Real-Time Photo-Realistic Augmented Reality Under Dynamic Ambient Lighting Conditions

Kamran Alipour and Jürgen P. Schulze
Department of Computer Science and Engineering, University of California San Diego, La Jolla, California

Abstract

Despite all major developments in graphics hardware, realistic rendering is still a computational challenge in real-time augmented reality (AR) applications deployed on portable devices. We have developed a real-time photo-realistic AR system which captures environment lighting dynamically, computes second order spherical harmonic (SH) coefficients of it on the CPU and the resulting nine coefficients on the AR device for real-time rendering. Our technique provides a very computationally efficient rendering procedure for diffuse objects in real-time rendering scenarios. We use two options for dynamic photometric registration of environments: a Ricoh Theta S 360° camera, and a Raspberry Pi Zero mini PC with a camera with a 180° fisheye lens. We tested our system successfully with software developed in Unity 3D with the Vuforia AR package, running on a Microsoft Hololens, and also an Oculus Rift DK2 with a stereo camera add-on.

Introduction

Consistent illumination and visual coherence are the two major technical challenges towards a seamlessly photorealistic Augmented Reality (AR) scenario. A constantly growing need for more realistic blending between real world and virtual objects in AR applications has provoked extensive studies towards photorealistic illumination simulation. Lighting measurement and simulation generally include two major components: direct or high frequency lighting that affect the object illumination directly; and indirect or low frequency lighting that is created mainly by the reflection of the light from objects in the scene and causes effects like color bleeding. While high frequency lighting is more crucial when dealing with near-field illumination scenarios, low frequency lighting is present among almost all scenarios in the form of ambient illumination. As a common base to almost all photorealistic lighting problems, this paper focuses on the simulation of low frequency lighting from ambient environment illumination.

Capturing and replicating environment illumination dynamically can be difficult technical challenge in real-time applications. Especially in scenarios where the environment lighting can change rapidly, the AR model needs to be able to capture and reflect those changes instantly.

Theoretically, we need an exact definition of positions, intensities, colors and the number of illumination sources to produce an accurate simulation of ambient light in the virtual scene. However, these are rarely readily available in real-world AR scenarios, and the rendering approach itself is quite computationally expensive.

Related Work

To address this challenge, [1] introduced Image Based Lighting (IBL) as a method to measure scene radiance, global illumination and model environment illumination on synthetic objects. This light-based model measures incident lighting at the location of the synthetic objects. This method has been widely used and improved upon by others, to generate environment maps [2] with various levels of glossiness and to render synthetic objects into dynamic real-world scenes [3] at interactive frame rates [11]. Moreover, Ramamoorthi and Hanrahan [4] developed an analytic relationship between radiance and irradiance to determine the illumination from images of a convex Lambertian object and then simplifed that relationship in terms of spherical harmonics (SH) coefficients of the lighting corresponding to the lowest-frequency modes of the illumination [5]. This analytic simplification was later used as a basis for the Precomputed Radiance Transfer (PRT) model in real-time rendering under dynamic, low-frequency illumination [6].

On the other hand, photometric registration is a crucial step of IBL, and multiple techniques have been introduced for generating such an omni-directional image. Different hardware setups for incident light acquisition have been proposed through the years with an effort to propose a technique with minimum scene invasion and hardware requirements. One popular approach is using mirror balls in the scene as light probes to capture a full high-resolution panoramic image of the environment. Kanbara and Yokoya [7] proposed a 3D marker which is a combination of a flat, square marker and a black mirror ball to cover both geometric and photometric aspects of the lighting environment. Supan et al. [8] proposed placing a perfect mirrored sphere in the scene and also a camera with a fisheye lens, which gave them the advantage of capturing specular reflections since incident lighting is acquired at higher resolution. The need to produce more specular reflections on synthetic objects has urged researchers to move from these mostly LDR registrations towards HDR mapping. Madsen and Laursen [9] use pre-acquired HDR environment maps for photometric registration in their AR implementation. Grosch and Eble [10] also use an HDR camera to capture the environment in real-time. However, their method limits the results to simulating indirect lighting in a controlled space, the Cornell Box. Moreover, in some more advanced cases, multiple HDR cameras have been setup in the scene, which can be impractical for portable and mobile scenarios.

