All-Frequency Precomputed Radiance Transfer using Spherical Radial Basis Functions and Clustered Tensor Approximation

Yu-Ting Tsai* Zen-Chung Shih†

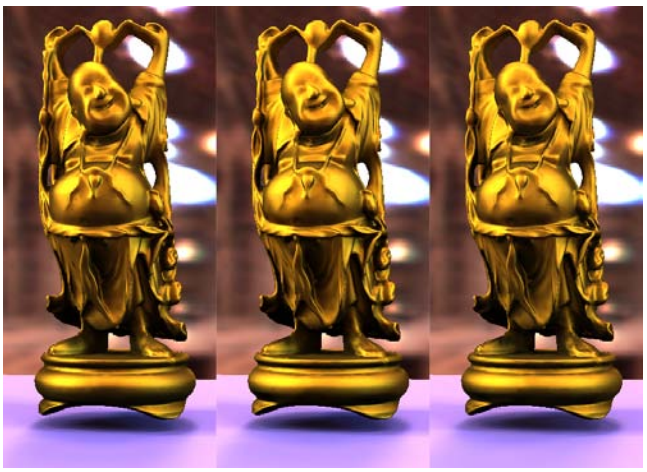
Department of Computer Science, National Chiao Tung University, Taiwan

Abstract

This paper introduces a new data representation and compression technique for *precomputed radiance transfer* (PRT). The light transfer functions and light sources are modeled with *spherical radial basis functions* (SRBFs). A SRBF is a rotation-invariant function that depends on the geodesic distance between two points on the unit sphere. Rotating functions in SRBF representation is as straightforward as rotating the centers of SRBFs. Moreover, high-frequency signals are handled by adjusting the bandwidth parameters of SRBFs. To exploit inter-vertex coherence, the light transfer functions are further classified iteratively into disjoint clusters, and tensor approximation is applied within each cluster. Compared with previous methods, the proposed approach enables real-time rendering with comparable quality under high-frequency lighting environments. The data storage is also more compact than previous all-frequency PRT algorithms.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture; G.1.2 [Numerical Analysis]: Approximation—Special function approximations; E.4 [Coding and Information Theory]: Data compaction and compression

Keywords: Real-Time Rendering, Illumination, Precomputed Radiance Transfer, Spherical Radial Basis Functions, Tensor Approximation, Non-Linear Optimization



(a) Raw PRT data (b) All-frequency CPCA (13.34 fps) (c) Our approach (51.12 fps)

Figure 1: Comparison of the rendered results between the all-frequency CPCA algorithm [Liu et al. 2004] and our approach. The lighting environment adopted in the result of all-frequency CPCA was approximated with 972 coefficients using the area-weighted wavelet method [Ng et al. 2003], whereas our approach modeled the environment with 162 scattered SRBFs (972 parameters). Note that our approach achieves real-time performance with image quality comparable to all-frequency CPCA.

1 Introduction

Real-time rendering of global light transport remains a major challenge in computer graphics. The difficulty lies in interactively synthesizing realistic images with limited computing power. The *precomputed radiance transfer* (PRT) algorithm [Sloan et al. 2002] has recently attracted much attention owing to its ability to allow real-time rendering of complex objects under dynamic lighting environments. However, previous PRT methods either only handle low-frequency lighting environments, or suffer from the unwieldy size of PRT data sets even after compression. For dynamic scenes, the amount of PRT data sets further expands to an impractical degree for real-time applications. The enormous PRT data sets often prohibit high-quality rendering, subsequently restricting practical applications of the PRT algorithm.

This paper presents a novel method to model and compress raw PRT data sets. The proposed approach is based on approximating light transfer functions with a set of *spherical radial basis functions* (SRBFs) in advance. To investigate inter-vertex data coherence, the approximated results of diffuse and glossy objects are further compressed using *clustered principal component analysis* (CPCA) [Sloan et al. 2003] and *clustered tensor approximation* (CTA), respectively. For glossy objects, the proposed CTA algorithm classifies the light transfer functions into groups to reduce inter-cluster variance. Since the PRT data sets within each cluster are intrinsically a multi-dimensional array, we retain their original structure, and analyze the coherence along each dimension with tensor approximation. Additionally, a technique for iteratively updating cluster members is introduced to minimize least-squares errors.

We adopt SRBFs to represent radiance functions for three reasons. First, the spatial localization property of SRBFs allows high-frequency signals to be handled efficiently. Second, SRBFs are circularly axis-symmetric and rotation-invariant functions defined on the unit sphere. Radiance functions can be modeled in their intrinsic domain, rather than a cubic or planar domain, to avoid the false boundaries and distortions that result from re-parameterizations in non-intrinsic domains. Furthermore, it is also simple to rotate and convolute functions represented in SRBFs. Third, similar to *radial basis functions* (RBFs), SRBFs are naturally applicable to interpolate and extrapolate scattered data. They can be non-uniformly distributed based on sampled data to obtain a compact representation for the original function.

In brief, this paper makes the following contributions:

*e-mail:hiei.cis91g@nctu.edu.tw

†e-mail:zcshih@cs.nctu.edu.tw

- A novel representation for PRT data and radiance functions based on SRBFs is introduced, which allows a compact data representation and real-time rendering of complex objects under high-frequency lighting environments.
- A new technique for analyzing multi-dimensional data is proposed, which iteratively re-classifies cluster members to search for a locally optimal solution.
- A new approach for modeling area light sources, such as *high dynamic range* (HDR) environment maps, with scattered SRBFs is also presented, which may be promising for other problems, including the importance sampling of spherical functions.

2 Related Work

2.1 Precomputed Radiance Transfer

The PRT algorithm can capture self-shadowing and self-interreflection effects from dynamic lighting environments. As a preprocess, PRT precomputes a solution to the light transport of a scene, and records the simulation results. To decrease data storage and computational costs, the recorded data are compressed for efficient rendering at run-time. The *low-frequency* PRT methods [Lehtinen and Kautz 2003; Sloan et al. 2002; Sloan et al. 2003] projected the per-vertex light transfer functions onto the *spherical harmonics* (SH) basis. The coherence among vertices was then exploited using *principal component analysis* (PCA) [Lehtinen and Kautz 2003] or CPCA [Sloan et al. 2003]. By contrast, the *all-frequency* PRT methods [Liu et al. 2004; Ng et al. 2003; Ng et al. 2004; Wang et al. 2004] approximated the densely-sampled PRT data with sophisticated compression techniques, such as non-linear wavelet approximation [Ng et al. 2003; Ng et al. 2004; Wang et al. 2004] and BRDF factorization [Liu et al. 2004; Wang et al. 2004]. However, previous compression schemes are inadequate for harnessing the power of PRT. Based on the SH basis, the low-frequency PRT algorithms may take tens of thousands of terms to represent regional lighting and shadowing effects. As for the all-frequency PRT algorithms, the compressed data are still cumbersome for real-time rendering of objects with glossy BRDFs.

Green et al. [2006] also proposed a hybrid PRT method for static scenes. While view-independent effects, including direct and indirect diffuse terms, were modeled with the SH or wavelet basis, high-frequency view-dependent signals, such as direct and indirect glossy terms, were approximated with Gaussian functions using non-linear optimization. However, this method is currently restricted to model specular effects with Gaussian functions, and it is unclear whether other all-frequency effects, such as all-frequency shadows, could be handled. By contrast, our approach allows rendering objects with all-frequency shadows in an unified framework.

Apart from static scenes and distant illumination, PRT has been extended to deformable objects [James and Fatahalian 2003; Sloan et al. 2005], dynamic scenes [Kautz et al. 2004; Zhou et al. 2005], and local lights [Kristensen et al. 2005; Zhou et al. 2005]. Sloan et al. [2005] adopted rotation-invariant harmonic functions, i.e. *zonal harmonics* (ZH), to model light transfer functions using non-linear optimization. Thus, rotating the transfer functions is simple and trivial. Although the ZH basis may yield a more compact representation than the SH basis, it is still restricted to low-frequency signals and lighting environments. Moreover, Kristensen et al. [2005] introduced the unstructured light clouds by precomputing light transfer functions with respect to densely-sampled local lights and then clustering the results based on heuristic metrics. Zhou et al. [2005] also presented a shadow approach, i.e. precomputed shadow fields, for dynamic scenes and local lights. The shadow field was built from concentric shells surrounding a light source or an object. At run-time, the shadow fields of each scene entity were combined to obtain incident radiance distributions for rendering.

