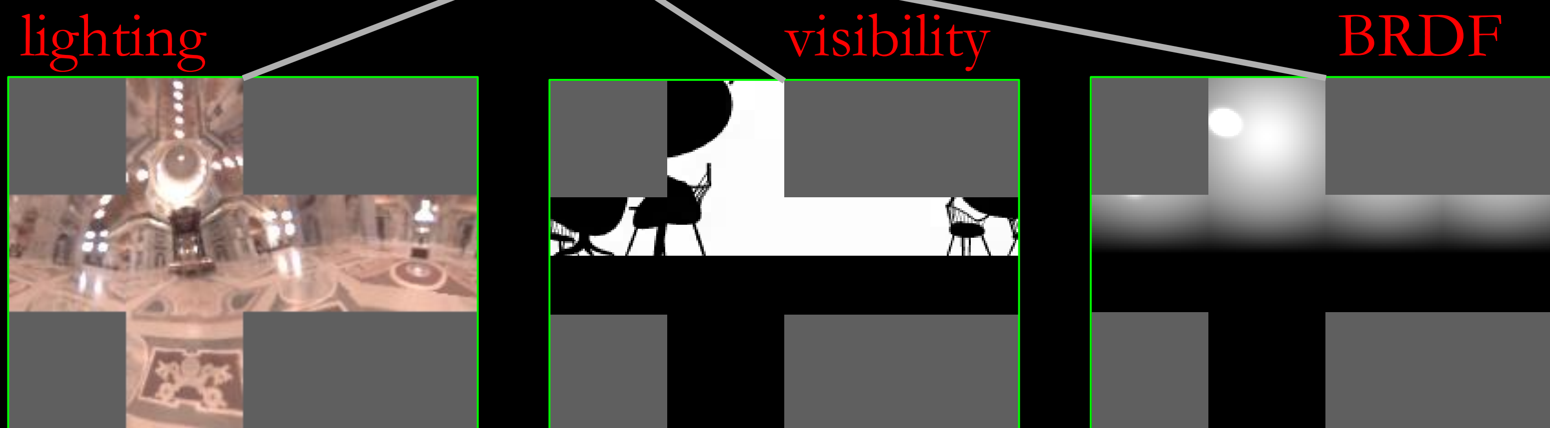
Rendering Tutorial 4: Precomputed Radiance Transfer (预计算辐射传输)

徐昆
清华大学

2018-10-11

Rendering under environment lighting

$$L(\mathbf{o}) = \int_{\Omega} \underline{L(\mathbf{i})}\,\underline{V(\mathbf{i})}\,\underline{\rho(\mathbf{i},\mathbf{o})}\,\max(0, \boldsymbol{n}\cdot\mathbf{i})\,\mathrm{d}\mathbf{i}$$



- $\mathbf{i}/\mathbf{o}$: incoming/view directions
- Brute-force computation
  - Resolution: 6*64*64
  - Needs 6*64*64 times for each point!

Precomputed Radiance Transfer (PRT)

- Introduced by Sloan in SIGGRAPH 2002
  - *Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments* [Sloan 02]

Basic idea of PRT [Sloan 02]

$$L(\mathbf{o}) = \int_{\Omega} \underbrace{L(\mathbf{i})}_{\text{lighting}} \underbrace{V(\mathbf{i})\rho(\mathbf{i},\mathbf{o}) \max(0, \boldsymbol{n} \cdot \mathbf{i})}_{\text{light transport}} \, \mathrm{d}\mathbf{i}$$

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

Diffuse Case

$$L(\mathbf{o}) = \rho \int_{\Omega} \underline{L(\mathbf{i})} V(\mathbf{i}) \max(0, \boldsymbol{n} \cdot \mathbf{i}) \, \mathrm{d}\mathbf{i}$$

$$L(\mathbf{i}) \approx \sum \underbrace{l_i}_{\text{lighting coefficient}} \underbrace{B_i(\mathbf{i})}_{\text{basis function}}$$

$$L(\mathbf{o}) \approx \rho \sum l_i \underbrace{\int_{\Omega} \mathrm{B}_i(\mathbf{i}) V(\mathbf{i}) \max(0, \boldsymbol{n} \cdot \mathbf{i}) \, \mathrm{d}\mathbf{i}}$$

Precompute

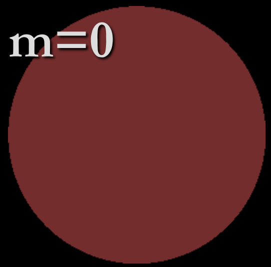
$$L(\mathbf{o}) \approx \rho \sum l_i T_i$$

- Reduce rendering computation to dot product

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

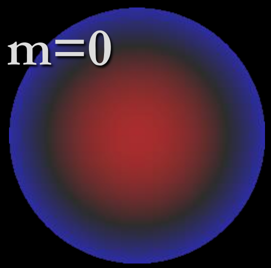
- Spherical Harmonics (SH)
- SH have nice properties:
  - orthonormal
  - simple projection/reconstruction
  - rotationally invariant (no aliasing)
  - simple rotation
  - simple convolution
  - few basis functions: low freqs

l=0 m=0

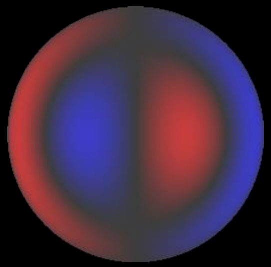


l=1 m=-1

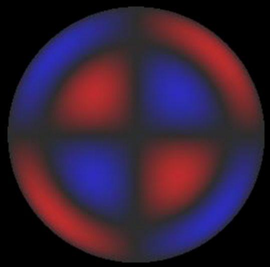
l=1 m=0



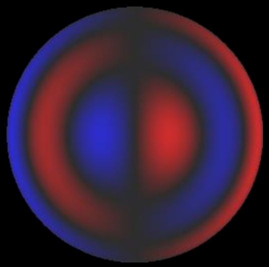
l=1 m=1



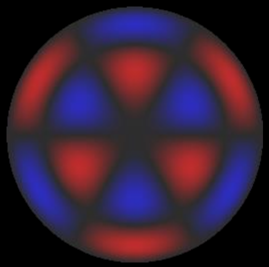
l=2 m=1



l=3 m=-1



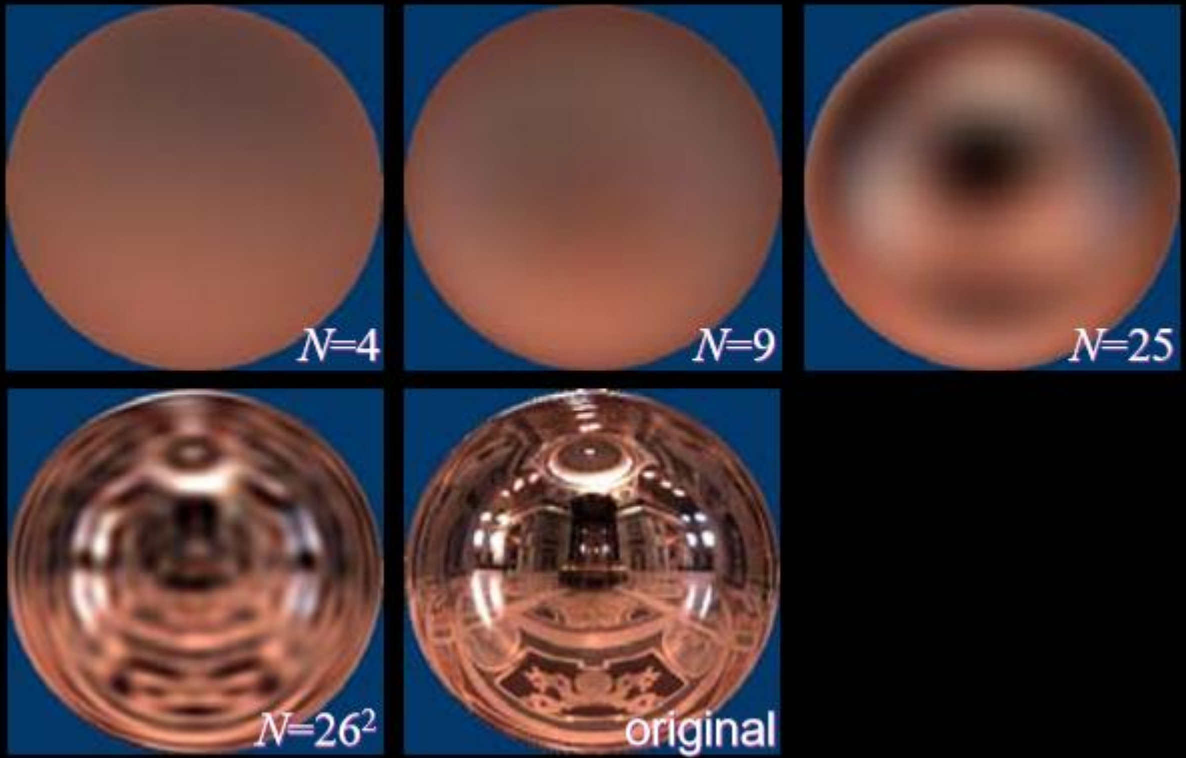
l=3 m=2



l=4 m=-2

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

- Spherical Harmonics (SH)
- Light Approximation Examples



Low frequency

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

- SH is orthonormal, we have:

$$\int_{\Omega} B_i(\mathbf{i}) \cdot B_j(\mathbf{i}) \mathrm{d}\mathbf{i} = \mathbf{1} \quad (\mathbf{i} = j)$$

$$\int_{\Omega} B_i(\mathbf{i}) \cdot B_j(\mathbf{i}) \mathrm{d}\mathbf{i} = \mathbf{0} \quad (\mathbf{i} \neq j)$$

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

Original space

lighting

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

SH space

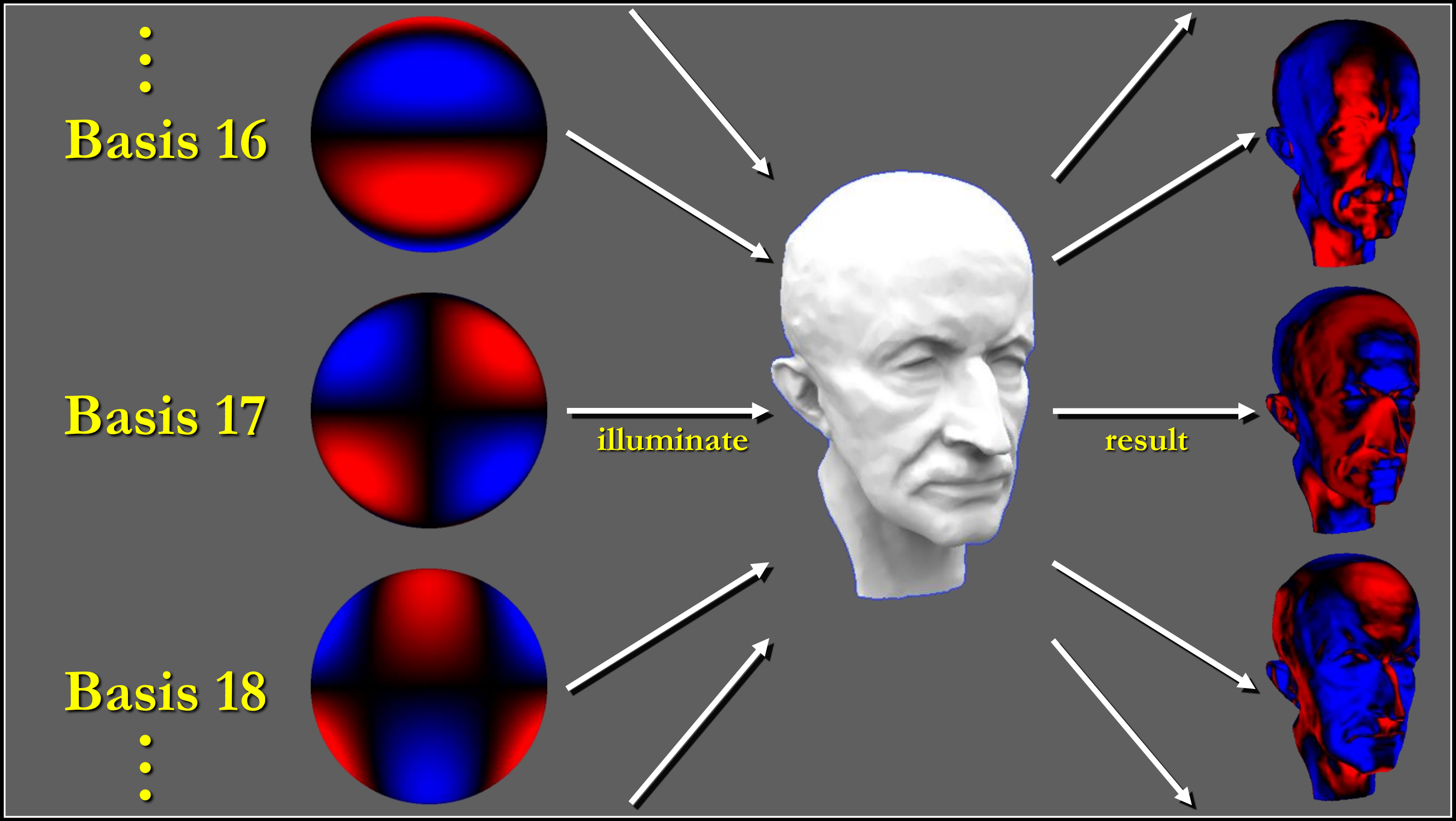
lighting coefficients

- Projection to SH space $l_i = \int_\Omega L(\mathbf{i}) \cdot B_i(\mathbf{i}) \mathrm{d}\mathbf{i}$
- Reconstruction $L(\mathbf{i}) \approx \sum l_i B_i(\mathbf{i})$

Precomputation

light transport $T_i \approx \int_{\Omega} \mathrm{B}_i(\mathbf{i}) \underline{V(\mathbf{i}) \max(0, \boldsymbol{n} \cdot \mathbf{i})}\, \mathrm{d}\mathbf{i}$

- No shadow/ shadow/ inter-reflection



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

Diffuse Rendering Results

No Shadows

Shadows

Shadows+Inter

Glossy Case

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

$$L(\mathbf{o}) \approx \sum l_i \underline{T_i(\mathbf{o})}$$

$$T_i(\mathbf{o}) \approx \sum \underbrace{t_{ij}}_{\text{transport matrix}} \underbrace{B_j(\mathbf{o})}_{\text{basis function}}$$

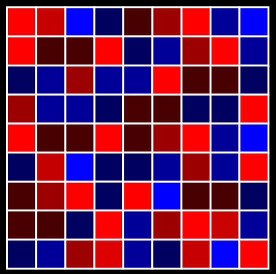
$$L(\mathbf{o}) \approx \sum \left(\sum l_i t_{ij} \right) B_j(\mathbf{o})$$

reflected radiance coefficient

$\approx$

light coefficient

$*$



transport matrix

- Rendering: vector-matrix multiplication

Time Complexity

- #SH Basis : 9/**16**/25
- Diffuse Rendering
  - At each point: dot-product of size 16
- Glossy Rendering
  - At each point: vector(16) * matrix (16*16)

Glossy Rendering Results

No Shadows/Inter

Shadows

Shadows+Inter

- Glossy object, 50K mesh
- Runs at 3.6 fps on 2.2Ghz P4, ATI Radeon 8500

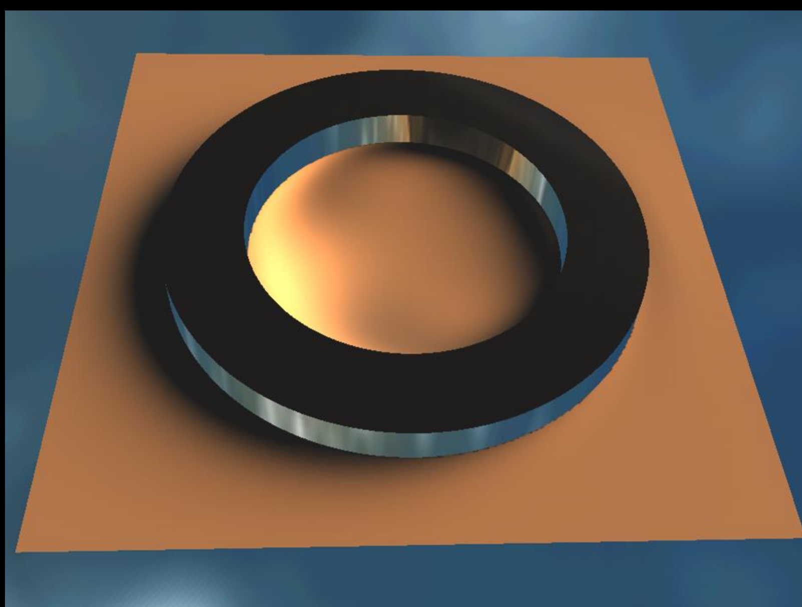
Interreflections and Caustics

interreflections

none

1 bounce

2 bounces



caustics

Transport Paths

$$LP$$

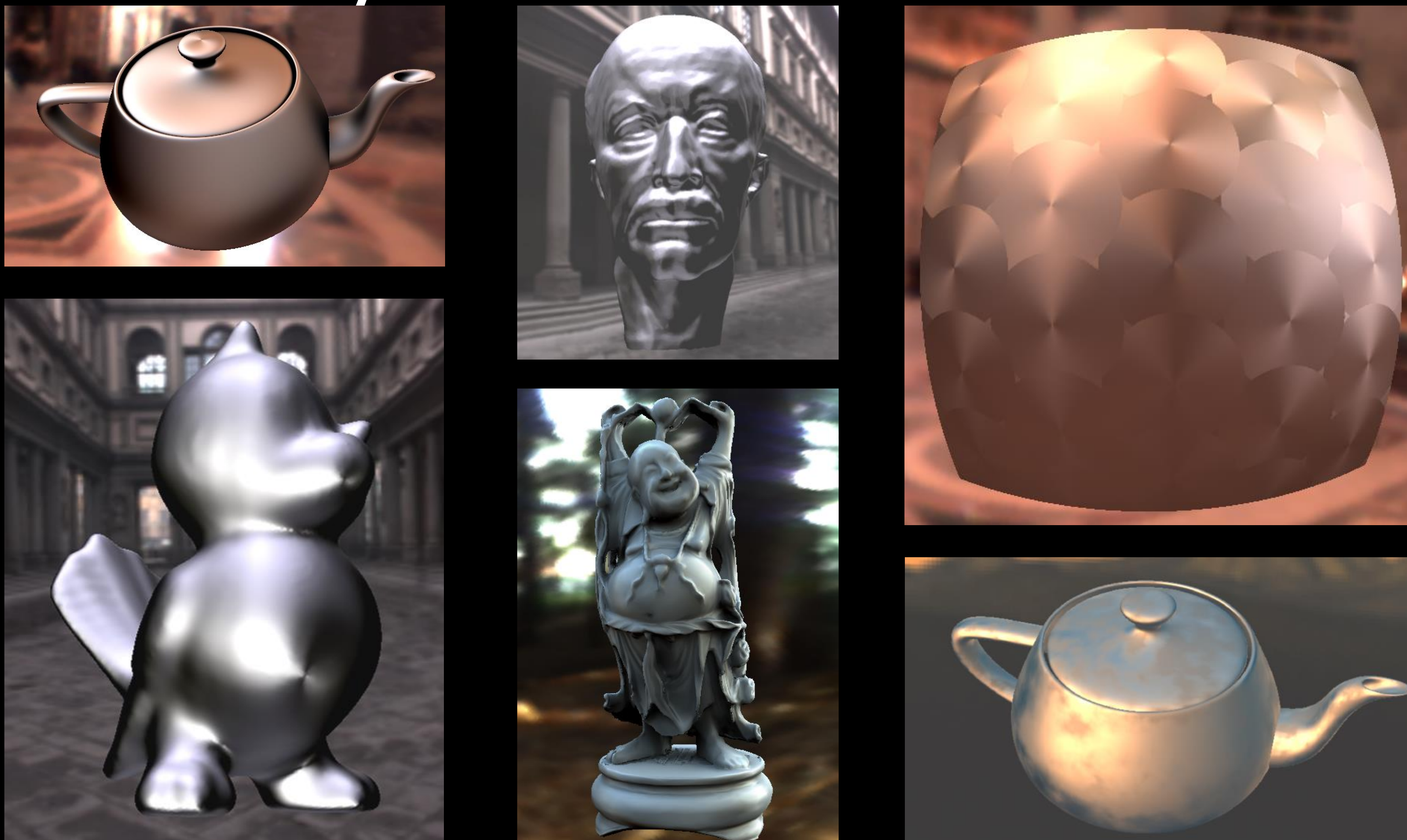
Runtime is independent of transport complexity

$$LGP$$

$$L(D|G)^* P$$

$$LS^*(S^*(D|G))^* P$$

Arbitrary BRDF Results



Anisotropic BRDFs **Other BRDFs** **Spatially Varying**

Results

Acquired Environments

Geometry: 50k vertex mesh

Summary of [Sloan 02]

