# Error Analysis of Precomputed Radiance Transfer for Deforming Geometry

João L. Cardoso [*†] Paul G. Kry [*]

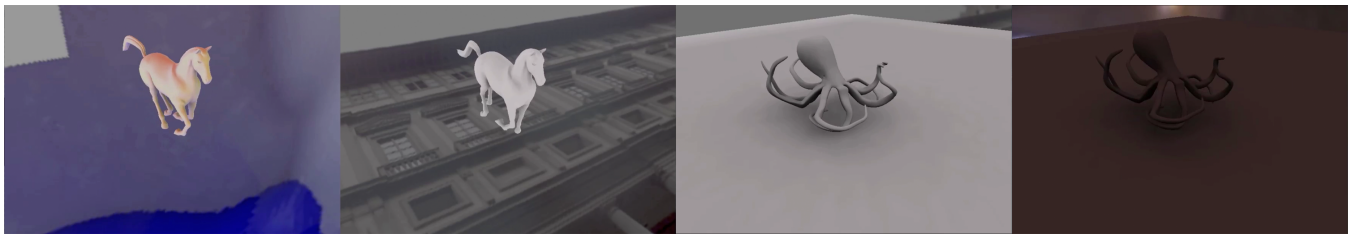

**Figure 1:** *Ambient light can be changed in real-time as the animation is rendered.*

## Abstract

In this project we explore the approximation errors associated with precomputed light transport techniques on lambertian surfaces.

Low frequency lighting is described on a spherical harmonic basis. Transfer functions for direct lighting components are computed for every pose on each vertex of the model. The resulting animation is then approximated to a smaller data set with less memory requirements using a principal component analysis of both the geometry and transfer functions across the set of poses.

At run-time, the geometry and transfer functions are reconstructed as necessary from the resulting singular value decomposition, and an environment light applied to the transfer functions. An analysis of the theoretical and visual errors due to the reconstruction and spherical harmonic approximations is then performed.

**Keywords:** radiance transfer, radiosity, global illumination, lambertian surfaces, principal component analysis

## 1 Introduction

Although the underlying principles of lighting are well understood, their efficient computation is still a challenging problem. Many features - such as diffuse and glossy shading, lighting from area sources, soft shadows, indirect illumination, caustics, subsurface scattering and interreflections - are so computationally expensive that they can take minutes or hours to compute on average complexity scenes [2010; 2012], making the general approaches unusable for real-time rendering.

Commonly, this is solved by resorting to precomputation based techniques, such as precomputed radiance transfer. This category of methods traditionally constraint the scene geometry to be rigid and, under this assumption, the surface reflection proprieties of the model can be computed, allowing for real-time rendering of features such as self-shadowing and self-interreflection on **any** lighting environment (figure 1) [2002; 2012].

Yet, this constraint greatly restricts the practical application of these techniques. For that reason, a number of techniques have been proposed over the past years, that cope precomputed radiance transfer techniques with animated geometry. [Sloan et al. 2005] proposed the use of zonal harmonics, which works particularly well for effects such as bumps or other details. [James and Fatahalian 2003] proposed the use of impulse response functions, which allow for physical interaction with the models under certain constraints. Our approach resembles more the work developed by [Fatahalian 2003], which resorts to a singular value analysis of the transfer functions computed for each possible poses.

## 2 Radiance Transfer

On a more physically based rendering pipeline, geometry is commonly lighted by computing the resulting color on some points on the model, and the remaining area by interpolation of these points. Precomputed radiance transfer techniques, on the other hand, turn this idea upside down, and compute the surface proprieties - how they reflect different types of light - on those points instead. Then, the ambient light can be approximated as a composition of these types and the color computed.

A pertinent question is how should these points be chosen. A common approach is sampling them across the surface using Monte-Carlo techniques [2010]. Yet, due to out interest in rendering a series of poses, we compute the transfer on each vertex, which ensures all rendered points match across the entire simulation.