2.2 Spherical Radial Basis Functions

Spherical radial basis functions (called *spherical basis functions* in [Narcowich and Ward 1996]) are special RBFs defined on the unit sphere. Their intrinsic nature in the spherical domain and other appealing properties, such as rotational invariance and positive definiteness, make them appropriate for modeling and analyzing spherical data without introducing any artificial boundaries or distortions. When combined with multi-resolution approaches, such as *spherical wavelets* (SW) [Freeden et al. 1998; Narcowich and Ward 1996], SRBFs become a powerful tool for analyzing scattered data on a sphere, including information measured by satellites and observed stations on the entire globe.

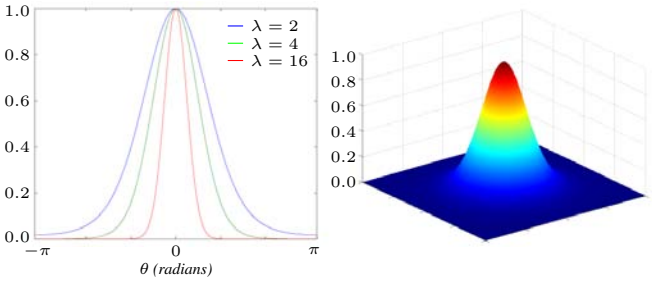
Computer graphics researchers are not unfamiliar with SRBFs. The generalized cosine lobe [Lafortune et al. 1997] and the isotropic Gaussian kernel of Ward model [Ward 1992] are two special cases of SRBFs. In both papers, they were applied to model BRDFs, often leading to a compact, expressive, and physically plausible representation of reflectance functions. Green et al. [2006] also applied Gaussian functions, which can be viewed as a special case of SRBFs, to model all-frequency glossy and mirroring effects with self-occlusions. In this paper, we adopt uniform SRBFs to model PRT data sets, and employ non-linear optimization to approximate lighting environments with scattered SRBFs.

2.3 Tensor Approximation

Dimensionality reduction techniques have been widely adopted to analyze and compress data in computer graphics. Perhaps the most popular approach is the traditional PCA [James and Fatahalian 2003; Lehtinen and Kautz 2003; Sloan et al. 2003], which is a linear model and often computed using *singular value decomposition* (SVD). In PCA, data samples are transformed from a high-dimensional space into another low-dimensional sub-space spanned by only a few *principal components* (PCs). Thus, the original samples can be approximated by projecting them onto these PCs. A major drawback of the traditional PCA is that data must be re-arranged into a standard two-mode matrix before analysis. However, in real cases, observations are frequently sampled under various conditions, and can be naturally classified into more than two modes. The original structure and important information of data may be lost after the re-arrangement.

In recent years, the multi-linear analysis [De Lathauwer et al. 2000] has been successfully applied to data-driven models [Vasilescu and Terzopoulos 2004; Wang et al. 2005] as well as the face transfer [Vlasic et al. 2005]. It can be viewed as a generalization of the traditional PCA, and permits separate dimensionality reduction. In [Vasilescu and Terzopoulos 2004], a set of *bidirectional-texture-function* (BTF) images was organized as a tensor. The basis matrices associated with each mode were then extracted using the traditional PCA by retaining the structure of that mode and unfolding others into a single vector. Wang et al. [2005] also proposed an out-of-core and block-wise technique based on the optimal N-mode SVD algorithm [De Lathauwer et al. 2000].

Although tensor approximation methods allow a higher compression ratio than the traditional PCA, directly adopting these methods to PRT data sets would be inadequate for our goal. Even after applying the optimal tensor decomposition, the reduced ranks of each mode are still high (30 views, 30 illuminations, and a 96×96 spatial resolution in [Wang et al. 2005]) for efficient reconstruction on



(a) (b)

Figure 2: (a) 2D plot of Gaussian SRBFs with different bandwidth parameters. (b) 3D plot of a Gaussian SRBF.

GPUs. To solve this problem, we group the PRT data sets into disjoint regions, within which the member data form a sub-tensor with lower variance, and then decompose the results to obtain fewer reduced ranks using the optimal N-mode SVD algorithm.

3 Overview

In the off-line process (Section 5), a solution to global illumination, i.e. light transfer functions, is precomputed, approximated with a set of uniformly distributed SRBFs (Section 5.1), and further compressed using either CPCA for diffuse objects or the proposed CTA algorithm for glossy objects (Section 5.2). The source radiance is also modeled with scattered SRBFs for efficient rendering at run-time (Section 5.3). For glossy objects, since approximating the raw PRT data with SRBFs and then compressing the results using CTA are computationally expensive, we factor BRDFs using SVD to reduce the cost.

In the run-time process (Section 6), the source radiance is rotated to align with objects and convoluted with the light basis matrix of each cluster. Finally, the shading color of each vertex is reconstructed from the vertex basis matrices, the projected weights, and the pre-filtered view-dependent textures.

4 SRBF Representation for PRT

4.1 Background of SRBFs

SRBFs are circularly axis-symmetric functions defined on $\mathbb{S}^m$, which is the unit sphere in $\mathbb{R}^{m+1}$. Let η and ξ denote two points on $\mathbb{S}^m$, and θ be the geodesic distance between η and ξ, i.e. $\arccos(\eta \cdot \xi)$. A SRBF is defined as a function that depends on θ, and can be expressed in terms of expansions in Legendre polynomials as

$$G(\cos\theta) = G(\eta \cdot \xi) = \sum_{l=0}^{\infty} G_l P_l(\eta \cdot \xi)\,, \tag{1}$$

where $P_l(\eta \cdot \xi)$ is the normalized *Legendre polynomial of degree* l, i.e. $P_l(1) = 1$, and the *Legendre coefficients* G_l satisfy $G_l \geqslant 0$ as well as $\sum_{l=0}^{\infty} G_l < \infty$. ξ is also known as the center of a SRBF.

Based on the orthogonal property of Legendre polynomials in the interval $[-1, +1]$, the Legendre expansions facilitate the convolution of two SRBFs, i.e. the *spherical singular integral*, by

$$\begin{aligned}(G *_m H)(\xi_g \cdot \xi_h) &= \int_{\mathbb{S}^m} G(\eta \cdot \xi_g) H(\eta \cdot \xi_h)\, d\omega(\eta) \\ &= \sum_{l=0}^{\infty} G_l H_l \frac{\omega_m}{d_{m,l}} P_l(\xi_g \cdot \xi_h)\,,\end{aligned} \tag{2}$$

where ω_m is the total surface area of $\mathbb{S}^m$, $d_{m,l}$ denotes the dimension of the space of order-l spherical harmonics on $\mathbb{S}^m$, and $d\omega$ is the differential surface element on $\mathbb{S}^m$. For more details about Eq. 2, please refer to [Narcowich and Ward 1996][1] and [Freeden and Windheuser 1997].

