All-Frequency Shadows Using Non-linear Wavelet Lighting Approximation

Ren Ng — *Stanford*

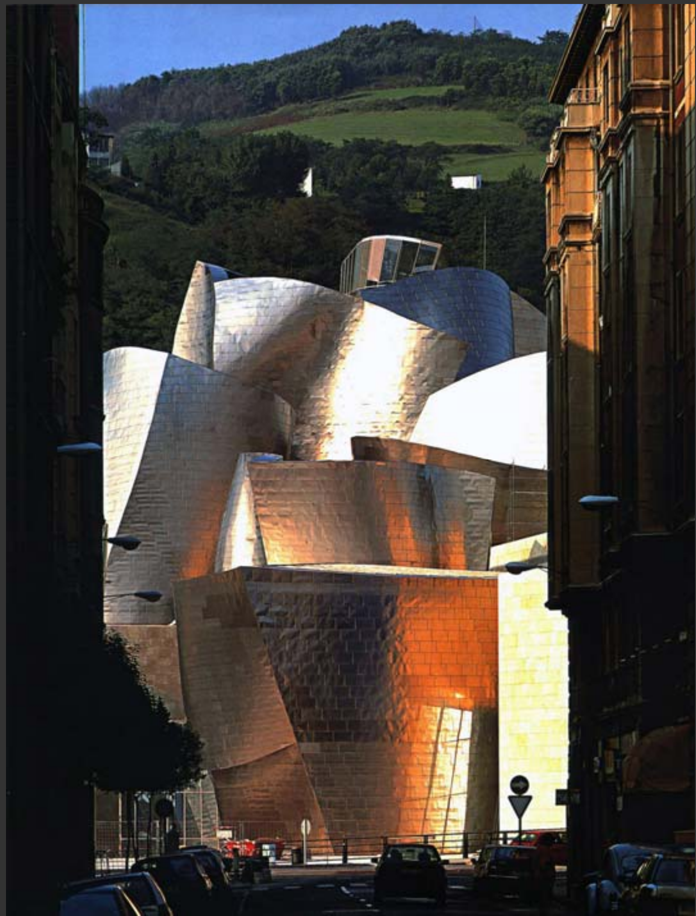
Ravi Ramamoorthi — *Columbia*

Pat Hanrahan — *Stanford*

SIGGRAPH 2003

Light on *Stone River* (Goldsworthy)

Lighting Design



From *Frank Gehry Architecture*, Ragheb ed. 2001

Existing Fast Shadow Techniques

We know how to render very hard and very soft shadows

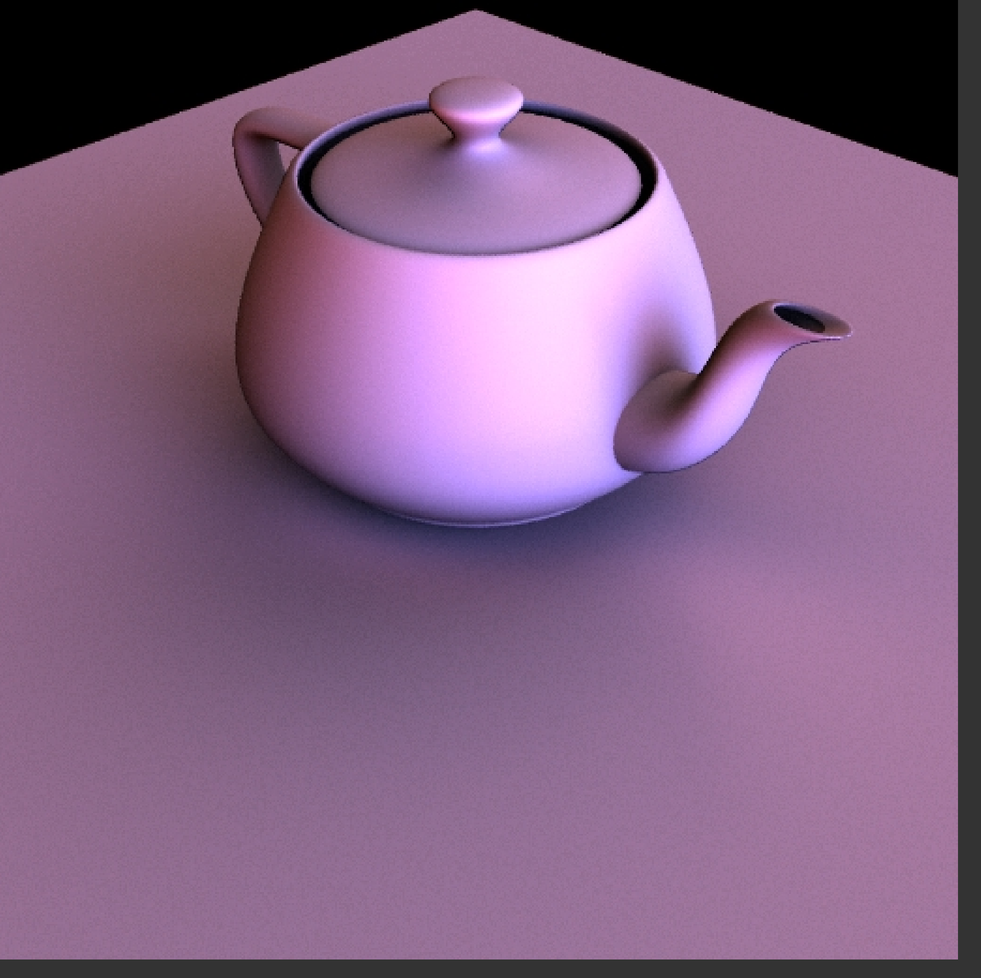
Sen, Cammarano, Hanrahan, 2003

Sloan, Kautz, Snyder 2002

Shadows from point-lights (shadow maps, volumes)

