Today’s slides courtesy of Kun Xu,
Tsinghua University
A Brief Recap

- Precomputed Radiance Transfer
- Basis functions
  - Spherical Harmonics
  - Wavelets
Rendering under environment lighting

\[ L(\mathbf{o}) = \int_{\Omega} L(\mathbf{i}) V(\mathbf{i}) \rho(\mathbf{i}, \mathbf{o}) \max(0, \mathbf{n} \cdot \mathbf{i}) \, d\mathbf{i} \]

- **i/o:** incoming/view directions
- **Brute-force computation**
  - Resolution: 6*64*64
  - Needs 6*64*64 times for each point!
Basic idea of PRT [Sloan 02]

\[ L(o) = \int_{\Omega} L(i)V(i)\rho(i, o) \max(0, n \cdot i) \, di \]

- Approximate lighting using basis functions
  - \( L(i) \approx \sum l_i B_i(i) \)
- Precomputation stage
  - compute light transport, and project to basis function space
- Runtime stage
  - dot product (diffuse) or matrix-vector multiplication (glossy)
Basis functions $B(i)$

- Spherical Harmonics (SH)
- Light Approximation Examples

Low frequency
Basis functions $B(\mathbf{i})$

- **Original space**

- **SH space**

\[ L(\mathbf{i}) \approx \sum l_i B_i(\mathbf{i}) \]

- **Projection to SH space**

- **Reconstruction**

\[ l_i = \int_{\Omega} L(\mathbf{i}) \cdot B_i(\mathbf{i}) d\mathbf{i} \]

\[ L(\mathbf{i}) \approx \sum l_i B_i(\mathbf{i}) \]
Run-time Rendering

\[ L(\mathbf{o}) \approx \rho \sum l_i T_i \]

- Rendering at each point is reduced to a dot product
  - First, project the lighting to the basis to obtain \( l_i \)
  - Or, rotate the lighting instead of re-projection
  - Then, compute the dot product

- Real-time: easily implemented in shader
Wavelet [Ng 03]

- 2D Haar wavelet

- Projection:
  - Wavelet Transformation
  - Retain a small number of non-zero coefficients

- A non-linear approximation

- All-frequency representation
Non-linear Wavelet Light Approximation

Wavelet Transform
Non-linear Wavelet Light Approximation

Retain 0.1% – 1% terms
# Matrix Row Wavelet Encoding

\[
\begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\
T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM}
\end{bmatrix}
\]
Matrix Row Wavelet Encoding

\[
\begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\
T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM}
\end{bmatrix}
\]

Wavelet Transform
Matrix Row Wavelet Encoding

\[
\begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & \ldots & T_{1M} \\
T_{21} & T_{22} & T_{23} & T_{24} & \ldots & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \ldots & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \ldots & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \ldots & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \ldots & T_{6M} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \ldots & T_{NM} \\
\end{bmatrix}
\]

Wavelet Transform
Matrix Row Wavelet Encoding

\[
\begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\
T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM}
\end{bmatrix}
\]

Wavelet Transform
Matrix Row Wavelet Encoding

\[
\begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\
T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM}
\end{bmatrix}
\]

Wavelet Transform
Matrix Row Wavelet Encoding

\[
\begin{bmatrix}
T_{11}' & 0 & 0 & T_{14}' & \cdots & 0 \\
T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM}
\end{bmatrix}
\]

Store Back in Matrix
Matrix Row Wavelet Encoding

\[
\begin{bmatrix}
T_{11}' & 0 & 0 & T_{14}' & \cdots & 0 \\
T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM}
\end{bmatrix}
\]

Only 3% – 30% are non-zero
Today

- More Basis functions
  - Zonal Harmonics
  - Spherical Gaussian
  - Piecewise Constant
  - ...

- Efforts to improve PRT
  - Dynamic objects
  - Translucent / Hair / etc.
  - Material Editing
  - Analytic Interreflection
  - ...