Another important factor is the reflection model to be used. In this project we focus on lambertian surfaces, as this model defines that the brightness of a surface to an observer is the same regardless of the observer's angle of view. Yet, other models could have been used, as shown by [Liu et al. 2004], which extended this idea to support all frequency transfer using a factorization of bidirectional reflectance distribution functions.

### 2.1 Spherical Harmonics

One must also chose a representation for the transfer functions. In this project we encode them using spherical harmonic coefficients, as described by [Sloan et al. 2002]. On a strict definition, their functions define an orthonormal basis over the unit sphere, analogous to the Fourier transform over that domain. Yet, the mathematical def-

[*]McGill University
[†]University of Coimbra

inition and algorithms used for the coefficient computation are out of the scope of these project, and hence will not be covered on this report.

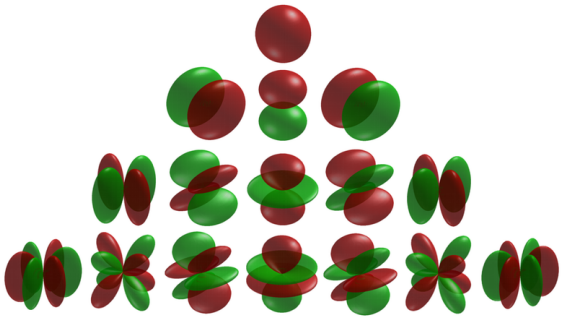

**Figure 2:** *Visual representations of the first few spherical harmonics. Red portions represent regions where the function is positive, and green portions represent regions where the function is negative.*

There is an infinite amount of spherical harmonic coefficients, ordered from low to high frequency basis functions over the sphere, and divided into a series of bands. Theoretically, computing this infinite recurrence would give an exact representation of the transfer. Computing a finite number constitutes is thereby a valid approximation, as long as it corresponds to a number of bands.

### 2.2 Reconstruction

Let $l$ be the maximum band to be used. Then $n$, the number of coefficients, is:

$$n = (l+1)^2 \tag{1}$$

Suppose $T = [T_0, T_1, ..., T_n]$ is the precomputed radiance transfer vector of some point $p$ on a surface, and $A = [A_0, A_1, ..., A_n]$ the approximation of the ambient light on the SH basis. Then the color $c$ of the surface on $p$ is obtained as:

$$c = T \cdot A \tag{2}$$

Gouraud shading is then used to interpolate the resulting colors on the vertices across the surface of the model.

## 3 Storage

The major drawback of precomputed radiance transfer on animated geometry - besides precomputation time - is the memory requirements. While precomputing the transfer for every pose is not much of an issue, storing and loading it into memory is impractical on current hardware.

For instance, the octopus is constituted by about 104000 vertices and 100 poses. Precomputing and storing the spheric harmonic coefficients for every pose using 3 bands would thereby require $104000 * 100 * 9 * 3 = 280800000$ floating point values just to represent the transfer functions the three RGB components. While this is not too large to fit in random access memory of the contemporary common computer, it is still too high to be used in practical situations.

Instead, we approximate separately the geometry and SH coefficients to a smaller data set.

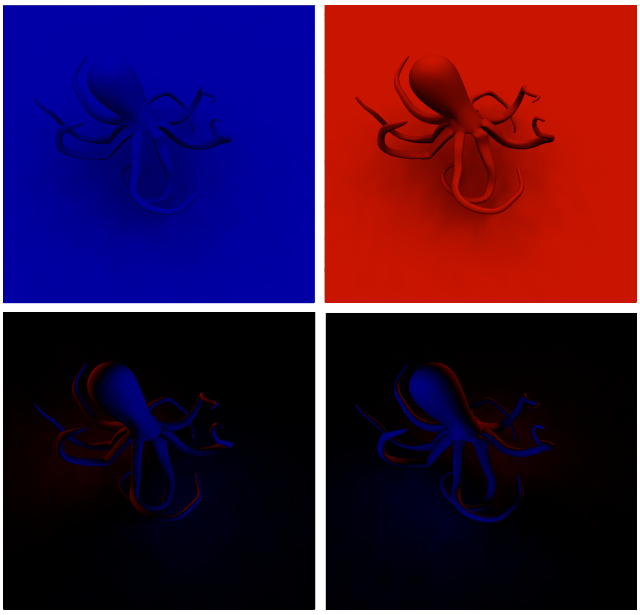

**Figure 3:** *The first four spherical harmonic coefficients, from top-left to bottom-right, as viewed using our renderer. Blue represents regions where the coefficient is positive, and red where is negative.*