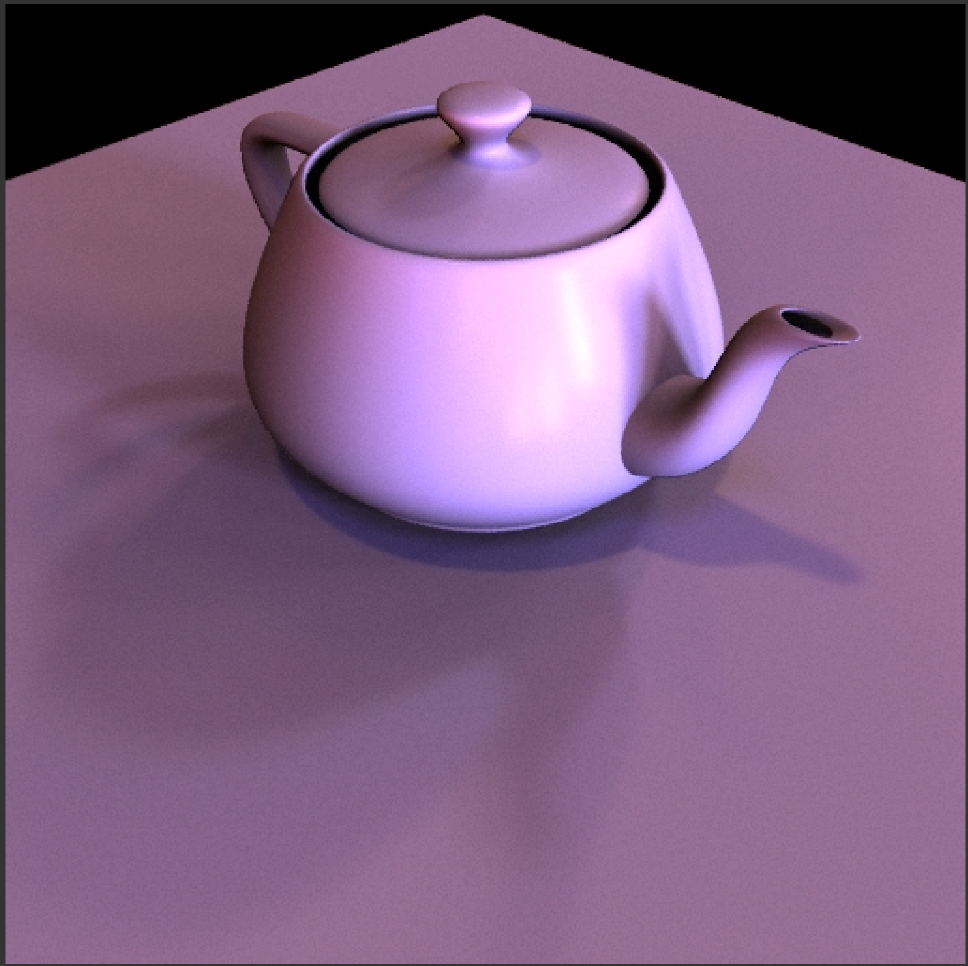
Shadows from smooth lighting (precomputed radiance transfer)

Soft Lighting



Teapot in Grace Cathedral

All-Frequency Lighting



Teapot in Grace Cathedral

Formulation

$$B(\mathbf{x},\omega_0) = \int_\Omega L(\mathbf{x},\omega) V(\mathbf{x},\omega) f(\mathbf{x},\omega,\omega_0)(\omega \cdot \mathbf{n}(\mathbf{x})) d\omega$$

Geometry relighting: assume diffuse

$$T(\mathbf{x},\omega) = V(\mathbf{x},\omega) f(\mathbf{x},\omega \rightarrow \omega_0)(\omega \cdot \mathbf{n}(\mathbf{x}))$$

Image relighting: fix view point

$$T(\mathbf{x},\omega) = V(\mathbf{x},\omega) f(\mathbf{x},\omega \rightarrow \omega_0(\mathbf{x}))(\omega \cdot \mathbf{n}(\mathbf{x}))$$

$$\longrightarrow \quad B(\mathbf{x}_i) = \sum_j T(\mathbf{x}_i,\omega_j) L(\omega_j)$$

$$\mathbf{B} = \mathbf{TL}$$

T could include global effects

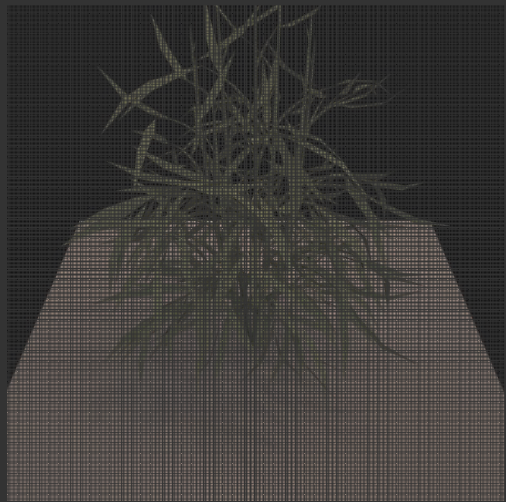
Relighting as Matrix-Vector Multiply

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \vdots \\ P_N \end{bmatrix}$$

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

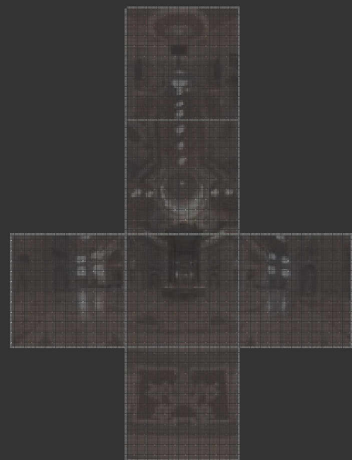
Relighting as Matrix-Vector Multiply

$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \vdots \\ P_N \end{bmatrix}$$



Output Image (Pixel Vector)

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



Input Lighting (Cubemap Vector)