Zonal Harmonics [Sloan 05]

- circularly symmetric functions
- Subset of SH basis (m=0)
- Low-frequency
- Rotational invariant
- Much more faster in rotation than SH
Spherical Gaussian (SG) [Tsai 06]

SGs (or Spherical Radial Basis Functions, SRBFs)

\[ G(\mathbf{v}; \mathbf{p}, \lambda) = e^{\lambda(\mathbf{v} \cdot \mathbf{p} - 1)} \]

center    bandwidth

varying center
Spherical Gaussian (SG) [Tsai 06]

- SGs (or Spherical Radial Basis Functions, SRBFs)

\[ G(\mathbf{v}; \mathbf{p}, \lambda) = e^{\lambda(\mathbf{v} \cdot \mathbf{p} - 1)} \]

- center
- bandwidth

Increasing bandwidth
Mathematical Properties of SGs

- Closed-form integral
  - The integral of an SG is closed-form

\[
\int_{\Omega} G(v; p, \lambda) \, dv = \frac{2\pi}{\lambda} (1 - e^{-2\lambda})
\]
Mathematical Properties of SGs

- Closed under multiplication
  
  The product of two SGs is also an SG

\[
G(v; p_1, \lambda_1) \cdot G(v; p_2, \lambda_2) = cG\left(v; \frac{\lambda_1 p_1 + \lambda_1 p_2}{|\lambda_1 p_1 + \lambda_1 p_2|}, |\lambda_1 p_1 + \lambda_1 p_2|\right)
\]

\[G_{iso}(v; p_1, \lambda_1) \cdot G_{iso}(v; p_2, \lambda_2) = \text{Product}\]
Mathematical Properties of SGs

- Closed under convolution approximately
  - The convolution of two SGs is still an SG

\[
\int_\Omega G(v; p_1, \lambda_1) \cdot G(v; p_2, \lambda_2) dv \approx c_3 G\left(p_1; p_2, \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}\right)
\]

\[G_{iso}(v; p_1, \lambda_1) \ast G_{iso}(v; p_2, \lambda_2) = \text{Convolution}\]
Summary of SGs

- Rotationally invariant
  - Lighting, BRDFs demand rotation
- Capable of representing all-frequency signals
  - All-frequency lighting/BRDFs
- Closed-form integral
  - Rendering is essentially integration [Kajiya 1986]
- Closed under multiplication
  - Multiplication of lighting, visibility and BRDFs
- Closed under convolution
  - Support for various applications

- SGs are non-orthogonal!
Lighting Approximation

- Non-linear process: iterative L-BFGS-B solver (slow)

\[ L(i) \approx \sum l_i G(i; p_i, \lambda_i) \]
BRDF Factorization [Wang 03, Liu 03]

- Precompute the factorization

\[ \rho(i, o) \approx \sum_{m} f_{m}(i) \cdot g_{m}(o) \]
Overall Rendering Algorithm

Derivation: Factorizing BRDF

\[
L(o) = \int_{\Omega} L(i)V(i)\rho(i, o) \max(0, n \cdot i) \, di
\]

\[
L(o) = \int_{\Omega} L(i)V(i) \left( \sum_{m} f_{m}(i) \cdot g_{m}(o) \right) \max(0, n \cdot i) \, di
\]

\[
L(o) = \sum_{m} g_{m}(o) \int_{\Omega} L(i)V(i)f_{m}(i) \max(0, n \cdot i) \, di
\]

Both represented using SGs
Overall Rendering Algorithm

- **Derivation:** projection to SGs

\[
L(o) = \sum_m g_m(o) \int_\Omega L(i)T(i) \, di
\]

- \(L(i) \approx \sum l_i G_i(i)\)
  - non-linear approx.

- \(T(i) \approx \sum t_j G_j(i)\)
  - pre. scattered approx.

