Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal

Volume 4 | Issue 2 Article 1

July 2017

An Overview of Modern Global Illumination

Skye A. Antinozzi
University of Minnesota, Morris

An Overview of Modern Global Illumination

An Overview of Modern Global Illumination

Skye A. Antinozzi
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
antin006@morris.umn.edu

ABSTRACT

Advancements in graphical hardware call for innovative solutions, which can improve the realism of computer generated lighting. These innovative solutions aim to generate state of the art computer generated lighting through a combination of intelligent global illumination models and the use of modern hardware. The solution described in this paper achieves global illumination by ray tracing over geometry within a 3D scene from distributed light field probes and proceeds to shade the scene with a deferred renderer. Such a solution provides the flexibility and robustness that many other global illumination models have previously lacked while still achieving realistic lighting that is representative of the capabilities of the operating hardware.

Keywords

global illumination, ray tracing, light field probes, rendering equation, BRDF, deferred rendering

1. INTRODUCTION

1.1 Light and Visual Perception

Light enables one to observe their surroundings. For example, one may see walls and a ceiling that are painted an off-white hue while seated in a library, or, perhaps one sees lush trees with green leaves woven into their branches that sway in the wind while seated at a park bench. One may also view this paper from either printed paper or a digital display in which the colored text can be read properly. In the library setting, perception may be enabled by the fluorescent bulbs mounted within the ceiling fixtures. Or, if one sits at a park bench during the day then perhaps their perception is enabled by the sun above them. The fluorescent bulbs and the sun both share at least one property in common - when they are given sufficient energy, they begin to emit light into their surrounding environment. Light, or visible light, is defined as electromagnetic radiation to which the organs of sight react, such as the human retina [3, 6]. It is through this process that one can observe their surrounding environment.

If an object has the ability to produce light then that object is known as an *emitter*. Emitters are the objects in our world that enable visual perception of our surrounding environments. Without them, we would have no light and, without light, we would live in a dark and colorless world devoid of any cues of visual perception. When an emitter emits light into its environment it emits elementary particles known as *photons* which can be emitted in any direction into 3D space and are practically infinite in number. When photons are introduced into an environment they propagate throughout it by interacting with the environment's *geometry*, where geometry describes objects in the scene that light can interact with. For instance, if one reads this paper from print then a light source may be emitting photons that bounce off of the paper in the direction of the retina, enabling the reader to observe the text that they read. This is the process of visual perception by light. It is important to note, however, that photons are explained by the field of quantum mechanics to exhibit a particle-wave duality where they can either be observed as a particle or as a wave [11]. The reason as to why is beyond the scope of this paper and this is to our benefit, as the remainder of this paper will ignore the wave-like properties of photons and refer to photons as only particles of light.

UMM CSci Senior Seminar Conference, April 2017 Morris, MN.

2. SIMULATION OF REAL-WORLD LIGHT

2.1 Visibility and Ray Tracing

One of the primary problems in computer graphics is attempting to simulate the phenomena of light explained above. In other words, a primary focus of computer graphics is solving the problem of how to simulate visible light using a digital machine. We live in a continuous world that we cannot completely discretize and simulate on a machine that works on discrete quantities. In reality, the amount of photons produced from any one light source is thought to be a practically infinite number where each photon moves throughout continuous space. As real-world light works in such a continuous and infinite model we must describe a discrete model that can be implemented on a digital machine.

The light that represents all that one sees does not only interact with the retinae but also interacts with the surrounding environment. In other words, objects that one is not currently viewing also have light interacting with them in a way similar to light interacting with the retinae. However, as far as we humans are concerned, the important geometry in our environment is defined by what we give our visual focus to. From this reasoning, we have grounds to begin understanding a common, yet powerful, technique in computer graphics known as *ray tracing*. Ray tracing provides us with

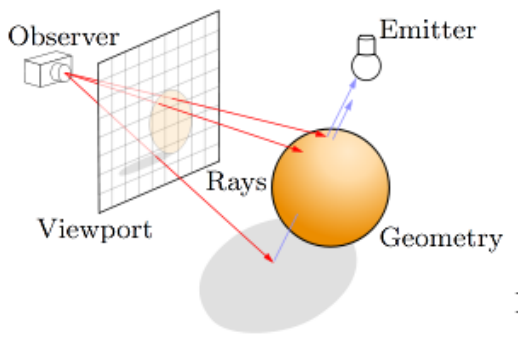

Figure 1: A concept visualization for a ray tracer is shown as rays are cast underneath the emitter from the observer viewport to intersect with geometry within the scene [9].

a fairly simple technique that solves the problem of determining what geometry within our environment is currently visible to us.