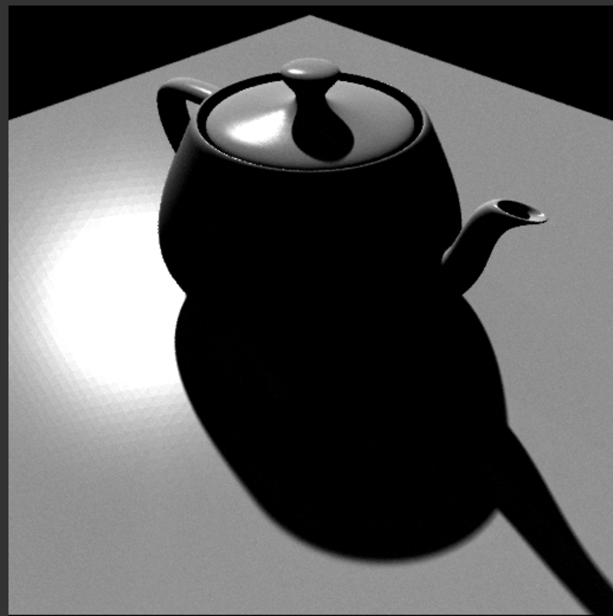
Transport Matrix

Ray-Tracing Matrix Columns

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

Ray-Tracing Matrix Columns

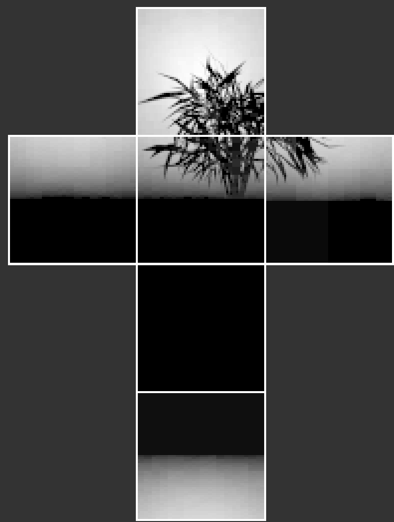
$$\begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1M} \\ T_{21} & T_{22} & \cdots & T_{2M} \\ T_{31} & T_{32} & \cdots & T_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ T_{N1} & T_{N2} & \cdots & T_{NM} \end{bmatrix}$$



Only works for image relighting

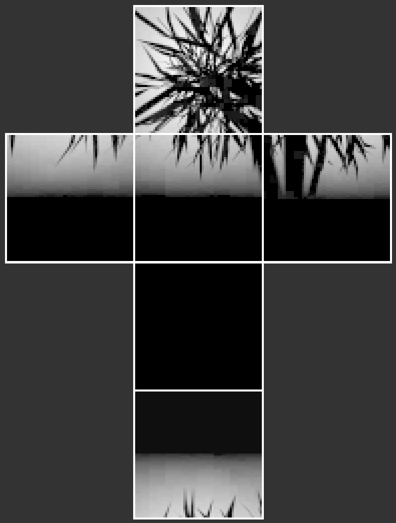
Light-Transport Matrix Rows

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



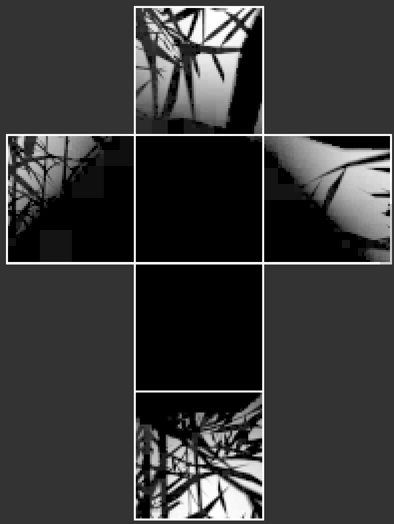
Light-Transport Matrix Rows

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



Light-Transport Matrix Rows

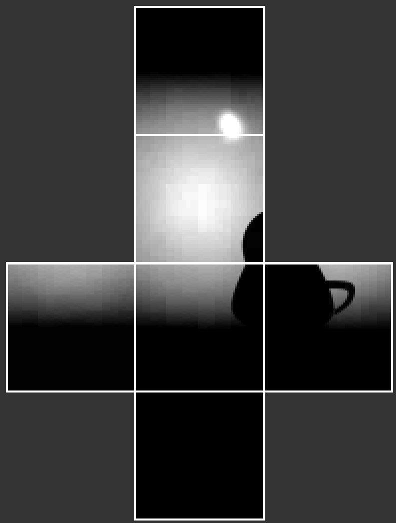
$$\begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1M} \\ T_{21} & T_{22} & \cdots & T_{2M} \\ T_{31} & T_{32} & \cdots & T_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ T_{N1} & T_{N2} & \cdots & T_{NM} \end{bmatrix}$$



Rasterizing Matrix Rows

Pre-computing rows

- Rasterize visibility hemicubes with graphics hardware
- Read back pixels and weight by reflection function



Matrix Multiplication is Enormous

Dimension

- 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*

Sparse Matrix-Vector Multiplication

Choose data representations with mostly zeroes

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

Matrix: Wavelet-encode transport rows

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

Non-linear Wavelet Light Approximation

Wavelet Transform

Non-linear Wavelet Light Approximation

$$\begin{bmatrix} 0 \\ L_2 \\ 0 \\ 0 \\ 0 \\ L_6 \\ \vdots \\ 0 \end{bmatrix}$$

Non-linear Approximation

Retain 0.1% – 1% terms

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**

Idea: Compress lighting by considering input data

Why Wavelets?

Wavelets provide dual space / frequency locality

- Large wavelets capture low frequency, area lighting
- Small wavelets capture high frequency, compact features

In contrast

- Spherical harmonics
  - Perform poorly on compact lights
- Pixel basis
  - Perform poorly on large area lights

Choosing Non-linear Coefficients

Three methods of prioritizing

1. Magnitude of wavelet coefficient
   - Optimal for approximating lighting
2. Transport-weighted
   - Biases lights that generate bright images
3. Area-weighted
   - Biases large lights

