a region of the final image. If many of these intersection points lie within the same light beam, one can decide to subdivide this light beam even further to more precisely capture its contribution to the illumination.

5.4 Almost Perfectly Specular Surfaces

The light beam structure characterizes light specularly reflected in the scene. This structure could reduce the list of potential combinations of specular surfaces reaching a 3D point. The optimization of Mitchell and Hanrahan [18] would thus avoid searching among all possible combinations of specular surfaces.

Although we developed the structure for perfectly specular surface interactions, it is possible to extend it to almost perfectly specular surfaces as long as the light distribution remains mostly directional. If we do so, the light distribution within a light beam is no longer as *regular*. However by sampling the contribution with rays back to the light source in some flavor of bi-directional technique [27, 17], we could simulate non-perfectly specular surface interactions.

6 Conclusion

We presented the use of a hierarchical light beam structure to capture the perfectly specular light flux. It provides accurate indirect specular illumination without some of the noise and blur artifacts that basic photon map techniques can exhibit (without the Monte Carlo phase).

Even though most of the techniques used here have been introduced in different contexts, we showed how they could be integrated into a hierarchy in order to handle some of the problems involved in complex light flux. Among the features investigated, our hierarchical structure allows for efficient capture of the 3D effects of participating media, incremental update of the structures in an animated sequence, memory space reduction via caching, and logarithmic detection of point-in-beam requests.

Some ideas to improve the efficiency and accuracy of the results have been proposed with importance-driven subdivision of the indirect specular light flux, and handling degenerate light beams and almost perfectly specular surfaces. Photon tracing techniques can not easily release memory, adapt the sampling distribution without bias, or be built incrementally for an animated sequence.

Our technique does not suit for rendering illumination effects where directionality is not a predominant factor. Such effects come from large extended light sources, volumetric multiple scattering, and indirect illumination due to interreflections on diffuse surfaces. However our approach shows nice properties to handle light specular interactions required to render illumination from caustics.

Acknowledgements

We acknowledge financial support from FCAR and NSERC. Special thanks to the anonymous reviewers for their numerous suggestions, and to Avalon Viewpoint Datalabs for making available the galleon and shark 3D models.

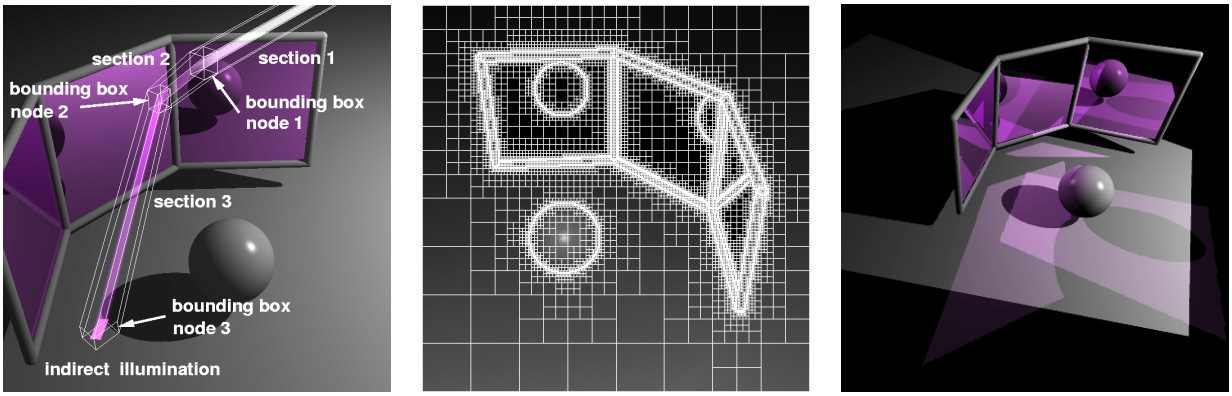

Figure 10: A 3D representation of light beams. Left: annotated 3D structure of shafts enclosing a light beam; Center: subdivided light image taken from the light source; Right: corresponding mirror reflections taken from the eye.