In a 3D modeled scene that is stored within computer memory, we define several properties of the scene and objects within the scene that help us to simulate real-world 3D space. One of these properties is the ordered-triple (x,y,z) that provides the numerical 3D coordinates of any point in discrete space within a 3D scene [4]. With ordered-triples it is possible to exactly identify where scene geometry and the *viewport* from which one views the scene both exist. To determine what geometry of the scene is visible within the 2D viewport, ray tracing works backwards by casting rays from the viewport into the scene (see Figure 1). More specifically, a ray tracer will iterate over the 2D grid defined by the viewport and cast a ray through each cell, or pixel, within the viewport grid [4]. Upon casting a ray, the ray tracer will use the stored ordered-triples and the casted ray to determine what scene geometry is currently visible, or *unoccluded* [2]. If scene geometry is occluded then the ray tracer can use the z-component of a relevant ordered-triple to resolve which geometry is closest, or visible, to the observer. This process is known as *depth testing*. Once a traced ray has been found to intersect with scene geometry then the traced ray is said to have become *incident* with the scene geometry. The tracer can then use the color of the geometry reported by the incident ray to provide a 2D representation of the point that can then be mapped and then rendered to the screen.

A caveat to the ray tracer proposed here is that it lacks the ability to simulate many realistic properties exhibited by the propagation of light particles throughout a scene. This proposed ray tracer only provides *direct illumination*, which describes geometry that is lit only by the first incident ray. In other words, once a ray has become incident with some scene geometry it does not interact with any other scene geometry and ceases to propagate throughout the scene. By limiting each traced ray to only one intersection with scene geometry the proposed ray tracer does not provide a level of realistic lighting that is reminiscent of the real world. Without expanding upon the ray tracer by simulating phenomena such as reflections, refractions and shadows, the scene will not exhibit the realistic qualities of light that an observer would expect to see.

2.2 Global Illumination

The challenge of simulating realistic lighting on a digital machine is a challenging problem which, when solved, has the potential to provide captivating lighting of scenes similar to the images shown in Figure 2. There are many facets to consider when realistically lighting a scene, such as how to handle reflections, refractions and shadows. To refer to simulating such qualities within a scene, the term *global illumination* is used as a descriptor of the collection of results produced by *indirect lighting* [4]. To further understand this, consider how light behaves as it reflects, refracts and produces shadows within an environment. Light reflects from many, if not all, surfaces within an environment by becoming incident with a surface to then bounce and become incident with yet another surface and so on. Sunlight may refract through the window pane of an office or through one's glasses as they read this paper - which may also cast a shadow onto the desk underneath the paper held in the reader's hand. Also, consider that the shadow cast by the paper is a soft shadow that is not actually black but a light grey hinting at the absence of light. The paper's shadow also does not abruptly transition from complete light to shadow at its edges but rather fades from light into darkness. It can be seen that the lighting of our environments exhibit subtle cues that constitute realistic properties of light. These subtle cues can be attributed to the illumination of surfaces caused not directly by an emitter but rather by the reflections and refractions of the light produced by the emitter. In other words, these illuminated surfaces are not lit directly by emitters but rather are indirectly illuminated by emitters. It is the property of indirect illumination that inspires the realistic qualities of light that global illumination models are designed to achieve [4].

2.2.1 Demand for Robust Lighting Solutions

As computer hardware continues to make progress in computational power the demand for robust computer generated global illumination solutions rises. Whether a designer is creating a scene for a 3D computer video game that runs in real-time, or, is carefully constructing a scene for an animated film that will take minutes to generate, the industry of computer imaging will always require innovative, flexible and robust lighting solutions to provide realism in ways that evolve beyond previous methods. From here, the remainder of this paper describes an insight into an implementation of a global illumination model using light field probes that provides a robust and realistic lighting solution for those working in computer aided design software (CAD), animated film creation, and the computer video game industry.

