Precomputation-Based Real-Time Rendering

Lecture #11: Wednesday, 7 October 2009
Lecturer: Ravi Ramamoorthi
Scribe: James O'Shea

Abstract

One of the goals in computer graphics is to generate photorealistic images. Despite the advent of faster computers and more sophisticated algorithms, high-quality results can be difficult or impossible to achieve using the traditional rendering pipeline. A relatively new trend in the field is to use acquired data from real images in order to produce high-quality renderings and photorealistic appearances. Commonly referred to as image-based rendering (IBR), this method (in its purest form) completely bypasses the processing of scene geometry. Given the recent successes of this technique, researchers have also started applying the idea to precomputed synthetic data in an effort to achieve offline-rendering quality with real-time rendering speeds. This lecture will explore the technique in detail.

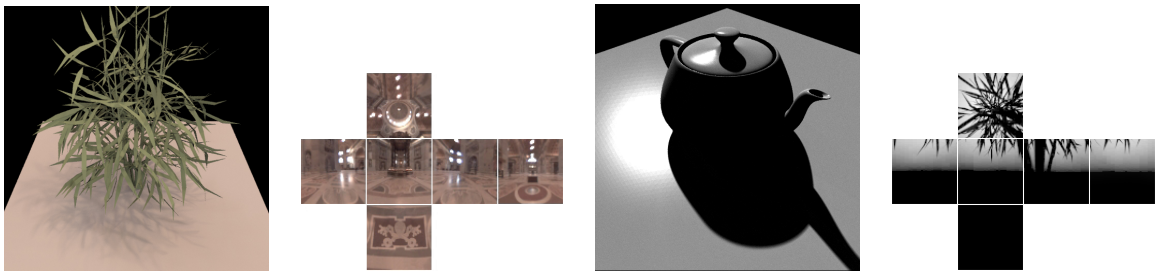
1 Introduction

Image-based rendering has recently emerged as a new method for generating realistic images in real-time frame rates. One of the main shortcomings of the technique is the need for real image data, which may be difficult, expensive, or impossible to acquire for all desired outputs. An alternative approach is to use high-quality synthetic data as the source imagery. This approach can take advantage of time-consuming rendering techniques to generate the images while still achieving real-time interactive results.

The main challenges with this method concern data storage and signal processing. For a highly-complex multi-dimensional input data set, the data representation and storage issues are critical. The data must often be stored efficiently while still providing immediate access to it. Because the images are pre-computed, some aspect of the scene must remain static. Most of the recent approaches assume the geometry is static and the lighting or viewpoint can change.

2 Precomputation-based Relighting

We first examine precomputation-based rendering through the problem of relighting a scene. In this type of situation, the geometry and viewpoint are both assumed to be static. Under these assumptions, the problem can be written as a product of matrices:



(a) Output image P (b) Input lighting cubemap L (c) Light transport matrix T column (d) Light transport matrix T row

Figure 1: *Example input and output data for precomputation-based relighting. Figure (a) represents an example output image P. Figure (b) is an example of the input lighting L. Figure (c) provides an example of one of the columns from the light transport matrix T. Each column represents all the pixel values for a single light value. Figure (d) provides an example of one of the rows of the light transport matrix. Each row represents the value for an individual pixel for all the light sources in the input cubemap L.*

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \vdots \\ P_N \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & \dots & T_{1M} \\ T_{21} & T_{22} & \dots & T_{2M} \\ T_{31} & T_{32} & \dots & T_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ T_{N1} & T_{N2} & \dots & T_{NM} \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ L_3 \\ \vdots \\ L_M \end{bmatrix} \tag{1}$$

In this equation P is the output image, T is the precomputed light transport matrix, and L is the input lighting for the scene (as a cubemap vector). See figure 1 for examples of the output image and the input lighting. Each column of the light transport matrix T represents all the pixel values for a single light value in L. Each row of the light transport matrix T represents the values for a single pixel in the image for all of the light values in the input lighting matrix L.

The main problem with this approach is that the matrices are very large. Even with a simple 6 x 64 x 64 cubemap and a modestly sized 512 x 512 output image, the light transport vector will have over 6.4×10^9 entries. Given this size, computing the output matrices is an intractable problem.