- Approximate Lighting and light transport using basis functions (SH)
  - Lighting -> lighting coefficients
  - light transport -> coefficients / matrices
- Precompute and store light transport
- Rendering reduced to:
  - Diffuse: dot product
  - Glossy: vector matrix multiplication

Limitations [Sloan 02]

- Low-frequency
  - Due to the nature of SH
- Dynamic lighting, but static scene/material
  - Changing scene/material invalidates precomputed light transport
- Big precomputation data

Follow up works

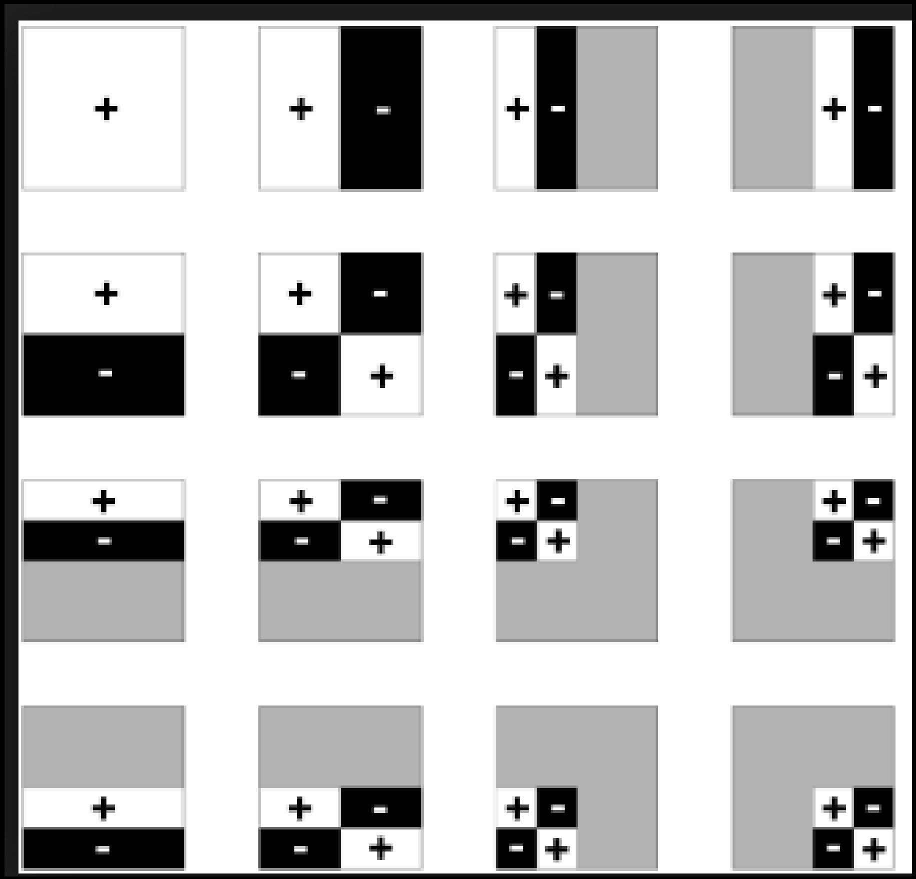
- More basis functions
- dot product => triple products
- Static scene => dynamic scene
- Fix material => dynamic material
- Other effects: translucent, hair, …
- Precomputation => analytic computation
- …

More basis functions

- Spherical Harmonics (SH)
- Wavelet
- Zonal Harmonics
- Spherical Gaussian (SG)
- Piecewise Constant

Wavelet [Ng 03]

- 2D Haar wavelet
- Projection:
  - Wavelet Transformation
  - Retain a small number of non-zero coefficients
- A non-linear approximation
- All-frequency representation



low frequency vs all frequency

Teapot in Grace Cathedral

Low frequency (SH)

All frequency (Wavelet)

Relighting as Matrix-Vector Multiply

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \mathrm{M} \\ P_N \end{bmatrix}$$

$$= \begin{bmatrix} T_{11} & T_{12} & \mathrm{L} & T_{1M} \\ T_{21} & T_{22} & \mathrm{L} & T_{2M} \\ T_{31} & T_{32} & \mathrm{L} & T_{3M} \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ T_{N1} & T_{N2} & \mathrm{L} & T_{NM} \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ \mathrm{M} \\ L_N \end{bmatrix}$$

Relighting as Matrix-Vector Multiply

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \mathrm{M} \\ P_N \end{bmatrix}$$

Output Image (Pixel Vector)

- Input Lighting (Cubemap Vector)

$$= \begin{bmatrix} T_{11} & T_{12} & \mathrm{L} & T_{1M} \\ T_{21} & T_{22} & \mathrm{L} & T_{2M} \\ T_{31} & T_{32} & \mathrm{L} & T_{3M} \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ T_{N1} & T_{N2} & \mathrm{L} & T_{NM} \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ \mathrm{M} \\ L_N \end{bmatrix}$$

Transport Matrix

Non-linear Wavelet Light Approximation

Wavelet Transform

Non-linear Wavelet Light Approximation

$$\begin{bmatrix} 0 \\ L_2 \\ 0 \\ 0 \\ 0 \\ L_6 \\ \mathrm{M} \\ 0 \end{bmatrix}$$

Non-linear Approximation

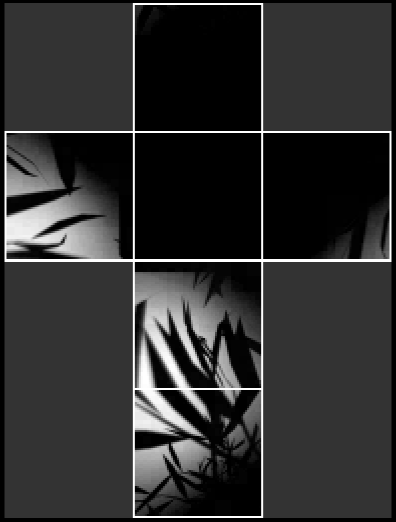
Retain 0.1% – 1% terms

Matrix Row Wavelet Encoding

$$
\begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & \mathrm{L} & T_{1M} \\
T_{21} & T_{22} & T_{23} & T_{24} & \mathrm{L} & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \mathrm{L} & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \mathrm{L} & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \mathrm{L} & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \mathrm{L} & T_{6M} \\
\mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & T_{7M} \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \mathrm{L} & T_{NM}
\end{bmatrix}
$$

Matrix Row Wavelet Encoding

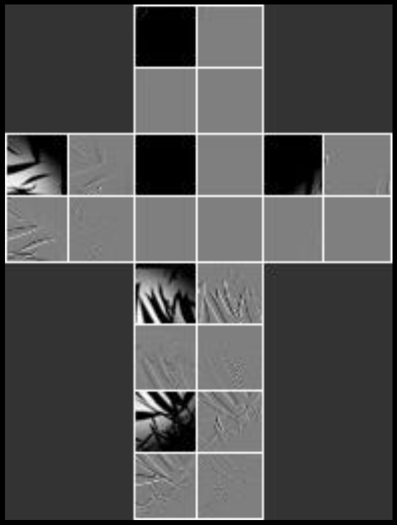
$$
\begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & \mathrm{L} & T_{1M} \\
T_{21} & T_{22} & T_{23} & T_{24} & \mathrm{L} & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \mathrm{L} & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \mathrm{L} & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \mathrm{L} & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \mathrm{L} & T_{6M} \\
\mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & T_{7M} \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \mathrm{L} & T_{NM}
\end{bmatrix}
$$



Extract Row

Matrix Row Wavelet Encoding

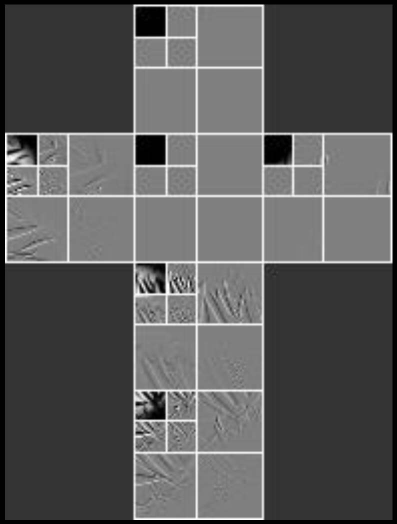
$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\ T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\ T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\ \vdots & \vdots & \vdots & \vdots & \ddots & T_{7M} \\ T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM} \end{bmatrix}$$



Wavelet Transform

Matrix Row Wavelet Encoding

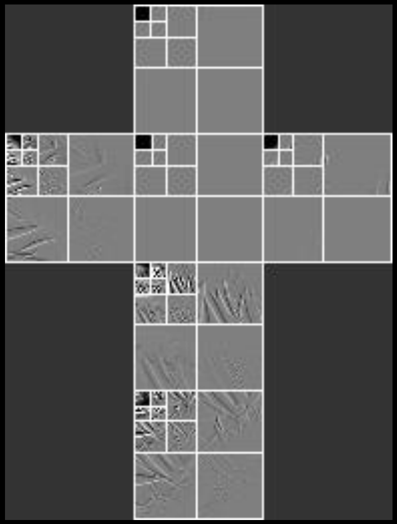
$$
\begin{bmatrix}
T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\
T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\
T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\
T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\
T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\
T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\
\vdots & \vdots & \vdots & \vdots & \ddots & T_{7M} \\
T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM}
\end{bmatrix}
$$



Wavelet Transform

Matrix Row Wavelet Encoding

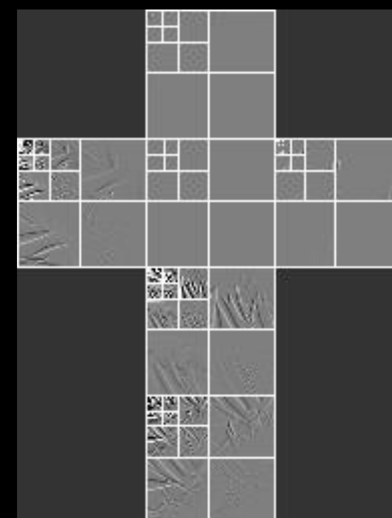
$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} & \mathrm{L} & T_{1M} \\ T_{21} & T_{22} & T_{23} & T_{24} & \mathrm{L} & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \mathrm{L} & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \mathrm{L} & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \mathrm{L} & T_{5M} \\ T_{61} & T_{62} & T_{63} & T_{64} & \mathrm{L} & T_{6M} \\ \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & T_{7M} \\ T_{N1} & T_{N2} & T_{N3} & T_{N4} & \mathrm{L} & T_{NM} \end{bmatrix}$$



Wavelet Transform

Matrix Row Wavelet Encoding

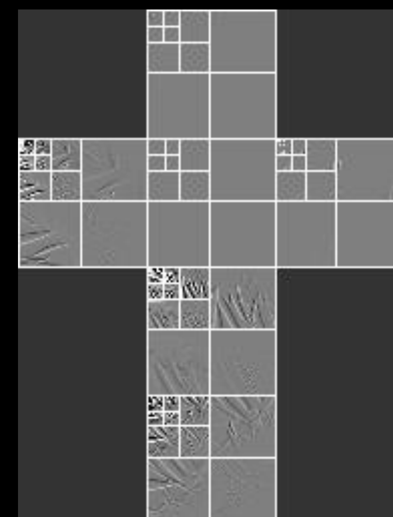
$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} & \mathrm{L} & T_{1M} \\ T_{21} & T_{22} & T_{23} & T_{24} & \mathrm{L} & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \mathrm{L} & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \mathrm{L} & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \mathrm{L} & T_{5M} \\ T_{61} & T_{62} & T_{63} & T_{64} & \mathrm{L} & T_{6M} \\ \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & T_{7M} \\ T_{N1} & T_{N2} & T_{N3} & T_{N4} & \mathrm{L} & T_{NM} \end{bmatrix}$$



Wavelet Transform

Matrix Row Wavelet Encoding

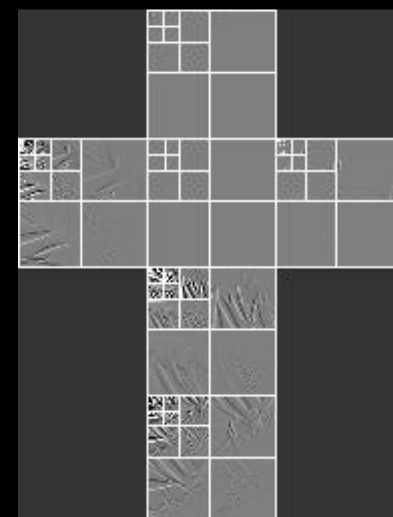
$$\begin{bmatrix} T_{11}' & 0 & 0 & T_{14}' & \mathrm{L} & 0 \\ T_{21} & T_{22} & T_{23} & T_{24} & \mathrm{L} & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \mathrm{L} & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \mathrm{L} & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \mathrm{L} & T_{5M} \\ T_{61} & T_{62} & T_{63} & T_{64} & \mathrm{L} & T_{6M} \\ \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{O} & T_{7M} \\ T_{N1} & T_{N2} & T_{N3} & T_{N4} & \mathrm{L} & T_{NM} \end{bmatrix}$$



Store Back in Matrix

Matrix Row Wavelet Encoding

$$\begin{bmatrix} T_{11}' & 0 & 0 & T_{14}' & \cdots & 0 \\ T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\ T_{61} & T_{62} & T_{63} & T_{64} & \cdots & T_{6M} \\ \vdots & \vdots & \vdots & \vdots & \ddots & T_{7M} \\ T_{N1} & T_{N2} & T_{N3} & T_{N4} & \cdots & T_{NM} \end{bmatrix}$$



Only 3% – 30% are non-zero

Why Non-linear Approximation?

- Linear
  - Use a fixed set of approximating functions
  - Precomputed radiance transfer uses 25 - 100 of the lowest frequency spherical harmonics
- Non-linear
  - Use a dynamic set of approximating functions (*depends on each frame's lighting*)
  - In our case: choose 10's - 100's from a basis of 24,576 wavelets

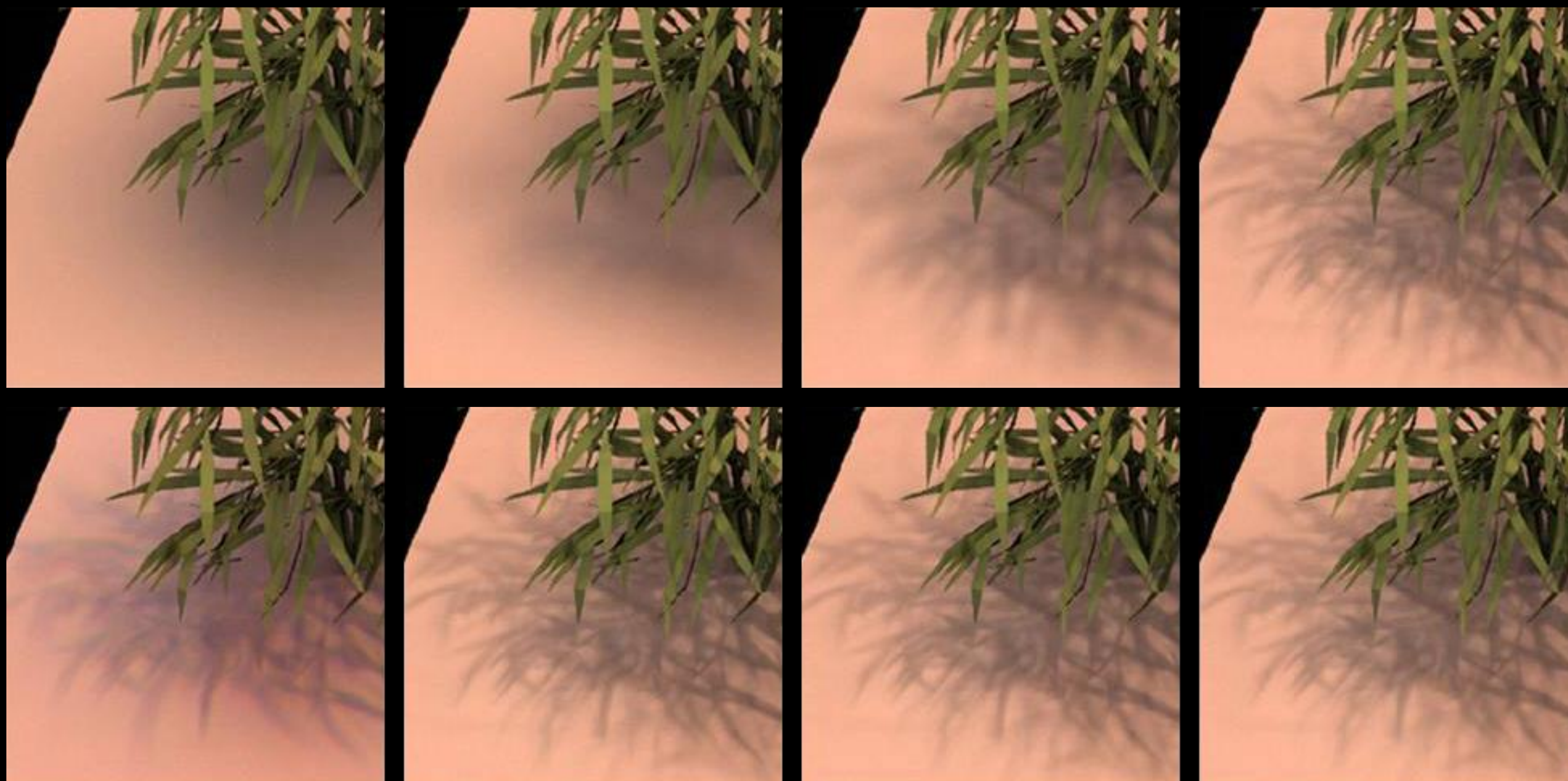
Overall Rendering Algorithm

- Pre-compute (per scene)
  - Compute matrix in pixel basis
  - Wavelet transform rows
  - Quantize, store
- Interactive Relighting (each frame)
  - Wavelet transform lighting
  - Prioritize and retain *N* wavelet coefficients
  - Perform sparse-matrix vector multiplication