3. LIGHT FIELD PROBE MODEL

3.1 Light Field Probes

To handle realistic properties of light, we require the collection of a fairly extensive set of data about the scene, its geometry, and the light sources within the scene that the computer operates on. In addition to the data required, we also need a way of efficiently operating on that data. There have been many proposed and implemented global illumination models that aim to approximate realistic lighting. From this collection of models, a recent method that employs *light field probes* shows great promise for advances in both the usability of global illumination implementation and these im-

Figure 2: An indoor scene illuminated by a specialized light field probe model in the computer video game *Tom Clancy's The Division* [8].

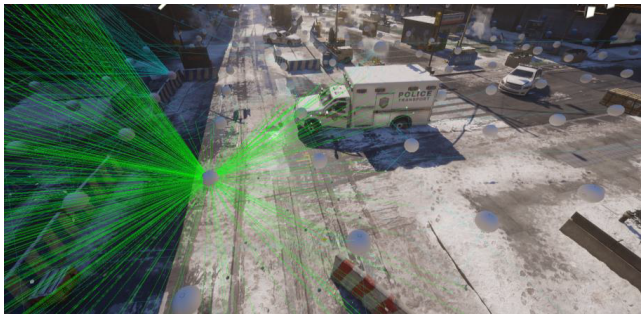

Figure 3: Distributed light field probes visualize their ray emission paths in a scene from the computer video game *Tom Clancy's The Division* [8].

plementations' computational efficiencies. Light field probes provide 3D scene designers with a flexible and robust solution for approximating realistic lighting through global illumination [7]. A light field probe can be thought of as an emitting sphere of light that can be suspended at any point in space within a 3D scene (see Figure 3). By suspending probes in 3D space, a scene designer that requires realistic lighting within a scene does not have to meticulously go through the scene and adjust the lighting by hand. Rather, they can simply place a light field probe where the realistic lighting is required. In addition to this, light field probes do not require manual placement by a designer but can be generated and distributed within a scene programmatically [7]. By packaging a subset of the complexity of global illumination into these compact probes, effort expended before by human hands has now become an automated process that can realistically light scenes and – if lighting requirements are not fulfilled by automated distribution – allows designers to intervene and tweak the probes as necessary.

3.1.1 Light Field Probe Construction

The construction of light field probes require that they each encode a scene's radiance and geometric information to be used by *light field* queries. Scenes managed by the light field probe model each contain a discrete light field L(x,w) that encodes the spatial-angular distribution of radiance for all points x $\in \mathbb{R}^3$ and outgoing directions $\omega \in S^2$ [7]. At each discrete sample position x', a light field probe is stored that encodes information about surrounding light sources and geometry. Specifically, each light field probe needs the following three pieces of information about the surrounding geometry. Each probe requires an image (similar to a viewport) of the light field at point x' L(x',ω), that it can use to determine visible geometry using ray tracing (see bottom of Figure 4). The probes also require the surface normals $\vec{n}_{x''}$ for each geometry point x'' closest to x' that will provide necessary information to introduce shadows, or *shading*, into the scene [1]. To be more clear, all visible geometry from x' is shown within the spherical image captured from the location of x'. Points x'' contained on each surface geometry within the image must have their surface normals calculated. Finally, the probes require the radial distances, in radians, $r_{x' \leftrightarrow x''}$ between x' and x'' to give further information for the shading of the scene geometry.

In addition to the encoded data for light field probes outlined here, these probes may also possess other encoded data that may be modified to affect the lighting of scene. For instance, the radiosity of a light field probe can be encoded to determine how well-lit scene geometry within its spherical proximity will be. This means that light field probes enable designers to provide varying levels in the intensity of light throughout a scene - a factor that can significantly contribute to the realism of the scene's lighting.

Finally, it is important to note that light field probes are known as proxy geometry and are not taken into account when light is propagated throughout a scene. In other words, light cannot interact with a light field probe but can only interact with other tangible geometry within the scene.

