Pre-computed Radiance Transfer

Jaroslav Křivánek, KSVI, MFF UK
Jaroslav.Krivanek@mff.cuni.cz
Goal

- Real-time rendering with complex lighting, shadows, and possibly GI
- Infeasible – too much computation for too small a time budget

Approaches
- Lift some requirements, do specific-purpose tricks
  - Environment mapping, irradiance environment maps
  - SH-based lighting
- Split the effort
  - Offline pre-computation + real-time image synthesis
  - “Pre-computed radiance transfer”
SH-based Irradiance Env. Maps

Incident Radiance (Illumination Environment Map)

Irradiance Environment Map
SH-based Irradiance Env. Maps

Images courtesy Ravi Ramamoorthi & Pat Hanrahan
SH-based Arbitrary BRDF Shading

- [Kautz et al. 2003]
- Arbitrary, dynamic env. map
- Arbitrary BRDF
- No shadows

SH representation
- Environment map (one set of coefficients)
- Scene BRDFs (one coefficient vector for each discretized view direction)
Rendering: for each vertex / pixel, do

\[ L_o(\omega_o) = \int_{\Omega} L_i(\omega_i) \cdot \text{BRDF}(\omega_i, \omega_o) \cdot \cos \theta_i \cdot d\omega_i \]

= coeff. dot product

\[ L_o(\omega_o) = \Lambda_{\text{intp}}(p) \cdot F(p, \omega_o) \]
Pre-computed Radiance Transfer
Pre-computed Radiance Transfer

- **Goal**
  - Real-time + complex lighting, **shadows, and GI**
  - Infeasible – too much computation for too small a time budget

- **Approach**
  - Precompute (offline) some information (images) of interest
  - Must assume something about scene is constant to do so
  - Thereafter real-time rendering. Often hardware accelerated
Assumptions

- Precomputation
- Static geometry
- Static viewpoint (some techniques)

- Real-Time Rendering (relighting)
  - Exploit linearity of light transport
Relighting as a Matrix-Vector Multiply

\[
\begin{bmatrix}
    P_1 \\
    P_2 \\
    \vdots \\
    P_N
\end{bmatrix}
= \begin{bmatrix}
    T_{11} & T_{12} & \cdots & T_{1M} \\
    T_{21} & T_{22} & \cdots & T_{2M} \\
    T_{31} & T_{32} & \cdots & T_{3M} \\
    \vdots & \vdots & \ddots & \vdots \\
    T_{N1} & T_{N2} & \cdots & T_{NM}
\end{bmatrix}
\begin{bmatrix}
    L_1 \\
    L_2 \\
    \vdots \\
    L_M
\end{bmatrix}
\]
Relighting as a Matrix-Vector Multiply

\[
\begin{bmatrix}
  P_1 \\
  P_2 \\
  P_3 \\
  \vdots \\
  P_N
\end{bmatrix}
\begin{bmatrix}
  T_{11} & T_{12} & \cdots & T_{1M} \\
  T_{21} & T_{22} & \cdots & T_{2M} \\
  T_{31} & T_{32} & \cdots & T_{3M} \\
  \vdots & \vdots & \ddots & \vdots \\
  T_{N1} & T_{N2} & \cdots & T_{NM}
\end{bmatrix}
\begin{bmatrix}
  L_1 \\
  L_2 \\
  \vdots \\
  L_M
\end{bmatrix}
\]

Output Image
(Pixel Vector)

Input Lighting
(Cubemap Vector)

Precomputed
Transport
Matrix
Matrix Columns (Images)

\[
\begin{bmatrix}
T_{11} & T_{12} & \cdots & T_{1M} \\
T_{21} & T_{22} & \cdots & T_{2M} \\
T_{31} & T_{32} & \cdots & T_{3M} \\
\vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & \cdots & T_{NM}
\end{bmatrix}
\]
Matrix is Enormous
- 512 x 512 pixel images
- 6 x 64 x 64 cubemap environments

Full matrix-vector multiplication is intractable
- On the order of $10^{10}$ operations per frame

How to relight quickly?
Outline

- **Compression methods**
  - *Spherical harmonics-based PRT* [Sloan et al. 02]
  - (Local) factorization and PCA
  - Non-linear wavelet approximation