Two examples of SRBFs are the Abel-Poisson SRBF kernel (Eq. 3) and the Gaussian SRBF kernel (Eq. 4):

$$G^{Abel}(\eta \cdot \xi; \lambda) = \frac{1 - \lambda^2}{\left[1 - 2\lambda(\eta \cdot \xi) + \lambda^2\right]^{3/2}}\,,\ 0 < \lambda < 1\,, \tag{3}$$

$$G^{Gau}(\eta \cdot \xi; \lambda) = e^{-\lambda} e^{\lambda(\eta \cdot \xi)}\,,\ 0 < \lambda\,, \tag{4}$$

where λ denotes the bandwidth parameter, and controls the coverage of a SRBF. By choosing an appropriate value for λ, a SRBF can be adaptive to the spatial variation of local region. Therefore, SRBFs not only overcome one of the major disadvantages of the SH basis, but also possess more degrees of freedom than the ZH basis. Furthermore, while the convolution of two Abel-Poisson SRBFs remains another Abel-Poisson SRBF, the spherical singular integral of two Gaussian SRBFs can be efficiently evaluated for small m. Their spherical singular integrals are

$$\begin{aligned}&\left(G^{Abel} *_m H^{Abel}\right)(\xi_g \cdot \xi_h; \lambda_g, \lambda_h) \\ &\qquad = \frac{1 - (\lambda_g \lambda_h)^2}{\left[1 - 2(\lambda_g \lambda_h)(\xi_g \cdot \xi_h) + (\lambda_g \lambda_h)^2\right]^{3/2}}\,,\end{aligned} \tag{5}$$

$$\begin{aligned}&\left(G^{Gau} *_m H^{Gau}\right)(\xi_g \cdot \xi_h; \lambda_g, \lambda_h) \\ &\quad = e^{-(\lambda_g + \lambda_h)} \omega_m\, \Gamma\!\left(\frac{m+1}{2}\right) I_{\frac{m-1}{2}}(\|r\|) \left(\frac{2}{\|r\|}\right)^{\frac{m-1}{2}}\,,\end{aligned} \tag{6}$$

where $r = \lambda_g \xi_g + \lambda_h \xi_h$. For the mathematical proof and details about Eq. 6, please refer to Appendix A.

Similar to RBFs in $\mathbb{R}^{m+1}$, given a set of distinct points $\boldsymbol{\Xi} = \{\xi_1, \ldots, \xi_n\}$ on $\mathbb{S}^m$, which is called the set of SRBF centers, and another set of real numbers $\boldsymbol{\Lambda} = \{\lambda_1, \ldots, \lambda_n\}$ in $\mathbb{R}$, which is named the set of SRBF bandwidth parameters, a spherical function $F(\eta)$ can be represented in SRBF expansions as

$$F(\eta) \approx \sum_{k=1}^{n} F_k G(\eta \cdot \xi_k; \lambda_k)\,. \tag{7}$$

SRBFs thus behave as reproducing kernels for interpolating $F(\eta)$ on $\mathbb{S}^m$, and the SRBF coefficients F_k can be obtained by *ordinary least-squares* (OLS) or *regularized least-squares* (RLS) projection [Ramsay and Silverman 2005].

4.2 PRT Representation in SRBFs

For distant illumination, the rendering equation [Kajiya 1986] is

$$B_x(\eta_o) = \int_{\mathbb{S}^2} L(\eta_i) \rho_x(\eta_i, \eta_o) V_x(\eta_i)(\eta_i \cdot n_x)\, d\omega(\eta_i)\,, \tag{8}$$

where $B_x(\eta_o)$ denotes the outgoing radiance from a point x in direction η_o, and $L(\eta_i)$ is the incident radiance in direction η_i. ρ_x, $V_x(\eta_i)$, and n_x denote the BRDF, the visibility function, and the surface normal at x, respectively.

[1] There is a minor error with ω_m in [Narcowich and Ward 1996]. The correct equation should be $\omega_m = \frac{2\pi^{(m+1)/2}}{\Gamma\left(\frac{m+1}{2}\right)}$.

After approximating the BRDF with several view-dependent functions $\phi_{x,k}(\eta_o)$ and light-dependent functions $\psi_{x,k}(\eta_i)$ by using the SVD algorithm [Liu et al. 2004; Wang et al. 2004], Eq. 8 becomes

$$\begin{aligned} B_x(\eta_o) &\approx \sum_{k=1}^{K} \Big[\phi_{x,k}(\eta_o) \int_{\mathbb{S}^2} T_{x,k}(\eta_i) L(\eta_i)\, d\omega(\eta_i) \Big] , \\ T_{x,k}(\eta_i) &= \psi_{x,k}(\eta_i) V_x(\eta_i)(\eta_i \cdot n_x) , \\ \rho_x(\eta_i, \eta_o) &\approx \sum_{k=1}^{K} \phi_{x,k}(\eta_o)\psi_{x,k}(\eta_i) = \boldsymbol{\Phi}_x \boldsymbol{\Psi}_x , \end{aligned} \tag{9}$$

where $T_{x,k}(\eta_i)$ is known as the light transfer function, and describes the transport of light, for the k-th light-dependent function $\psi_{x,k}(\eta_i)$, between the distant illumination source and the incident radiance to x in direction η_i.

Let $\boldsymbol{\Xi}_1 = \{\xi_{1,1}, \ldots, \xi_{1,n_t}\}$ and $\boldsymbol{\Xi}_2 = \{\xi_{2,1}, \ldots, \xi_{2,n_l}\}$ be two sets of distinct points on $\mathbb{S}^2$, where $\xi_{1,1} \neq \ldots \neq \xi_{1,n_t}$ and $\xi_{2,1} \neq \ldots \neq \xi_{2,n_l}$ are satisfied, and $\boldsymbol{\Lambda}_1 = \{\lambda_{1,1}, \ldots, \lambda_{1,n_t}\}$ and $\boldsymbol{\Lambda}_2 = \{\lambda_{2,1}, \ldots, \lambda_{2,n_l}\}$ denote two sets of real numbers in $\mathbb{R}$. Our approach represents the light transfer function $T_{x,k}(\eta_i)$ and the distant illumination $L(\eta_i)$ in SRBF expansions as

$$\begin{aligned} T_{x,k}(\eta_i) &\approx \sum_{l=1}^{n_t} T_{x,k,l} G(\eta_i \cdot \xi_{1,l}; \lambda_{1,l}) . \\ L(\eta_i) &\approx \sum_{j=1}^{n_l} L_j G(\eta_i \cdot \xi_{2,j}; \lambda_{2,j}) , \end{aligned} \tag{10}$$

By combining Eq. 9 with Eq. 10, the outgoing radiance at x is approximated by

$$\begin{aligned} B_x(\eta_o) \approx \sum_{k=1}^{K} \Big[\phi_{x,k}(\eta_o) \sum_{j=1}^{n_l} \sum_{l=1}^{n_t} T_{x,k,l}\, L_j \\ \int_{\mathbb{S}^2} G(\eta_i \cdot \xi_{1,l}; \lambda_{1,l}) G(\eta_i \cdot \xi_{2,j}; \lambda_{2,j})\, d\omega(\eta_i) \Big] . \end{aligned} \tag{11}$$

Eq. 11 can be simply re-written in matrix notation as

$$\mathbf{B}_x \approx \boldsymbol{\Phi}_x\, \mathbf{T}_x\, \mathbf{A}_x\, \mathbf{L} , \tag{12}$$

where $\mathbf{A}_x$ is known as the interpolation matrix between the light transfer matrix $\mathbf{T}_x$ and the incident illumination vector $\mathbf{L}$. The component of $\mathbf{A}_x$ in the m-th row and the n-th column is given by

$$\big(\mathbf{A}_x\big)_{mn} = \int_{\mathbb{S}^2} G(\eta_i \cdot \xi_{1,m}; \lambda_{1,m}) G(\eta_i \cdot \xi_{2,n}; \lambda_{2,n})\, d\omega(\eta_i) . \tag{13}$$

Since SRBFs are non-orthonormal basis functions, $\mathbf{A}_x$ is necessary to account for the spherical singular integrals.

5 Off-Line Process

5.1 PRT Precomputation

For an input scene, our system precomputes the light transfer matrix $\mathbf{T}_x$ at each vertex. The visibility function at x, $V_x(\eta_i)$, is obtained by rendering the scene into a cube map with flat shading. To avoid aliasing artifacts, the cube map is super-sampled with a $6 \times 128 \times 128$ resolution, and then down-sampled to $6 \times 32 \times 32$ pixels. Currently, our system only takes direct illumination into account. However, inter-reflections can be handled by applying GPUs to render triangle ID maps. We intend to integrate full global illumination solutions in the future.

The BRDF at x, given by ρ_x, is assumed to be the same everywhere on an object without losing generality, and approximated using the SVD algorithm by retaining K terms of ϕ_x and ψ_x with the first K largest singular values. This scheme will provide a preliminary data reduction for the raw PRT data sets while preserving most of the important information for further analysis in the following CTA stage. In our current configuration, K simply equals to 1 for diffuse objects, whereas K is set to 16 for glossy objects. The value of K depends on the complexity of a BRDF, and a setting of $K \leqslant 16$ is typically sufficient for analytic BRDFs [Liu et al. 2004; Wang et al. 2004]. For complex BRDFs, we can set K to a value such that the sum of the first K largest singular values exceeds a certain percentage, for example 90%, of the sum of all singular values.