3.2 Spherical to Octahedral Mappings

A primary idea behind this method is that a ray trace is performed over the spherical image of the light field at point x' to determine visible geometry surrounding the probe located at x'. A problem that naturally arises from this is determining how to ray trace, or march, over a spherical image. Modern ray tracing techniques are typically limited to performing a ray trace over a rectangular image where the bounds of the image are easily determined. From this, the algorithm must either be able to provide a surface parameterization of the light field probe's sphere to determine the bounds of the image for ray tracing, or, the algorithm must utilize some other solution that does not determine surface parameterizations for the light field probe spheres. To address this problem, this method utilizes a transformation of spherical to square images using octahedral mappings [7]. This is done by mapping the octants of a spherical image to the faces of an octahedron where the faces of the octahedron are then projected onto the 2D plane and then unfolded into a unit square (see Figure 4). Once this process has completed, the result is a square image representation of the probe's surrounding geometry that was previously encoded by a spherical image. With a square image of the surrounding geometry, it is now possible to perform a ray trace over this new representation of the probe's viewport to determine what geometry is visible to the probe.

3.3 Multiple Probe Visibility

If scene geometry has been determined as visible from the viewport of a light field probe then it is known that the incident surface must be lit by the probe at the point of intersection. However, it is possible that more than one light field probe can incident the same surface at the same point. In other words, all probes within the scene affect the

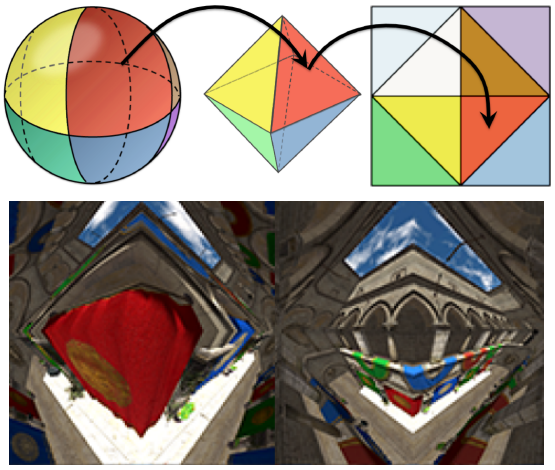

Figure 4: The top image shows the transformation from sphere to octahedron to unit square to allow ray tracing of a probe's visible geometry [1]. The bottom image shows two spherical images from the *Sponza Atrium* scene as their final square image represenations [7].

scene's lighting so this model requires a way to iterate over all of the relevant probes contained within a scene. What is now required is a *light field ray tracer* that can light a scene by accounting for all probes that light geometry within the observer's viewport. For instance, it may be that this method is being used within a real-time application such as a computer video game. In such a case, the only probes that the algorithm would be concerned with are the ones that help to determine what is currently visible within the viewport. Keep in mind that in smaller scenes, however, it may be the case that all light field probes within the scene affect the current visible geometry.

3.3.1 Light Field Ray Tracing

The light field ray tracer algorithm works by iterating over the relevant probes in a scene by first casting a ray into the light field from the viewport. Once a ray has been cast, the probe whose center lies closest to the ray is selected as the first probe used to evaluate visible geometry that the ray may intersect. By selecting a probe in this manner, we are likely to select the correct probe that will light the geometry that has become incident within the viewport. This has the advantage of minimizing the number of probes that need to be checked when determining which geometry should be lit by probes in a scene. Once the probe and the viewport have been determined to have visibility of the same incident geometry, the light field ray tracer proceeds to consider other probes that affect the lighting at the incident point. The light field ray tracer does this by recognizing that each ray cast into a scene must be contained within a cage, or *bounding box*, of eight light field probes. Due to close spatial locality of the probes, it is reasonable to assume that these probes will also share common visible geometry with the probe that recorded the first point of incidence from the traced light field ray. This idea helps to define a heuristic that the ray tracer can use to iterate through the eight probes that define the corners of the bounding box in order to check their visibility of the incident point (see Figure 5).