- Changing view as well as lighting
  - Clustered PCA
  - Triple Product Integrals

- Handling Local Lighting
  - Direct-to-Indirect Transfer
SH-based PRT

- Better light integration and transport
  - dynamic, env. lights
  - self-shadowing
  - interreflections

- For diffuse and glossy surfaces

- At real-time rates

- Sloan et al. 02
SH-based PRT: Idea
PRT Terminology

Source Radiance (spherical environment map)

inter-reflection

Transferred Incident Radiance

self-shadowing

p
## Relation to a Matrix-Vector Multiply

### a) SH coefficients of transferred radiance

### b) Irradiance (per vertex)

\[
\begin{bmatrix}
T_{11} & T_{12} & \cdots & T_{1M} \\
T_{21} & T_{22} & \cdots & T_{2M} \\
T_{31} & T_{32} & \cdots & T_{3M} \\
\vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & \cdots & T_{NM}
\end{bmatrix}
\begin{bmatrix}
L_1 \\
L_2 \\
\vdots \\
L_M
\end{bmatrix}
\]

SH coefficients of EM (source radiance)
The $L$ vector is projected onto low-frequency components (say 25). Size greatly reduced.

Hence, only 25 matrix columns

But each pixel/vertex still treated separately
- One RGB value per pixel/vertex:
  - Diffuse shading / arbitrary BRDF shading w/ fixed view direction
  - SH coefficients of transferred radiance (25 RGB values per pixel/vertex for order 4 SH)
    - Arbitrary BRDF shading w/ variable view direction

Good technique (becoming common in games) but useful only for broad low-frequency lighting
Diffuse Transfer Results

No Shadows/Inter  Shadows  Shadows+Inter
SH-based PRT with Arbitrary BRDFs

- Combine with Kautz et al. 03
- Transfer matrix turns SH env. map into SH transferred radiance
- Kautz et al. 03 is applied to transferred radiance
Arbitrary BRDF Results

Anisotropic BRDFs

Other BRDFs

Spatially Varying
Outline

- Compression methods
  - Spherical harmonics-based PRT [Sloan et al. 02]
  - (Local) factorization and PCA
  - Non-linear wavelet approximation

- Changing view as well as lighting
  - Clustered PCA
  - Triple Product Integrals

- Handling Local Lighting
  - Direct-to-Indirect Transfer
PCA or SVD factorization

• SVD:

\[ \mathbf{I}_j^{p \times n} = \mathbf{E}_j^{p \times p} \cdot \mathbf{S}_j^{p \times n} \cdot \mathbf{X} \]

-diagonal matrix (singular values)

• Applying Rank b:

\[ \mathbf{I}_j^{p \times n} \approx \mathbf{E}_j^{p \times b} \cdot \mathbf{X} \]

• Absorbing \( \mathbf{S}_j \) values into \( \mathbf{C}_j^T \):

\[ \mathbf{I}_j^{p \times n} \approx \mathbf{E}_j^{p \times b} \cdot \mathbf{L}_j^{b \times n} \]
Idea of Compression

- Represent matrix (rather than light vector) compactly
- Can be (and is) combined with SH light vector
- Useful in broad contexts.
  - BRDF factorization for real-time rendering (reduce 4D BRDF to 2D texture maps) McCool et al. 01 etc
  - Surface Light field factorization for real-time rendering (4D to 2D maps) Chen et al. 02, Nishino et al. 01
  - BTF (Bidirectional Texture Function) compression

- Not too useful for general precomput. relighting
  - Transport matrix not low-dimensional!!
Local or Clustered PCA

- Exploit local coherence (in say 16x16 pixel blocks)
  - Idea: light transport is locally low-dimensional.
  - Even though globally complex
  - See Mahajan et al. 07 for theoretical analysis

- Clustered PCA [Sloan et al. 2003]
  - Combines two widely used compression techniques: Vector Quantization or VQ and Principal Component Analysis
Compression Example