Output Image Comparison

SH

wavelet



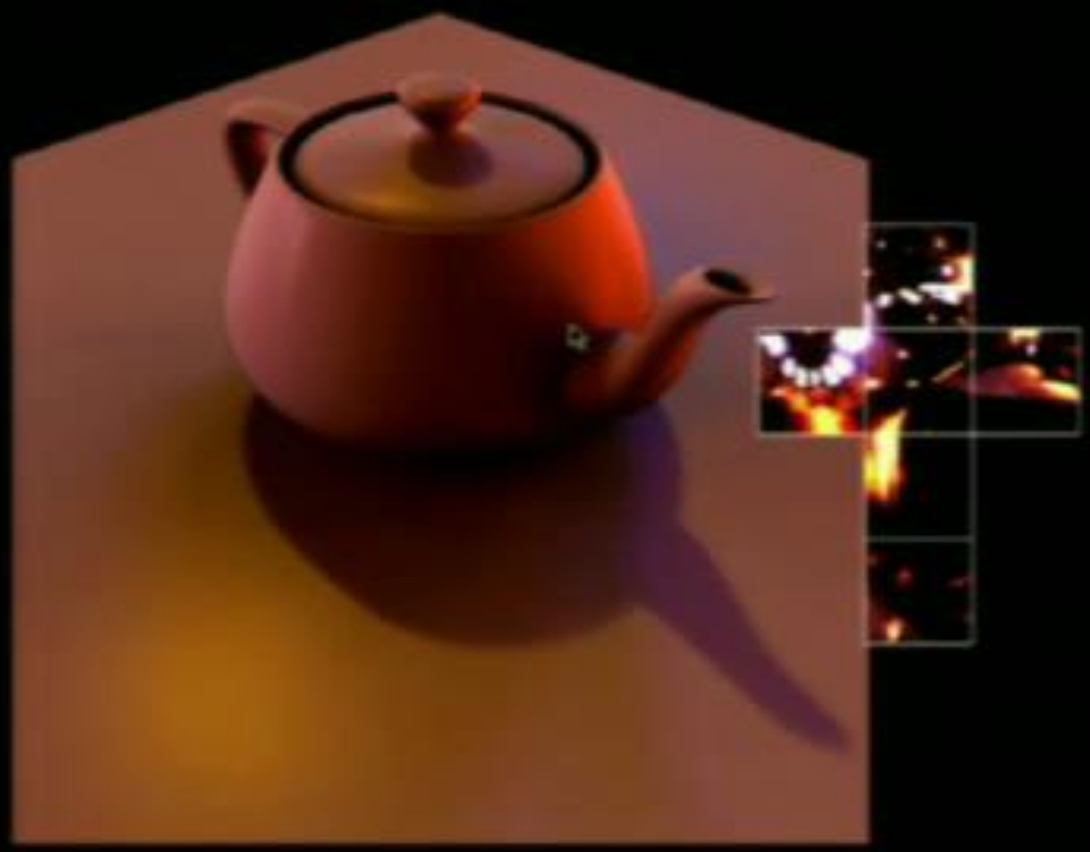
25 200 2,000 20,000

Limitation

- Wavelet: not rotational invariant
  - Re-projection at each frame
  - Results in flicking

- Only support dot-product operator
  - Limited to diffuse or fix-view glossy

Results



Zonal Harmonics [Sloan 05]

- circularly symmetric functions
- Subset of SH basis (m=0)

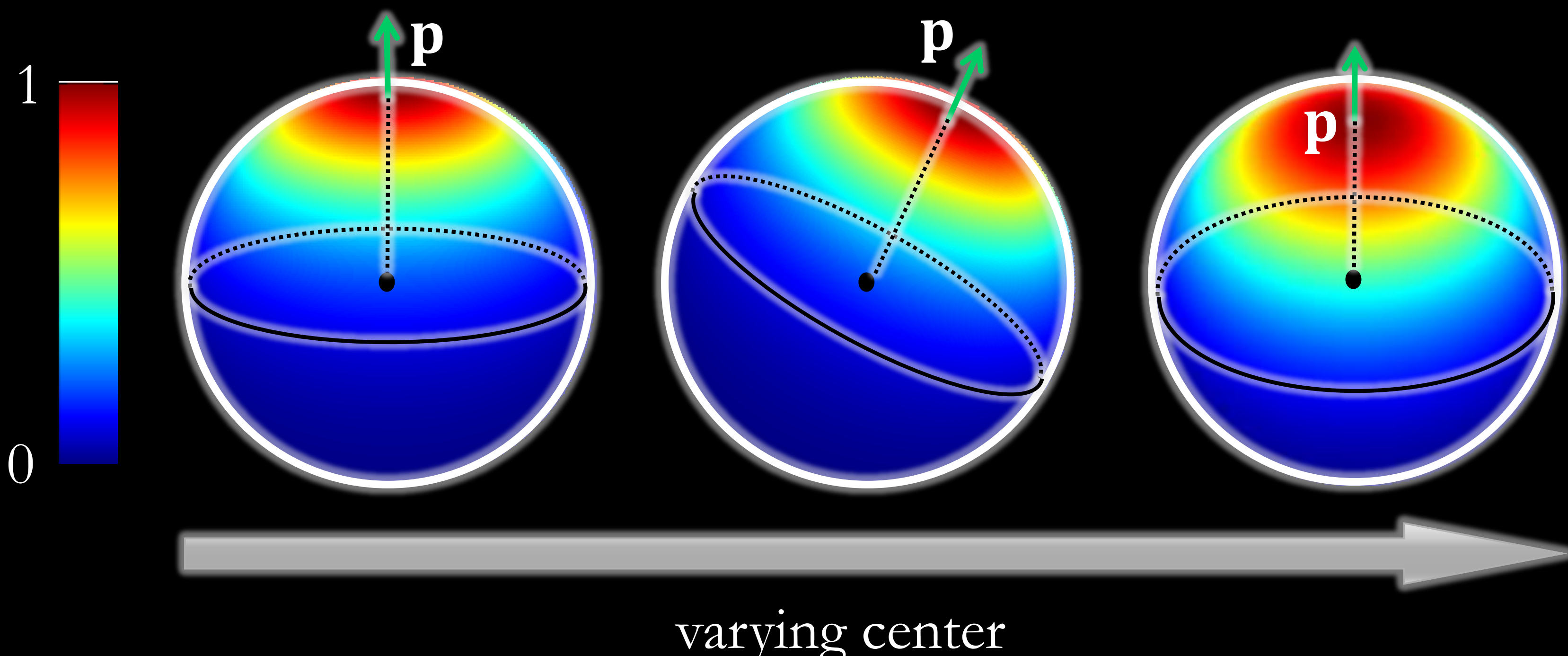
- Low-frequency
- Rotational invariant
- Much more faster in rotation than SH

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

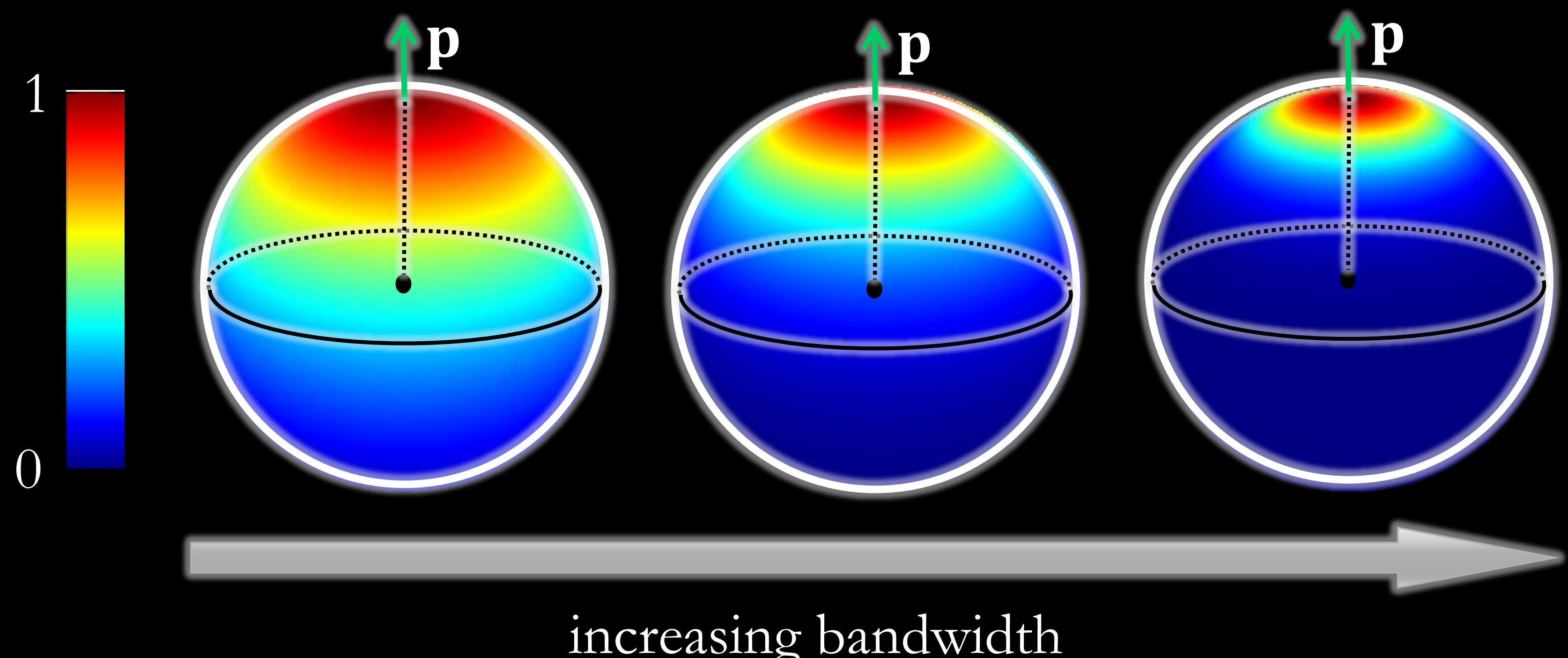


Spherical Gaussian (SG) [Tsai 06]

- SGs (or Spherical Radial Basis Functions, SRBFs)

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

center bandwidth



Mathematical Properties of SGs

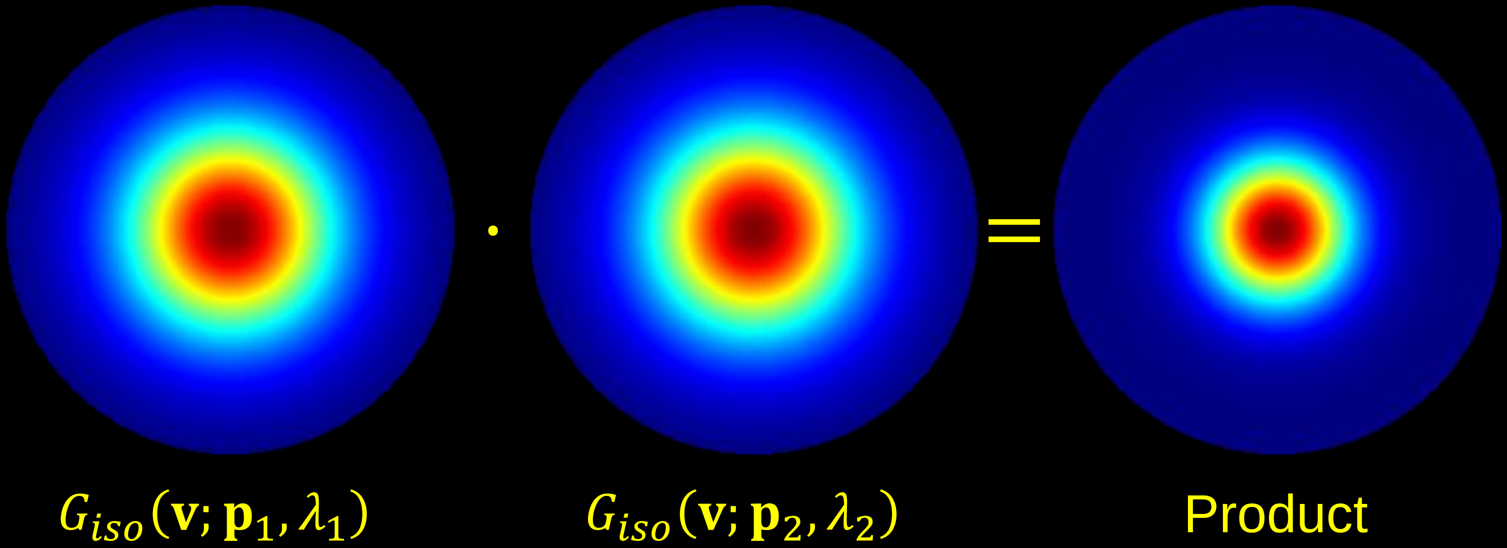
- Closed-form integral
  - The integral of an SG is closed-form

$$\int_{\Omega} G(\mathbf{v};\mathbf{p},\lambda)\mathrm{d}\mathbf{v} = \frac{2\pi}{\lambda}\left(1 - e^{-2\lambda}\right)$$

Mathematical Properties of SGs

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

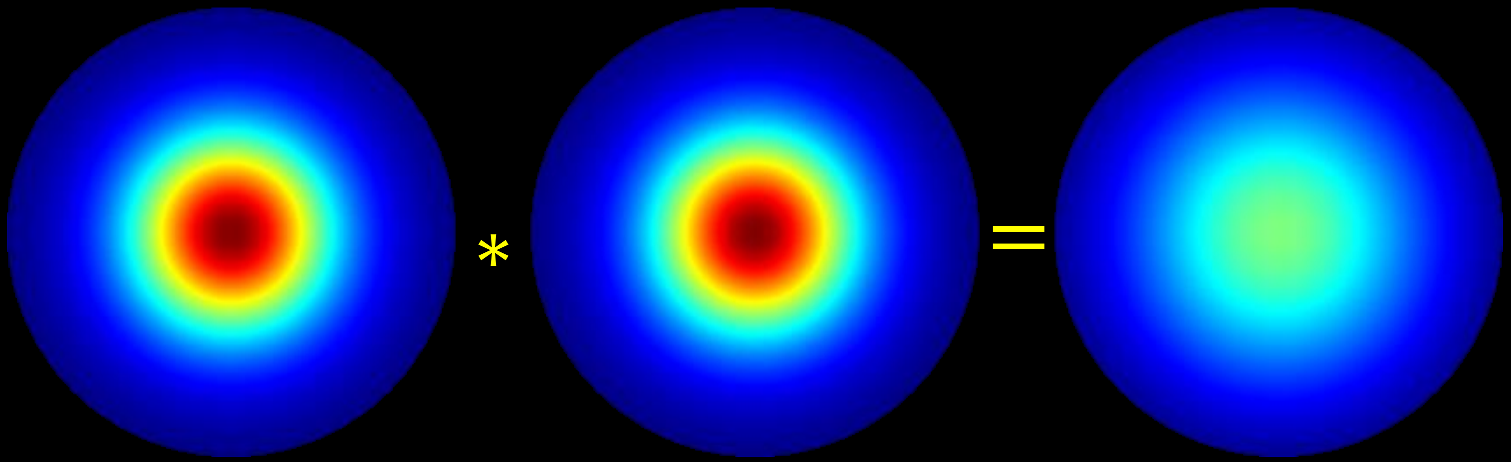
$$\mathrm{G}(\mathbf{v};\mathbf{p}_1,\lambda_1)\cdot\mathrm{G}(\mathbf{v};\mathbf{p}_2,\lambda_2) = c\mathrm{G}\left(\mathbf{v};\frac{\lambda_1\mathbf{p}_1+\lambda_1\mathbf{p}_2}{|\lambda_1\mathbf{p}_1+\lambda_1\mathbf{p}_2|},|\lambda_1\mathbf{p}_1+\lambda_1\mathbf{p}_2|\right)$$



Mathematical Properties of SGs

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

$$\int_{\Omega} G(\mathbf{v};\mathbf{p}_1,\lambda_1)\cdot G(\mathbf{v};\mathbf{p}_2,\lambda_2)\mathrm{d}\mathbf{v} \approx c_3 G\left(\mathbf{p}_1;\mathbf{p}_2,\frac{\lambda_1\lambda_2}{\lambda_1+\lambda_2}\right)$$