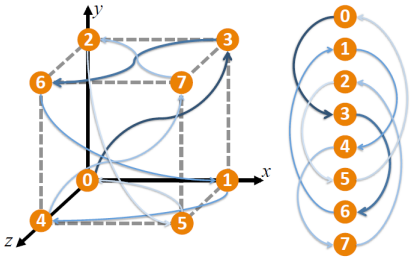

Figure 5: A possible traversal of the eight light field probe cage used for light field ray tracing [7].

As a result, any corner probe from the bounding box that has visibility of the incident point will be a probe that also lights the incident point.

3.4 Visibility of Off-Screen Geometry

Light field probes also provide the subtle but very important aspect of realistic lighting known as world-space ray tracing. World-space ray tracing is different from traditional ray tracing in that it takes into account geometry that is located outside of the viewport. To better understand this, notice that Figure 6 shows an off-screen curtain being reflected by the television screen. It's important to realize that the curtain, which is visible through the reflection, has no geometry actually visible from the viewport. This means that this model of global illumination takes into account all relevant geometry within a scene and not just the geometry directly viewable from the viewport. Because the model takes into account both the viewport and light field probes, the system is able to reflect off-screen objects that are visible to one or more distributed light field probes and include them within rendered images.

4. SCENE SHADING

4.1 Achieving Realistic Lighting

In order to determine how light is distributed from geometry within a scene, the global illumination model can provide multiple generalized light models that manage how light interacts with different surface materials. For instance, light reflecting from a mirror would have the mirror recognized as a glossy surface with a property of high *specular reflection*. Surface geometry produces a specular reflection when light is reflected in a nearly perfect "mirror bounce." [4] Or, consider a curtain hanging in a room that is recognized as an opaque, or matte, surface material. This curtain would be described to have a property of *highly diffuse lambertian reflectance* [4]. Surface geometry produces a diffuse lambertian reflection when the incident ray is reflected in many different directions rather than just one. Finally, consider an example of refraction where light passes through an object, such as a glass pane, and reflects inside of the object multiple times before exiting the object in some direction. Interactions between surface materials and light, such as these, all call for individual computational lighting models that can properly simulate their distributions of reflected light.

4.2 Modeling Reflected Light

In order to provide these models of reflected light distribution, the global illumination model turns to generalized

Figure 6: World-space ray tracing is demonstrated by the reflection of the off-screen curtain from the TV screen [7].

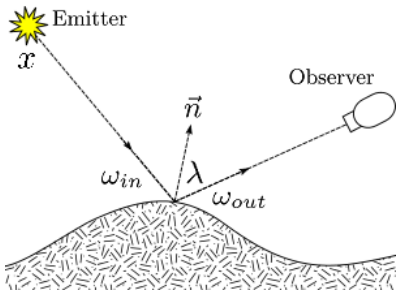

Figure 7: A concept visualization of the BRDF that shows the incident ray ω_{in} from the emitter being reflected about the surface normal $\vec{n}$ as ω_{out}, with light color λ, in the direction of the observer [10].

forms of the *bidirectional reflectance distribution function* (BRDF). Simply put, the BRDF answers the question of how reflected light is distributed from a surface. Its most basic form describes the likelihood that a ray of light with encoded color λ incident at point x from direction ω_{in} will be reflected in the outgoing direction ω_{out} (see Figure 7). Therefore, the BRDF can be written as the function L(x, ω_{in}, ω_{out}, λ) [4]. It is important to point out that λ is encoded as a property of surface geometry and can be used to determine the color of light that has become incident with scene geometry.

The proposed global illumination model can be extended to include proper shading of scene geometry by making use of generalized forms of the BRDF. For instance, a BRDF could be used to determine how light is reflected from the wood, TV screen or curtain in Figure 7. Because of the introduction of shading, the global illumination model must also be aware of what surface materials it operates on. This requirement calls for an encoding of surface materials to define each from one another as glossy, lambertian or any other surface type. Properties such as these must be available to the algorithm as they are crucial for determining which BRDF to apply to each surface material.