### 3.1 Singular Value Decompostion

Principal component analysis is used to obtain a basis for a data set such that variation of the data is maximized along the coordinate axes. Data reduction is achieved by ignoring the axes of lowest variation, thereby projecting the original data into a lower dimensional space with minimal loss of accuracy. On a loose sense, this can be seen as redefining the animation as a smaller set of features, which can be interpolated to reconstruct the original animation.

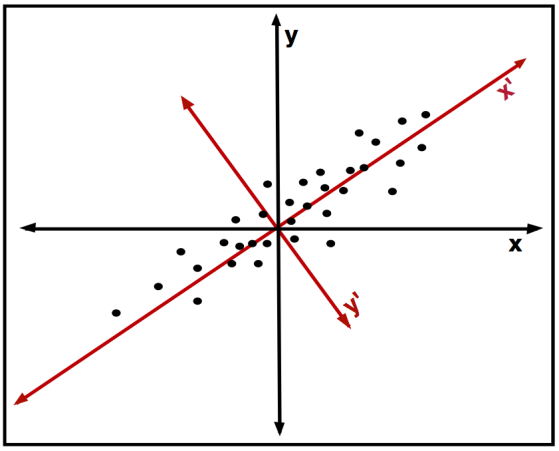


**Figure 4:** *Variation of the data is greatest along the newly defined axes than on the canonical ones. $y'$ could be ignored to perform a data reduction, thus projecting it into a 1D data set.*

To make use of this algorithm, we first define two new matrixes - one for the geometry and another for the transfer. The operations performed on both are the same, and hence we will reference an abstract matrix $A$ from this point forward.

Let $P_i$ be a vector containing the data relative to the ith pose of the

model. Let $A_{m\times n}$ be our new matrix, where $m$ is the size of $P_i$ and $n$ the number of poses. Then $A$ is defined as:

$$A = \begin{bmatrix} P_0^T & \cdots & P_n^T \end{bmatrix} \tag{3}$$

For example, the $A$ matrix for the geometry has the following representation:

$$A_{geometry} = \begin{pmatrix} x & \cdots & x \\ y & \cdots & y \\ z & \cdots & z \\ \vdots & \ddots & \vdots \\ x & \cdots & x \\ y & \cdots & y \\ z & \cdots & z \end{pmatrix} \tag{4}$$

Factorizing $A$ directly is not very efficient, as that would make the most important features of the data set to be it's average. Hence, we first calculate the mean of every row, $M$, and subtract it from $A$, so that every value is the distance from the mean:

$$M_i = \sum_{k=0}^{n} \frac{A_{i,k}}{n} \tag{5}$$

$$A'_{\text{row i}} = A_{\text{row i}} - M_i \tag{6}$$

We then perform the singular value decomposition:

$$A'_{m\times n} = U_{m\times n} \cdot S_{n\times n} \cdot V_{n\times n} \tag{7}$$

where $U$ and $V^T$ are two unitary matrixes and $S$ ia diagonal matrix whose entries contain the singular values of $A'$. These values indicate the variation of the corresponding basis and are ordered in decreasing magnitude. One might thus discard the $n-t$ lower variation axes, where $t$ is an integer, by performing a truncation of the three matrixes.

$$A'_{m\times t} = U_{m\times t} \cdot S_{t\times t} \cdot V_{t\times t} \tag{8}$$