Surface is curve, signal is normal

Following couple of slides courtesy P.-P. Sloan
Compression Example

Signal Space
Cluster normals
VQ

Replace samples with cluster mean

\[ \mathbf{M}_p \approx \tilde{\mathbf{M}}_p = \mathbf{M}_{C_p} \]
PCA

Replace samples with mean + linear combination

\[ \tilde{M}_p \approx \tilde{M}_p = M^0 + \sum_{i=1}^{N} w_p^i M^i \]
CPCA

Compute a linear subspace in each cluster

\[ M_p \approx \tilde{M}_p = M^0_{C_p} + \sum_{i=1}^{N} w^i_p M^i_{C_p} \]
CPCA

- Clusters with low dimensional affine models
- How should clustering be done?
  - $k$-means clustering
- Static PCA
  - VQ, followed by one-time per-cluster PCA
  - optimizes for piecewise-constant reconstruction
- Iterative PCA
  - PCA in the inner loop, slower to compute
  - optimizes for piecewise-affine reconstruction
Static vs. Iterative
Equal Rendering Cost

VQ  PCA  CPCPA
Outline

- **Compression methods**
  - Spherical harmonics-based PRT [Sloan et al. 02]
  - (Local) factorization and PCA
  - *Non-linear wavelet approximation*

- Changing view as well as lighting
  - Clustered PCA
  - Triple Product Integrals

- Handling Local Lighting
  - Direct-to-Indirect Transfer
**Sparse Matrix-Vector Multiplication**

Choose data representations with mostly zeroes

**Vector:** Use *non-linear wavelet approximation* on lighting

**Matrix:** Wavelet-encode transport rows

\[
\begin{bmatrix}
T_{11} & T_{12} & \cdots & T_{1M} \\
T_{21} & T_{22} & \cdots & T_{2M} \\
T_{31} & T_{32} & \cdots & T_{3M} \\
\vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & \cdots & T_{NM}
\end{bmatrix}
\begin{bmatrix}
L_1 \\
L_2 \\
\vdots \\
L_M
\end{bmatrix}
\]
Haar Wavelet Basis
Non-linear Wavelet Approximation

Wavelets provide dual space / frequency locality
- Large wavelets capture low frequency area lighting
- Small wavelets capture high frequency compact features

Non-linear Approximation
- Use a dynamic set of approximating functions (depends on each frame’s lighting)
- By contrast, linear approx. uses fixed set of basis functions (like 25 lowest frequency spherical harmonics)
- We choose 10’s - 100’s from a basis of 24,576 wavelets (64x64x6)
Non-linear Wavelet Light Approximation

Wavelet Transform
Non-linear Wavelet Light Approximation

Retain 0.1% – 1% terms
Error in Lighting: St Peter’s Basilica

\[ 100 \quad 80 \quad 60 \quad 40 \quad 20 \quad 0 \]

Relative \( L^2 \) Error (%)

1 \quad 10 \quad 100 \quad 1000 \quad 10000

Approximation Terms

Sph. Harmonics

Non-linear Wavelets

Ng, Ramamoorthi, Hanrahan 03
Output Image Comparison

Top: Linear Spherical Harmonic Approximation
Bottom: Non-linear Wavelet Approximation

25 200 2,000 20,000
Outline

- Compression methods
  - Spherical harmonics-based PRT [Sloan et al. 02]
  - (Local) factorization and PCA
  - Non-linear wavelet approximation

- **Changing view as well as lighting**
  - **Clustered PCA**
  - Triple Product Integrals

- Handling Local Lighting
  - Direct-to-Indirect Transfer
SH + Clustered PCA

- Described earlier (combine Sloan 03 with Kautz 03)
  - Use low-frequency source light and transferred light variation (Order 5 spherical harmonic = 25 for both; total = 25*25=625)
  - 625 element vector for each vertex
  - Apply CPCA directly (Sloan et al. 2003)
  - Does not easily scale to high-frequency lighting
    - Really cubic complexity (number of vertices, illumination directions or harmonics, and view directions or harmonics)
  - Practical real-time method on GPU
Outline

- Compression methods
  - Spherical harmonics-based PRT [Sloan et al. 02]
  - (Local) factorization and PCA
  - Non-linear wavelet approximation

- *Changing view as well as lighting*
  - Clustered PCA
  - *Triple Product Integrals*

- Handling Local Lighting
  - Direct-to-Indirect Transfer
Problem Characterization

6D Precomputation Space

- Distant Lighting (2D)
- View (2D)
- Rigid Geometry (2D)

With ~ 100 samples per dimension
~ $10^{12}$ samples total!!: Intractable computation, rendering
Factorization Approach