- \(L(o) = \sum_m g_m(o) \sum_{i,j} l_i t_j \int_\Omega G_i(i)G_j(i) \, di\)
  - analytic solution

- **Timing:** \(O(N*N*M)\), non-orthogonal
Results

Figure 6: Rendered results of the teapot model.
Piecewise Constant [Xu 08]

- Spherical Piecewise Constant Basis Function (SPCBF)
  - Split sphere into regions
  - Each region is represented by a constant

- Property
  - All-frequency
  - Rotation-Invariant
  - Multi-product
  - Fast projection
Piecewise Constant [Xu 08]

- Light Projection
  - Bottom-up optimization
Piecewise Constant [Xu 08]

- Projection of visibility and BRDFs
  - BRDF
    - using summed area table
  - Visibility
    - Using visibility distance table
Results
## Comparison of Basis Functions

<table>
<thead>
<tr>
<th>Feature</th>
<th>SH</th>
<th>Wavelet</th>
<th>SG</th>
<th>SPCBF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orthogonal</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>All-frequency</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Rotation invariant</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Multiple product</td>
<td>✓</td>
<td>✓</td>
<td>✓ ?</td>
<td>✓</td>
</tr>
<tr>
<td>Compact Representation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
</tr>
</tbody>
</table>
Triple Product

- Original PRT: light * light transport

\[ L(o) = \int_{\Omega} L(i) V(i) \rho(i, o) \max(0, n \cdot i) \, di \]

- Triple Product

- ... Multiple Product

- Not Flexible
Wavelet Triple Product [Ng 04]

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

\[
B = \int_{S^2} L(\omega) V(\omega) \tilde{\rho}(\omega) d\omega
\]

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

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

\[
= \sum_i \sum_j \sum_k L_i V_j \tilde{\rho}_k C_{ijk}
\]
Wavelet Triple Product [Ng 04]

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

<table>
<thead>
<tr>
<th>Basis Choice</th>
<th>Number Non-Zero ( C_{ijk} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>General (e.g. PCA)</td>
<td>( O(N^3) )</td>
</tr>
<tr>
<td>Pixels</td>
<td>( O(N) )</td>
</tr>
<tr>
<td>Fourier Series</td>
<td>( O(N^2) )</td>
</tr>
<tr>
<td>SH</td>
<td>( O(N^{5/2}) )</td>
</tr>
<tr>
<td>Haar Wavelets</td>
<td>( O(N \log N) )</td>
</tr>
</tbody>
</table>
Wavelet Triple Product [Ng 04]
SG Triple Product

- **Analytic Computation**
  - The product of two SGs is also an SG
    
    \[
    G(v; p_1, \lambda_1) \cdot G(v; p_2, \lambda_2) = cG\left(v; \frac{\lambda_1 p_1 + \lambda_1 p_2}{|\lambda_1 p_1 + \lambda_1 p_2|}, |\lambda_1 p_1 + \lambda_1 p_2|\right)
    \]

Could be easily extended to multiple product
SH Triple Product

- Precompute all triple products of SH basis

\[ C_{ijk} = \int_{\Omega} B_i(i)B_j(i)B_k(i) \, d\mathbf{i} \]

- Compute the product of two functions directly in SH space:

Original space

\[ L(i) \approx \sum l_i B_i(i) \]

\[ T(i) \approx \sum t_j B_j(i) \]

\[ L(i) \cdot T(i) \approx \sum l_t k B_k(i) \]

SH space

- Could be easily extended to multiple product
Shadow Field [Zhou05]

- PRT
  - Handle only static scenes

- Shadow Field
  - Handle moving light sources & objects
  - Rigid objects + dynamic scene configuration
  - Capture SRF/OOF around lights/objects
Sampling & Compression (Low Frequency)

- 16 spheres
- 6x16x16 samples per sphere
- OOFs: 384 KB
- SRFs: 1 MB

Spherical Harmonics
Rendering: Products

BRDF
***
OOF-1
***
OOF-3
**
SRF-4
= Reflected radiance
Results
SH Exponential [Ren 06]

- **Shadow Field**
  - Rigid objects
  - Computation of SH multiple product is still costly

- **SH Exponential**
  - Dynamic, deformation scene (objects)
  - Derive Exp/Log operators in SH space
  - Convert costly multiple product to summation in log space
Blocker Geometry Approximation