5. COMPLETING THE LIGHTING MODEL

5.1 A Global Illumination Rendering Pipeline

With visible geometry information for each probe and BRDF models to define how light is distributed from surfaces within a scene, the global illumination model can finally combine these two processes into a rendering model known as *deferred rendering* (or *deferred shading*) [7]. A deferred renderer allows for the construction and rendering of geometry and lighting within 3D scenes by working in two passes. During the first pass, the deferred renderer stores all surface information used for determining visibility and lighting calculations into multiple buffers. One buffer of particular importance and relevance is known as the *G-buffer* (or *geometry buffer*) [4]. The G-buffer stores geometry primitives which makes the buffer responsible for holding information about surfaces closest to the eye, such as ordered-triples of points on the surface, surface normals, radial distances and encoded material properties that are necessary for lighting calculations. Next, the second pass uses the collected scene information and the BRDF models to perform lighting calculations for the scene geometry contained within the bounds of the viewport. Shading is only calculated for geometry contained within the bounds of the viewport to minimize the amount of unnecessary lighting calculations for a scene in order to improve the performance of the second pass of the deferred renderer. Once the deferred renderer has completed both passes, the system's platform can then render the data stored within the buffers onto the supported display device. From this point, an observer of such a device would be able to view the constructed 3D scene from their own 2D viewport and, if done correctly, they would witness phenomena of realistic lighting such as reflections, refractions and soft shadows that 3D global illumination has to offer (see Figure 2).

5.2 Approximating the Rendering Equation

Global illumination models, including the one described in this paper, all aim to solve the problem of providing realistic lighting within 3D scenes. Specifically, this problem is formally defined mathematically by the *rendering equation*. The rendering equation describes the color, λ, of radiance leaving the point x in the direction of ω_{out} as L_{out}(x, ω_{out}, λ) in units of radiosity, or brightness per unit area.

It describes this behavior by first determining the radiance emitted from point x, with color λ, in the direction of ω_{out} to produce the function L_{emis}(x, ω_{out}, λ). The value of emitted radiance is then summed together with the light reflected from x in the direction of ω_{out} to determine L_{out}(x, ω_{out}, λ). The amount of radiance reflected from each ray in outgoing directions can be described by the previously discussed BRDF, or L(x, ω_{in}, ω_{out}, λ). By multiplying the BRDF with the radiosity of incident rays of color λ at points x from directions ω_{in}, or L_{in}(x, ω_{in}, λ), the result is a product that nearly represents the radiosity of a reflected ray.

What remains in order to describe the radiosity of a reflected ray is to account for the *Lambert factor*, which describes that the amount of reflected light at an incident point depends on the angle of incidence [4]. This means that as the angle of incidence increases from the surface normal that less light will be reflected from the surface. More specifically, as ω_{in} becomes incident with a surface, the dot product of the negated angle of incidence and the surface normal, or $-\omega_{in} \cdot \vec{n}$, accounts for the distribution of incident light at an angle. The Lambert factor retains a maximum value of one and approaches zero as the incident angle increases from the surface normal – eventually producing no reflected light once the incident angle reaches a value of zero.

We have described all of the components required for computing the radiosity of any one reflected ray. What is now required is the ability to account for the sum of infinitely many rays being reflected from the incident surface. To calculate this sum, an integral can be defined that repeatedly sums the product of the emitted radiance, the BRDF and the Lambert factor. The limit of integration can be defined as Ω to represent the hemisphere of all possible incoming directions that surround the point of incidence. From this point, it is now possible to assemble the entirety of the rendering equation. To represent the amount of radiance emitted from an incident point, the emission function and reflectance distribution integral can be combined to describe the radiosity present at an incident point in the following equation [4].

$$L_{out}(x, \omega_{out}, \lambda) = L_{emis}(x, \omega_{out}, \lambda) + \int_{\Omega} L_{in}(x, \omega_{in}, \lambda) f(x, \omega_{in}, \omega_{out}, \lambda)(-\omega_{in} \cdot \vec{n})\, d\omega_{in}$$

Even though it may not be immediately apparent, models of global illumination (and simulated lighting models in general) all strive to simulate the properties defined within the formal rendering equation in order to approximate a solution that best captures the essence of realistic lighting.