### 3.2 Reconstruction

While one could reconstruct the entire matrix $A'$, that would defeat the entire purpose of the decomposition. Instead, we will reconstruct and store in memory one pose at the time:

$$P_i \simeq M + U \cdot S \cdot V_{\text{column i}} \tag{9}$$

Furthermore, $S$ and $V$ are square matrixes of the same size, and thus can be premultiplied before rendering:

$$V' = S \cdot V \tag{10}$$

$$P_i \simeq M + U \cdot V'_{\text{column i}} \tag{11}$$

## 4 Error Analysis

Regarding the singular value decompositions of the geometry and transfer coefficients, both theoretical and visual relative errors were computed. The first is the expected error that, given a set of singular values, would come from truncating $S$. It is computed has:

$$E_{theoretical} = \frac{\sum_{i=t+1}^{n} S_{i,i}}{\sum_{i=0}^{n} S_{i,i}} \tag{12}$$

where $t$ is the number of components retained. Visual errors are computed by simple comparison of the resulting images, pixel by pixel. Let $I_f$ be the resulting image using all components or bands, and $I_p$ using a specified number:

$$E_{visual} = \frac{|I_f - I_p|}{|I_f|} \tag{13}$$

Using the above formulas, the results obtained for the octopus model are presented in the table below:

| Geometry Components | 103 | 20 | 7 | 4 | 2 |
|---|---|---|---|---|---|
| Theoretical Error | 0% | 0.58% | 2.23% | 4.68% | 8.3% |
| Visual Error | 0% | 2.83% | 4.41% | 5.37% | 6% |

| Transfer Components | 103 | 20 | 7 | 4 | 2 |
|---|---|---|---|---|---|
| Theoretical Error | 0% | 7.67% | 15.56% | 20.24% | 25.5% |
| Visual Error | 0% | 0.64% | 1.2% | 1.75% | 2.22% |

| Spherical Bands | 7 | 4 | 2 |
|---|---|---|---|
| Visual Error | 0%* | 0.5% | 1.6% |

As we can see, the expected error was smaller than the perceptible one for the geometry. On the other hand, the visible error when reducing the transfer dimensional space is much smaller than what expected. This fits the common sense reasoning that one is much more likely to notice an object is out of place, than its color is slightly off. It is also very favorable to our needs, as the transfer matrix tends to be much larger than the geometry: each vertex is represented with three floats for the geometry, but three are also required for each transfer coefficient.

We have assumed the results obtained from using seven spherical bands to be the ground truth. This is obviously not correct, but it is close enough to provide us a good approximation regarding the visual errors when reducing the number of bands.

## 5 Implementation

The implementation of the project required the development and extension of a number of independent applications.

### 5.1 Pbrt

For the transport precomputation, we have used a modified version of *pbrt2* [2010], an open source implementation of a physically based renderer, as described in the second edition of the book. Although the renderer supports the computation of transport for direct lighting, the user is given no control on how these should be computed, nor the ability to perform operations on such values - they are only used for rendering images. So it had to be extended with new rendering modes to:

- Compute the direct radiance transfer functions on specified point/normal pairs of a scene into a readable file.