2.1 Compression Methods

To reduce the size of the precomputed light transport matrices, we can compress the data in order to reduce its complexity. Several signal processing techniques have been adapted for this purpose in order to approximate the complex light transport data using a small subset of components.

Figure 2: *Example output image using precomputed radiance transfer based on 25 spherical harmonic components. This image was generating with shadows and interreflections.*

2.1.1 Spherical Harmonics

One of the first approaches to compress the light transport function was to use spherical harmonics to approximate the lighting in the scene. Prior work has demonstrated that the first few spherical harmonics can be used to effectively approximate the lighting in low-frequency conditions [5]. This method relies on the linearity of light transport. By using 25 of the spherical harmonic components, the complexity of the lighting is reduced and the light transport matrix has only 25 columns. Each pixel in the output image can now be computed using a simple dot product between two matrices of 25 entries each, which can easily be done in hardware. See figure 2 for an example result using this technique. This technique is simple and effective, but its application is primarily limited to low-frequency lighting environments.

2.1.2 PCA or SVD Factorization

Another compression approach is to factor the transport matrix into its principal components and use those to compute the lighting. As long as T has a low rank, this technique can reduce the complexity of the lighting to allow real-time rendering frame rates. This idea has been applied to factoring 4D BRDFs, surface light fields, and orientation light fields, but it is not useful as a general precomputation approach to relighting since the light transport matrix is not normally low-dimensional.

For highly complex lighting which includes all frequencies, the number of components required to reliably approximate the lighting will often be too high to make it a useful approach. One way around this limitation is to use apply the PCA locally to clusters. Although light transport is high-dimensional at a global level, it is locally low-dimensional [6].

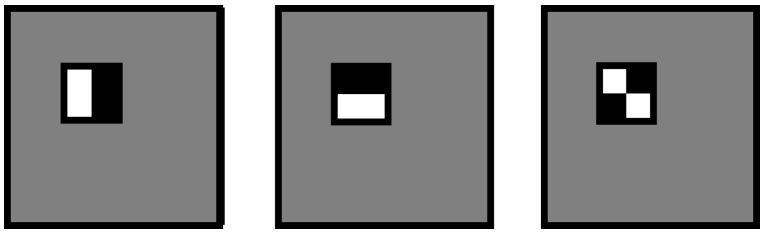
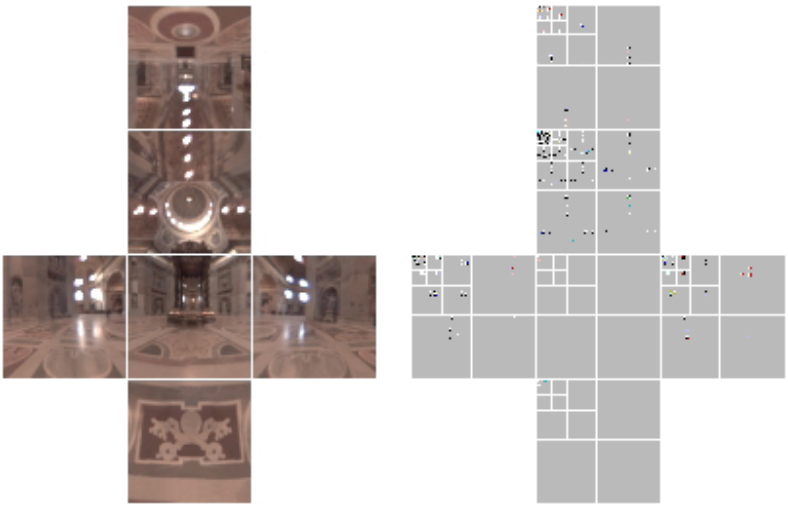


Figure 3: *Set of Haar wavelets used to compress the light transport matrix.*



(a) Lighting environment (b) Wavelet approximation

Figure 4: *Example lighting environment and its wavelet approximation*

2.1.3 Wavelets

A final compression approach is to use wavelets to factor the rows of the light transport matrix [2]. Wavelets have the advantage that they can be applied at different scales: small wavelets for high-frequency lighting and large wavelets for low-frequency lighting. In this context, Haar wavelets are typical employed (Figure 3). They can also be applied dynamically for each frame to better approximate the lighting, as opposed to the spherical harmonics approach which uses the same 25 basis functions for all frames. Figure 4 illustrates how a lighting environement can be approximated using wavelets. Figure 5 compares this method to the spherical harmonic approach.