Another alternative method to light acquisition is light estimation based on a 2D input image. Finlayson and Fredembach [13] introduced one of the earliest implementations using two images (one filtered). Later, Lalonde et al. [14] introduced a similar method using a single image and a probability distribution based

on the position of the sun for outdoor light estimation, where this method proved most effective. This work was followed by other methods focused on natural light estimation using either sun position probability [15] or other deep learning methods [16, 17]. These techniques show very promising results in outdoor scenarios where there is usually one major direct light source. However, in other situations (e.g., indoors and in environments with multiple light sources) they are not as effective.

A common feature between all the above mentioned approaches is that they are very computationally intensive and use the GPU to produce realistic lighting simulations, which can be rather costly for many smaller devices. However, in a lot of scenarios producing relighting based on only ambient lighting and being able to compute that using only the CPU can provide a more practical framework for device-based applications. This assumption also significantly reduces the required quality for the environment lighting maps. As a result, the lighting module can work perfectly with an LDR sensor, which makes the solution both cost efficient and flexible.

In this publication, we present a real-time photorealistic AR module using an analytic expression for the irradiance in terms of the SH coefficients of the lighting. By computing only nine coefficients based on low frequencies in the lighting, the SH technique provides an efficient rendering algorithm for diffuse objects to be utilized for real-time applications. By lowering the computational cost compared to other solutions we have made our approach applicable to devices with only CPU resources and limited power consumption. Also by excluding complicated aspects like specular illumination, we have been able to simulate ambient light using LDR sensors. The simplicity of its architecture makes this AR module compatible with AR technologies including Oculus Rift, HTC Vive (both with stereo cameras such as the ZED Mini) and the Microsoft Hololens. We successfully implemented our algorithm in C++ with OpenCV, and integrated it with OpenGL as well as Unity and Vuforia to produce AR applications for the Oculus Rift DK2 and the Microsoft Hololens.

Spherical Harmonics

Based on the lighting technique introduced by Ramamoorthi and Hanrahan [5], we use an analytic expression for the irradiance in terms of SH coefficients of the lighting. The method reaches an average error as low as 1% by computing only 9 light coefficients corresponding to low frequency modes of the illumination. In this section we cover a brief review of the technique.

By neglecting the cast shadows and near field illumination, this analytical model defines the irradiance E as a function of surface normal n which is given by the integral over the upper hemisphere (n):

$$E(n) = \int_{\omega(n)} L(\omega)(n.\omega)d\omega \tag{1}$$

Where n and ω are unit direction vectors. Based on this analogy, E and L are parameterized by a direction (θ,ϕ) on the unit sphere. The radiosity factor B which defines the image intensity is defined by scaling irradiance with albedo $\rho(p)$:

$$B(p,n) = \rho(p)E(n) \tag{2}$$

where p is position.

An analytical formula has been provided for the irradiance in terms of SH coefficients. SHs Y_{lm}, with $l \geq 0$ and $-l \leq m \leq l$, are the analog on the sphere to the Fourier basis on the line or circle. The first 9 SHs (with $l \leq 2$) are simply polynomials of the cartesian components and are numerically calculated (for more details see Ramamoorthi and Hanrahan [5]).

Eventually, $E(\theta,\phi)$ and $L(\theta,\phi)$ are represented by the coefficients E_{lm} and L_{lm} in their SH expansion:

$$E(n) = \sum_{l,m} L(\omega)(\theta,\phi) \qquad L(n) = \sum_{l,m} L_{lm}Y_{lm}(\theta,\phi) \tag{3}$$