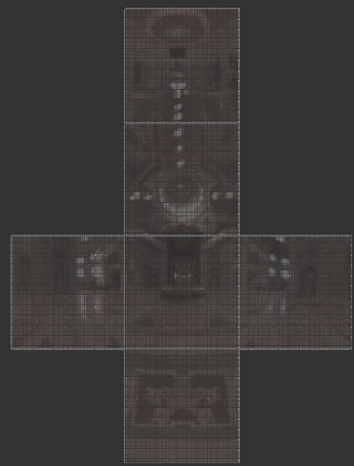
Sparse Matrix-Vector Multiplication

Choose data representations with mostly zeroes

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

Matrix: Wavelet-encode transport rows

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

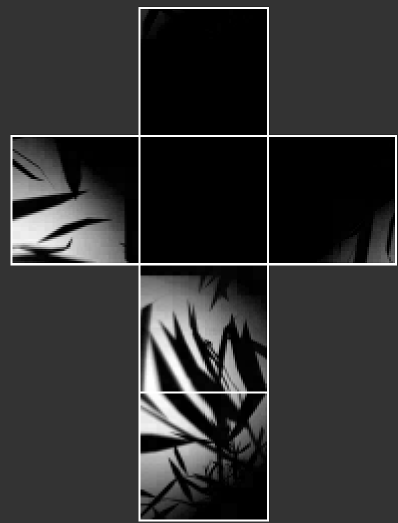


Matrix Row Wavelet Encoding

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

Matrix Row Wavelet Encoding

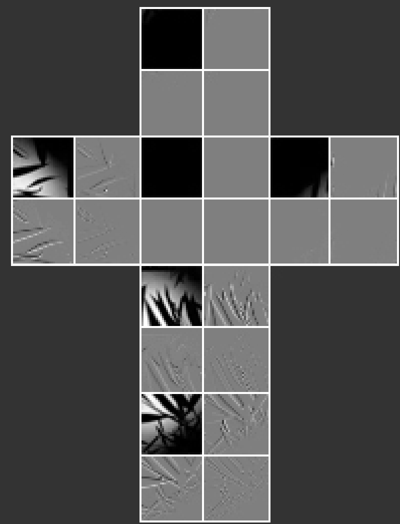
$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\ T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\ T_{61} &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



Extract Row

Matrix Row Wavelet Encoding

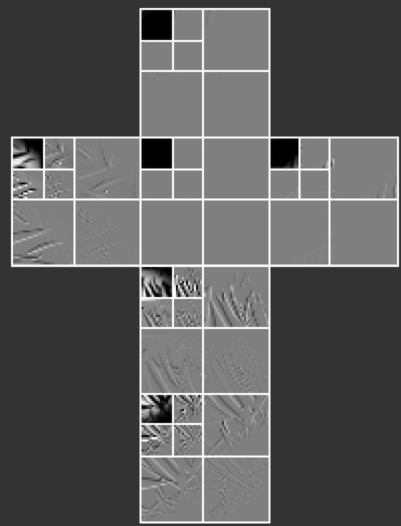
$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\ T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\ T_{61} &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



Wavelet Transform

Matrix Row Wavelet Encoding

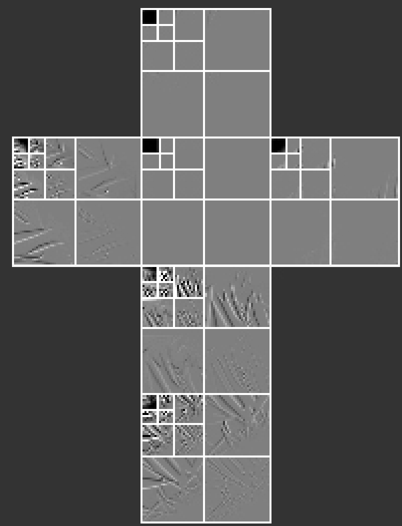
$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\ T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\ T_{61} &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



Wavelet Transform

Matrix Row Wavelet Encoding

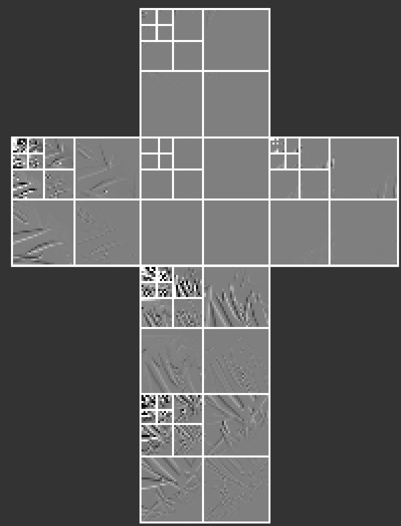
$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\ T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\ T_{61} &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



Wavelet Transform

Matrix Row Wavelet Encoding

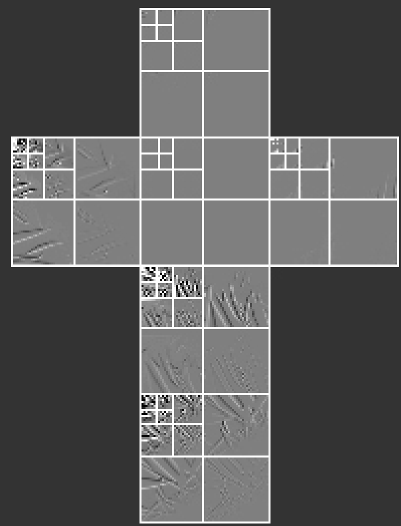
$$\begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} & \cdots & T_{1M} \\ T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\ T_{61} &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



Wavelet Transform

Matrix Row Wavelet Encoding

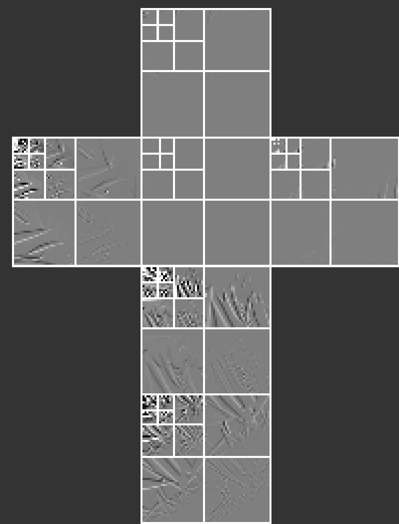
$$\begin{bmatrix} T_{11}' & 0 & 0 & T_{14}' & \cdots & 0 \\ T_{21} & T_{22} & T_{23} & T_{24} & \cdots & T_{2M} \\ T_{31} & T_{32} & T_{24} & T_{34} & \cdots & T_{3M} \\ T_{41} & T_{42} & T_{43} & T_{44} & \cdots & T_{4M} \\ T_{51} & T_{52} & T_{53} & T_{54} & \cdots & T_{5M} \\ T_{61} & T_{62} & T_{63}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



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

Total Compression

Lighting vector compression

- Highly lossy
- Compress to 0.1% – 1%

Matrix compression

- Essentially lossless encoding
- Represent with 3% – 30% non-zero terms

Total compression in sparse matrix-vector multiply

- 3 – 4 orders of magnitude less work than full multiplication

Error Analysis

Compare to linear spherical harmonic approximation

- [Sloan, Kautz, Snyder, 2002]