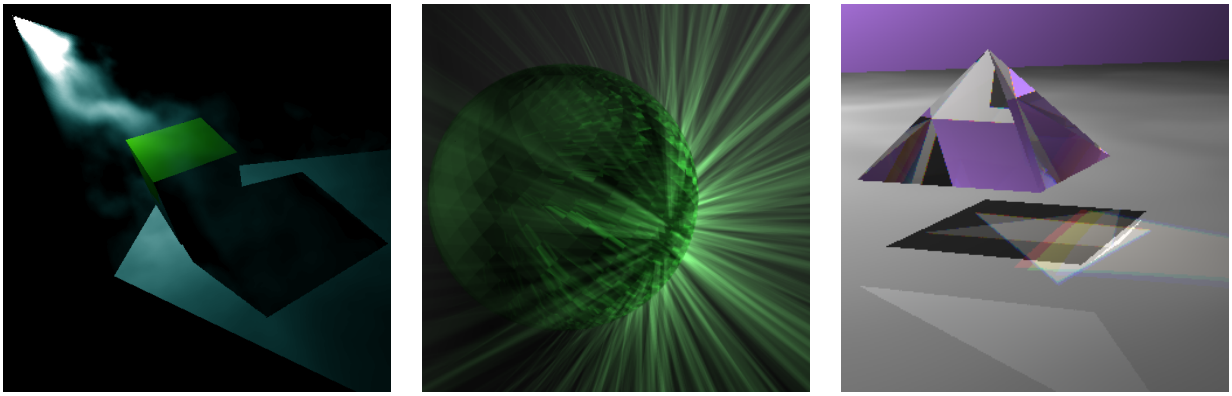

Figure 11: Left: non-uniform smoke with sampled attenuation; Center: uniform smoke illuminated with reflected beams only; Right: refraction at three wavelengths in a crystal pyramid.

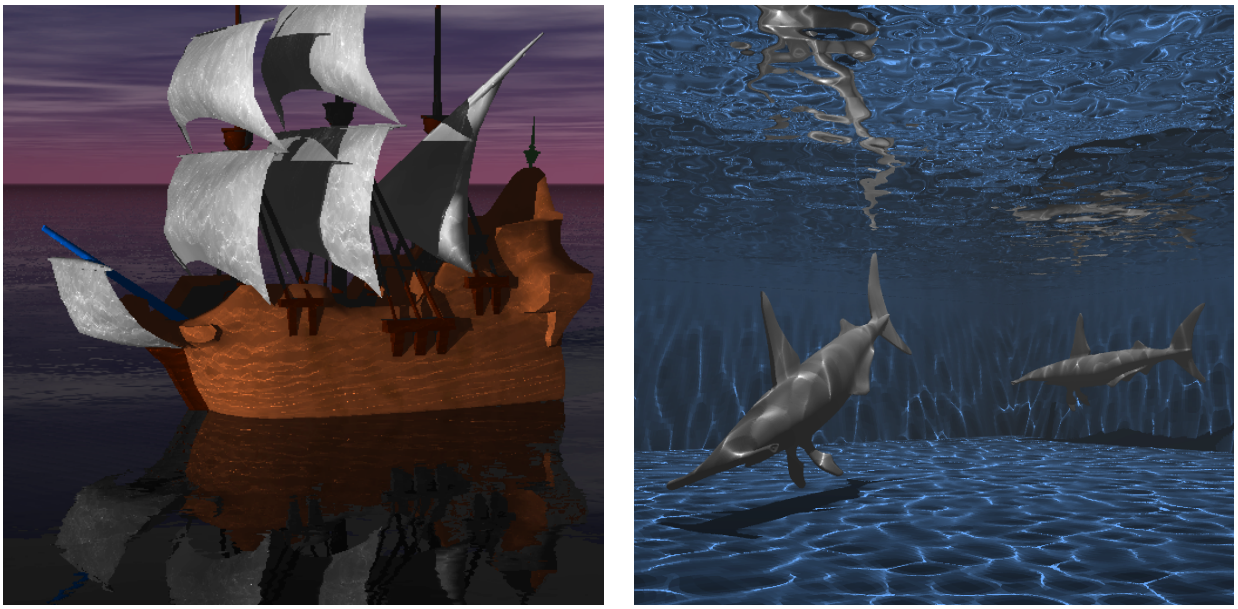

Figure 12: Water worlds.

References

[1] J. Arvo. Backward ray tracing. In *SIGGRAPH '86 Developments in Ray Tracing seminar notes*, volume 12, August 1986.

[2] N. Brière and P. Poulin. Hierarchical view-dependent structures for interactive scene manipulation. In *SIGGRAPH 96 Conference Proceedings*, Annual Conference Series, pages 83–90, August 1996.

[3] S.E. Chen, H.E. Rushmeier, G. Miller, and D. Turner. A progressive multi-pass method for global illumination. In *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 165–174, July 1991.