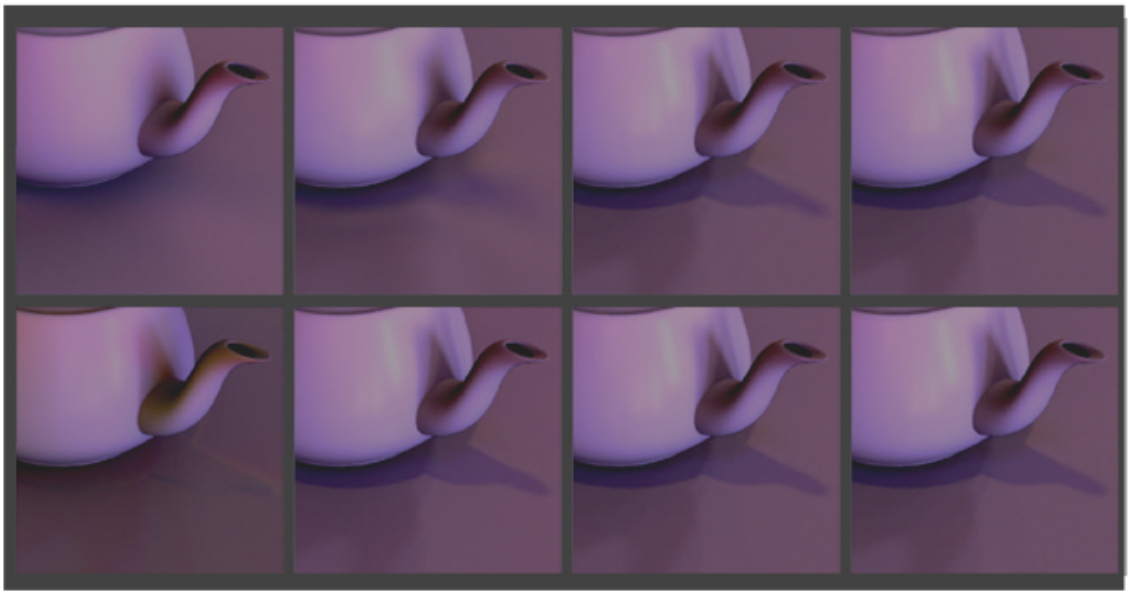


Figure 5: *Comparison of results from spherical harmonics approximation (top row) and wavelet approximation (bottom row). The number of components used to approximate the lighting increases from left to right.*

3 Precomputation-based Change of View

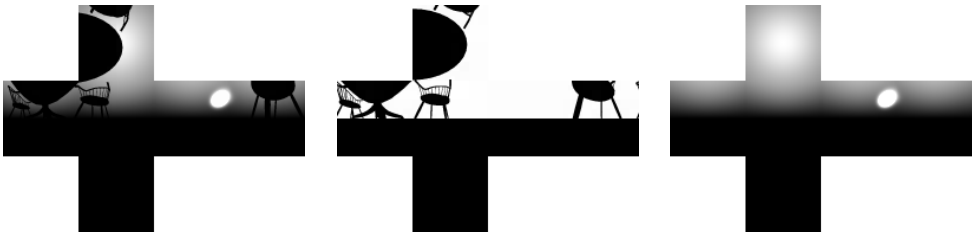
The approaches to relighting a scene can also be applied to changing the view of a scene. When combined with relighting, this problem requires a 6D light transport space: 2D for view, 2D for lighting, and 2D for geometry.

3.1 Spherical Harmonics

Low frequency changes in lighting and changes in view can be approximated using the first 25 spherical harmonic components for each. This leads to a $25 \times 25 = 625$ element vector for each vertex. As with the original application in approximating lighting, this method fails to reliably reproduce high-frequencies.