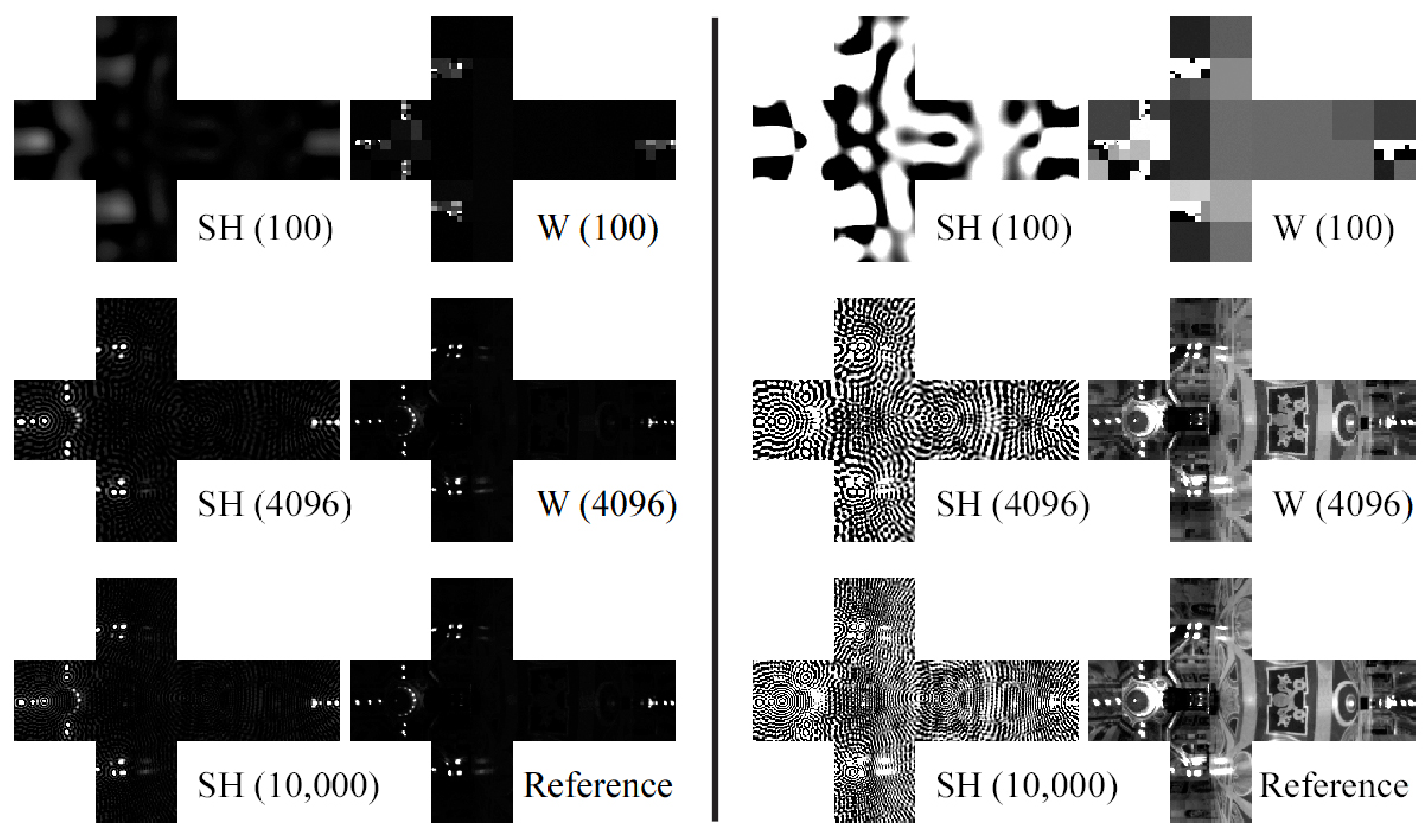
Measure approximation quality in two ways:

- Error in lighting
- Error in output image pixels

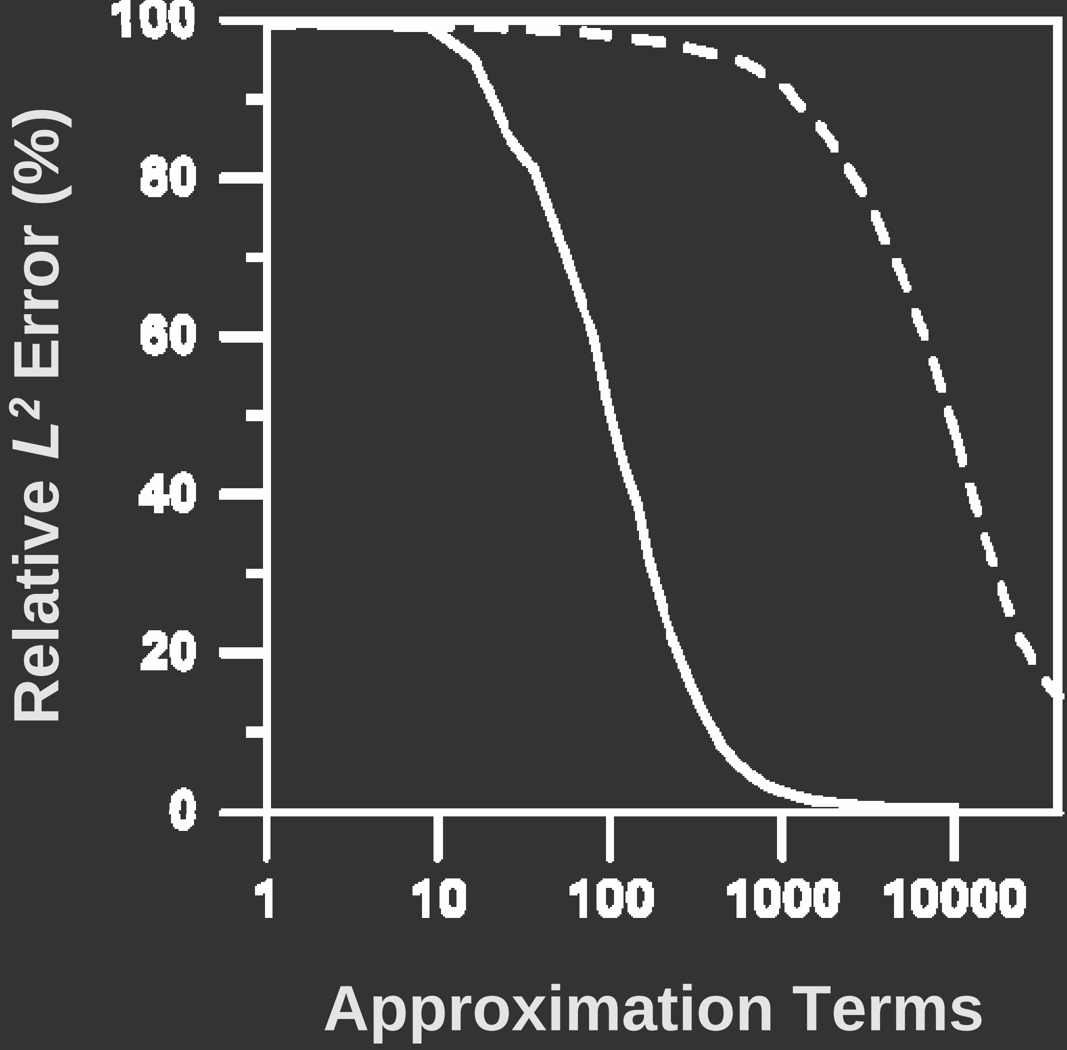
Main point (for detailed natural lighting)