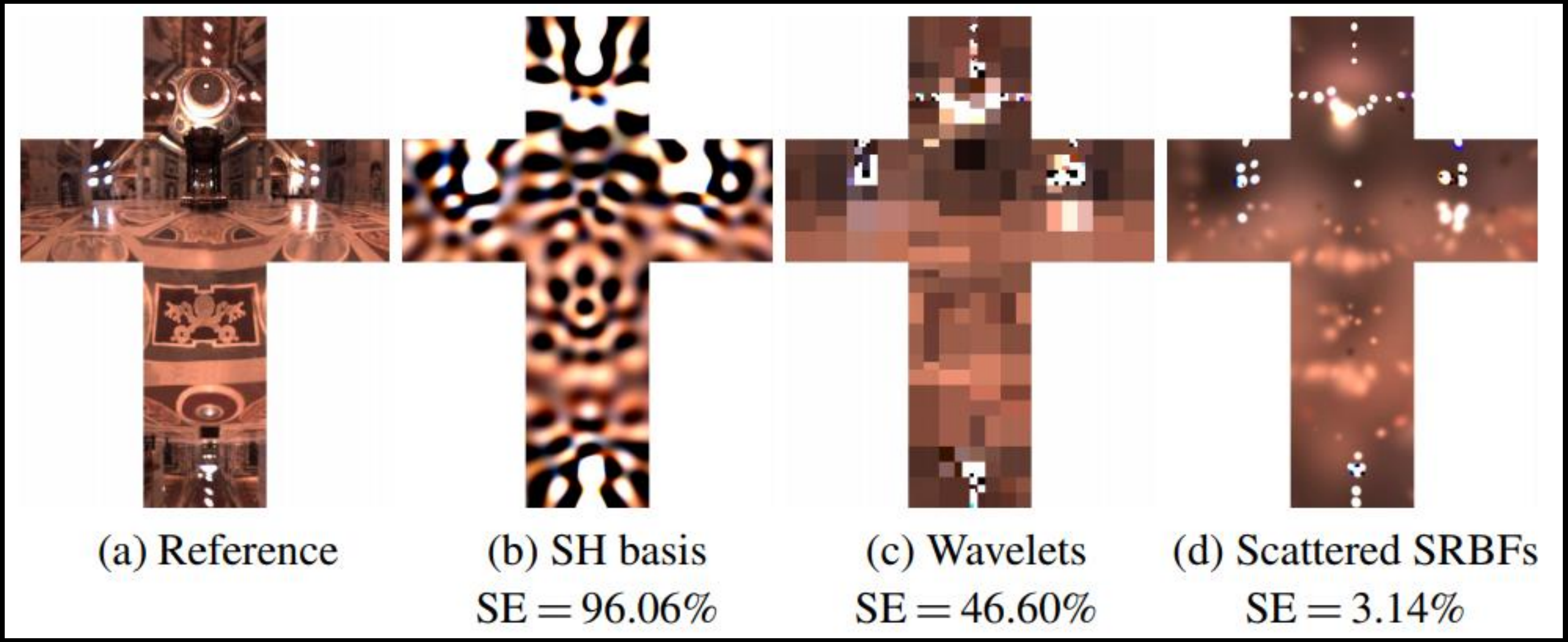
$G_{iso}(\mathbf{v};\mathbf{p}_1,\lambda_1)$ $G_{iso}(\mathbf{v};\mathbf{p}_2,\lambda_2)$ Convolution

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



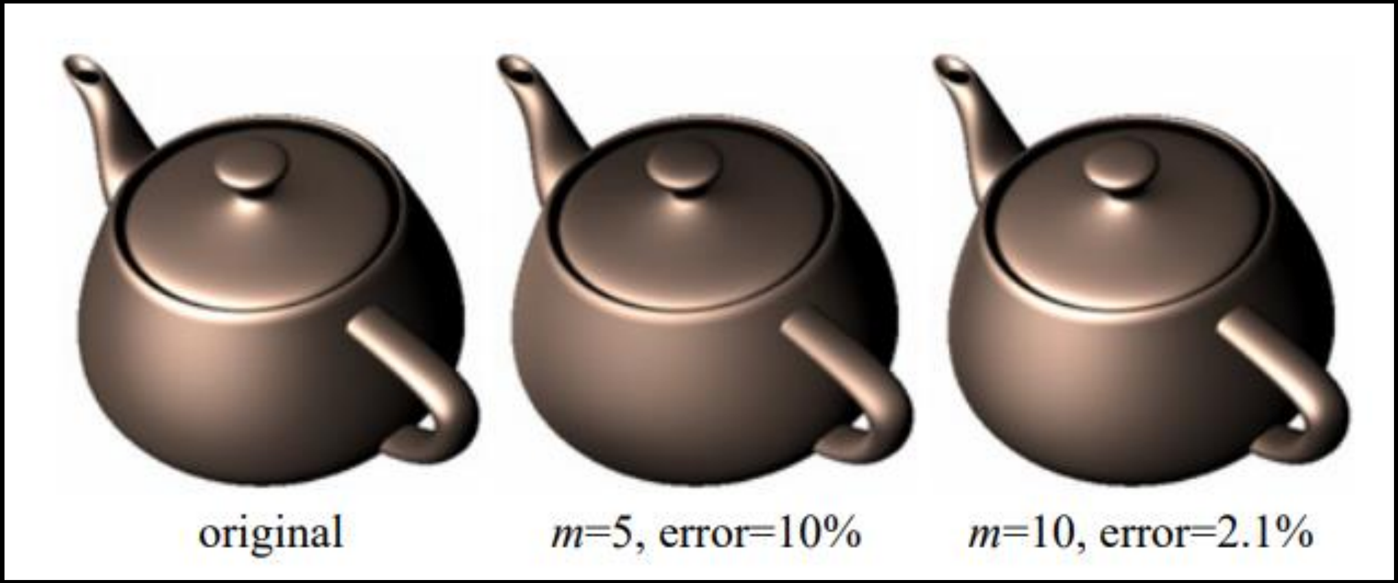
- non-linear process: iterative L-BFGS-B solver (slow)

$$L(\mathbf{i}) \approx \sum l_i G(\mathbf{i}; \mathbf{p}_{\mathrm{i}}, \lambda_{\mathrm{i}})$$

BRDF Factorization [Wang 03, Liu 03]

- Precompute the factorization

$$\rho(\mathbf{i},\mathbf{o}) \approx \sum_{\mathrm{m}} f_{\mathrm{m}}(\mathbf{i}) \cdot g_{\mathrm{m}}(\mathbf{o})$$



Overall Rendering Algorithm

- Derivation: Factorizing BRDF

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

$$\Downarrow$$

$$L(\mathbf{o}) = \int_{\Omega} L(\mathbf{i})V(\mathbf{i})\left(\sum_{\mathrm{m}} f_{\mathrm{m}}(\mathbf{i})\cdot g_{\mathrm{m}}(\mathbf{o})\right)\max(0,\boldsymbol{n}\cdot\mathbf{i})\,\mathrm{d}\mathbf{i}$$

$$\Downarrow$$

$$L(\mathbf{o}) = \sum_{\mathrm{m}} g_{\mathrm{m}}(\mathbf{o})\int_{\Omega} \underline{L(\mathbf{i})}\,\underline{\overbrace{V(\mathbf{i})f_{\mathrm{m}}(\mathbf{i})\max(0,\boldsymbol{n}\cdot\mathbf{i})}^{T(\mathbf{i})}\,\mathrm{d}\mathbf{i}}$$

Both represented using SGs

Overall Rendering Algorithm

- Derivation: projection to SGs

$$L(\mathbf{o}) = \sum_{\mathrm{m}} g_{\mathrm{m}}(\mathbf{o}) \int_{\Omega} \underline{L(\mathbf{i})}\,\underline{T(\mathbf{i})}\ \mathrm{d}\mathbf{i}$$

$L(\mathbf{i}) \approx \sum l_i G_i(\mathbf{i})$

non-linear approx.

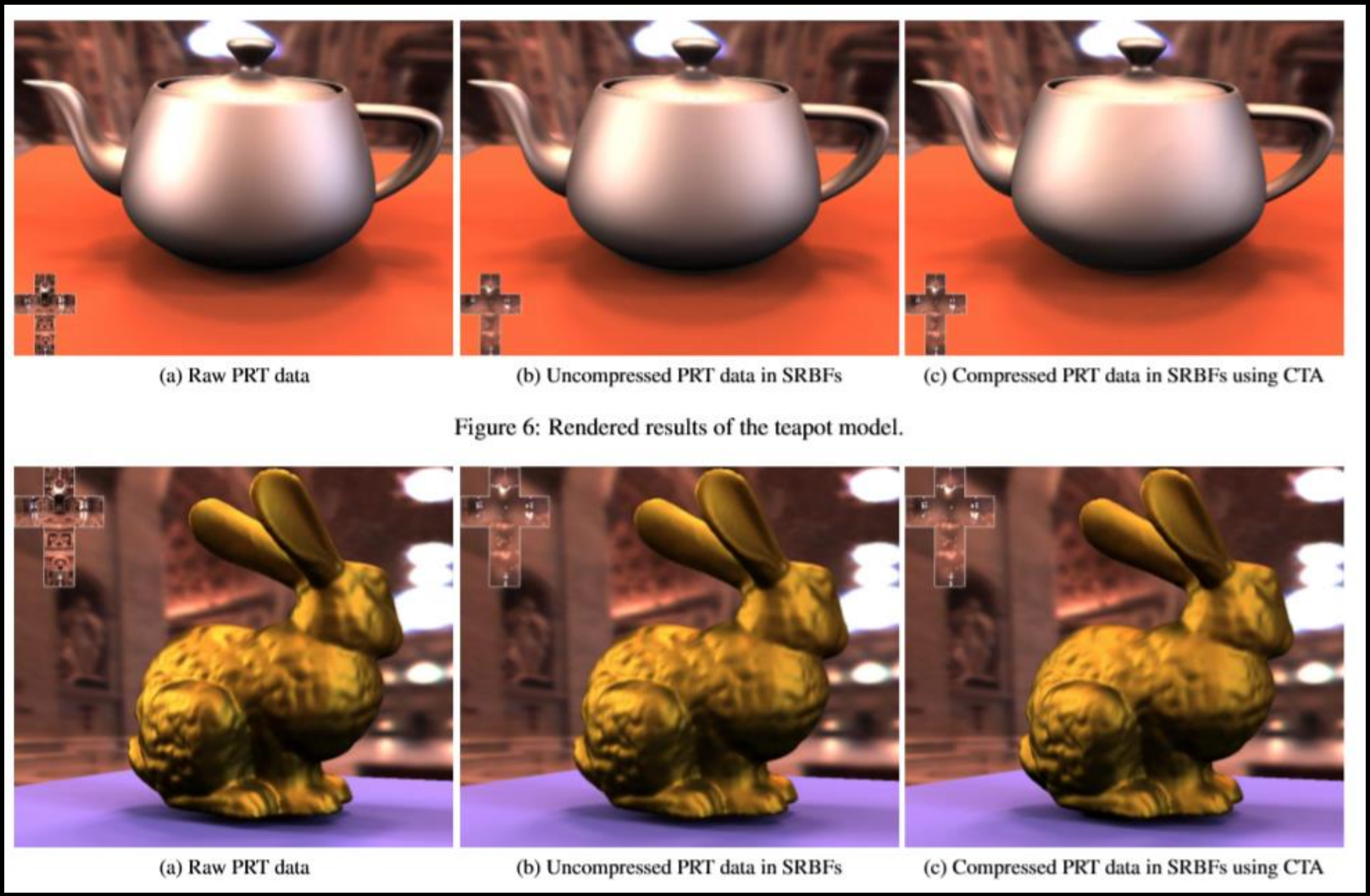
$T(\mathbf{i}) \approx \sum t_j G_j(\mathbf{i})$

pre. scattered approx.

$$L(\mathbf{o}) = \sum_{\mathrm{m}} g_{\mathrm{m}}(\mathbf{o}) \sum_{i,j} l_i t_j \underbrace{\int_{\Omega} G_i(\mathbf{i}) G_j(\mathbf{i})\ \mathrm{d}\mathbf{i}}_{\text{analytic solution}}$$

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

Results



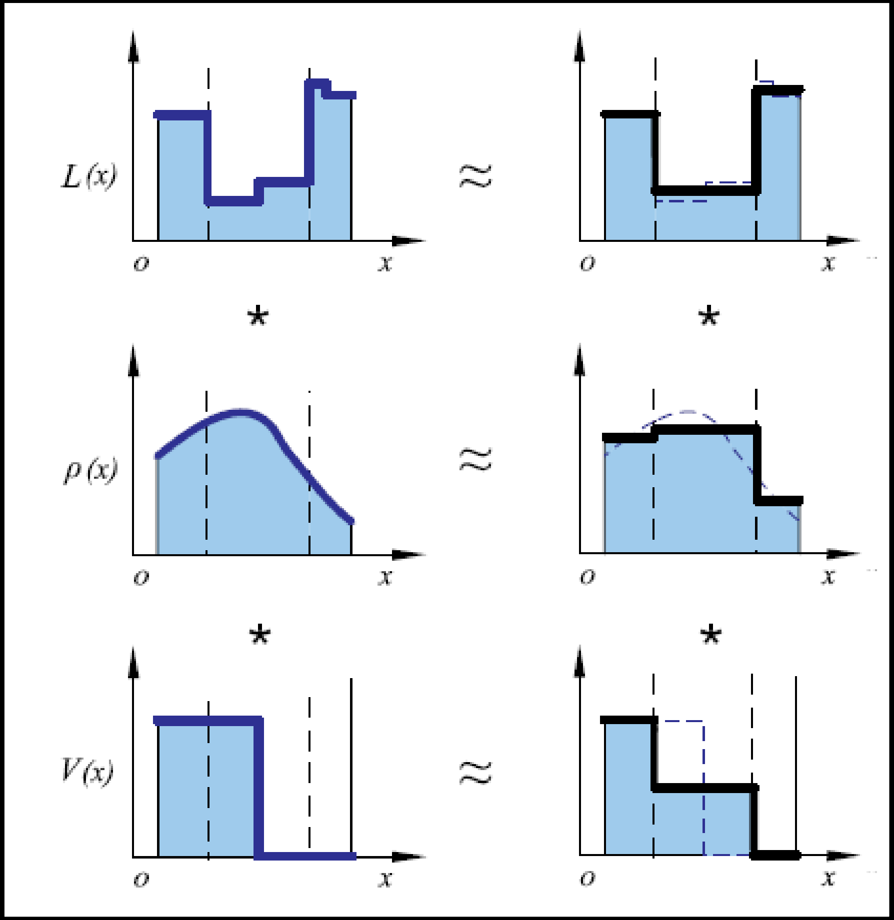
(a) Raw PRT data (b) Uncompressed PRT data in SRBFs (c) Compressed PRT data in SRBFs using CTA

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

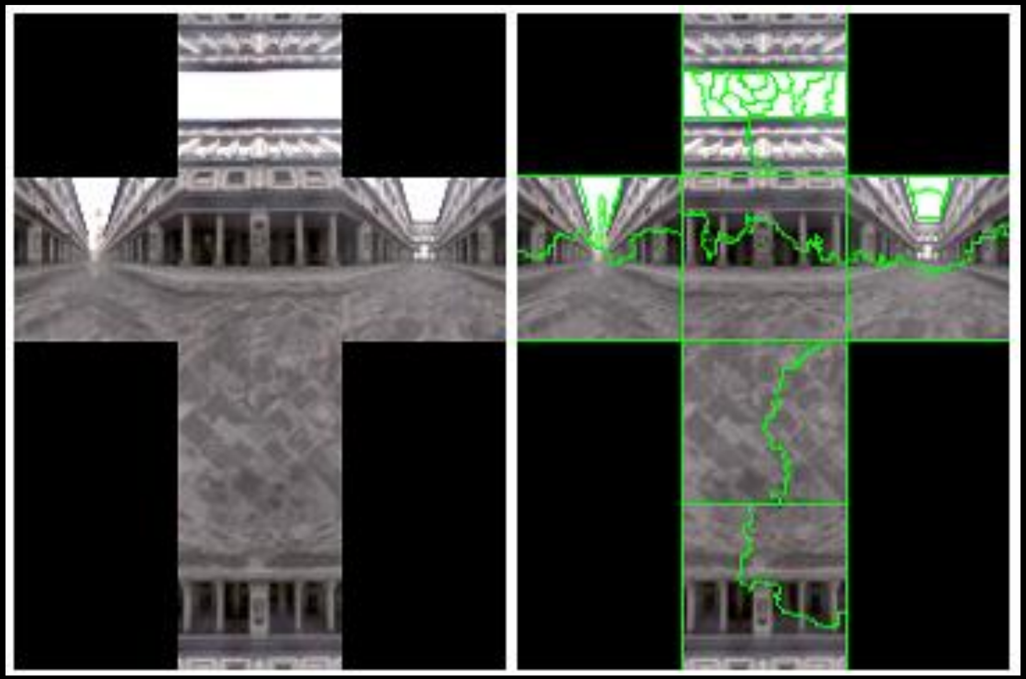
Piecewise Constant [Xu 08]

- 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



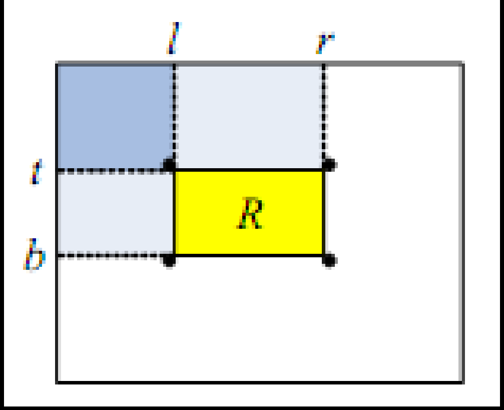
Piecewise Constant [Xu 08]

- Light Projection
  - Bottom-up optimization

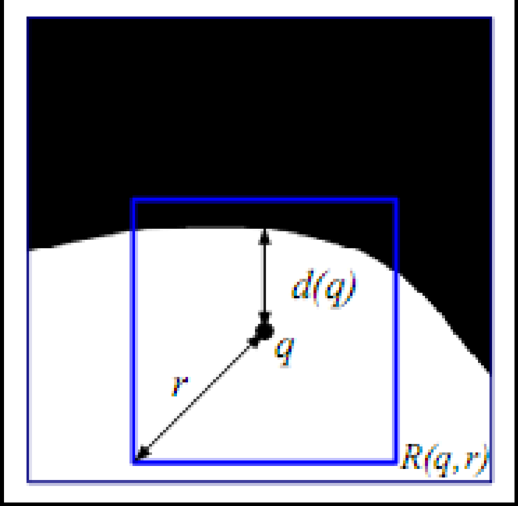


Piecewise Constant [Xu 08]

- Projection of visibility and BRDFs
  - BRDF
    - using summed area table



  - Visibility
    - Using visibility distance table



Results

Comparison of Basis Functions

	SH	Wavelet	SG	SPCBF
Orthogonal	√	√	×	√
All-frequency	×	√	√	√
Rotation invariant	√	×	√	√
Multiple product	√	√	√ ?	√
Compact Representation	√	√	√	×

Triple Product

- Original PRT: light * light transport

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

lighting light transport

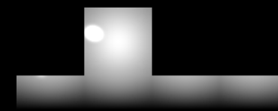
Not Flexible

- Triple Product

lighting visibility BRDF $\times$ cosine

- … Multiple Product

Wavelet Triple Product [Ng 04]

$\int_{S^2}$  $*$  $*$  $d\omega$

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

Wavelet Triple Product [Ng 04]