Also, $A = (n.\omega)$ with coefficients A_l:

$$A(\theta) = max[cos\theta, 0] = \sum_{l} A_l Y_{l0}(\theta) \tag{4}$$

By neglecting the cast shadows and near field illumination, this analytic model defines the irradiance E as a function of surface normals given by the integral over the upper hemisphere. For rendering purposes, we only consider low frequency lighting coefficients of order 2 or less. Based on this assumption, irradiance is approximated by only 9 parameters (1 for $l = 0$, $m = 3$, 3 for $l = 1$, $-1 \leq m \leq 1$, and 5 for l=2, $-2 \leq m \leq 2$); and we can write irradiance E as:

$$E(n) = n^t M n; \quad n^t = (x \quad y \quad z \quad 1). \tag{5}$$

M is a symmetric 4x4 matrix for each color channel and can be written as:

$$M = \begin{bmatrix} c_1L_{22} & c_1L_{2-2} & c_1L_{21} & c_2L_{11} \\ c_1L_{2-2} & -c_1L_{22} & c_1L_{2-1} & c_2L_{1-1} \\ c_1L_{21} & c_1L_{2-1} & c_3L_{20} & c_2L_{10} \\ c_2L_{11} & c_1L_{1-1} & c_2L_{10} & c_4L_{00} - c_5L_{20} \end{bmatrix} \tag{6}$$

In Eq.(6), c_i is analytically calculated once and $L_{l,m}$ will be updated on each frame based on the light map. This concise implication of environment illumination drastically simplifies the rendering process and makes it suitable in a variety of applications.

AR Implementation

With the simplified lighting computations, our algorithm can run on the CPU with limited computational resources. Also, as mentioned earlier, the SH representation of environment lighting reduces the output size into only nine coefficients, which make the results transferable to multiple devices instantly. Based on these intuitions, we propose two hardware setups for our AR system: a server solution and an edge solution. In both methods, the light acquisition sensors should be ideally placed at the position of the augmented object. This positioning is measured by using either an AR marker or AR spatial sensors based on the AR devices. For our server solution we use a Ricoh Theta S 360° camera to capture a live panoramic video stream of the environment, which the rendering units use as input for their lighting calculations. On the other hand, in our edge solution we use a 180° fisheye camera which covers the upper hemisphere of the environment map. The camera is connected to a Raspberry Pi Zero mini PC where the light coefficients are computed (Figure 1).

Figure 1. *Environemntal light acquisition hardware. Left: 360° Ricoh Theta camera used for the server solution. Right: Raspberry Pi Zero and 180° fisheye camera used in the edge solution.*

In our server solution pipeline, the 360° camera streams the panoramic video of the environment to a local server through its Wi-Fi connection. The lighting coefficients based on the panoramic video from the panoramic camera are calculated on the server and transmitted to all participating AR output devices. The connection between the server and the AR can be either through Wi-Fi (e.g., for the Microsoft Hololens), or a cable (e.g., HDMI for the Oculus Rift). Figure 2 depicts the hardware setup of our server solution.

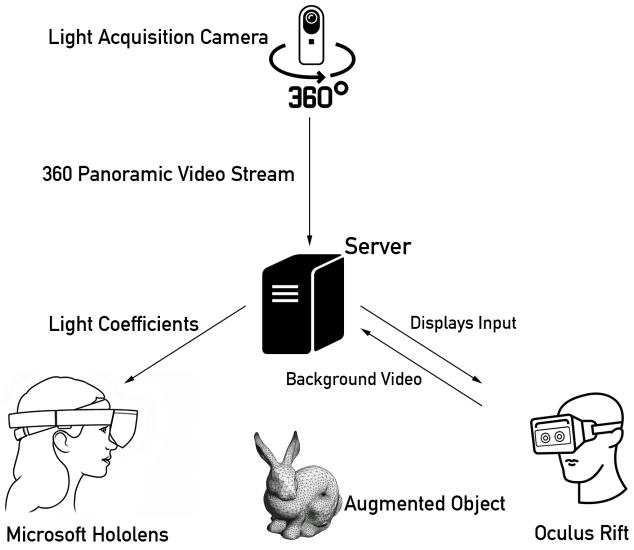

Figure 2. *Components of the overall implementation and their interactions for our server solution.*

As an alternative approach, we deployed our light acquisition and calculation algorithms to a Raspberry Pi Zero with a 180° fisheye camera (Figure 3). The change from panoramic view to hemispherical view is based on the notion that in most scenarios the major light sources are located in the upper half of the environment map. The device can provide lighting calculation results through its Wi-Fi connection. This setting can turn our solution into a standalone system with the ability to provide all VR devices with real-scene lighting information on an augmented Object.