- Non-linear wavelets converge exponentially faster than linear harmonics

Lighting Error



Error in Lighting: St Peter's Basilica

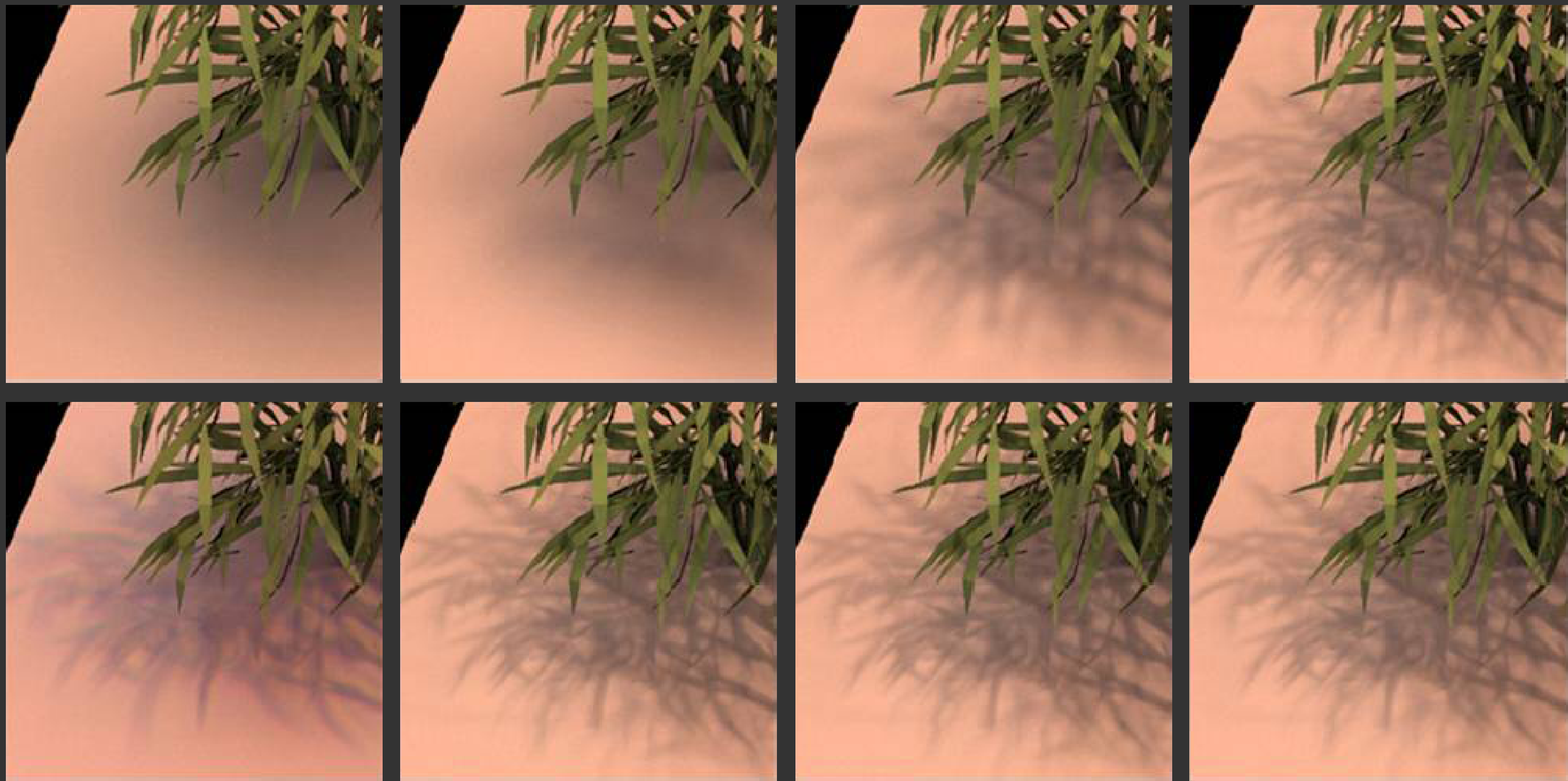


Sph. Harmonics

Non-linear Wavelets

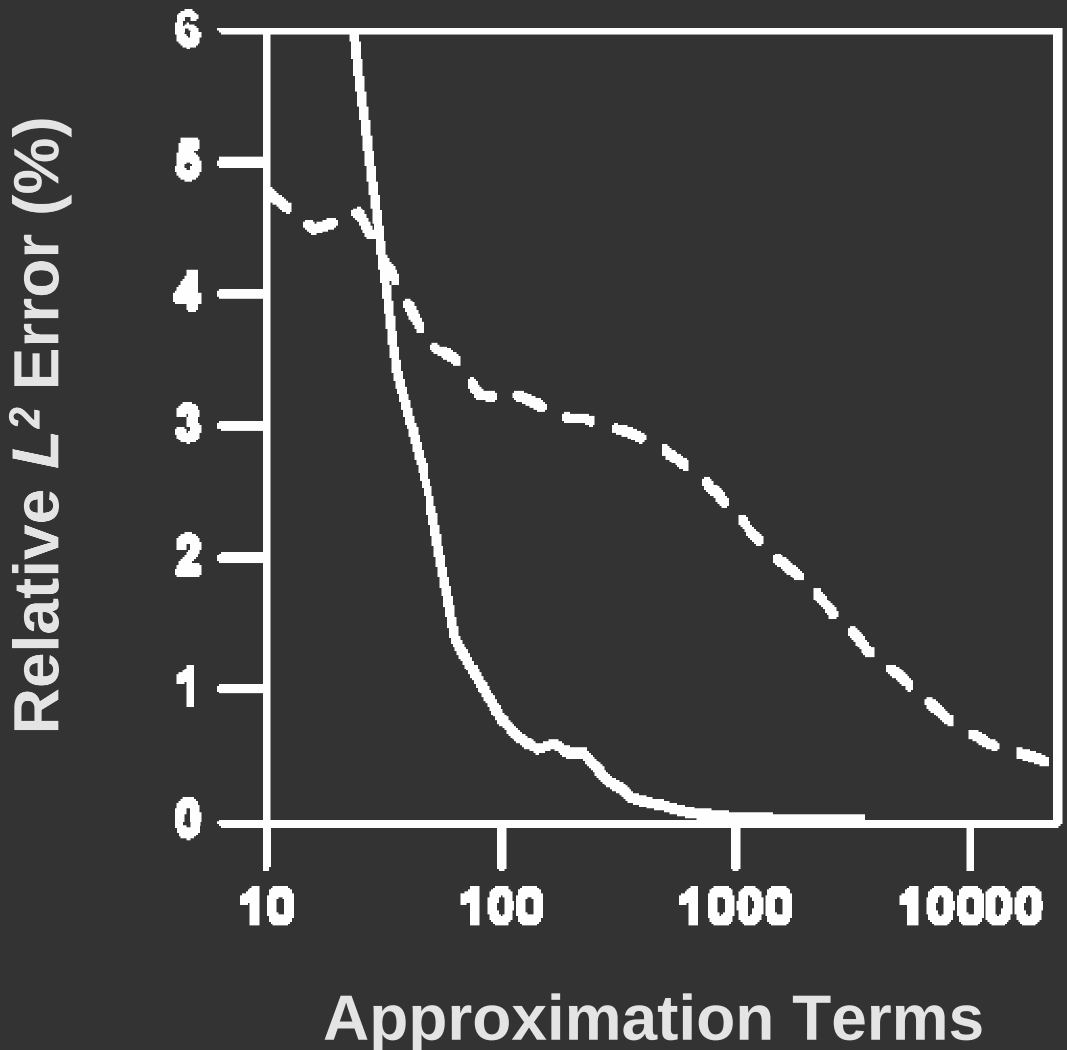
Output Image Comparison

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