3.2 Factored BRDFs

Another approach is to break the problem into manageable subproblems which can reliably be approximated using existing methods. Several recent papers describe a technique which factors the BRDF to depend on incident and outgoing light. The incident components are used to approximate the view-independent relighting, whilte the outgoing components take view into account. These can be linearly combined to produce an output image. This method is practical, but the factorization is not necessarily optimal. For Phong BRDFs, Mahajan et al. showed that the number of terms needed to accurately



(a) 6D Light Transport (b) 4D Visibility (c) 4D BRDF

Figure 6: *Example showing how the full 6D light transport matrix (left) can be broken down to the 4D visibility matrix (middle), and the 4D BRDF matrix (right).*

approximate BRDF grows linearly with the Phong exponent [1]. In general, the number of terms needed is related to the frequency content of the BRDF along the half-angle direction.

3.3 Triple Products

A more sophisticated approach is known as the triple product integral [3]. This method breaks the 6D light transport precomputation into the product of the 4D visibility matrix, and the 4D BRDF matrix (Figure 6). The light transport is reorganized in this way in order to make it possible to deal with the large matrices. Rather than deal with a matrix with 10^{12} samples, we break it up into two matrices that have 10^8 samples each (for a 512×512 image). This makes the problem tractable.

To explain how this method works, we first examine the reflection equation:

$$B = \int_{S^2} L(\omega)\, V(\omega)\, \tilde{\rho}(\omega)\, d\omega \tag{2}$$

In this equation, the reflectance B is a function of the lighting $L(\omega)$, the viewpoint $V(\omega)$, and the BRDF $\tilde{\rho}(\omega)$. After expanding the angular dependence of lighting, visibility, and BRDF in terms of basis functions $\Psi(\omega_i)$, we get the following:

$$B = \int_{S^2} \Big(\sum_i L_i \Psi_i(\omega)\Big) \Big(\sum_j V_j \Psi_j(\omega)\Big) \Big(\sum_k \tilde{\rho}_k \Psi_k(\omega)\Big)\, d\omega \tag{3}$$

We can simply this equation by moving the summations and coefficients that do not depend on ω outside of the integral

$$\begin{aligned} B &= \sum_i \sum_j \sum_k L_i V_j \tilde{\rho}_k \int_{S^2} \Psi_i(\omega)\Psi_j(\omega)\Psi_k(\omega) d\omega \\ &= \sum_i \sum_j \sum_k L_i V_j \tilde{\rho}_k C_{ijk} \end{aligned}$$

Figure 7: *Example image demonstrating the type of output possible using the triple product method. The output scene is rendered on the left. An example lighting cubemap, 4D visibility cubemap, and 4D BRDF cubemap are shown on the right. The pixel for which these cubemaps apply is shown in red in the lefthand image of the scene.*

The C_{ijk} term are known as the tripling coefficients. This method breaks the problem up into manageable pieces which are still within the space and computation limits of current computers. This approach is able to handle all frequencies effectively while still providing frame rates on the order of 3-5 seconds per frame. Figure 7 shows an example output image using this method.

4 Summary

The use of precomputed light transport matrices for real-time rendering is essentially a data-management and signal processing problem. A 6D precomputation space is necessary for changing lighting and viewpoint, but the resulting light transport matrices are too large for solving in real time. There have been several approaches based on existing data compression methods, and new techniques have been developed as well. Ramamoorthi presents a more thorough discussion of these techniques in his recent survey of precomputation- based rendering [4].

References

[1] D Mahajan, Y Tseng, and R Ramamoorthi. An analysis of the in-out brdf factorization for view-dependent relighting. *Computer Graphics Forum*, Jan 2008.

[2] R Ng, R Ramamoorthi, and P Hanrahan. All-frequency shadows using non-linear wavelet lighting approximation. *ACM Transactions on Graphics (TOG)*, Jan 2003.

[3] R Ng, R Ramamoorthi, and P Hanrahan. Triple product wavelet integrals for all-frequency relighting. *International Conference on Computer Graphics and ...*, Jan 2004.

[4] Ravi Ramamoorthi. Precomputation-based rendering. *Found. Trends. Comput. Graph. Vis.*, 3(4):281–369, 2009.

[5] P Sloan, J Kautz, and J Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *Proceedings of the 29th annual conference on ...*, Jan 2002.

[6] P Sloan, J Snyder, and J Hall. Clustered principal components for precomputed radiance transfer. *US Patent App. 10/641,472*, Jan 2003.