6. RESULTS

By using light field probes, BRDFs for each surface type, and a deferred renderer, this global illumination model provides an efficient method for achieving realistic lighting. Figure 1 depicts a living room scene where global illumination was computed in 4.7 ms by using light field probes. However, it must be noted that the full implementation required to achieve such a result extends beyond the scope of this paper. For instance, an efficient implementation for gathering probe surrounding geometry data (images, surface normals, etc.) builds upon the methods presented in this paper and, when described, could easily extend beyond this paper's length. Nonetheless, such efficient implementations utilize relevant optimizations to increase runtime performance for global illumination.

An example optimization is gathering surrounding probe geometry before runtime in order to quickly globally illuminate a scene [7]. In scenes with no moving geometry, the deferred renderer's G-buffer can be precomputed allowing only runtime lighting calculations to be considered and results in increased performance. In addition, scenes with moving geometry can have the G-buffer recomputed once geometry has changed position within the scene [7]. By limiting the G-buffer to recomputing only when geometry has moved within the scene, redundant fetches of surrounding probe geometry information are eliminated. Precomputing and recomputing the G-buffer are both examples of optimizations that can greatly increase the model's runtime performance.

The light field probe model is able to produce realistic lighting similar to previous global illumination methods while presenting several new advantages (see [2]). Primarily, light field probes allow for the simple and flexible placement of probes within a scene to provide realistic lighting with a low computational overhead [7]. With the overview this paper describes and advanced optimizations, such as precomputation of the G-buffer, the light field probe model contributes a powerful method for globally illuminating a scene.

7. CONCLUSION

The light field probe model for global illumination provides a flexible and robust solution that addresses the need for realistic lighting within 3D scenes. Solutions such as these present efficient designs that can be implemented on modern hardware and can be easily utilized by designers of 3D scenes. However, the implementation discussed within this paper is only one specific model of many that achieve global illumination and does not specifically consider other global illumination models used to produce images like those in Figure 2. Nonetheless, the overviewed global illumination model this paper discusses provides the basis of a powerful global illumination model that can be expanded upon to introduce elements that more vividly capture the nature of light. Instead of attempting to tackle many different models of global illumination and the plethora of advanced techniques therein, this paper has aimed to form the foundation of understanding the behavior of light by modeling it within a single global illumination model that can be implemented on a digital machine. From here, it is now possible to explore complementary and expanding techniques of global illumination, such as *shading of separate Lambertian and glossy surfaces*, *spherical harmonics* and *physically based lighting* in order to further understand the robustness and flexibility that global illumination models provide [5, 7].

8. REFERENCES

[1] Z. Cigolle, S. Donow, D. Evangelakos, M. Mara, M. McGuire, and Q. Meyer. A Survey of Efficient Representations for Independent Unit Vectors. *Computer Graphics Techniques*, 3:1–30, 2014.

[2] C. Crassin, F. Neyret, M. Sainz, S. Green, and E. Eisemann. Interactive Indirect Illumination Using Voxel Cone Tracing. In *Computer Graphics Forum*, Oxford, UK and Maiden, MA, 2011. Eurographics Association and Blackwell Publishing.

[3] Dictionary.com. Definition of light, 2017. [Online; accessed 11-April-2017].

[4] F. Dunn and I. Parberry. *3D Math Primer for Graphics and Game Development*. A K Peters/CRC Press, Boca Raton, FL, 2011.

[5] C. Hery and R. Villemin. Physically based lighting at pixar, 2013. [Online; accessed 11-April-2017].

[6] Hyperphysics.phy-aster.gsu.edu. The retina. [Online; accessed 12-April-2017].

[7] M. McGuire, M. Mara, D. Nowrouzezahrai, and D. Luebke. Real-time Global Illumination using Precomputed Light Field Probes. In *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, New York, NY, 2017. ACM.

[8] N. Stefanov. Global Illumination in Tom Clancy's The Division. Game Developers Conference, 2016. [Online; accessed 11-April-2017].

[9] Wikipedia.com. Ray tracing (graphics). [Online; accessed 12-April-2017].

[10] Wikipedia.com. Bidirectional reflectance distribution function, 2017. [Online; accessed 12-April-2017].

[11] Wikipedia.com. Wave-particle Duality, 2017. [Online; accessed 11-April-2017].