25 | 200 | 2,000 | 20,000

Error in Output Image: Plant Scene



Results

SIGGRAPH 2003 video

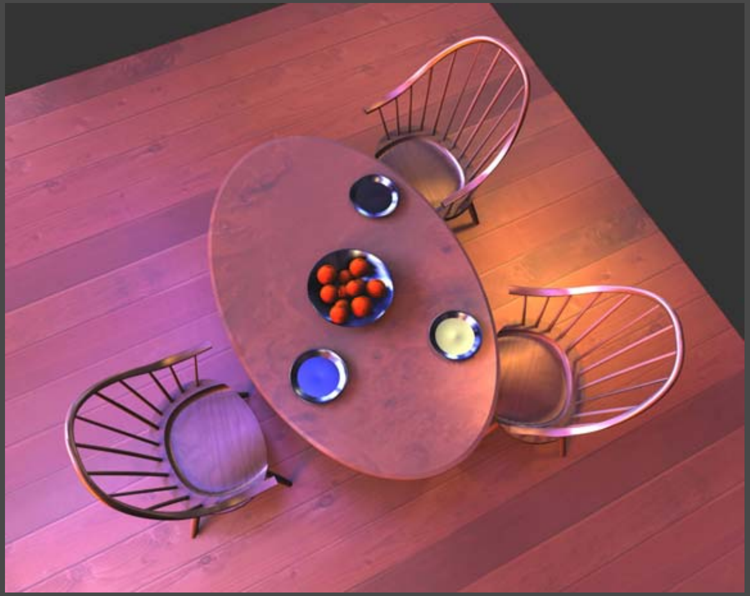
Summary

A viable representation for all-frequency lighting

- Sparse non-linear wavelet approximation
- 100 – 1000 times information reduction