The set of SRBF centers $\boldsymbol{\Xi}_1$ on $\mathbb{S}^2$ is generated by repeatedly subdividing an icosahedron, while the set of SRBF bandwidth parameters $\boldsymbol{\Lambda}_1$ is assigned with the same value determined from the minimum geodesic distance between two centers, which is related to the variance with respect to the center of a SRBF [Freeden and Windheuser 1997; Narcowich and Ward 1996]. Our system currently subdivides an icosahedron into a mesh with 642 vertices as the SRBF centers, and sets the variance to $\pi/40$ radians for deriving the SRBF bandwidth parameters. For more details about the relationship between the variance and the bandwidth parameters for Abel-Poisson and Gaussian SRBFs, please refer to [Freeden and Windheuser 1997; Narcowich and Ward 1996]. Although this configuration is sufficient for general cases according to our experiments, a finer mesh on $\mathbb{S}^2$ as well as a smaller value of variance (a larger value of bandwidth parameter) can always be adopted to handle more complicated $T_{x,k}(\eta_i)$. Finally, each $T_{x,k}(\eta_i)$ is projected onto the SRBFs, and the resulting coefficients are quantized to 16-bits of precision.

The light transfer functions are not modeled with scattered SRBFs for two reasons. First, the optimization process (Section 5.3) is computationally too expensive to be performed at each vertex. Second, after encoding the PRT data sets using CPCA or CTA, since the SRBF centers of each vertex are usually different, we can not take advantage of the per-cluster operations (Section 6) at run-time, and must perform the convolutions vertex by vertex. We aim to investigate these issues in the future.

5.2 Clustered Tensor Approximation

After approximating the light transfer matrix $\mathbf{T}_x$ with SRBFs, the size of the PRT data sets is still bulky for real-time applications. To solve this problem, the CPCA algorithm [Sloan et al. 2003] is applied to find inter-vertex data redundancy for a diffuse object, while the transfer matrices of a glossy object are organized as a 3^{rd}-order tensor $\boldsymbol{\mathcal{T}}$ (Fig. 3), and compressed using the proposed CTA algorithm. A simple and intuitive method for CTA is to classify vertices into disjoint regions according to heuristic objective functions, and then decompose the sub-tensor within each region using the optimal N-mode SVD algorithm [De Lathauwer et al. 2000]. However, it is often hard to define objective functions consistent with the approximation errors of tensor decomposition. Therefore, our approach updates cluster members based on approximation errors at the end of each iteration, and guarantees at least a locally optimal solution.

5.2.1 Overview

The proposed CTA algorithm consists of three phases: the initialization, clustering, and approximation phase. In the initialization phase, an initial solution to cluster members is obtained. Each vertex is then iteratively re-classified into the cluster with the minimum approximation error until convergence in the clustering phase. Since the number of member vertices varies from cluster to cluster, straightforwardly employing the optimal N-mode SVD algorithm

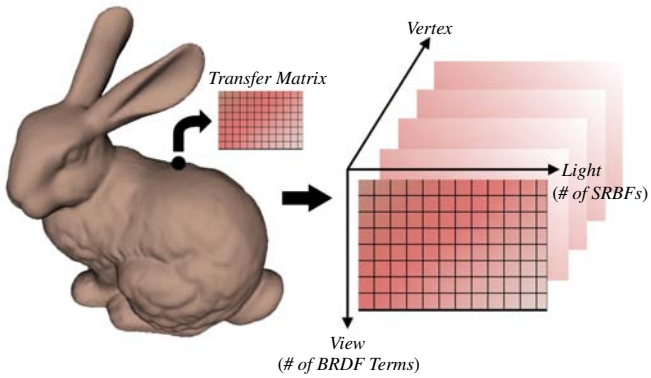


Figure 3: The tensor representation for precomputed light transfer matrices. The transfer matrices are organized as a 3^{rd}-order tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I_{view} \times I_{light} \times I_{vert}}$.

would be difficult to compute approximation errors. To solve this issue, we also derive the *dual* mode-3 basis matrices. Finally, the optimal basis matrices of each cluster are extracted in the approximation phase.

Let I_{view} denote the number of BRDF approximation terms, i.e. K, I_{light} be the number of SRBFs for modeling the light transfer functions, i.e. n_t, and I_{vert} represent the total number of vertices. The proposed CTA algorithm classifies $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I_{view} \times I_{light} \times I_{vert}}$ into a set of sub-tensors $\boldsymbol{\mathcal{T}} = \{\boldsymbol{\mathcal{T}}_1, \ldots, \boldsymbol{\mathcal{T}}_{n_c}\}$, where n_c denotes the desired number of clusters, and $\boldsymbol{\mathcal{T}}_c \in \mathbb{R}^{I_{view} \times I_{light} \times I_{vert,c}}$ is the sub-tensor within cluster c whose number of member vertices is $I_{vert,c}$. Let R_{view}, R_{light}, and R_{vert} be the reduced ranks of each mode. Each sub-tensor is decomposed by extracting the basis matrices of each mode, which are denoted by $\mathbf{U}_{view} \in \mathbb{R}^{I_{view} \times R_{view}}$, $\mathbf{U}_{light,c} \in \mathbb{R}^{I_{light} \times R_{light}}$, and $\mathbf{U}_{vert,c} \in \mathbb{R}^{I_{vert,c} \times R_{vert}}$. To iteratively update cluster members and facilitate the rendering process at run-time, for each cluster, another orthonormal vertex basis matrix $\mathbf{V}_{vert,c} \in \mathbb{R}^{R_{vert} \times (R_{view} R_{light})}$, which we call the dual mode-3 basis matrix, is derived from $\mathbf{U}_{vert,c}$.

Note that for the 1^{st} mode, i.e. the view mode, due to performance issues at run-time (Section 6), only a single basis matrix is determined for all clusters, rather than an individual basis matrix for each cluster. This scheme generally does not result in significant loss of information, since the view mode is highly dependent on BRDFs. For objects with multiple BRDFs, $\boldsymbol{\mathcal{T}}$ can be split into multiple groups, one group for each BRDF, and CTA is then performed within each group. To avoid confusion, although the block-based mode-n product [Wang et al. 2005] is applied, we use the notation as if $\boldsymbol{\mathcal{T}}$ were not partitioned into blocks.

5.2.2 Initialization Phase

The proposed CTA algorithm starts with an initial solution to cluster members. After randomly selecting vertices as cluster representatives and unfolding $\mathbf{T}_x$ into a vector $uf(\mathbf{T}_x)$, each vertex is classified by performing several iterations of the ELBG algorithm [Patanè and Russo 2001]. $\boldsymbol{\mathcal{T}}$ is then partitioned into blocks for efficient computation of tensor approximation. The mode-1 basis matrix $\mathbf{U}_{view}$ is also extracted with all other basis matrices initialized to the identity matrix $\mathbf{I}$, and a reduced core tensor $\boldsymbol{\mathcal{Z}}_{view} \in \mathbb{R}^{R_{view} \times I_{light} \times I_{vert}}$ for the clustering phase is determined by

$$\boldsymbol{\mathcal{Z}}_{view} = \boldsymbol{\mathcal{T}} \times_1 \mathbf{U}_{view}^T . \tag{14}$$

5.2.3 Clustering Phase

In this phase, each vertex is iteratively re-classified into the cluster with the minimum approximation error $\|uf(\mathbf{T}_x) - uf(\tilde{\mathbf{T}}_{x,c})\|^2$, where $\tilde{\mathbf{T}}_{x,c}$ denotes the approximated light transfer matrix of vertex x with respect to cluster c.