- Using sphere sets
- Dynamically update at each frame

original

$E=0.35$, $n_s=64$

our method
Rendering Computation

- Multiple Product (a very big number)
  
  Light * Self_Vis * Occlusion1 * Occlusion2 * .... * OcclusionN * BRDF ***

- Approach

\[ f_1 \times f_2 \times \cdots \times f_n \]

\[ \exp(\log f_1 + \log f_2 + \cdots + \log f_n) \]

- Implement \( \exp/\log \) directly in SH space
- Much faster
Results

Dinosaur Demo

120k vertices (75k static, 45k dynamic)
500 spheres in blocker approximation
250 receiver clusters
12.6 Hz average frame rate
More Rendering Applications

- Translucent Rendering
- Hair Rendering
- BRDF Editing
- Translucent Editing
- Hair Editing
Translucent Rendering [Wang05]

- Extend PRT to handle translucent materials

\[ L(x_o, o) = \int_A \int_\Omega L(x_i, i) S(x_i, i; x_o, o) \max(0, n \cdot i) \, di \]

- Precompute the transport for multiple scattering and single scattering separately
Results

- Dynamic environment lighting
- Fix: geometry + materials
- Real-time
Hair Rendering [Ren 10]

- Extend PRT to handle hair rendering
  - Support environment lighting

- Single Scattering
  - Self-shadowing
  - Fiber scattering
  - Transparency
- Multiple Scattering
- Natural Illumination
Single Scattering Computation

\[ L(o) = D \int_{\Omega} L(i) T(i) S(i, o) \max(0, n \cdot i) \, di \]

- Approximate \( L(o) \) by \( N \) SGs
- Move \( T \) out from the integral
  - small variation of \( T \)

\[ L(o) = D \sum_{j=1}^{N} L_j \tilde{T} \int_{\Omega} G_j(i) S(i, o) \max(0, n \cdot i) \, di \]

Precompute as 4D table
Results

- Dynamic lighting, geometry
- Fix hair scattering parameters
- Interactive framerates
BRDF Editing [Ben-Artzi 06]

**PRT**
- dynamic lighting + precompute light transport
- Fix: material + geometry

\[ L(o) = \int_\Omega L(i)V(i) \max(0, n \cdot i) \rho(i, o) di \]

**PRT based BRDF editing**
- dynamic material + precompute material transport
- Fix: lighting + geometry + viewpoint
BRDF Editing [Ben-Artzi 06]

- **Approach:** parameterize BRDF as 1D curve
  \[ \rho(i, o) = \rho_q(i, o)f(\gamma(i, o)) \]
  
  - Quotient term
  - 1D curve
  
  \[ f(\gamma) \approx \sum c_j b_j(\gamma) \]

- **Rendering Algorithm**
  - Precompute:
    \[ T_j = \int_{\Omega} L(i)V(i) \max(0, n \cdot i) \rho_q(i, o)b_j(\gamma(i, o)) \, di \]
  - Runtime: (viewing direction \( o \) is fixed)
    \[ L(o) \approx \sum c_j T_j \]
Results
BRDF Editing with interreflection [Sun06]

- dynamic lighting + viewpoint + material
- Fix: geometry
- all-frequency one bounce interreflection
- Introduce PTT: precomputed transfer tensors
Results

- Interactive rates
Translucent Editing [Xu 07]

- Combine the ideas in “BRDF editing” and in “translucent rendering”
  - dynamic dipole parameters + precompute material transport
  - Compute single/multiple scattering separately
  - Basis Function: piecewise linear
Results

- Real-time, environment lighting
- Fix: lighting + geometry
- Changing scattering parameters
PRT vs analytic integration

Rendering Integral \[ \int_{\Omega} L(i)V(i)\rho(i)\,di \]

- **PRT (Precomputation)**
  - Long precomputation time, large storage
  - Bake geometry/material/lighting into precomputation, needs to fix them

- **Analytic Computation**
  - No (or small) precomputation
  - Everything dynamic, could be run-time changed
SG based analytic Integration