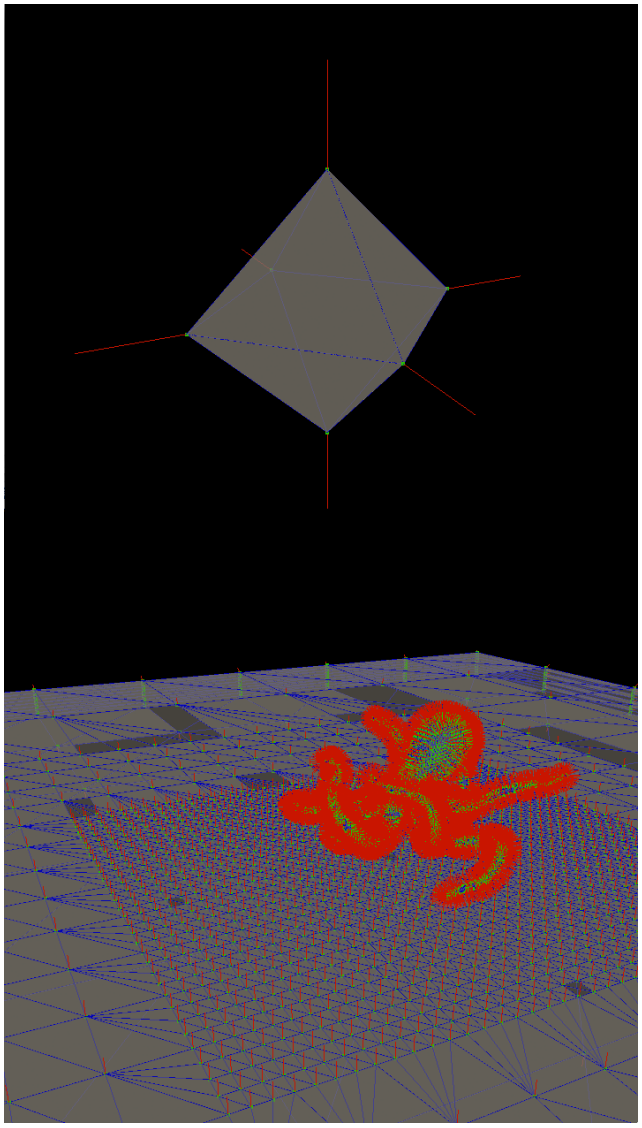

**Figure 5:** *(top) the normal of each vertex of the octahedron is computed as an average of all the vertex normals in the faces; (bottom) the obj renderer was constructed with the purpose of analyzing a model geometry and normals in mind.*

- Approximate an ambient map on a spherical harmonic basis into a new file.

Other minor changes were made, such as adding the ability to automatically render several scenes in sequence - vital for computing several poses.

### 5.2 Obj Models

The poses we wanted to use were stored in the *obj* format, and thus an *obj* loader was developed. Let $F_v$ be the faces a vertex $v$ belongs to, and $N(v, f)$ the normal of a vertex on a given face. For the purposes of transport precomputation, the loader computes $n$, the normal of $v$ as:

$$n = \frac{\sum_{f \in F_v} N(v, f)}{|n|} \tag{14}$$

Using the loader, an application was made to export the poses as *pbrt* scenes, so that the transfer functions could be computed. For debugging purposes, an *obj* renderer was also created.

Additionally, a small script that automatically adds common geometry (e.g: a floor) to a set of poses was created to modify our models.

### 5.3 Paim Format

After *pbrt*'s precomputation has taken place, another exporter reads the poses and transfer files, and groups the information as the geometry and a transfer matrixes. It then performs the singular value decomposition as explained in section 3.1, for which we have used *EJML*, a java matrix library. The results are stored in a binary format file, which was labeled *paim* (pre-animated and illuminated model).

Just as for the *obj* files, a *paim* renderer was created. Using this application one might dynamically set the number of geometry and transfer components and spherical harmonics bands to be used. The model can then be viewed on any environment light, and controls for both exposure and gamma are provided. Alternatively, each spheric harmonic coefficient can be visualized as a red and blue color palette over the surface (figure 3).

The error analysis was performed using Matlab. For the purpose of computing visual errors, the renderer also supports taking a snapshot of the color buffer and storing into a readable text file for comparison.

## 6 Acknowledgements

We would like to thank Derek Nowrouzezahrai for his guidance regarding methods for light precomputation.

## References

FATAHALIAN, K. 2003. *Real-Time Global Illumination of Deformable Objects*. Master's thesis, Carnegie Mellon University.

JAMES, D. L., AND FATAHALIAN, K. 2003. Precomputing interactive dynamic deformable scenes. *ACM Trans. Graph. 22*, 3 (July), 879–887.

LIU, X., SLOAN, P.-P., SHUM, H.-Y., AND SNYDER, J. 2004. All-frequency precomputed radiance transfer for glossy objects. In *Proceedings of the Fifteenth Eurographics conference on Rendering Techniques*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, EGSR'04, 337–344.