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

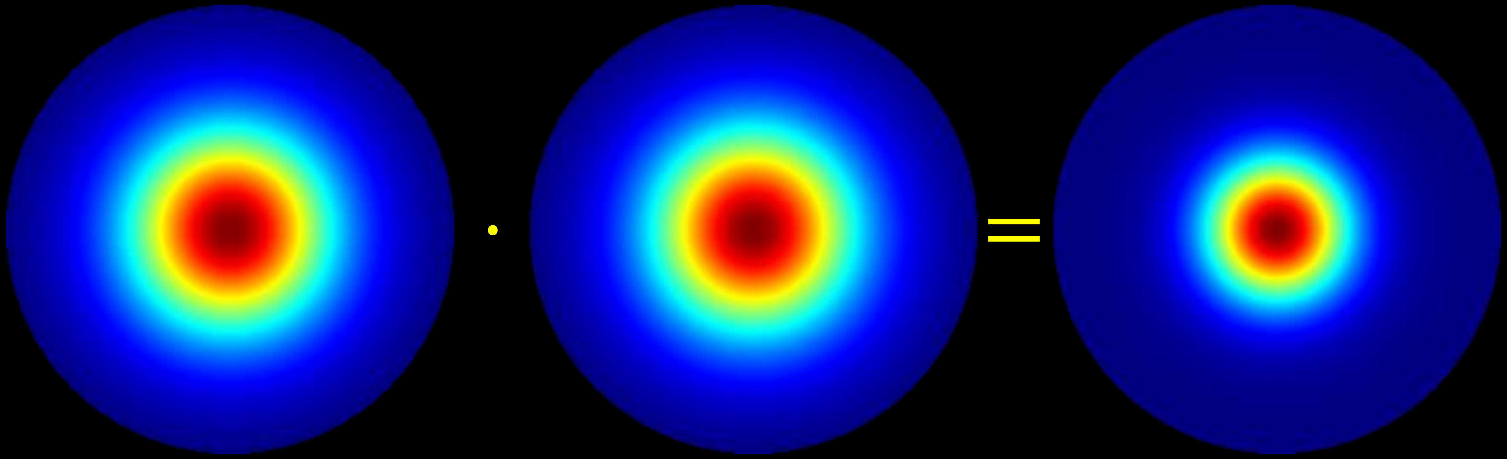
Basis Choice	Number Non-Zero C_{ijk}
General (e.g. PCA)	$O(N^3)$
Pixels	$O(N)$
Fourier Series	$O(N^2)$
SH	$O(N^{5/2})$
Haar Wavelets	$O(N \log N)$

Wavelet Triple Product [Ng 04]

SG Triple Product

- Analytic Computation
  - The product of two SGs is also an SG

$$\mathrm{G}(\mathbf{v};\mathbf{p}_1,\lambda_1)\cdot\mathrm{G}(\mathbf{v};\mathbf{p}_2,\lambda_2)=c\mathrm{G}\left(\mathbf{v};\frac{\lambda_1\mathbf{p}_1+\lambda_1\mathbf{p}_2}{|\lambda_1\mathbf{p}_1+\lambda_1\mathbf{p}_2|},|\lambda_1\mathbf{p}_1+\lambda_1\mathbf{p}_2|\right)$$



$G(\mathbf{v};\mathbf{p}_1,\lambda_1)$ $G(\mathbf{v};\mathbf{p}_2,\lambda_2)$ Product

Could be easily extended to multiple product

SH Triple Product

- Precompute all triple products of SH basis

$$C_{ijk} = \int_\Omega B_i(\mathbf{i})B_j(\mathbf{i})B_k(\mathbf{i})\mathrm{d}\mathbf{i}$$

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

Original space — SH space

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

$$T(\mathbf{i}) \approx \sum t_j B_j(\mathbf{i})$$

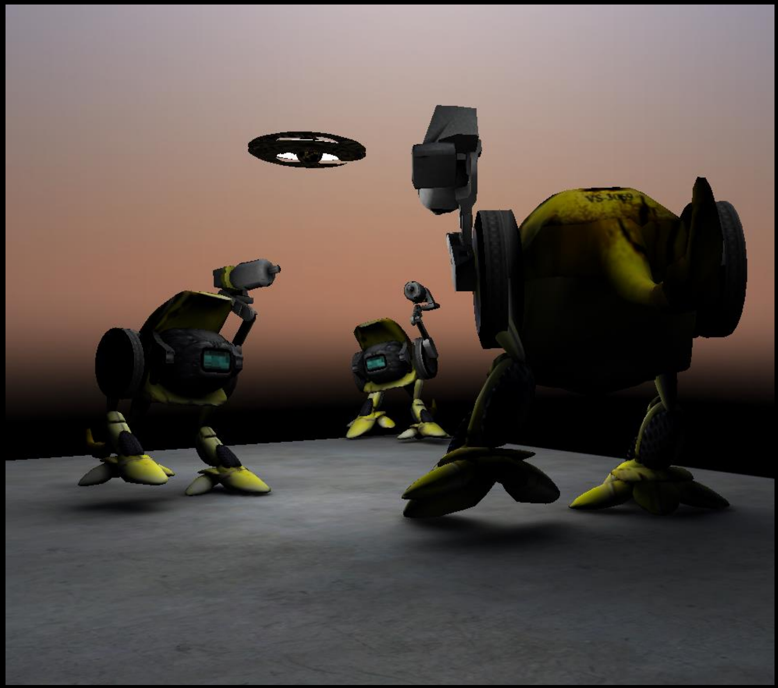
$$L(\mathbf{i}) \cdot T(\mathbf{i}) \approx \sum lt_k B_k(\mathbf{i})$$

$$lt_k = \sum_{i,j} l_i t_j C_{ijk}$$

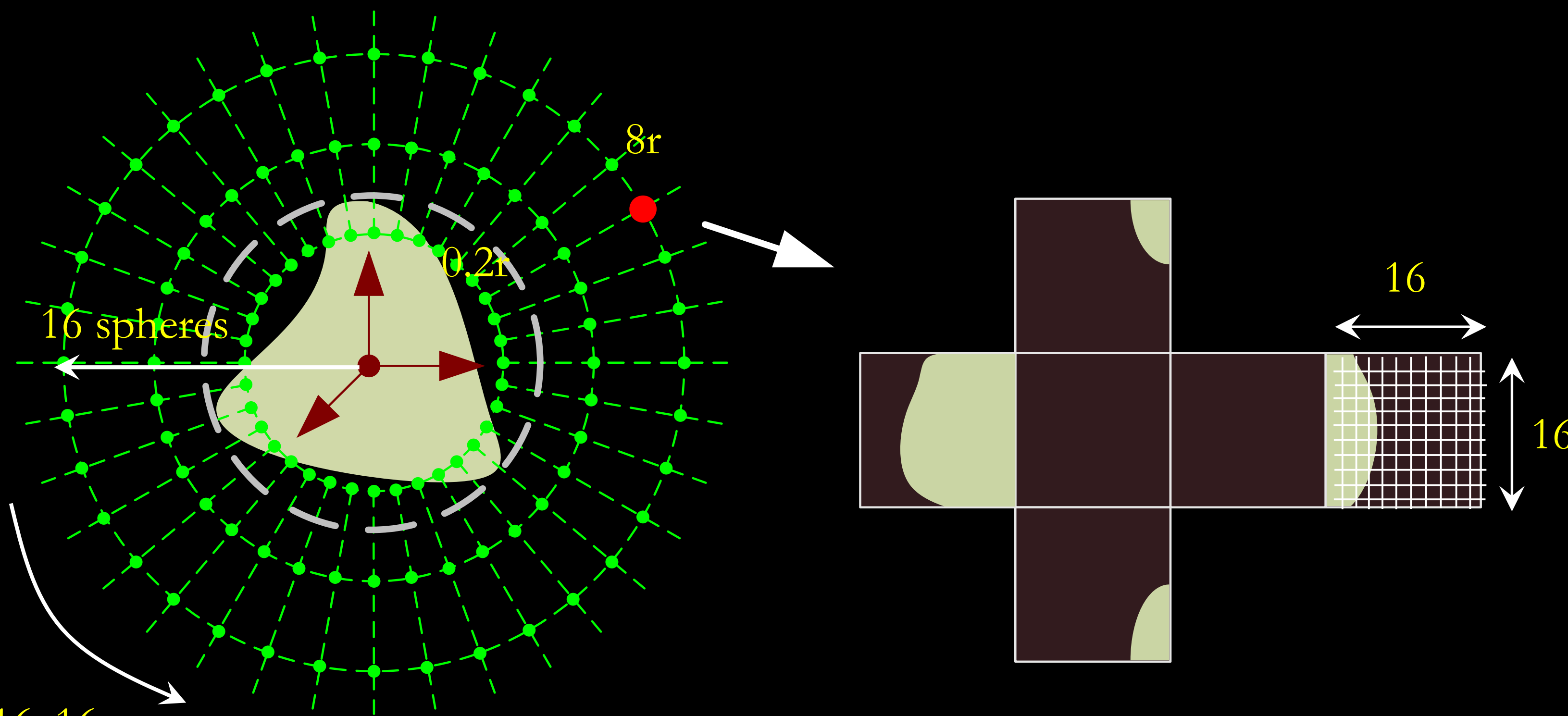
- Could be easily extended to multiple product

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



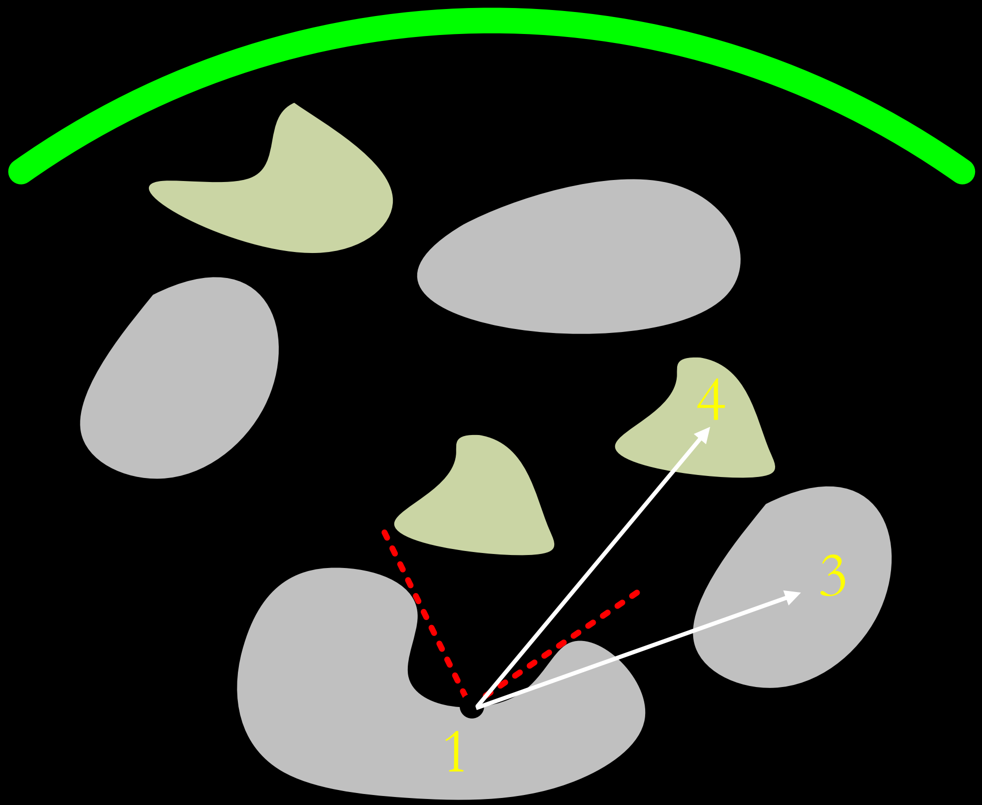
Sampling & Compression (Low Frequency)



6x16x16
samples
per sphere

Spherical Harmonics
OOFs: 384 KB
SRFs: 1 MB

Rendering: Products



BRDF

***

OOF-1

***

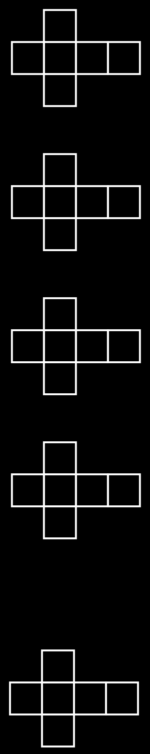
OOF-3

**

SRF-4

=

Reflected radiance



Results

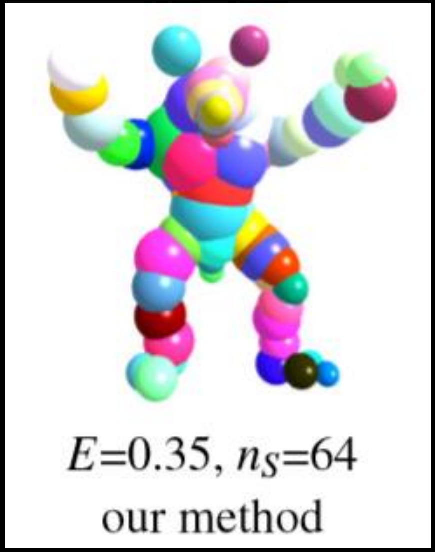
SH Exponential [Ren 06]

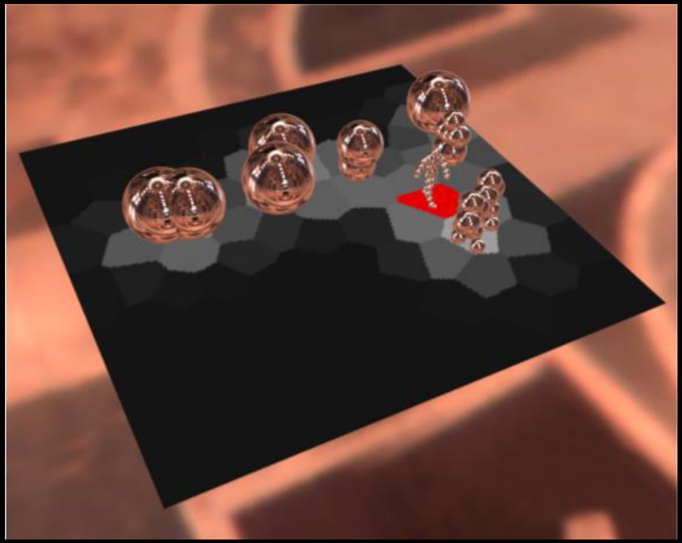
- Shadow Field
  - Rigid objects
  - Computation of SH multiple product is still costly

- 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





Rendering Computation

- Multiple Product (a very big number)

Light * Self_Vis * Occlusion1 * Occlusion2 * …. * OcclusionN * BRDF ***

- Approach

$$\boldsymbol{f_1} * \boldsymbol{f_2} * \cdots * \boldsymbol{f_n}$$

$$\exp(\log \boldsymbol{f_1} + \log \boldsymbol{f_2} + \cdots + \log \boldsymbol{f_n})$$

  - 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

$$L(x_o, \mathbf{o}) = \int_A \int_\Omega L(x_i, \mathbf{i}) S(x_i, \mathbf{i}; x_o, \mathbf{o}) \max(0, \boldsymbol{n} \cdot \mathbf{i}) \, \mathrm{d}\mathbf{i}$$

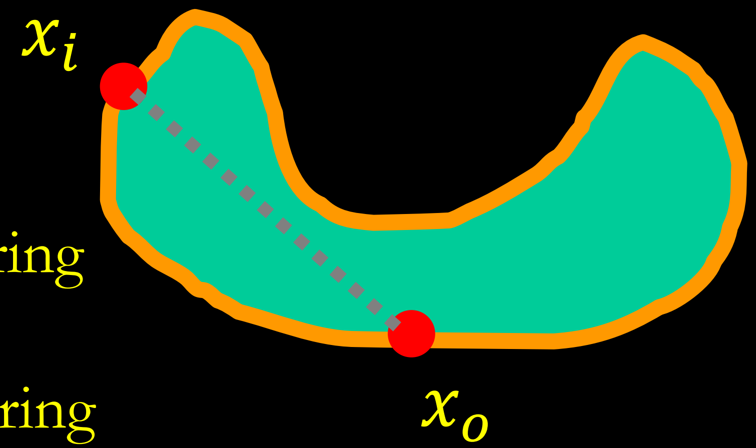
BSSRDF

||

factorized single scattering

\+

diffuse multiple scattering



- Precompute the transport for multiple scattering and single scattering separately

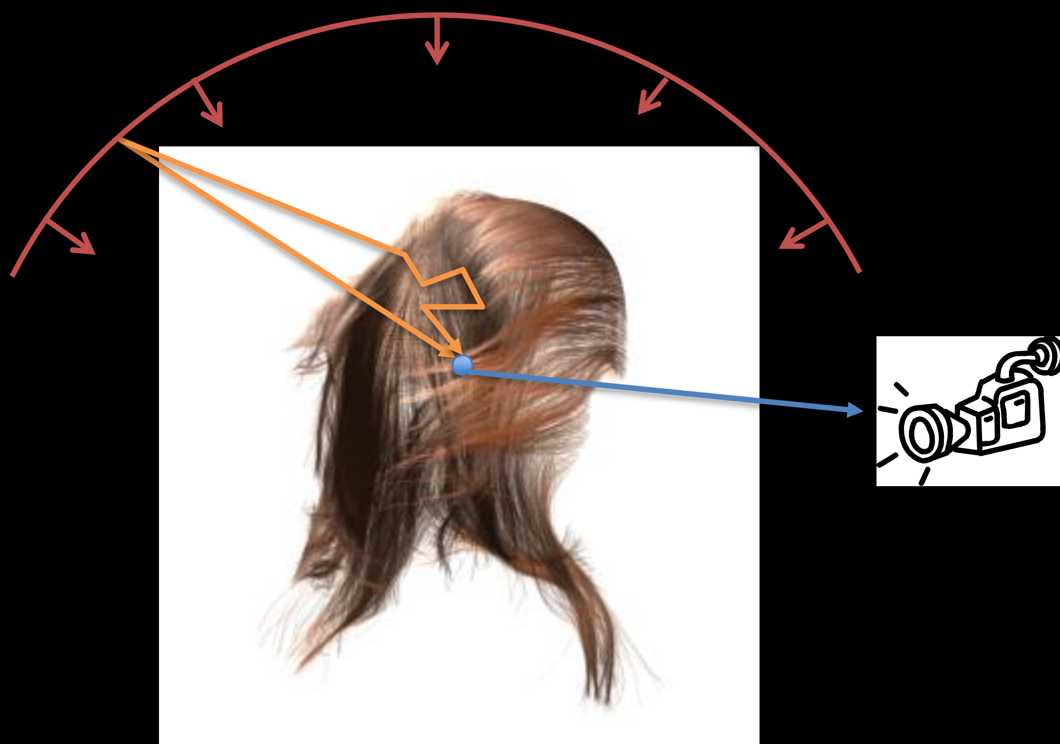
Results

- Dynamic environment lighting
- Fix: geometry + materials
- Real-time

Multiple Scattering | Single Scattering | Multiple and Single Scattering | Diffuse BRDF

Hair Rendering [Ren 10]

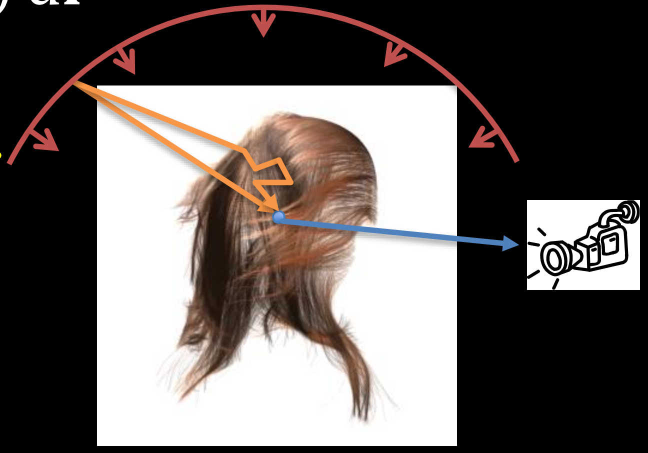
- Extend PRT to handle hair rendering
  - Support environment lighting