Given $\boldsymbol{\mathcal{Z}}_{view}$ and an initial solution to cluster members, $\boldsymbol{\mathcal{Z}}_{view}$ is first decomposed into isolated sub-tensors, and the sub-tensor within cluster c is denoted by $\boldsymbol{\mathcal{Z}}_{view,c}$. The block-based optimal N-mode SVD algorithm [Wang et al. 2005] is then applied to extract both $\mathbf{U}_{light,c}$ and $\mathbf{U}_{vert,c}$ from $\boldsymbol{\mathcal{Z}}_{view,c}$. To facilitate the computation of approximation errors for vertices outside cluster c, the dual mode-3 basis matrix $\mathbf{V}_{vert,c}$ is derived as

$$\boldsymbol{\mathcal{Z}}_c = \boldsymbol{\mathcal{Z}}_{view,c} \times_2 \mathbf{U}_{light,c}^T \times_3 \mathbf{U}_{vert,c}^T , \tag{15}$$

$$\mathbf{V}_{vert,c} = \left\{ uf_3(\boldsymbol{\mathcal{Z}}_c) \left[uf_3(\boldsymbol{\mathcal{Z}}_c) \right]^T \right\}^{-\frac{1}{2}} uf_3(\boldsymbol{\mathcal{Z}}_c) , \tag{16}$$

where $uf_3(\boldsymbol{\mathcal{Z}}_c) \in \mathbb{R}^{R_{vert} \times (R_{view} R_{light})}$ denotes the unfolded matrix along the 3^{rd} mode, i.e. the vertex mode. Since the rows of $uf_3(\boldsymbol{\mathcal{Z}}_c)$ are orthogonal, Eq. 16 is equivalent to the normalization of each row of $uf_3(\boldsymbol{\mathcal{Z}}_c)$.

After that, each vertex is re-classified into the cluster with the minimum approximation error, and $\tilde{\mathbf{T}}_{x,c}$ is computed by

$$\mathbf{W}_{x,c} = \mathbf{V}_{vert,c}\, uf\left(\mathbf{U}_{view}^T \mathbf{T}_x \mathbf{U}_{light,c}\right) , \tag{17}$$

$$\tilde{\mathbf{T}}_{x,c} = \mathbf{U}_{view}\, uf^{-1}\left(R_{light}, \mathbf{V}_{vert,c}^T \mathbf{W}_{x,c}\right) \mathbf{U}_{light,c}^T , \tag{18}$$

where $\mathbf{W}_{x,c} \in \mathbb{R}^{R_{vert}}$ is called the projected weights of vertex x with respect to cluster c, and $uf^{-1}(R_{light}, \mathbf{V}_{vert,c}^T \mathbf{W}_{x,c})$ represents the inverse operation of unfolding, which refolds the vector $\mathbf{V}_{vert,c}^T \mathbf{W}_{x,c}$ into a matrix in $\mathbb{R}^{R_{view} \times R_{light}}$, whose components in the i-th row coincide with the $((i-1)\cdot R_{light} + 1)$-th to $(i \cdot R_{light})$-th elements of $\mathbf{V}_{vert,c}^T \mathbf{W}_{x,c}$. The entire process of this phase iterates until no vertices are assigned to another cluster, or a maximum iteration count is reached.

5.2.4 Approximation Phase

Based on the clustered results in the clustering phase, we perform an additional approximation phase to obtain the optimal basis matrices $\mathbf{U}_{view}$, $\mathbf{U}_{light,c}$, $\mathbf{U}_{vert,c}$, and $\mathbf{V}_{vert,c}$.

Given $\mathbf{U}_{light,c}$ and $\mathbf{U}_{vert,c}$ from the last iteration of the clustering phase, each $\boldsymbol{\mathcal{T}}_c$ is first projected onto the high-dimensional space spanned by $\mathbf{U}_{light,c}$ and $\mathbf{U}_{vert,c}$ to obtain $\boldsymbol{\mathcal{Z}}_{light,vert,c}$, where

$$\boldsymbol{\mathcal{Z}}_{light,vert,c} = \boldsymbol{\mathcal{T}}_c \times_2 \mathbf{U}_{light,c}^T \times_3 \mathbf{U}_{vert,c}^T . \tag{19}$$

The columns of $\mathbf{U}_{view}$ are then updated by extracting the first R_{view} columns of the left singular matrix from the SVD of $uf_1(\boldsymbol{\mathcal{Z}}_{light,vert,c})$. Next, $\boldsymbol{\mathcal{Z}}_{view}$ is re-calculated as Eq. 14, and decomposed into $\boldsymbol{\mathcal{Z}}_{view,c}$ for each cluster, which is utilized to update $\mathbf{U}_{light,c}$ and $\mathbf{U}_{vert,c}$. This process iterates until all the basis matrices converge, or the iteration count exceeds a threshold. Finally, the dual mode-3 matrix $\mathbf{V}_{vert,c}$ of each cluster and the projected weights of each vertex $\mathbf{W}_{x,c}$ are obtained by Eq. 16 and Eq. 17.

5.3 Modeling Lighting Environments with Scattered SRBFs

To efficiently shade objects whose light transfer functions are represented in SRBFs, the lighting environment $L(\eta_i)$ is also modeled with SRBFs. Although $L(\eta_i)$ can be approximated by adopting a

(a) Reference (b) SH basis SE = 96.06% (c) Wavelets SE = 46.60% (d) Scattered SRBFs SE = 3.14%

Figure 4: Comparison of the approximated results for (a) the *St. Peter's Basilica* HDR lighting environment using (b) OLS projection onto the SH basis, (c) area-weighted wavelet technique, and (d) the scattered SRBF fitting algorithm. Each result is reconstructed with 972 coefficients or parameters.

set of uniform SRBFs, we take a step further to obtain a compact set of scattered SRBFs for approximating $L(\eta_i)$. The entire process is formulated as an optimization problem, and handled by minimizing squared approximation errors.

Given a desired number of SRBFs n_l, we intend to learn three sets of parameters: the set of SRBF coefficients $\boldsymbol{L} = \{L_1, \ldots, L_{n_l}\}$, the set of SRBF centers $\boldsymbol{\Xi}_2$, and the set of SRBF bandwidth parameters $\boldsymbol{\Lambda}_2$, such that the following objective function is minimized:

$$\{\boldsymbol{L}, \boldsymbol{\Xi}_2, \boldsymbol{\Lambda}_2\} = \underset{\{\boldsymbol{L}, \boldsymbol{\Xi}_2, \boldsymbol{\Lambda}_2\}}{\arg\min} \int_{\mathbb{S}^2} \left| L(\eta_i) - \tilde{L}(\eta_i) \right|^2 d\omega(\eta_i) \,, \quad (20)$$

$$\tilde{L}(\eta_i) = \sum_{j=1}^{n_l} L_j G(\eta_i \cdot \xi_{2,j}; \lambda_{2,j}) \,. \quad (21)$$

Instead of solving all the three sets of parameters together, we propose an iterative approach, which optimizes only one set of parameters at each step, while leaving the others fixed. This scheme often yields better results, since the three sets of parameters are highly coupled with each other. The proposed scattered SRBF fitting algorithm is summarized as follows:

1. Given an initial guess or the results from previous iteration, the L-BFGS-B solver [Zhu et al. 1997] is performed to optimize the set of SRBF centers followed by the set of SRBF bandwidth parameters.
2. The set of SRBF coefficients is then obtained using OLS or RLS projection.
3. This process iterates until the difference of squared errors between current and previous iterations falls below a threshold, or a user-defined iteration count is reached.
4. Finally, an additional stage can be performed to optimize all the parameters simultaneously.

Since the approximated results often depend on the initial seeds of parameters, our system automatically determines a reasonable guess based on heuristics in the hope of achieving smaller approximation errors and accelerating the computation. Given a lighting environment, we first estimate the coverage of each direction in a dense set of unit directions $\boldsymbol{E}_i = \{\eta_{i,1}, \ldots, \eta_{i,n}\}$, and construct a priority queue. To prevent the initial SRBF centers from concentrated in local regions, after selecting a direction $\eta_{i,j}$ as an initial SRBF center, all the directions which locate within the coverage of $\eta_{i,j}$ are marked such that they will not be chosen in subsequent steps. This process continues until all the initial SRBF centers are selected. If all directions are marked before all the SRBF centers are determined, we reduce the coverage of each direction and the selected centers, update the marks, and re-loop over the priority queue to choose the remaining centers. Finally, the initial SRBF bandwidth parameters are derived from the coverage of the selected directions, while the initial guess about the set of SRBF coefficients is obtained by projecting the lighting environment onto the initial SRBFs in the OLS or RLS sense.