PHARR, M., AND HUMPHREYS, G. 2010. *Physically Based Rendering, Second Edition: From Theory To Implementation*, 2nd ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

RITSCHEL, T., DACHSBACHER, C., GROSCH, T., AND KAUTZ, J. 2012. The state of the art in interactive global illumination. *Comput. Graph. Forum 31*, 1 (Feb.), 160–188.

SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. 21*, 3 (July), 527–536.

SLOAN, P.-P., LUNA, B., AND SNYDER, J. 2005. Local, deformable precomputed radiance transfer. In *ACM SIGGRAPH 2005 Papers*, ACM, New York, NY, USA, SIGGRAPH '05, 1216–1224.

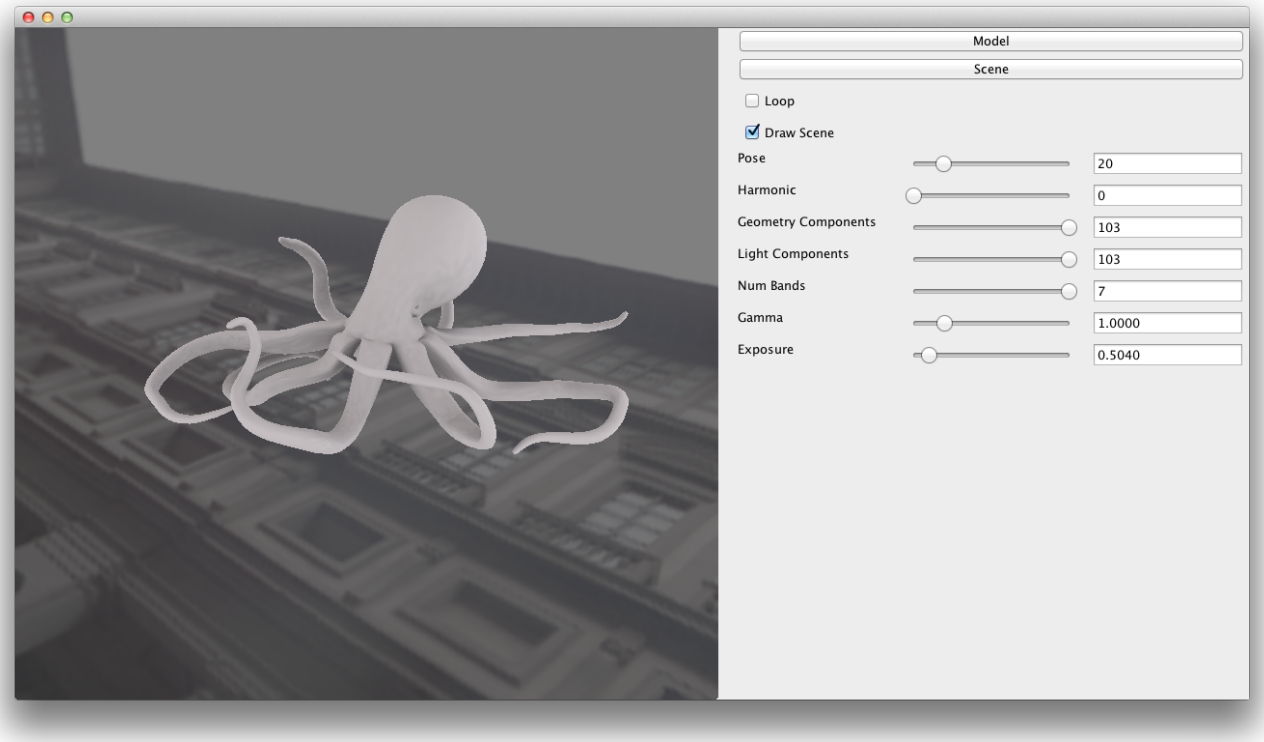


**Figure 6:** *our* paim *model renderer, displaying an octopus model on a street environment light. Exposure is reduced for a clear visualization of self-occlusion.*