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

Single Scattering Computation

$$L(\mathbf{o}) = D \int_{\Omega} L(\mathbf{i}) \underbrace{T(\mathbf{i})}_{\text{self shadow}} \underbrace{S(\mathbf{i}, \mathbf{o})}_{\text{hair scattering func.}} \max(0, \boldsymbol{n} \cdot \mathbf{i}) \, \mathrm{d}\mathbf{i}$$



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

$$L(\mathbf{o}) = D \sum_{j=1}^{N} L_j \tilde{T} \underbrace{\int_{\Omega} G_j(\mathbf{i}) S(\mathbf{i}, \mathbf{o}) \max(0, \boldsymbol{n} \cdot \mathbf{i}) \, \mathrm{d}\mathbf{i}}$$

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

$$L(\mathbf{o}) = \int_{\Omega} \rlap{$\displaystyle\underbrace{\phantom{L(\mathbf{i})V(\mathbf{i})\max(0,\boldsymbol{n}\cdot\mathbf{i})}}_{\text{material transport}}$} \overset{\text{lighting}}{\underline{L(\mathbf{i})}} \overbrace{V(\mathbf{i})\max(0,\boldsymbol{n}\cdot\mathbf{i})\,\underset{\text{material}}{\underline{\rho(\mathbf{i},\mathbf{o})}}}^{\text{light transport}} \mathrm{d}\mathbf{i}$$

- PRT based BRDF editing
  - dynamic material + precompute material transport
  - Fix: lighting + geometry + viewpoint

BRDF Editing [Ben-Artzi 06]

- Approach: parameterize BRDF as 1D curve

$$\rho(\mathbf{i},\mathbf{o}) = \underbrace{\rho_q(\mathbf{i},\mathbf{o})}_{\text{quotient term}}\underbrace{f\big(\gamma(\mathbf{i},\mathbf{o})\big)}_{\text{1D curve}}$$

$$f(\gamma) \approx \sum c_j\, \underbrace{b_j(\gamma)}_{\text{wavelet basis}}$$

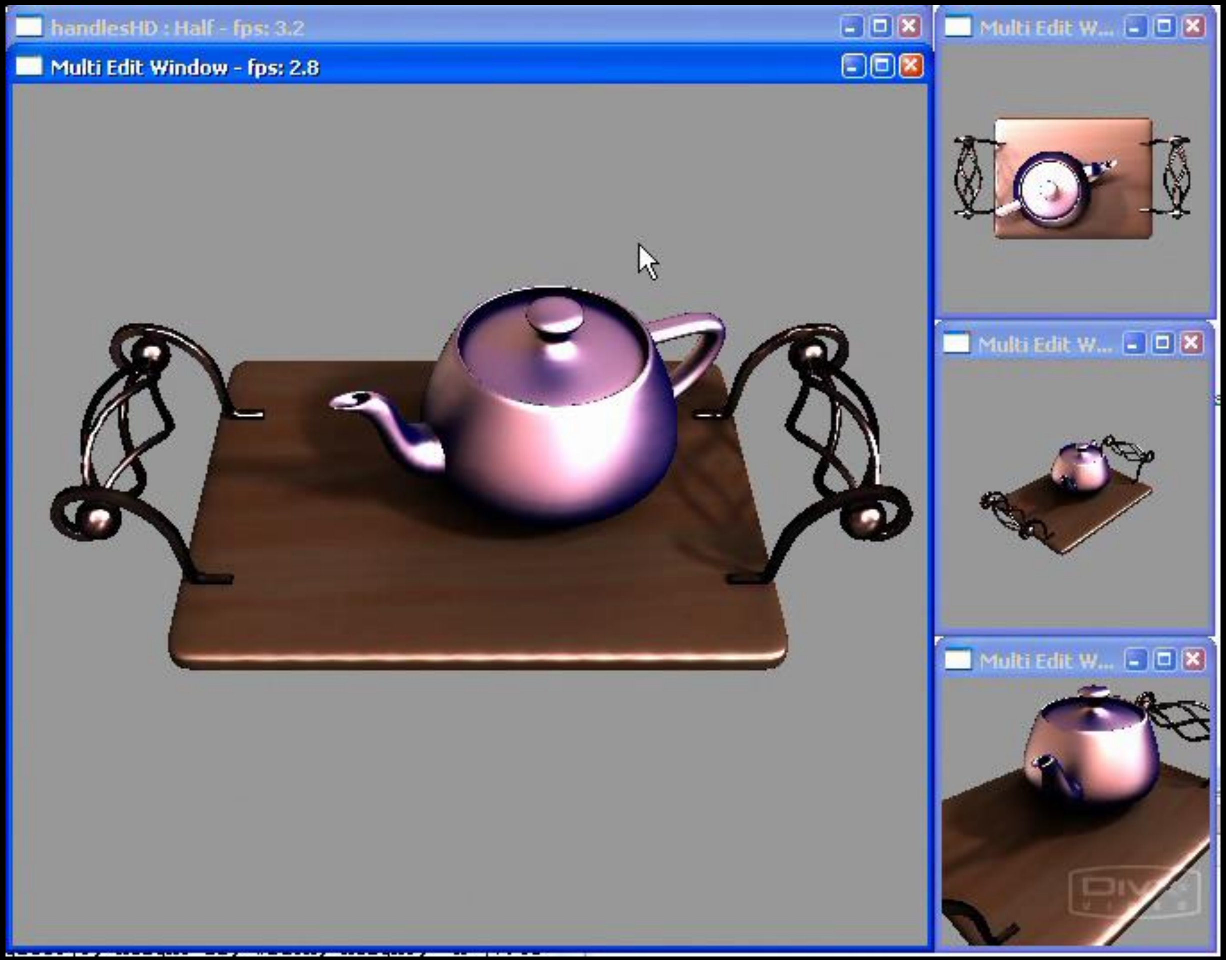
- Rendering Algorithm
  - Precompute:

$$T_j = \int_\Omega L(\mathbf{i})V(\mathbf{i})\max(0,\boldsymbol{n}\cdot\mathbf{i})\,\rho_q(\mathbf{i},\mathbf{o})b_j\big(\gamma(\mathbf{i},\mathbf{o})\big)\,\mathrm{d}\mathbf{i}$$

  - Runtime: (viewing direction $\mathbf{o}$ is fixed)

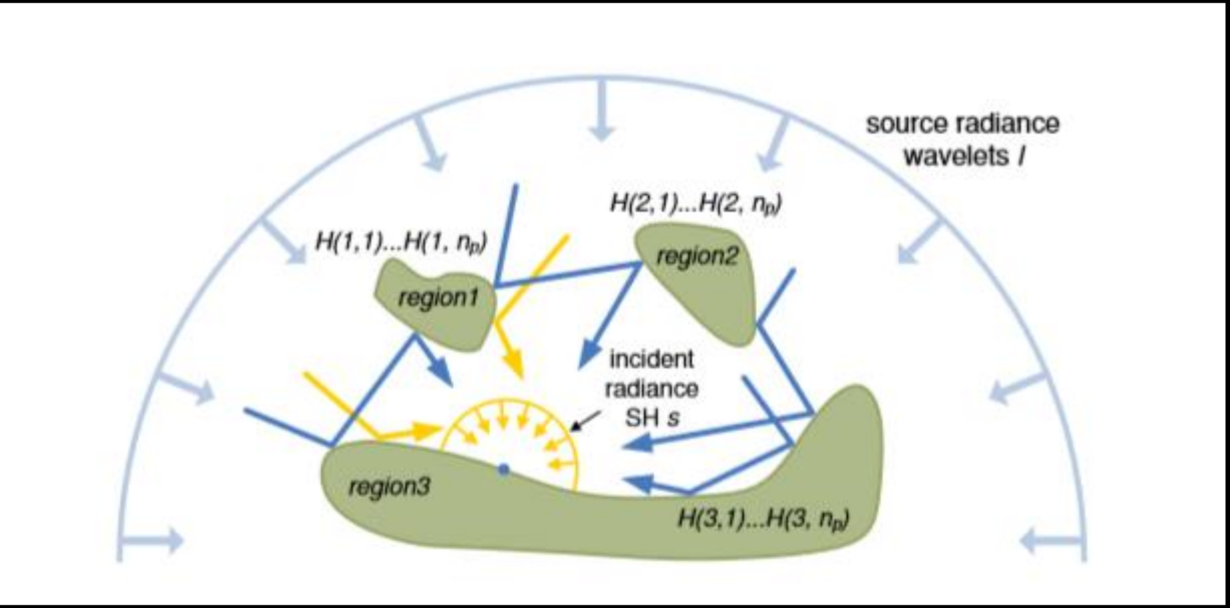
$$L(\mathbf{o}) \approx \sum c_j\, T_j$$

Results


handlesHD : Half - fps: 3.2
Multi Edit Window - fps: 2.8
Multi Edit W...
Multi Edit W...
Multi Edit W...

BRDF Editing with interreflection [Sun06]

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



Results

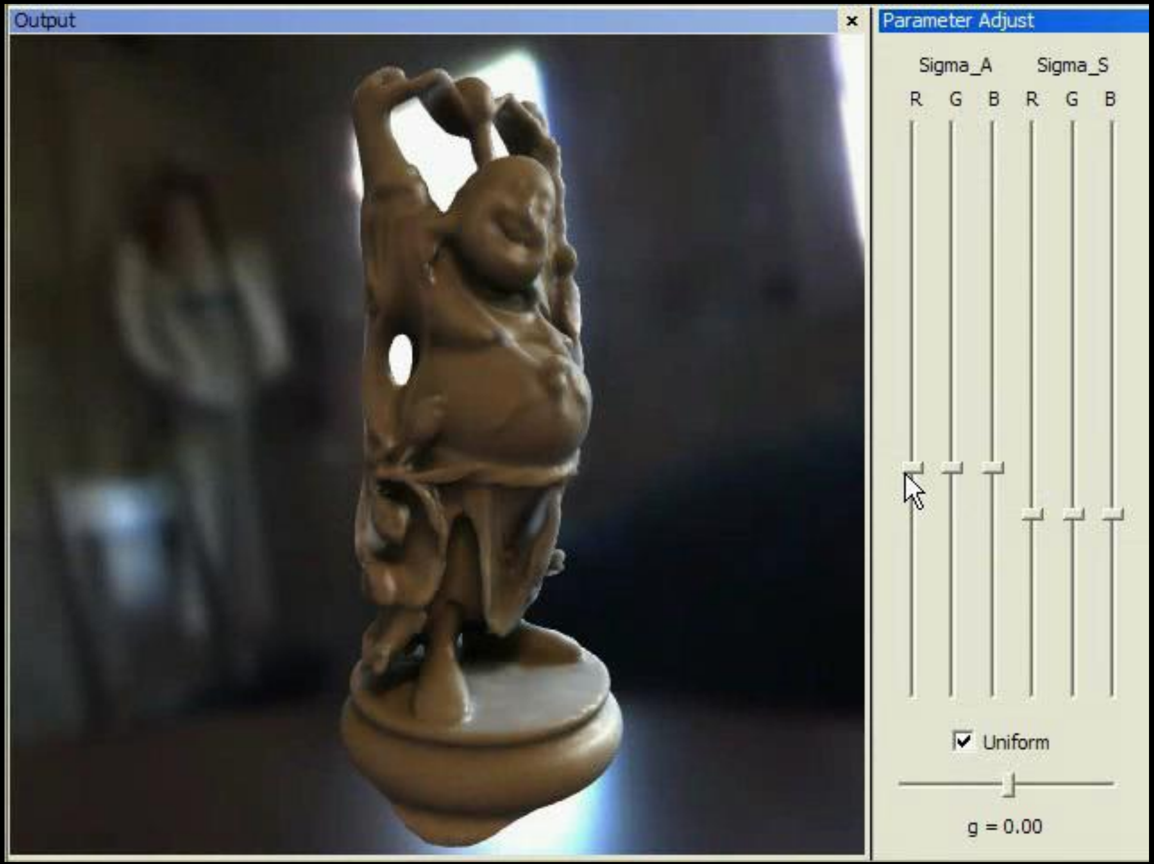
- Interactive rates

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 $\int_{\Omega} L(\mathbf{i})V(\mathbf{i})\rho(\mathbf{i})d\mathbf{i}$

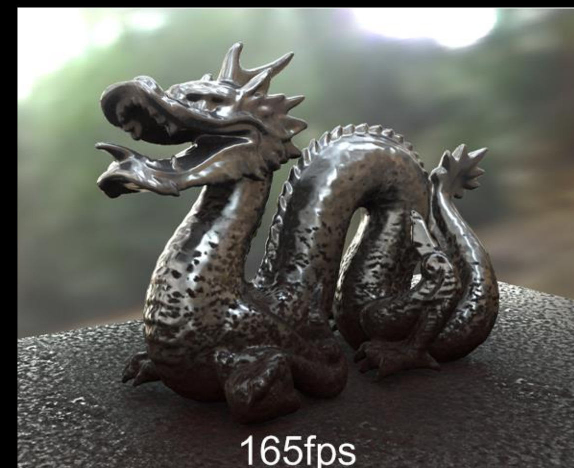
- PRT (Precomputation)
  - Long precomputation time, large storage
  - Bake geometry/material/lighting into precomputation, needs to fix them
- Analytic Computation
  - No (or small) precomputation
  - Everything dynamic, could be run-time changed

SG based analytic Integration

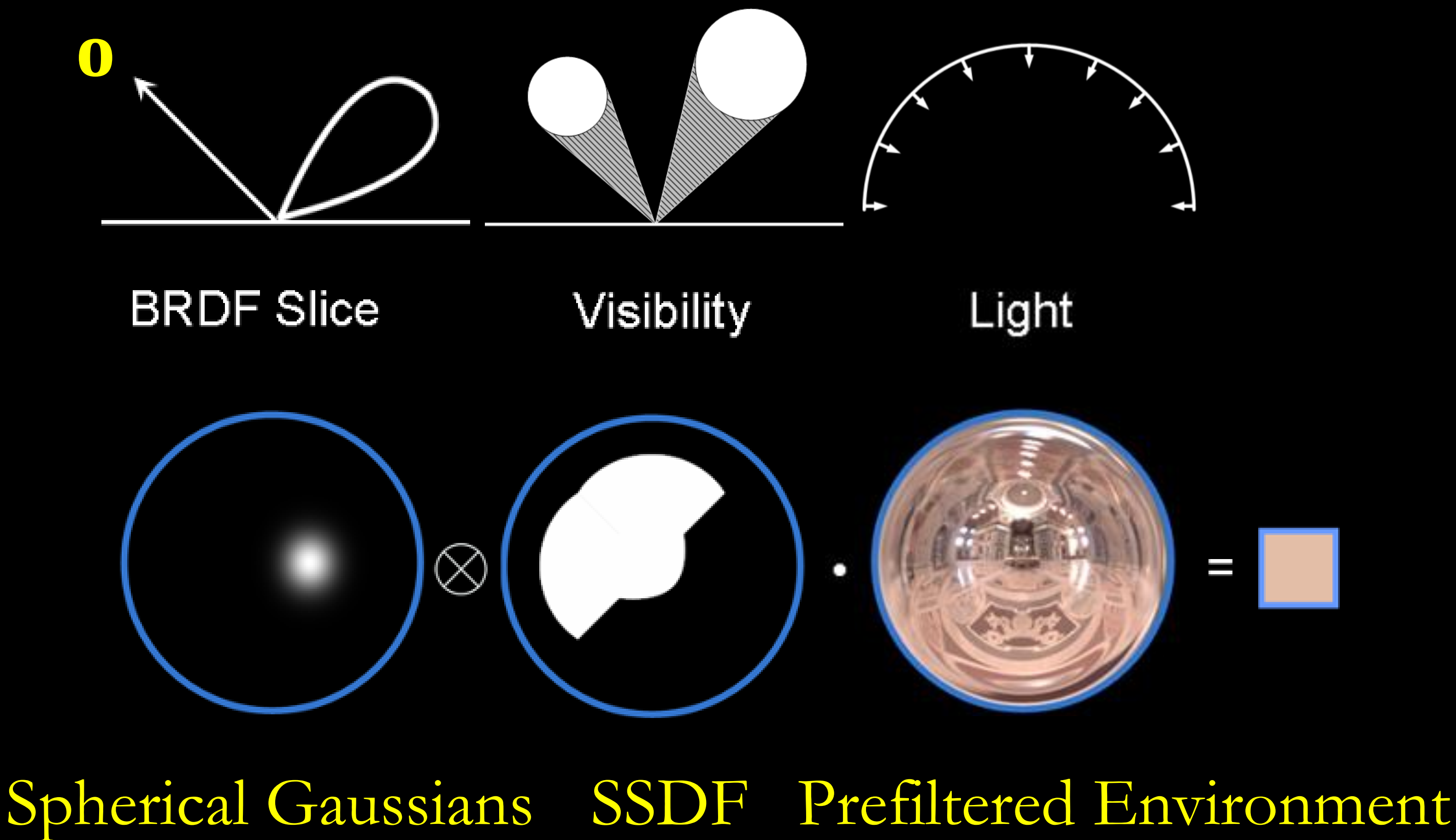
- SG as a PRT basis [Tsai 2006]
- rendering wi ynamic BR [Wang 20

Rendering with dynamic BRDFs [Wang09]