The coverage of a direction $\eta_{i,j}$ is estimated from its resemblance to nearby sampling directions, and can be computed by a heuristic approach. At first, we loop over the directions $\eta_{i,j'}$ in $\boldsymbol{E}_i$, in the order from nearest to farthest with respect to $\eta_{i,j}$, until the ratio of nearby directions, whose squared difference of the incident radiance between $\eta_{i,j}$ and $\eta_{i,j'}$, i.e. $|L(\eta_{i,j}) - L(\eta_{i,j'})|^2$, exceeds a user-defined threshold, is reached. This set of nearby directions forms a cone-shaped region centered on $\eta_{i,j}$. The coverage of $\eta_{i,j}$ is then defined as the geodesic distance between $\eta_{i,j}$ and the farthest $\eta_{i,j'}$ within this cone, and set as the variance of a SRBF for computing its initial bandwidth parameter. Based on the coverage of each direction, the priority queue is constructed by sorting the coverage-weighted incident radiance of each sampling direction.

Experimental results indicate that this heuristic guess of initial parameters generally works better than taking a set of uniform SRBFs to initialize the optimization process, and reduces the computation time by a factor of 2-4. Typically, it takes less than 9 minutes to approximate a $6 \times 32 \times 32$ cube map with 162 SRBFs (972 parameters). To accelerate the process of fitting a high-resolution cube map, the map can be down-sampled and approximated using the proposed algorithm. The resulting parameters are then taken as an initial solution to approximate the high-resolution map.

Figure 4 compares the reconstructed results of a HDR lighting environment using the SH basis, the area-weighted wavelets [Ng et al. 2003], and our approach. The proposed approach captures most features with only 162 SRBFs, whereas the result of wavelet method fails to distinguish some features, especially the high-energy lights at the ceiling. For the SH basis, high-energy signals dominate over low-energy ones, producing some ringing effects.

6 Run-Time Rendering Process

This section describes the rendering process for glossy objects whose light transfer functions are compressed using the proposed CTA algorithm. For diffuse objects, since the PRT data sets are encoded using CPCA, the rendering process is the same as that in [Sloan et al. 2003].

At run-time, the shading color of a point x belonging to cluster c is reconstructed by

$$C_{x,\eta_o} = \mathbf{E}_{\eta_o}\big(\tilde{\mathbf{B}}_x\big) = \mathbf{E}_{\eta_o}\big(\boldsymbol{\Phi}_x\, \tilde{\mathbf{T}}_{x,c}\big)\, \mathbf{A}\, \mathbf{L} \,, \quad (22)$$

where $\mathbf{E}_{\eta_o}(\mathbf{F})$ denotes the evaluation of $\mathbf{F}$ in direction η_o, and $\mathbf{A}$ is the interpolation matrix. Since the same set of SRBFs is applied to model $\mathbf{T}_x$ at each vertex, The subscript x of the interpolation matrix is omitted in Eq. 22. By substituting Eq. 18 into Eq. 22, we can derive the following shading equation:

$$\begin{aligned} C_{x,\eta_o} &= \mathbf{E}_{\eta_o}\big(\boldsymbol{\Phi}'_x\big)\, uf^{-1}\big(R_{light}\,, \mathbf{V}^T_{vert,c}\, \mathbf{W}_{x,c}\big)\, \mathbf{L}'_c \\ &= \mathbf{E}_{\eta_o}\big(\boldsymbol{\Phi}'_x\big) \sum_{j=1}^{R_{vert}} \Big\{ \big(\mathbf{W}_{x,c}\big)_j\, uf^{-1}\Big(R_{light}\,, \big(\mathbf{V}^T_{vert,c}\big)_j\Big) \mathbf{L}'_c \Big\}, \\ \boldsymbol{\Phi}'_x &= \boldsymbol{\Phi}_x\, \mathbf{U}_{view} \,, \quad \mathbf{L}'_c = \mathbf{U}^T_{light,c}\, \mathbf{A}\, \mathbf{L} = \mathbf{U}^T_{light,c}\, \mathbf{L}' \,, \end{aligned} \quad (23)$$

where $\big(\mathbf{V}^T_{vert,c}\big)_j$ and $\big(\mathbf{W}_{x,c}\big)_j$ denote the j-th column of $\mathbf{V}^T_{vert,c}$ and the j-th component of $\mathbf{W}_{x,c}$.

Model	Teapot	Buddha	Bunny
Vertices	41k	52k	61k
BRDF Terms	16	16	16
Clusters	110	140	170
Reduced Ranks R_{view}	8	8	8
Reduced Ranks R_{light}	48	48	48
Reduced Ranks R_{vert}	24	24	24
Raw PRT Data	15.11 GB	19.12 GB	22.35 GB
PRT Data in SRBFs	0.79 GB	1.00 Gb	1.17 GB
CTA Compressed Data	10.37 MB	13.18 MB	15.89 MB
SE Ratio (SRBF)	1.23%	1.41%	1.28%
SE Ratio (SRBF+CTA)	3.91%	3.75%	3.64%
PRT Computation Time	1 hr 53 min	2 hr 19 min	2 hr 47 min
CTA Compression Time	16 hr 6 min	24 hr 16 min	29 hr 2 min
Run-Time FPS	74.67/15.83	51.12/13.27	48.65/10.42

Table 1: Statistics and timing measurements of the proposed approach using the Cook-Torrance BRDF model [Cook and Torrance 1982] and the *St. Peter's Basilica* HDR lighting environment. The run-time FPS lists the rendering performance when the viewpoint changes or the lighting environment changes (view/light). The SE ratio is the ratio of reconstructed squared errors between the compressed data and the raw data.

The run-time PRT rendering process thus consists of the following four steps:

1. Align $\mathbf{L}$ with objects. This includes rotating the set of SRBF centers $\boldsymbol{\Xi}_2$, performing the spherical singular integrals to acquire $\mathbf{A}$, and computing the product $\mathbf{L}' = \mathbf{A}\,\mathbf{L}$.
2. For each cluster c, obtain the representative light transfer matrix $\mathbf{R}_c$, whose j-th column is $uf^{-1}(R_{light}, \left(\mathbf{V}^T_{vert,c}\right)_j)\,\mathbf{L}'_c$.
3. For each vertex, reconstruct the light transfer vector $\mathbf{R}_x$ by the weighted summation $\mathbf{R}_c\,\mathbf{W}_{x,c}$.
4. For each shading point, evaluate the view-dependent BRDF vector $\mathbf{E}_{\eta_o}(\mathbf{\Phi}'_x)$. The final shading color is then the dot product of $\mathbf{E}_{\eta_o}(\mathbf{\Phi}'_x)$ and $\mathbf{R}_x$.

We perform Step 1 and Step 2 on CPUs, whereas the other two steps are executed on GPUs. Step 1 is a per-object process, which is performed when the alignment of a lighting environment with an object is required. Step 2 is a per-cluster operation rather than a per-vertex one. Since the number of clusters is much smaller than the number of vertices, Step 2 greatly reduces the computation time. The representative light transfer matrix of each cluster is then transferred to GPUs to execute Step 3 in the vertex shader and Step 4 in the pixel shader. Note that the evaluation of $\mathbf{E}_{\eta_o}(\mathbf{\Phi}'_x)$ in Step 4 is conducted using a pre-filtered texture. To prevent aliasing artifacts, this texture is super-sampled, and then filtered down to a desired resolution.

7 Experimental Results and Discussion

7.1 Experimental Results

Table 1 lists the statistics and timing measurements of the proposed approach for various models. All of the simulation timings were measured on an AMD Athlon64 FX-55 PC with an NVIDIA GeForce 6800 GPU. The Gaussian SRBFs were adopted to represent the light transfer functions as well as the lighting environments. To improve rendering performance, the super-clustering technique [Sloan et al. 2003] was also applied to decrease the number of redrawn triangles. Table 1 shows that our approach can achieve high compression ratios and real-time rendering performance using modern GPUs. For our configurations, the SRBF representation followed by quantization reduce the raw data by 94.7%, and CTA further encodes the approximated results by a factor of 73 on average. The total compression ratio is thus beyond 1000:1.