- SG as a PRT basis [Tsai 2006]
- Rendering with dynamic BRDFs [Wang 2009]
- Frequency domain normal mapping [Han 2007]
- Rendering and appearance editing of hairs [Xu 2011]
- Bi-scale BRDF editing [Iwasaki 2012]
- Real-time rendering of rough refractions [Rousiers 2012]
- One-bounce interreflection [Xu 2014]
- Anisotropic spherical Gaussians [Xu 2014]
Rendering with dynamic BRDFs [Wang09]

- static scene, dynamic lighting, dynamic BRDF
- BRDF: microfacet model
  - parametric $\leftrightarrow$ measured
  - isotropic $\leftrightarrow$ anisotropic
  - glossy $\leftrightarrow$ mirror-like
Algorithm Overview

- BRDF Slice
- Visibility
- Light

Spherical Gaussians  SSDF  Prefiltered Environment
Results
Rendering and appearance editing of hairs [Xu 2011]

Single scattering

\[ L(\omega_o) = D \int_{\Omega} L(\omega_i) T(\omega_i) S(\omega_i, \omega_o) \cos \theta_i \, d\omega_i \]

- \( L(\omega_i) \): environment lighting
- \( T(\omega_i) \): self shadowing
- \( S(\omega_i, \omega_o) \): hair scattering function
Rendering and appearance editing of hairs \([Xu\ 2011]\)

Single scattering

\[
L(\omega_o) \approx D \sum_j \left( \sum_{\omega_i} \left( T_F(\omega_o, \omega_i) T_I(\omega_o, \omega_i) G_j(\omega_i) \right) \right) \cos \theta_i d\omega_i
\]

- Approximate \(L(\omega_i)\) by a set of SGs \(G_j(\omega_i)\) \([Tsai\ and\ Shih\ 2006]\)
Rendering and appearance editing of hairs [Xu 2011]

Single scattering

\[ L(\omega_o) \approx D \sum_j l_j \frac{1}{\lambda} \int \int \int \frac{G_j(\omega_i) T(\omega_i) S(\omega_i, \omega_o, \omega_c) \cos \theta_i}{d\omega_i} d\omega_i \]

- Approximate \( L(\omega_i) \) by a set of SGs \( G_i(\omega_i) \) [Tsai and Shih 2006]
- Move T out from the integral [Ren 2010]

Problem: evaluate scattering Integral
Single Scattering Integral

\[ \int_{\Omega} G_j(\omega_i) S(\omega_i, \omega_o) \cos \theta_i \, d\omega_i \]

- Previous Approach [Ren 2010]
  - Precompute the integral into 4D table

- Our key insight
  - Is there an approximated analytic solution?
  - YES
    - Decompose SG \( G_j(\omega_i) \) into products of \textit{circular Gaussians}
    - Approximate scattering function \( S(\omega_i, \omega_o) \) by \textit{circular Gaussians}
Results

- No precomputation
- all (geometry, lighting, hair scattering param.) dynamic
One-bounce interreflection [Xu 14]

- Aim at accurately and efficiently computing one-bounce interreflections with *all-frequency* BRDFs
- SG-based representation of BRDFs and lighting
- A novel *analytic* rendering formula
One-bounce Interreflection Model

\[
L_x(o) = \int_{\Omega_T} \rho_x(-r,o) \max(-r \cdot n_x, 0) \int_{\Omega} L_l(i) \rho_T(i, r) \max(i \cdot n_T, 0) \, di \, dr
\]

Configuration:
- Single triangle reflector
- Distant lighting
- No occlusion between the light, the reflector, and the receiver
- Ignore textures on the reflector
- Uniform BRDF (reflector)

Triangle \( T \) (reflector)

Light \( l \)