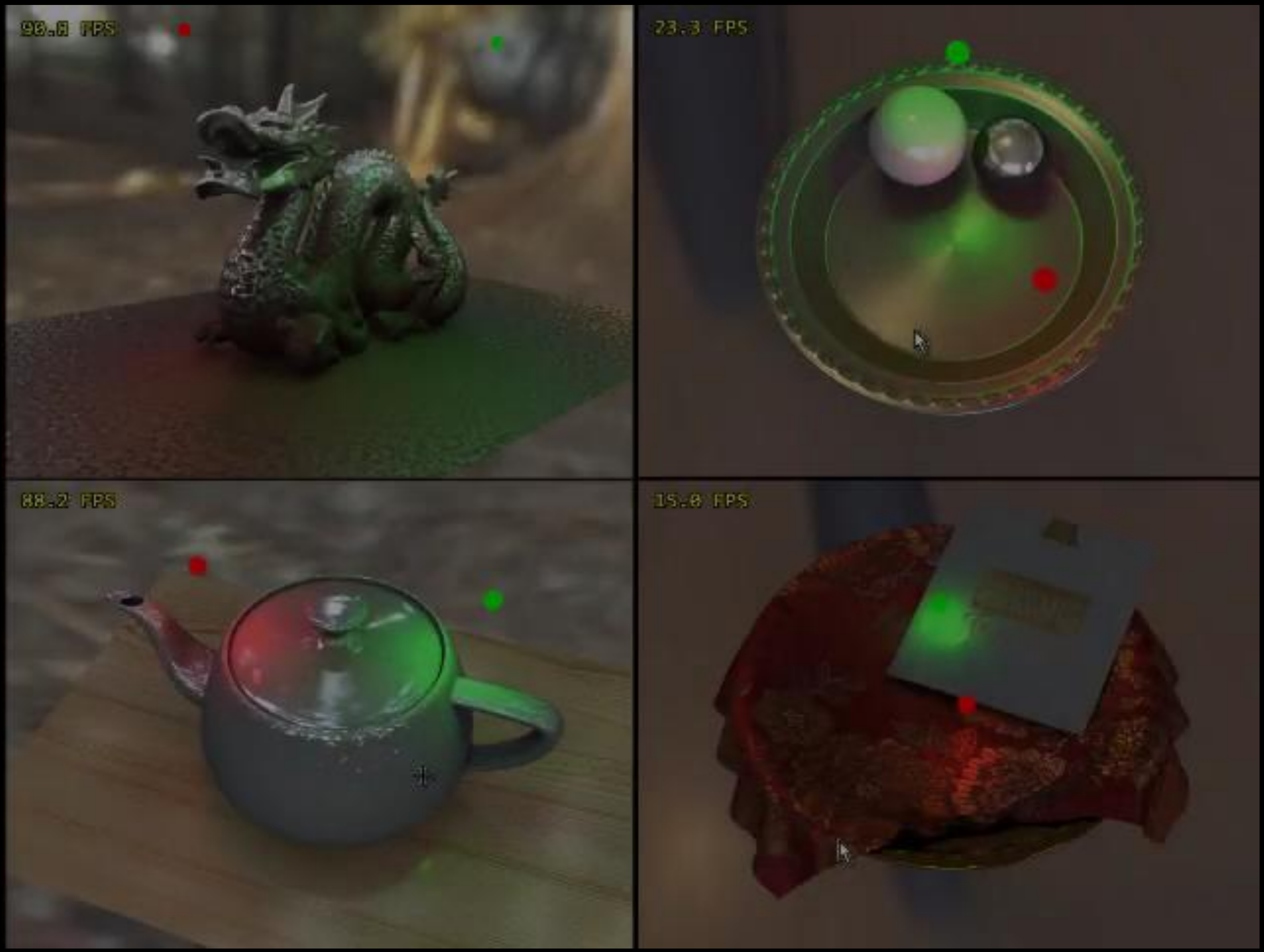
- static scene, dynamic lighting, dynamic BRDF
- BRDF: microfacet model
  - parametric ↔ measured
  - isotropic ↔ anisotropic
  - glossy ↔ mirror-like



Algorithm Overview



Results



Rendering and appearance editing of hairs [Xu 2011]

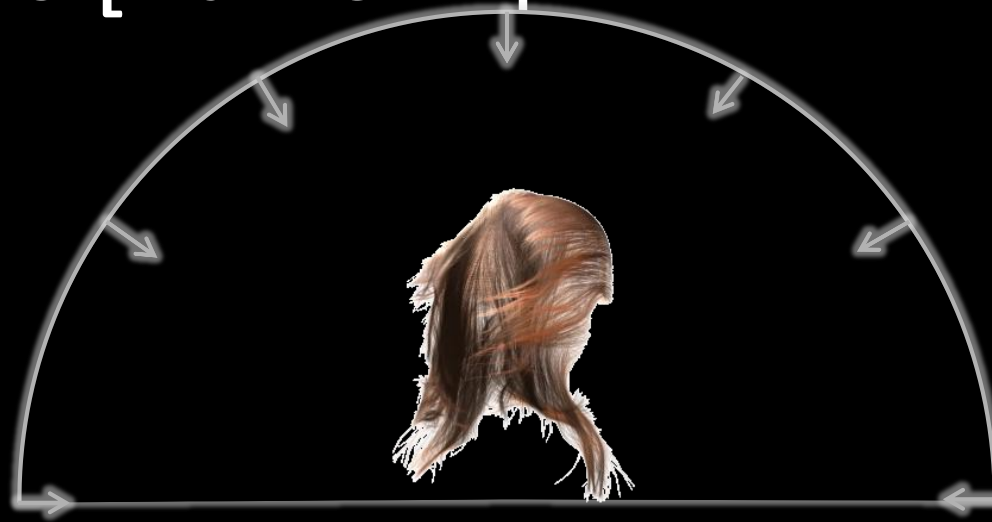
Single scattering

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

- $L(\omega_i)$: 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

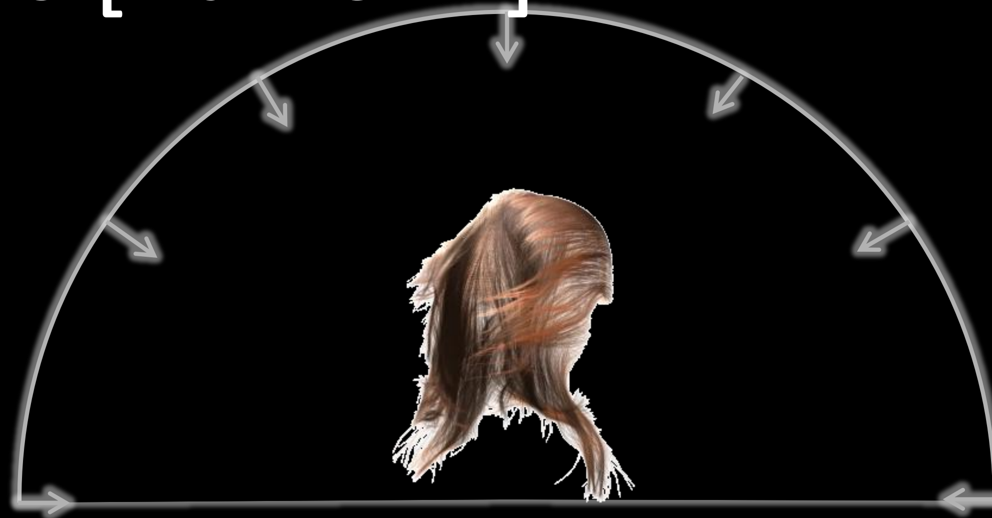


$$L(\omega_o) \approx D \sum_j \left(\int_{\Omega} G_j(\omega_i)\, T(\omega_i, \omega_o)\, S(\omega_i, \omega_o) \cos\theta_i \, d\omega_i \right)$$

- 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



$$L(\omega_o) \approx D \sum_j l_j \tilde{T} \int_{\Omega} G_j(\omega_i)\, S(\omega_i, \omega_o) \cos\theta_i \, d\omega_i$$

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

Problem:
evaluate scattering Integral

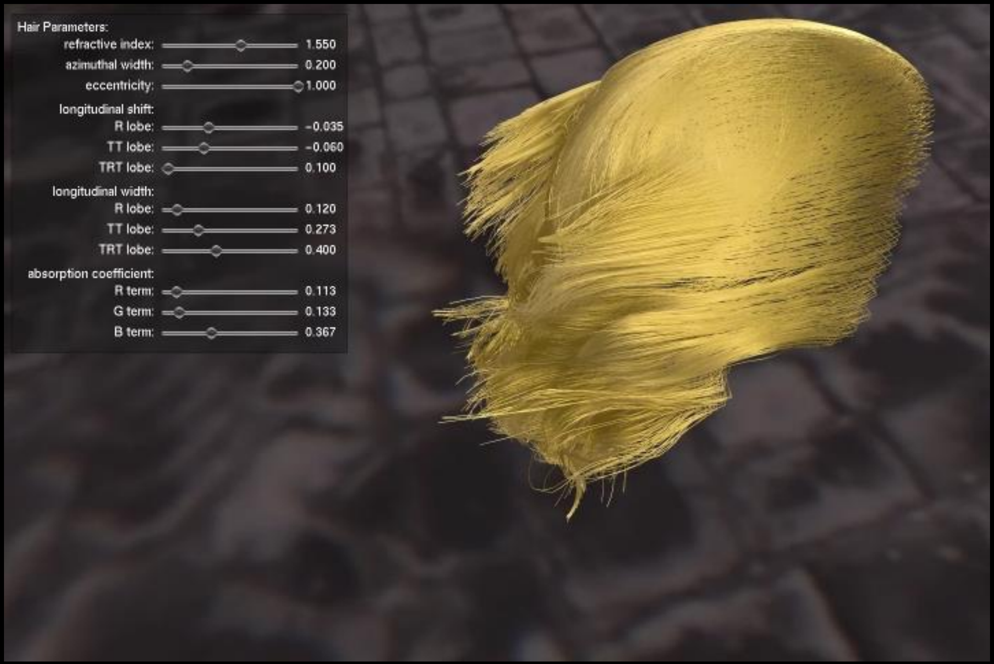
Single Scattering Integral

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

- 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 **circular Gaussians**
    - Approximate scattering function $S(\omega_i, \omega_o)$ by **circular Gaussians**

Results

- No precomputation
- all (geometry, lighting, hair scattering param.) dynamic

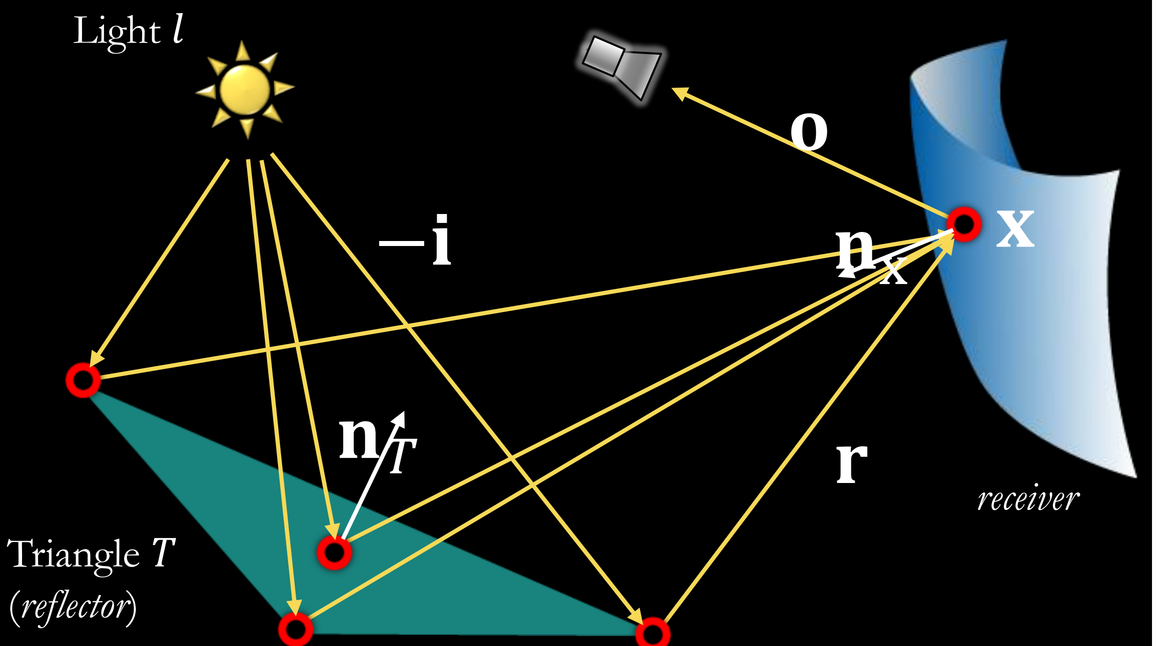


One-bounce interreflection [Xu 14]

- 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

$$L_{\mathrm{x}}(\mathbf{o}) = \int_{\Omega_T} \rho_{\mathrm{x}}(-\mathbf{r},\mathbf{o}) \max(-\mathbf{r}\cdot\mathbf{n}_{\mathrm{x}},0) \int_{\Omega} L_l(\mathbf{i})\,\rho_T(\mathbf{i},\mathbf{r}) \max(\boldsymbol{i}\cdot\mathbf{n}_{\mathrm{T}},0)\,\mathrm{d}\mathbf{i}\,\mathrm{d}\mathbf{r}$$



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

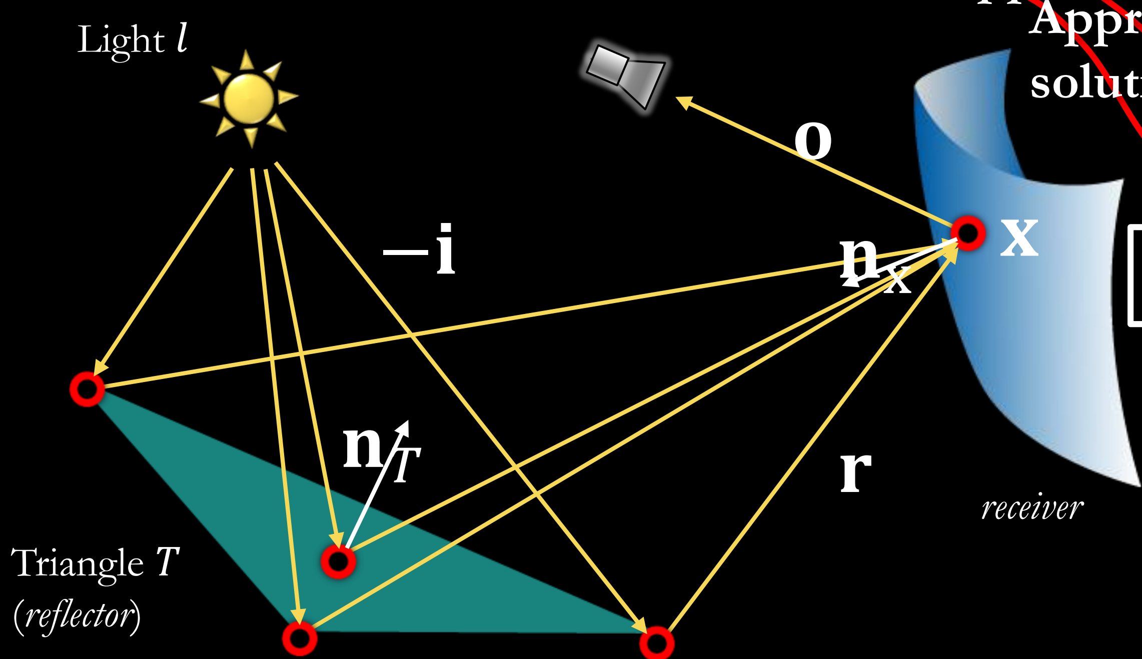
$$L_{\mathrm{x}}(\mathbf{o}) = \boxed{\int_{\Omega_T} \boxed{\rho_{\mathrm{x}}(-\mathbf{r},\mathbf{o})} \max(-\mathbf{r}\cdot\mathbf{n}_{\mathrm{x}},0) \boxed{\int_{\Omega} \boxed{L_l(\mathbf{i})}\boxed{\rho_T(\mathbf{i},\mathbf{r})} \max(\boldsymbol{i}\cdot\mathbf{n}_{\mathrm{T}},0)\, \mathrm{d}\mathbf{i}} \mathrm{d}\mathbf{r}}$$

Piecewise linear approximation

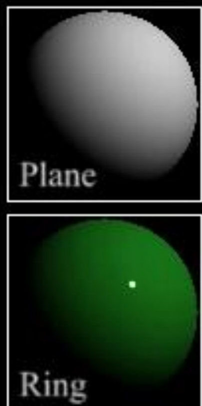
Approximated closed solution

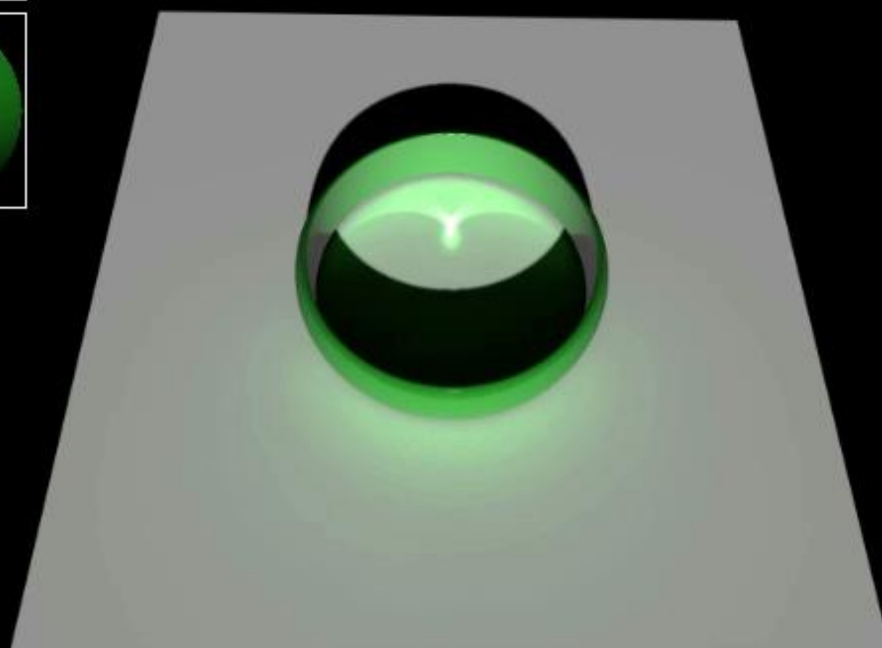
Represented

Analytically Evaluated !