Algorithm	All-Frequency CPCA	Our Approach
Light Basis Dimensions	0	642×48
Vertex Basis Dimensions	256×8	48×8
Projected Weights	12	24
Clusters	140×24	140
Compressed Data	48.41 MB	13.18 MB
SE Ratio	3.09%	3.75%
Run-Time FPS	13.34/1.65	51.12/13.27

Table 2: Comparison of statistics and timing measurements between our approach and the all-frequency CPCA algorithm for the buddha model. The raw PRT data for deriving the SE ratio are precomputed by setting the BRDF terms to 16.

In all rendered results, a diffuse plane was placed under each model to demonstrate the ability of SRBFs to handle all-frequency shadows. The SRBF representation typically provides visually pleasant shadows with hundreds of SRBFs. Although occasionally the contours of shadow boundaries appear to be too smooth such that some sharp features of shadowing objects are lost, a set of denser SRBFs can always be adopted to faithfully model the raw data at the cost of more compression time and slower rendering performance.

Table 2 and Figure 1 compare our approach with the all-frequency CPCA algorithm [Liu et al. 2004]. While the proposed scheme renders objects with comparable quality in real time, the performance of all-frequency CPCA is only at interactive rates. There are two main reasons that enable our approach to achieve real-time performance. First, we adopt SRBFs to approximate light transfer functions with much fewer coefficients. Second, to accelerate compression time and allow CPCA adaptive to the large dimension of the light mode, all-frequency CPCA statically partitioned the light mode into several segments. However, while performing per-vertex operations on GPUs at run-time, objects were rendered several times for each segment separately. Memory bandwidths were consumed for transferring the partially reconstructed data of each segment to GPUs. By contrast, our approach does not partition the light mode, since usually the dimension of this mode is not so large. Instead, we increase the reduced rank of the vertex mode to obtain lower approximation errors. Therefore, objects can be rendered on GPUs at real-time rates, without much overhead.

The main cost of our approach lies in the clustering phase of CTA (Section 5.2.3), since we search for a locally optimal solution to the basis matrices of each cluster. Other heuristic clustering approaches can be applied to decrease the compression time, while increasing the number of clusters or the values of reduced ranks to obtain similar image quality. However, this scheme generally leads to lower compression ratios as well as slower run-time performance. We intend to reduce the compression time, or develop heuristic error metrics to approximate a locally optimal solution in the future.

7.2 Discussion

Compared with previous all-frequency PRT algorithms, the advantages of our approach are as follows:

- Our method is able to render all-frequency effects from compact compressed data at real-time rates.

- With the SRBF representation, radiance functions can be modeled in their intrinsic domain to avoid false boundaries, distortions, and unnecessary re-parameterizations.
- SRBFs behave as noise filters to prevent aliasing artifacts, and usually lead to visually pleasant results.
- Our CTA algorithm further reduces the light mode to exploit the coherence that can not be found by all-frequency CPCA.

When lighting environments change, CTA also allows efficient computation of Step 2 (Section 6), whose cost depends on $I_{light} \times R_{light}$ and $R_{light} \times R_{vert}$. For the all-frequency CPCA algorithm, the environments are convoluted with the vertex basis matrices of each cluster. Therefore, the computational cost depends on $I_{light} \times I_{view} \times R_{vert}$, which is more expensive than our approach.

There are also some disadvantages of our approach:

- The off-line costs are higher than previous algorithms.
- Unlike ZH [Sloan et al. 2005], per-vertex rotations of light transfer functions currently can not be handled.
- After modeling light transfer functions with uniform SRBFs, some sharp features, such as all-frequency shadows, may be more approximate than wavelet methods.
- For highly specular BRDFs, such as all-frequency mirroring effects, the technique proposed by Green et al. [2006] may be better than our approach.
- Our method does not support fully dynamic changes of lighting environments at real-time rates, whereas previous algorithms do. The environments are only allowed to rotate efficiently at run-time.

However, in Section 5.3, we showed that the scattered SRBF representation can outperform the area-weighted wavelet technique for modeling lighting environments. If the scattered SRBF representation is applied to each vertex, the second and third disadvantages could be overwhelmed. Our current scheme is merely a trade-off between approximation errors and computational costs. As for highly specular BRDFs, the technique of Green et al. and our approach could be integrated as a two-pass rendering process to overcome the fourth disadvantage.

8 Conclusions

The proposed approach overcomes the major drawbacks of previous all-frequency PRT algorithms, and achieves real-time rendering performance without sacrificing much image quality. Experimental results reveal that SRBFs are effective in dealing with high-frequency signals, and provide an intrinsic representation for PRT data sets in the spherical domain. The proposed SRBF fitting algorithm also allows lighting environments to be modeled with a compact set of scattered SRBFs. Although the PRT data sets in SRBF representation are still cumbersome, they were successfully compressed to less than 0.1% of the raw data using the proposed CTA algorithm. With some modifications, we believe that CTA may be applicable to dynamic PRT data sets and other data-driven models, such as *image-based rendering*.

There are many possible applications of SRBFs and the scattered SRBF fitting algorithm. For example, we may adopt SRBFs to represent other spherical functions, such as BRDFs and BTFs, and then apply the results to efficiently render objects. Additionally, since a SRBF is a distribution on $\mathbb{S}^m$, learning the parameters of scattered SRBFs for a spherical function is equivalent to exploring the distribution of this function. Thus, SRBFs may be incorporated into importance sampling techniques to decrease the computation time of ray tracing algorithms.

In the future, we intend to improve the performance and quality of the scattered SRBF fitting algorithm, and solve the rendering issues on GPUs at run-time. Thus, the scattered SRBF representation can be applied to model the light transfer functions at each vertex. We are also interested in extending this work to compress the PRT data sets of dynamic scenes, such as the precomputed shadow fields [Zhou et al. 2005].

Acknowledgements

We would like to thank Chih-Hao Chen for providing his PRT code, the members of NCTU Computer Graphics Laboratory for profound discussions, the SIGGRAPH anonymous reviewers for valuable comments and suggestions. This work was supported by the National Science Council of Taiwan under Grant No. NSC94-2213-E009-092 and a Microsoft Research Asia "Gaming & Graphics Theme 2005" award.

References

COOK, R. L., AND TORRANCE, K. E. 1982. A Reflectance Model for Computer Graphics. *ACM Transactions on Graphics 1*, 1, 7–24.

DE LATHAUWER, L., DE MOOR, B., AND VANDEWALLE, J. 2000. On the Best Rank-1 and Rank-(R_1, R_2,...,R_n) Approximation of Higher-Order Tensors. *SLAM Journal on Matrix Analysis and Applications 21*, 4, 1324–1342.

FREEDEN, W., AND WINDHEUSER, U. 1997. Combined Spherical Harmonic and Wavelet Expansion—A Future Concept in Earth's Gravitational Determination. *Applied and Computational Harmonic Analysis 4*, 1, 1–37.

FREEDEN, W., GERVENS, T., AND SCHREINER, M. 1998. *Constructive Approximation on the Sphere*. Oxford University Press.

GREEN, P., KAUTZ, J., MATUSIK, W., AND DURAND, F. 2006. View-Dependent Precomputed Light Transport Using Nonlinear Gaussian Function Approximations. In *Proceedings of ACM Symposium on Interactive 3D Graphics and Games 2006*, 7–14.

JAMES, D. L., AND FATAHALIAN, K. 2003. Precomputing Interactive Dynamic Deformable Scenes. *ACM Transactions on Graphics 22*, 3, 879–887.

KAJIYA, J. T. 1986. The Rendering Equation. In *Proceedings of SIGGRAPH 1986*, 143–150.

KAUTZ, J., LEHTINEN, J., AND AILA, T. 2004. Hemispherical Rasterization for Self-Shadowing of Dynamic Objects. In *Rendering Techniques 2004*, 179–184.