[4] P.H. Christensen, D. Lischinski, E.J. Stollnitz, and D.H. Salesin. Clustering for glossy global illumination. *ACM Transactions on Graphics*, 16(1):3–33, January 1997.

[5] J.H. Chuang and S.A. Cheng. Computing caustic effects by backward beam tracing. *The Visual Computer*, 11(3):156–166, 1995.

[6] M.F. Cohen and J.R. Wallace. *Radiosity and Realistic Image Synthesis*. Academic Press Professional, 1993.

[7] S. Collins. Adaptive splatting for specular to diffuse light transport. In *Fifth Eurographics Workshop on Rendering*, pages 119–135, June 1994.

[8] S. Collins. Reconstruction of indirect illumination from area luminaires. In *Eurographics Workshop on Rendering*, pages 274–283, June 1995.

[9] G. Drettakis and F.X. Sillion. Interactive update of global illumination using A line-space hierarchy. In *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 57–64, August 1997.

[10] E.A. Haines and J.R. Wallace. Shaft culling for efficient ray-traced radiosity. In *Eurographics Workshop on Rendering*, pages 122–138, June 1991.

[11] P.S. Heckbert. Adaptive radiosity textures for bidirectional ray tracing. In *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 145–154, August 1990.

[12] P.S. Heckbert and P. Hanrahan. Beam tracing polygonal objects. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 119–127, July 1984.

[13] Homan Igehy. Tracing ray differentials. In *SIGGRAPH 99 Conference Proceedings*, Annual Conference Series, pages 179–186, August 1999.

[14] H.W. Jensen. Global illumination using photon maps. In *Eurographics Workshop on Rendering*, pages 21–30, June 1996.

[15] H.W. Jensen. Rendering caustics on non-lambertian surfaces. *Computer Graphics Forum*, 16(1):57–64, 1997.

[16] H.W. Jensen and P.H. Christensen. Efficient simulation of light transport in scenes with participating media using photon maps. In *SIGGRAPH 98 Conference Proceedings*, Annual Conference Series, pages 311–320, July 1998.

[17] E.P. Lafortune and Y.D. Willems. A theoretical framework for physically based rendering. *Computer Graphics Forum*, 13(2):97–107, June 1994.

[18] D.P. Mitchell and P. Hanrahan. Illumination from curved reflectors. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 283–291, July 1992.

[19] T. Nishita and E. Nakamae. Method of displaying optical effects within water using accumulation buffer. In *Proceedings of SIGGRAPH '94*, Annual Conference Series, pages 373–381, July 1994.

[20] M. Shinya, T. Saito, and T. Takahashi. Rendering techniques for transparent objects. In *Proceedings of Graphics Interface '89*, pages 173–182, June 1989.

[21] M. Shinya, T. Takahashi, and S. Naito. Principles and applications of pencil tracing. In *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 45–54, July 1987.

[22] P. Shirley. A ray tracing method for illumination calculation in diffuse-specular scenes. In *Proceedings of Graphics Interface '90*, pages 205–212, May 1990.

[23] P. Shirley, C.Y. Wang, and K. Zimmerman. Monte carlo techniques for direct lighting calculations. *ACM Transactions on Graphics*, 15(1):1–36, January 1996.

[24] F.X. Sillion, J.R. Arvo, S.H. Westin, and D.P. Greenberg. A global illumination solution for general reflectance distributions. In *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 187–196, July 1991.

[25] F.X. Sillion and C. Puech. A general two-pass method integrating specular and diffuse reflection. In *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23, pages 335–344, July 1989.

[26] F.X. Sillion and C. Puech. *Radiosity and Global Illumination*. Morgan Kaufmann, 1994.

[27] E. Veach and L. Guibas. Bidirectional estimators for light transport. In *Eurographics Workshop on Rendering*, pages 147–162, June 1994.

[28] E. Veach and L.J. Guibas. Metropolis light transport. In *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 65–76, August 1997.

[29] J.R. Wallace, M.F. Cohen, and D.P. Greenberg. A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods. In *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 311–320, July 1987.

[30] B. Walter, P.M. Hubbard, P. Shirley, and D.F. Greenberg. Global illumination using local linear density estimation. *ACM Transactions on Graphics*, 16(3):217–259, July 1997.

[31] M. Watt. Light-water interaction using backward beam tracing. In *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 377–385, August 1990.

[32] K. Zimmerman and P. Shirley. A two-pass realistic image synthesis method for complex scenes. In *Eurographics Workshop on Rendering*, pages 284–295, June 1995.