- \( o \)
- \(-i\)
- \( n_T \)
- \( r \)
- receiver

Configuration:
- Single triangle reflector
- Distant lighting
- No occlusion between the light, the reflector, and the receiver
- Ignore textures on the reflector
- Uniform BRDF (reflector)
One-bounce Interreflection Model

\[ L_x(o) = \int_{\Omega_T} \rho_x(-r,o) \max(-r \cdot n_x, 0) \int_{\Omega} L_l(i) \rho_T(i, r) \max(i \cdot n_T, 0) \, di \, dr \]

piecwise linear approximation

Approximated closed solution

Represented by SGs

light \( l \)

triangle \( T \)

receiver

reflector

analytically evaluated!
Results
Limitation of SGs

- Representing real functions
  - A mixture model of $n$ scattered SGs are required
- Poor scalability
  - More anisotropic functions require more SGs
- Making Trade-off
  - Larger $n \rightarrow$ more accuracy, more cost
  - Smaller $n \rightarrow$ less accuracy, less cost

An example
Anisotropic SG [Xu 14]

\[ G(\mathbf{v}; [\mathbf{x}, \mathbf{y}, \mathbf{z}], [\lambda, \mu]) = \max(\mathbf{v} \cdot \mathbf{z}, 0) \cdot e^{-\left(\lambda (\mathbf{v} \cdot \mathbf{x})^2 + \mu (\mathbf{v} \cdot \mathbf{y})^2\right)} \]

- tangent
- bi-tangent
- lobe
- bandwidth for \( \mathbf{x} \)-axis
- bandwidth for \( \mathbf{y} \)-axis
- smooth term
- exponential term

An ASG example with \( \lambda = 4 \) and \( \mu = 1 \)

3D view

Top view (2D)
ASGs

- Desired operators
  - Closed-form integral
  - Closed-form product
  - Closed-form convolution
Integral of an ASG

- Integral

\[ \int_{\Omega} G(v) dv \]

\[ = \int_{\phi=0}^{2\pi} \left( \int_{\theta=0}^{\pi/2} e^{-\lambda (\sin \theta \cos \phi)^2 - \mu (\sin \theta \sin \phi)^2} \sin \theta \cos \theta \, d\theta \right) d\phi \]

- Our approximation

\[ \int_{\Omega} G(v) dv \approx \frac{\pi}{\sqrt{\lambda \mu}} \]

- Accurate (error < 0.68%) when \( \lambda, \mu > 5 \)
Product of two ASGs

- Our approximation: \( G(v; A_1) \cdot G_2(v; A_2) \approx S(z_3; z_1, z_2) \cdot G(v; A_3) \)

- Validation
  - 1\(^{\text{st}}\) ASG: \( G(v; A_1) \)
  - 2\(^{\text{nd}}\) ASG: \( G(v; A_2) \)

Approximated product
Ground truth product
Convolution of an ASG with an SG

- Our approximation: $C(p) \approx \frac{\pi}{\sqrt{(\lambda+\nu)(\mu+\nu)}} \cdot G\left(p; [x, y, z], \left[\frac{\nu\lambda}{\nu+\lambda}, \frac{\nu\mu}{\nu+\mu}\right]\right)$
Results
Linearly Transformed Cosines [Heitz 18]

- Approximate BRDFs using *Linearly Transformed Cosines Functions*
- Analytical integration on spherical polygons
Misc

- Compression
  - VQ, PCA, Clustered PCA [Sloan 03]
- Meshless [Lehtinen 08]
- Image space
  - Direct-to-indirect Transfer [Hašan 06]
Misc

- Neural network as a basis
  - Radiance Regression Functions [Ren 2013]

- Deep Shading [Nalbach 2017]
Reading Materials

- SIGGRAPH 2005 Course, by Jan Kautz et al
  - Precomputed Radiance Transfer: Theory and Practice
    www0.cs.ucl.ac.uk/staff/j.kautz/PRTCourse/
- PRT survey, 2007, by Ravi Ramamoorthi
  - Precomputation-Based Rendering
- EG STAR 2012 Report, by Ritschel et al
  - The State of the Art in Interactive Global Illumination
Conclusion

- Precomputed Radiance Transfer
  - Project light/transport to basis function space
  - Precompute and save the transport
  - Efficient computing at run-time

- Various rendering applications/features
  - environment lighting, local lighting
  - BRDFs/ translucent
  - Material editing
  - Static/dynamic scenes
  - Interreflection
  - ...
Thanks!

Questions?