KRISTENSEN, A. W., AKENINE-MÖLLER, T., AND JENSEN, H. W. 2005. Precomputed Local Radiance Transfer for Real-Time Lighting Design. *ACM Transactions on Graphics 24*, 3, 1208–1215.

LAFORTUNE, E. P. F., FOO, S.-C., TORRANCE, K. E., AND GREENBERG, D. P. 1997. Non-Linear Approximation of Reflectance Functions. In *Proceedings of SIGGRAPH 1997*, 117–126.

LEHTINEN, J., AND KAUTZ, J. 2003. Matrix Radiance Transfer. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics*, 59–64.

LIU, X., SLOAN, P.-P. J., SHUM, H.-Y., AND SNYDER, J. 2004. All-Frequency Precomputed Radiance Transfer for Glossy Objects. In *Rendering Techniques 2004*, 337–344.

NARCOWICH, F. J., AND WARD, J. D. 1996. Nonstationary Wavelets on the m-Sphere for Scattered Data. *Applied and Computational Harmonic Analysis 3*, 4, 324–336.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2003. All-Frequency Shadows Using Non-Linear Wavelet Lighting Approximation. *ACM Transactions on Graphics 22*, 3, 376–381.

NG, R., RAMAMOORTHI, R., AND HANRAHAN, P. 2004. Triple Product Wavelet Integrals for All-Frequency Relighting. *ACM Transactions on Graphics 23*, 3, 477–487.

PATANÈ, G., AND RUSSO, M. 2001. The Enhanced LBG Algorithm. *Neural Networks 14*, 9, 1219–1237.

RAMSAY, J. O., AND SILVERMAN, B. W. 2005. *Functional Data Analysis*, 2nd ed. Springer-Verlag.

SLOAN, P.-P. J., KAUTZ, J., AND SNYDER, J. 2002. Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments. *ACM Transactions on Graphics 21*, 3, 527–536.

SLOAN, P.-P. J., HALL, J., HART, J. C., AND SNYDER, J. 2003. Clustered Principal Components for Precomputed Radiance Transfer. *ACM Transactions on Graphics 22*, 3, 382–391.

SLOAN, P.-P. J., LUNA, B., AND SNYDER, J. 2005. Local, Deformable Precomputed Radiance Transfer. *ACM Transactions on Graphics 24*, 3, 1216–1224.

VASILESCU, M. A. O., AND TERZOPOULOS, D. 2004. TensorTextures: Multilinear Image-Based Rendering. *ACM Transactions on Graphics 23*, 3, 336–342.

VLASIC, D., BRAND, M., PFISTER, H., AND POPOVIĆ, J. 2005. Face Transfer with Multilinear Models. *ACM Transactions on Graphics 24*, 3, 426–433.

WANG, R., TRAN, J., AND LUEBKE, D. P. 2004. All-Frequency Relighting of Non-Diffuse Objects using Separable BRDF Approximation. In *Rendering Techniques 2004*, 345–354.

WANG, H., WU, Q., SHI, L., YU, Y., AND AHUJA, N. 2005. Out-of-Core Tensor Approximation of Multi-Dimensional Matrices of Visual Data. *ACM Transactions on Graphics 24*, 3, 527–535.

WARD, G. J. 1992. Measuring and Modeling Anisotropic Reflection. In *Proceedings of SIGGRAPH 1992*, 265–272.

WATSON, G. N. 1944. *A Treatise on the Theory of Bessel Functions*, 2nd ed. Cambridge University Press.

ZHOU, K., HU, Y., LIN, S., GUO, B., AND SHUM, H.-Y. 2005. Precomputed Shadow Fields for Dynamic Scenes. *ACM Transactions on Graphics 24*, 3, 1196–1201.

ZHU, C., BYRD, R. H., LU, P., AND NOCEDAL, J. 1997. Algorithm 778: L-BFGS-B: Fortran Subroutines for Large-Scale Bound-Constrained Optimization. *ACM Transactions on Mathematical Software 23*, 4, 550–560.

Appendix

A The Convolution of Two Multi-Scale Gaussian SRBFs

For the convolution of two single-scale Gaussian SRBFs, i.e. Gaussian SRBFs with the same bandwidth parameters, Narcowich and Ward [1996] derived an equation. However, single-scale SRBFs become inadequate to model lighting environments with scattered SRBFs, whose bandwidth parameters should be adaptive to the input data. In this appendix, we introduce the spherical singular integral of two multi-scale Gaussian SRBFs, and derive an equation for efficient computation. The case of multi-scale Abel-Poisson SRBFs is much simple to be handled by Eq. 5.

Suppose that we intend to compute the convolution of two multi-scale Gaussian SRBFs, which corresponds to

$$\int_{\mathbb{S}^m} G^{Gau}(\eta\cdot\xi_g;\lambda_g)H^{Gau}(\eta\cdot\xi_h;\lambda_h)\,d\omega(\eta) = e^{-(\lambda_g+\lambda_h)}\int_{\mathbb{S}^m} e^{\eta\cdot(\lambda_g\xi_g+\lambda_h\xi_h)}\,d\omega(\eta)\,. \tag{24}$$

Since this convolution is rotation-invariant, we can replace r, where $r=\lambda_g\xi_g+\lambda_h\xi_h$, with $r'=(\|r\|,0,\ldots,0)$ without losing generality. Rewrite the integral on the right side of Eq. 24 in terms of spherical coordinates in $\mathbb{R}^{m+1}$, and substitute r with r' to obtain

$$\begin{aligned}&\int_{\mathbb{S}^m} e^{\eta\cdot(\lambda_g\xi_g+\lambda_h\xi_h)}\,d\omega(\eta)\\ &= \int_0^{2\pi}\cdots\int_0^{\pi} e^{(\eta\cdot r')}\sin^{m-1}\theta_1\cdots\sin\theta_{m-1}\,d\theta_1\cdots d\theta_m\\ &= \omega_{m-1}\int_0^{\pi} e^{(\|r\|\cos\theta_1)}\sin^{m-1}\theta_1\,d\theta_1\\ &= \omega_{m-1}\int_{-1}^{1} e^{(\|r\|t)}\left(1-t^2\right)^{\frac{m-2}{2}}dt\,,\quad t=\cos\theta_1\,.\end{aligned} \tag{25}$$

From [Watson 1944], the *modified Bessel function of the first kind of order* ν can be written in the integral form as

$$I_\nu(x)=\frac{\left(\frac{1}{2}x\right)^\nu}{\Gamma\left(\nu+\frac{1}{2}\right)\Gamma\left(\frac{1}{2}\right)}\int_{-1}^{1} e^{-xz}\left(1-z^2\right)^{\nu-\frac{1}{2}}dz\,. \tag{26}$$

By combining the last line of Eq. 25 with Eq. 26 and substituting the result into Eq. 24, Eq. 6 can be derived. Additionally, for the special case of $m=2$, we can further simplify Eq. 6 into

$$\left(G^{Gau}*_2 H^{Gau}\right)(\xi_g\cdot\xi_h;\lambda_g,\lambda_h)=4\pi\frac{\sinh\left(\|r\|\right)}{\|r\|}\,. \tag{27}$$

(a) Raw PRT data (b) Uncompressed PRT data in SRBFs (c) $R_{light} = 48$, $R_{vert} = 24$ (15.89 MB, 48.65/10.42 fps)

(d) $R_{light} = 24$, $R_{vert} = 24$ (9.40 MB, 49.21/16.45 fps) (e) $R_{light} = 48$, $R_{vert} = 12$ (13.00 MB, 53.47/11.13 fps) (f) $R_{light} = 24$, $R_{vert} = 12$ (7.26 MB, 53.96/16.84 fps)

Figure 5: Rendered results with various configurations of CTA. The compressed data size and the run-time FPS are shown in parentheses.

(a) Raw PRT data (b) Uncompressed PRT data in SRBFs (c) Compressed PRT data in SRBFs using CTA

Figure 6: Rendered results of the teapot model.

(a) Raw PRT data (b) Uncompressed PRT data in SRBFs (c) Compressed PRT data in SRBFs using CTA

Figure 7: Rendered results of the bunny model.