Our algorithm is intended to integrate with different AR frameworks and devices including the Oculus Rift and the Microsoft Hololens (Figure 4). For the Oculus Rift DK2, we use an OVRVision stereo camera which is mounted on the front of the headset. It captures the background video and feeds it to the rendering module. Using the headsets location and orientation in world coordinates, the server renders the synthetic object over the background video and displays the result in the headset.

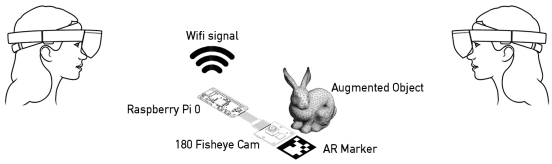

Figure 3. *Components of the overall implementation and their interactions for our server solution.*

Figure 4. *AR Units. Left: Oculus Rift DK2 with mounted OvrVision Unit. Right: Microsoft Hololens.*

We have developed an application for the Microsoft Hololens using the Unity 3D authoring system. This application receives the light coefficients from the server and uses them in shaders that generate an overlay image for AR. Using OpenCV and OpenGL libraries, we have also created an AR application which renders the AR scene for an Oculus Rift headset. The output of this application is discussed in the next section.

Results

In this section we present the results for our two proposed solutions. For the server solution, we run our local server on a PC with an Intel Xeon E5-2660 CPU. The 360° Ricoh camera provides a panoramic stream with a resolution of 640 x 480 pixels at 15 frames per second (fps). Using OpenCV, the frame size is reduced to 64 x 48 pixels before entering SH computation. On the other hand, the edge solution uses a Raspberry Pi Zero unit which is equipped with a Broadcom BCM2835 1GHz single-core CPU.

Server Solution

The Raspberry Pi Zero can calculate the SH coefficients at 30 fps which is a sufficiently interactive rate for real-time applications. Figure 5 shows the output for the Oculus Rift. The result depicts a believable blending of an augmented object within the real scene. In this setting the positioning of the augmented object is handled through the Oculus Rift's spatial sensor.

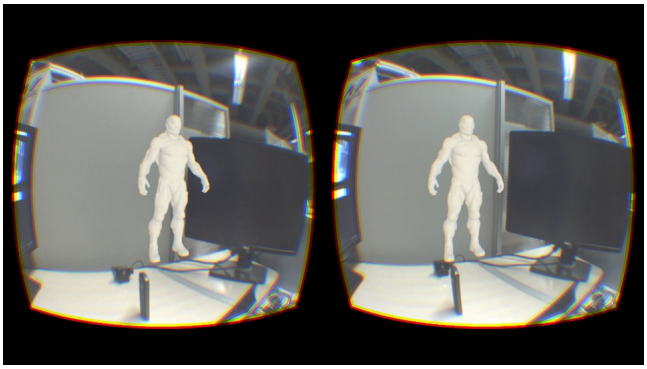

Figure 5. *Oculus Rift display output in server solution.*

We also compared the shading created by the algorithm to the real scene using a 3D printed version of our virtual model. This test is designed for the Microsoft Hololens for which we created an AR application within the Unity 3D platform. To position the virtual object accurately within the real scene we used the Vuforia package.

Figure 6. *Test setup for Hololens output in server solution.*

Figure 6 shows the setup of the test, and Figure 7 illustrates the 360° camera feed which is used by the lighting algorithm.

Figure 7. *Panoramic light camera feed within test setup.*

In a close-up view (Figure 8) we compare the lighting on the virtual object to its 3D printed replica in the real scene. As shown in the image, we achieve a fairly compelling resemblance between simulated and real lighting.

Edge Solution

We created a similar verification scenario for our edge solution where we use a Raspberry Pi Zero unit connected to a 180° fisheye camera to simulate diffuse lighting conditions around an AR marker attached to the unit. This system was able to compute lighting coefficients at 5-10 fps. The camera feed in this case only covers the upper hemisphere of the scene environment (Figure 9).