6D Transport

\[ \approx 10^{12} \text{ samples} \]

\[ = \]

\[ \approx 10^{8} \text{ samples} \]

4D Visibility

\[ \ast \]

4D BRDF

\[ \approx 10^{8} \text{ samples} \]
Triple Product Integral Relighting
Relit Images (3-5 sec/frame)
\[ B = \int_{S^2} L(\omega) V(\omega) \tilde{\rho}(\omega) \, d\omega \]

\[ = \int_{S^2} \left( \sum_i L_i \Psi_i(\omega) \right) \left( \sum_j V_j \Psi_j(\omega) \right) \left( \sum_k \tilde{\rho}_k \Psi_k(\omega) \right) \, d\omega \]

\[ = \sum_i \sum_j \sum_k L_i V_j \tilde{\rho}_k \int_{S^2} \Psi_i(\omega) \Psi_j(\omega) \Psi_k(\omega) \, d\omega \]

\[ = \sum_i \sum_j \sum_k L_i V_j \tilde{\rho}_k C_{ijk} \]
Basis Requirements

\[ B = \sum_{i} \sum_{j} \sum_{k} L_i V_j \tilde{\rho}_k C_{ijk} \]

1. Need few non-zero “tripling” coefficients

\[ C_{ijk} = \int_{S^2} \Psi_i(\omega) \Psi_j(\omega) \Psi_k(\omega) \, d\omega \]

2. Need sparse basis coefficients

\( L_i, V_j, \tilde{\rho}_k \)
1. Number Non-Zero Tripling Coeffs

\[ C_{ijk} = \int_{S^2} \Psi_i(\omega) \Psi_j(\omega) \Psi_k(\omega) \, d\omega \]

<table>
<thead>
<tr>
<th>Basis Choice</th>
<th>Number Non-Zero Coeffs</th>
</tr>
</thead>
<tbody>
<tr>
<td>General (e.g. PCA)</td>
<td>( O(N^3) )</td>
</tr>
<tr>
<td>Sph. Harmonics</td>
<td>( O(N^{5/2}) )</td>
</tr>
<tr>
<td>Haar Wavelets</td>
<td>( O(N \log N) )</td>
</tr>
</tbody>
</table>
2. Sparsity in Light Approx.

![Graph showing relative $L^2$ error (%) vs. approximation terms for Pixels and Wavelets.](image)
Summary of Wavelet Results

- Derive direct $O(N \log N)$ triple product algorithm
- Dynamic programming can eliminate $\log N$ term
- Final complexity linear in number of retained basis coefficients
Outline

- Compression methods
  - Spherical harmonics-based PRT [Sloan et al. 02]
  - (Local) factorization and PCA
  - Non-linear wavelet approximation

- Changing view as well as lighting
  - Clustered PCA
  - Triple Product Integrals

- **Handling Local Lighting**
  - *Direct-to-Indirect Transfer*
Direct-to-Indirect Transfer

- Lighting non-linear w.r.t. light source parameters (position, orientation etc.)
- Indirect is a linear function of direct illumination
  - Direct can be computed in real-time on GPU
  - Transfer of direct to indirect is pre-computed
- Hašan et al. 06
  - Fixed view – cinematic relighting with GI
**DTIT: Matrix-Vector Multiply**

\[
\begin{bmatrix}
P_1 \\
P_2 \\
P_3 \\
\vdots \\
P_N
\end{bmatrix}
\begin{bmatrix}
T_{11} & T_{12} & \cdots & T_{1M} \\
T_{21} & T_{22} & \cdots & T_{2M} \\
T_{31} & T_{32} & \cdots & T_{3M} \\
\vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & \cdots & T_{NM}
\end{bmatrix}
\begin{bmatrix}
L_1 \\
L_2 \\
\vdots \\
L_M
\end{bmatrix}
\]

Direct illumination on a set of samples distributed on scene surfaces

Compression: Matrix rows in Wavelet basis
DTIT: Demo
Summary

- Really a big data compression and signal-processing problem
- Apply many standard methods
  - PCA, wavelet, spherical harmonic, factor compression
- And invent new ones
  - VQPCA, wavelet triple products
- Guided by and gives insights into properties of illumination, reflectance, visibility
  - How many terms enough? How much sparsity?