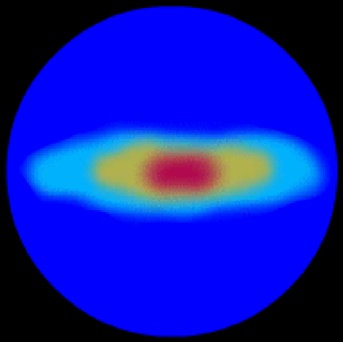
Results





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 → more accuracy, more cost
    - Smaller n → less accuracy, less cost



An example

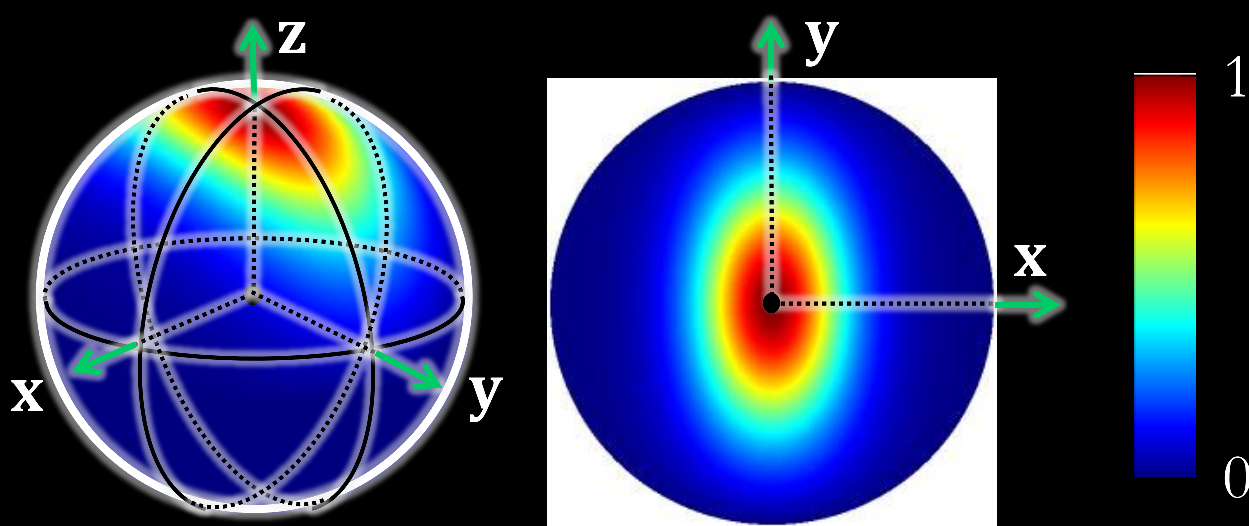
Anisotropic SG [Xu 14]

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

tangent　bi-tangent　lobe　bandwidth for **x**-axis　bandwidth for **y**-axis

An ASG example with

$$\lambda = 4$$
$$\mu = 1$$



3D view　Top view (2D)

ASGs

- Desired operators
  - Closed-form integral
  - Closed-form product
  - Closed-form convolution

Integral of an ASG

- Integral

$$\int_\Omega G(\mathbf{v})\mathrm{d}\mathbf{v}$$

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

- Our approximation

$$\int_\Omega G(\mathbf{v})\mathrm{d}\mathbf{v} \approx \frac{\pi}{\sqrt{\lambda\mu}}$$

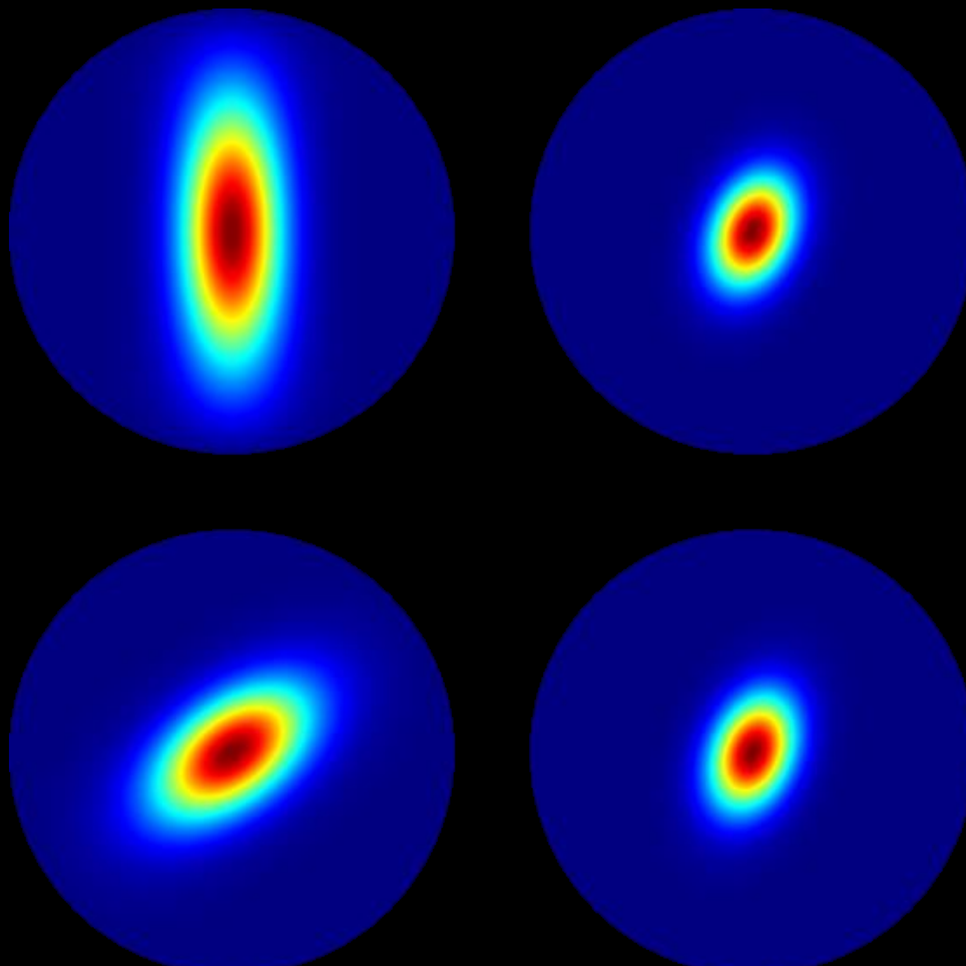
  - Accurate (error < 0.68%) when $\lambda, \mu > 5$

Product of two ASGs

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

1st ASG $G(\mathbf{v}; A_1)$

2nd ASG $G(\mathbf{v}; A_2)$



Approximated product

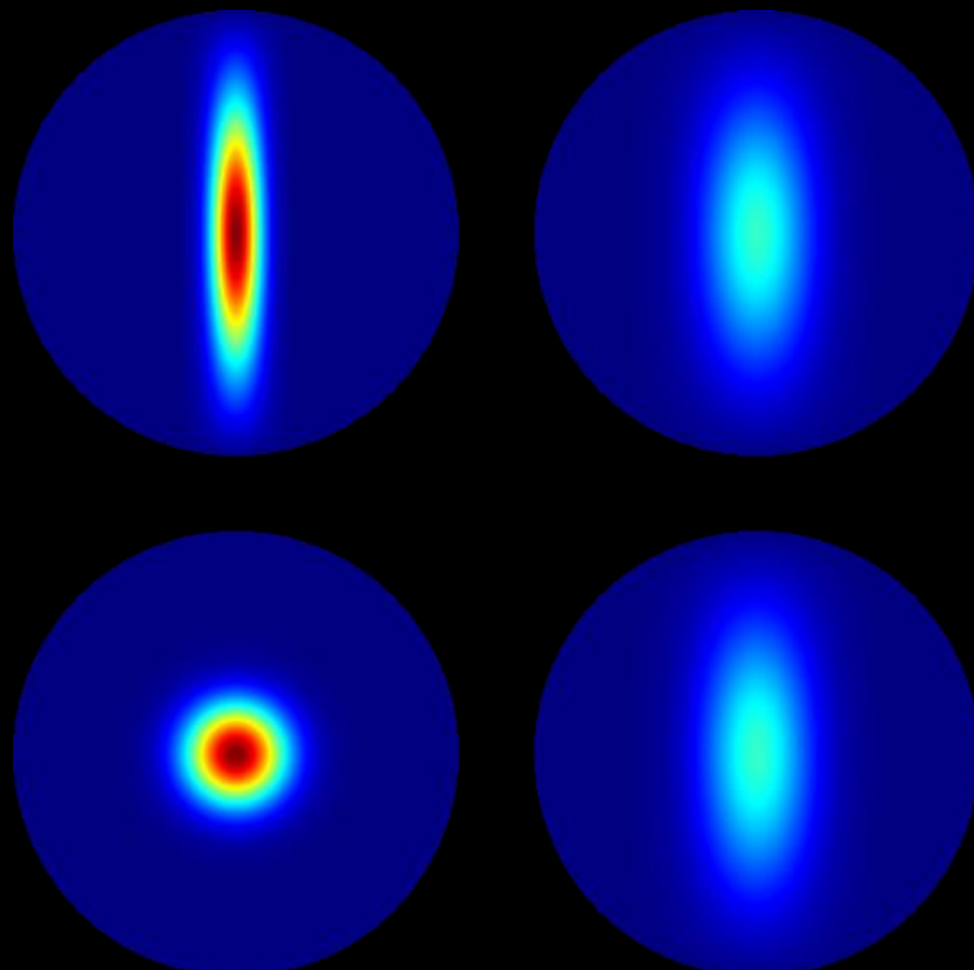
Ground truth product

Convolution of an ASG with an SG

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

ASG

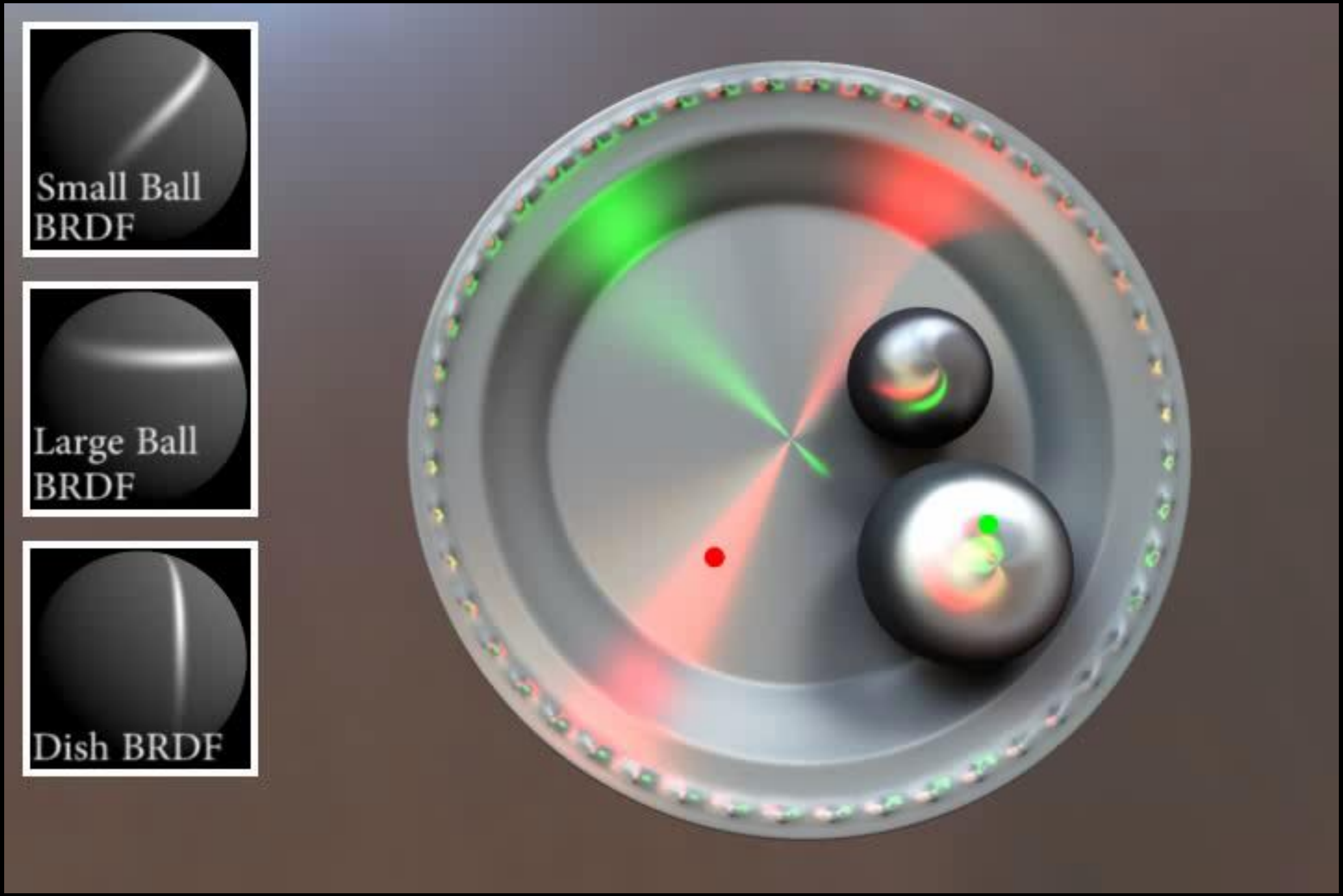
Approximated convolution



Convolution kernel (SG)

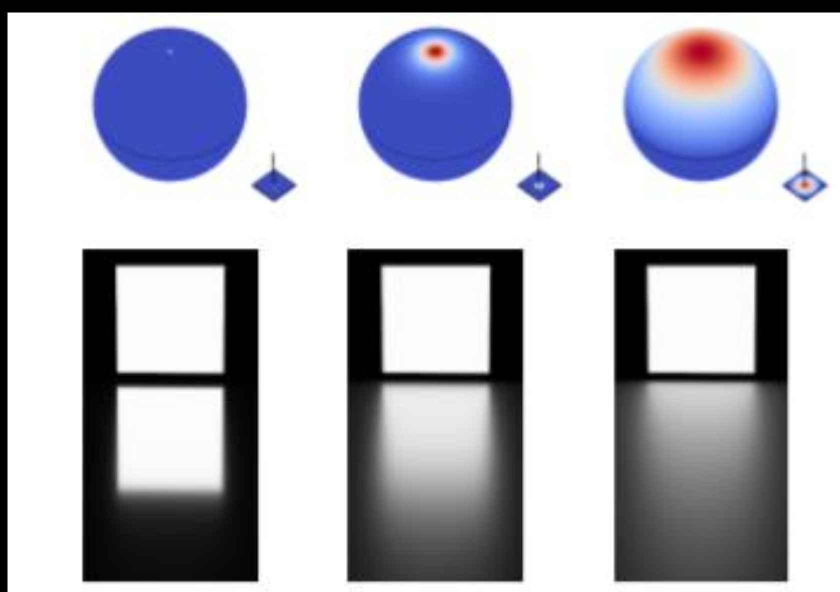
Ground truth convolution

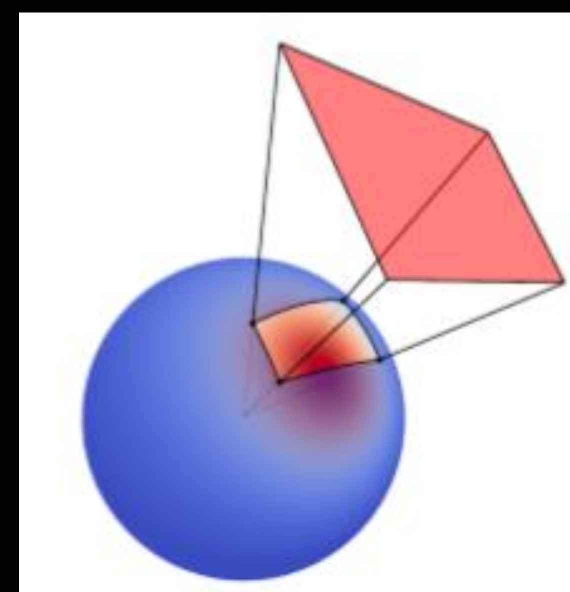
Results



Linearly Transformed Cosines [Heitz 18]

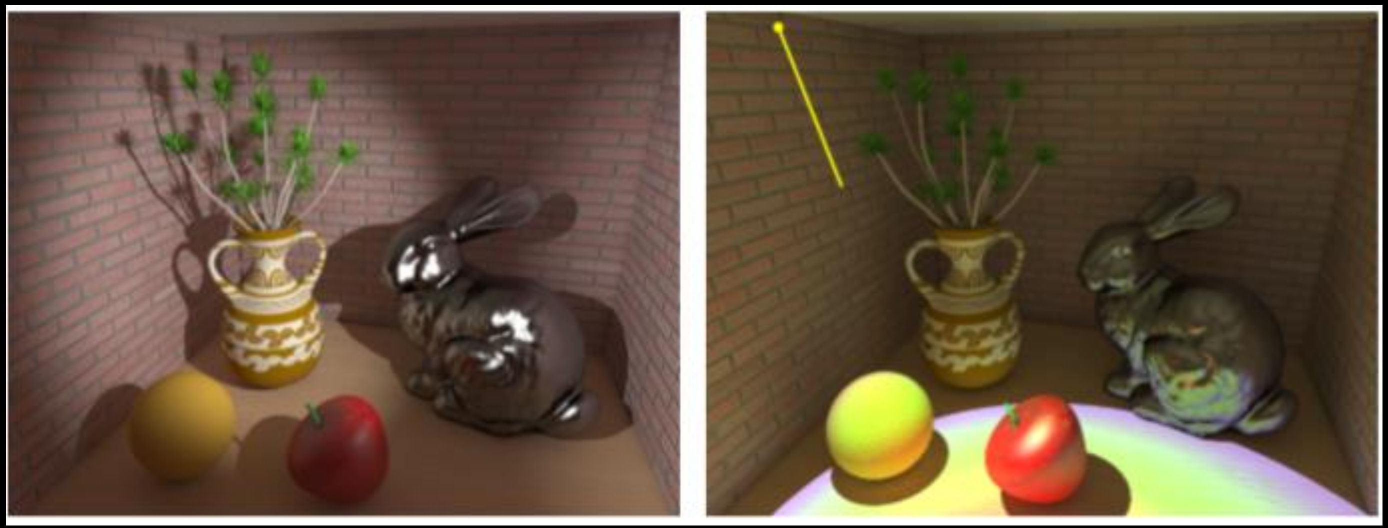
- Approximate BRDFs using *Linearly Transformed Cosines Functions*
- analytical integration on spherical polygons





Misc

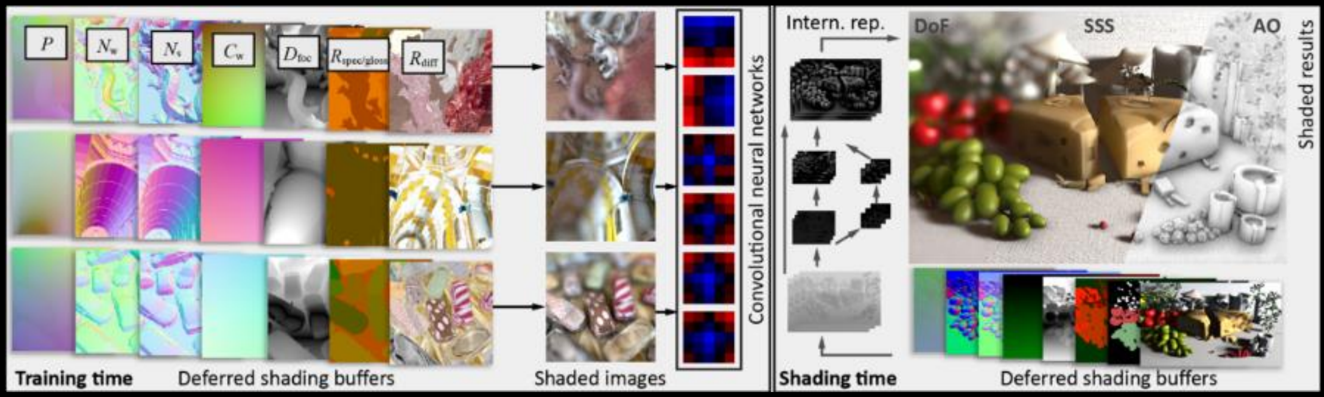
- Compression
  - VQ, PCA, Clustered PCA [Sloan 03]
- Meshless [Lehtinen 08]
- Image space
  - Direct-to-indirect Transfer [Hašan 06]



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?