The results from the edge solution show consistent lighting of the synthetic object based on the surrounding real scene (figure 10). To compare the two approaches we identified the wireless connection between server and camera as the main technical challenge in server solution as it does not always guarantee a consistent video stream, while the edge solution solves this issue with a wired connection between camera and processing unit. However, the edge solution faces difficulties in terms of frame rate as it has limited computational power compared to the server solution.

Figure 8. *Comparison of lighting on real and virtual models.*

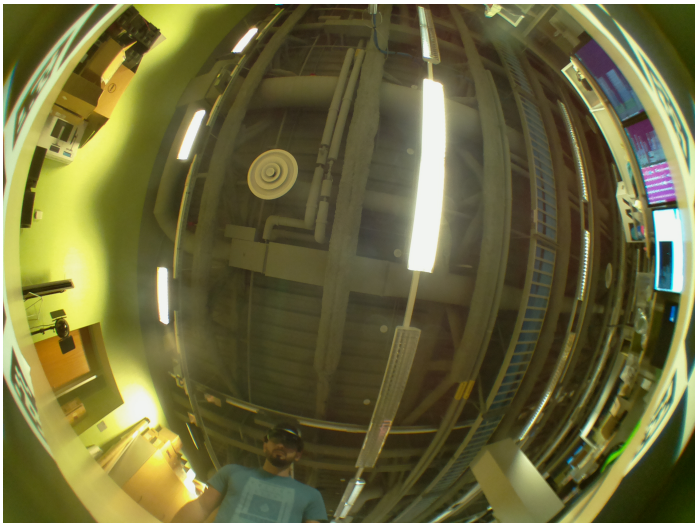

Figure 9. *Light feed from Raspberry Pi with fisheye camera in test setup.*

Conclusions

We introduced an architecture for an AR system which can simulate diffuse lighting under dynamic environment lighting conditions with minimum computational and hardware cost. By taking advantage of the SH representation of the environment map, our algorithm produces a photorealistic real-time AR experience for multiple users at interactive frame rates. Using this technique, multiple rendering units can use the same coefficients to replicate diffuse reflection in their AR environments instead of calculating them separately.

A generic architecture and a simple lighting algorithm make our solution compatible with various compute platforms and AR

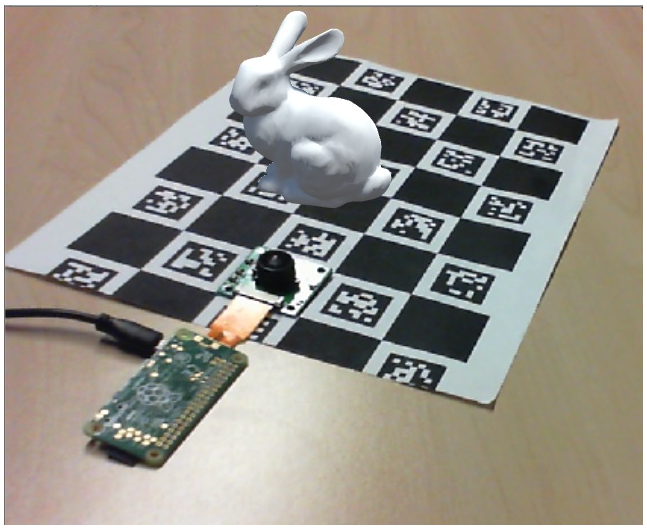

Figure 10. *Light reflection on a synthetic object in the test setup for the edge solution.*

technologies. We integrated our system with the popular Unity 3D platform and Vuforia, and tested AR applications for different AR devices, including the Microsoft Hololens and the Oculus Rift. We implemented our solution with two hardware setups, a server and an edge type approach. The results of our tests show reasonable reflection similarities between real and virtual models and seamless augmentation of synthetic objects.

We believe our method provides a useful framework for producing real-time AR at minimum computational cost. This efficiency provides opportunities for future efforts towards photorealistic AR applications on portable devices with limited computational resources.

References

[1] P. E. Debevec, Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography, Proc. of the 25th annual conference on Computer graphics and interactive techniques, p. 189 ff. (1998).

[2] Agusanto, Kusuma, Li Li, Z. Chuangui, N. W. Sing. Photorealistic rendering for augmented reality using environment illumination. Proc. the 2nd IEEE/ACM international symposium on mixed and augmented reality, p. 208 ff. (2003).

[3] S. Pessoa, G. Moura, J. Lima, V. Teichrieb, J. Kelner, Photorealistic Rendering for Augmented Reality: A Global Illumination and BRDF Solution, Virtual Reality Conference (VR) IEEE, p. 3 ff. (2010).

[4] R. Ramamoorthi, P. Hanrahan, On the relationship between radiance and irradiance: determining the illumination from images of a convex Lambertian object, J. Optical Society of America, 18(10), 2448-2459 (2001).

[5] R. Ramamoorthi, P. Hanrahan, An efficient representation for irradiance environment maps, Proc. of the 28th annual conference on Computer graphics and interactive techniques, p. 189 ff. (2001).

[6] P. P. Sloan, J. Kautz, J. Snyder, Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments, J. ACM Transactions on Graphics (TOG) Vol. 21, No. 3, p. 527 ff. (2002).

[7] M. Kanbara, N. Yokoya, Real-time Estimation of Light Source Environment for Photorealistic Augmented Reality, Proc. of the Pattern Recognition, 17th International Conference, Vol. 2, p. 911 ff. (2004).

[8] P. Supan, I. Stuppacher, M. Haller, Image based shadowing in real-time augmented reality, J. Virtual Reality, 5(3):17 (2006).

[9] C. B. Madsen and R. E. Laursen. A scalable gpu-based approach to shading and shadowing for photorealistic real-time augmented reality. In GRAPP (GM/R), p. 252261 (2007).

[10] T. Grosch, T. Eble, S. Mueller. Consistent interactive augmentation of live camera images with correct near-field illumination, Proc. the 2007 ACM symposium on Virtual reality software and technology, p. 125 ff. (2007).

[11] P. Supan, I. Stuppacher, Interactive image based lighting in augmented reality, Central European Seminar on Computer Graphics. Vol. 17, p. 18 ff. (2006).

[12] K. Rohmer, W. Bschel, R. Dachselt, T. Grosch, Interactive Near-Field Illumination for Photorealistic Augmented Reality with Varying Materials on Mobile Devices. IEEE transactions on visualization and computer graphics, 21(12), p. 1349-1362. (2015).

[13] G. Finlayson, C. Fredembach, M. S. Drew, Detecting illumination in images, In Computer Vision, IEEE 11th International Conference on Computer Vision (p. 1-8). (2007).

[14] J. F. Lalonde, A. A Efros., S. G. Narasimhan, Estimating natural illumination from a single outdoor image, Computer Vision, IEEE 12th International Conference, (pp. 183-190). (2009).

[15] J. F. Lalonde, A. A Efros., S. G. Narasimhan, Estimating the natural illumination conditions from a single outdoor image. International Journal of Computer Vision, 98(2), pp. 123-145. (2012).

[16] Y. Hold-Geoffroy, K. Sunkavalli, S. Hadap, E. Gambaretto, J. F. Lalonde, Deep outdoor illumination estimation. In IEEE Conference on Computer Vision and Pattern Recognition (Vol. 1, No. 2), pg. 6. (2017, July).

[17] J. Zhang, J. F. Lalonde. Learning High Dynamic Range from Outdoor Panoramas. arXiv preprint arXiv:1703.10200 (2017).

Author Biography

Kamran Alipour received his B.Sc. degree in aerospace engineering from KNTU, Tehran, Iran and his M.Sc. in aerospace engineering from Sharif University of Technology, Tehran Iran. Has been a Ph.D. student in the Department of Computer Science at UC San Diego since 2016. His areas of study include virtual reality, augmented reality, and image-based global illumination in real-time applications.

Dr. Jürgen P. Schulze is an Associate Research Scientist at UCSD's Qualcomm Institute, and an Associate Adjunct Professor in the computer science department, where he teaches computer graphics and virtual reality. His research interests include applications for virtual and augmented reality systems, 3D human-computer interaction, and medical data visualization. Dr. Schulze is the director of the Immersive Visualization Laboratory at UCSD's Qualcomm Institute.