Efficient relighting from detailed environment maps

Triple Product Wavelet Integrals For All-Frequency Relighting



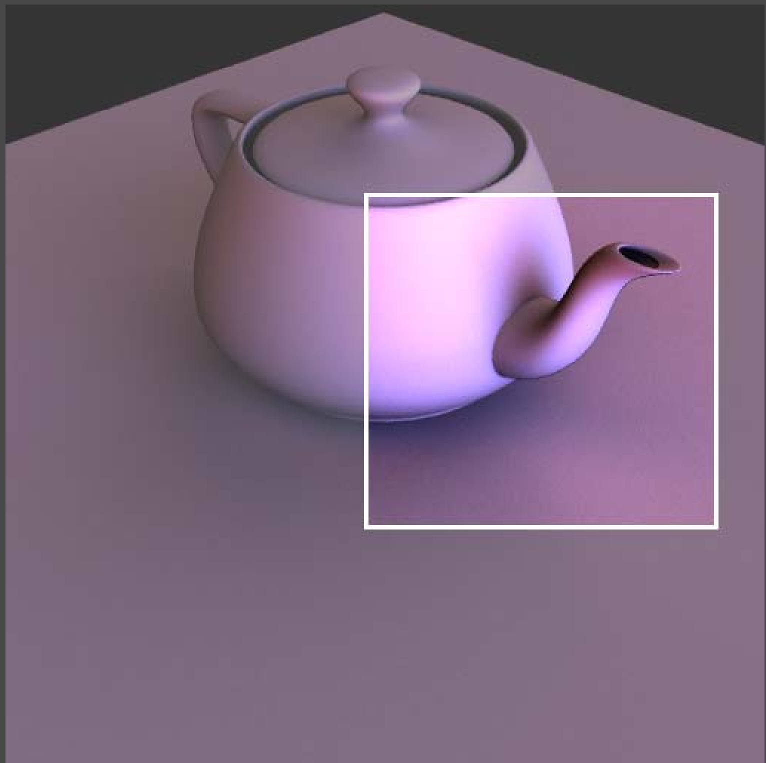
Ren Ng
Stanford

Ravi Ramamoorthi
Columbia

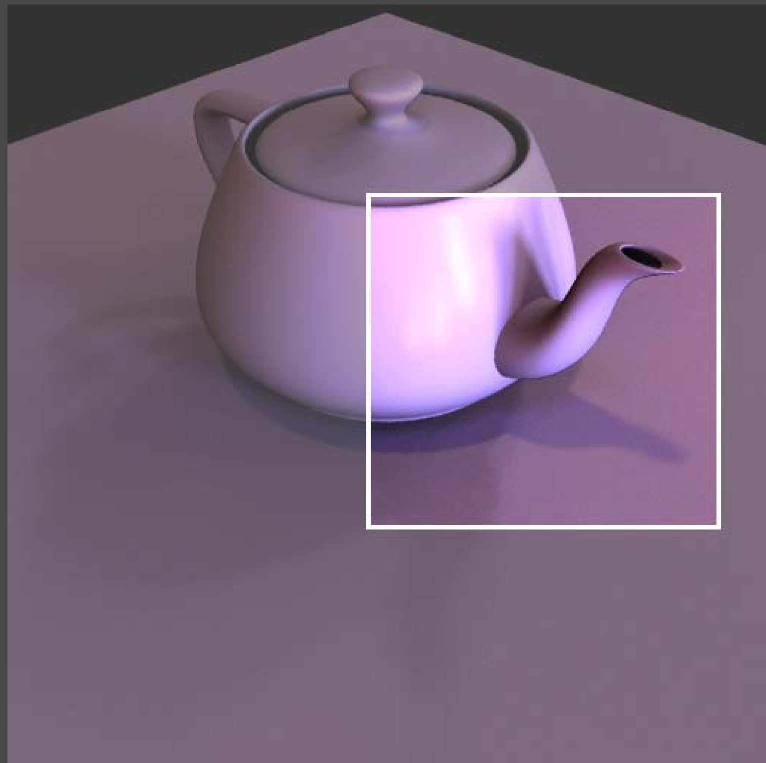
Pat Hanrahan
Stanford

SIGGRAPH 2004

Precomputed Relighting Techniques



Low-Frequency Lighting,
Dynamic View
[Sloan et al 2002, 2003]



All-Frequency Lighting,
Fixed View
[Ng et al 2003]

Formulation

$$B(\mathbf{x}, \omega_0) = \int_\Omega L(\mathbf{x}, \omega) V(\mathbf{x}, \omega) \rho(\mathbf{x}, \omega, \omega_0)(\omega \cdot \mathbf{n}(\mathbf{x})) d\omega$$

Simplifications:

1. **Incorporate cosine term into BRDF**
2. **Assume the lighting is distant, so L depends only on** ω
3. **Uniform BRDF over the surface**
4. **Expressed in the global frame; BRDF has to be rotated**

$$B(\mathbf{x}, \omega_0) = \int_\Omega L(\omega) V(\mathbf{x}, \omega) \tilde{\rho}(\mathbf{x}, \omega, \omega_0) d\omega$$

For a fixed point,

$$B = \int_\Omega L(\omega) V(\omega) \tilde{\rho}(\omega) d\omega$$

Double product

$$B = \int_\Omega L(\omega) \boxed{V(\omega)\tilde{\rho}(\omega)} d\omega$$

$$T(\omega)$$

$$L(\omega) = \sum_i L_i \Psi_i(\omega)$$

$$T(\omega) = \sum_j T_j \Psi_j(\omega)$$

Let $C_{ij} = \int_\Omega \Psi_i(\omega)\Psi_j(\omega)d\omega$

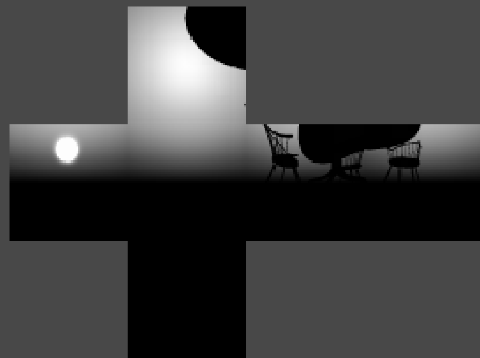
$$B = \int_\Omega \left(\sum_i L_i \Psi_i(\omega) \right)\left(\sum_j T_j \Psi_j(\omega) \right) d\omega$$

$$= \sum_i \sum_j L_i T_j \int_\Omega \Psi_i(\omega)\Psi_j(\omega)d\omega$$

$$= \sum_i \sum_j L_i T_j C_{ij} = \sum_i \sum_j L_i T_j \delta_{ij} = \sum_i L_i T_i = \mathbf{T} \cdot \mathbf{L}$$

Double Product Integral Relighting

Lighting

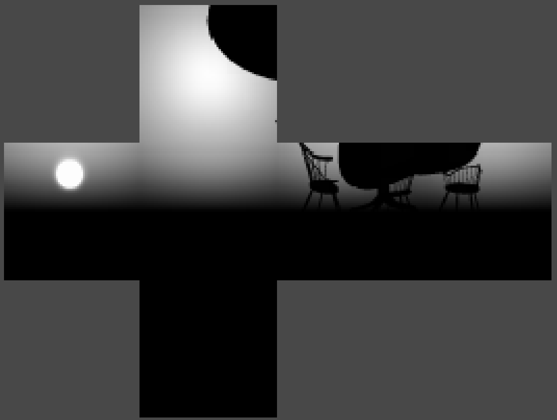


Transport

Problem Characterization

6D Precomputation Space

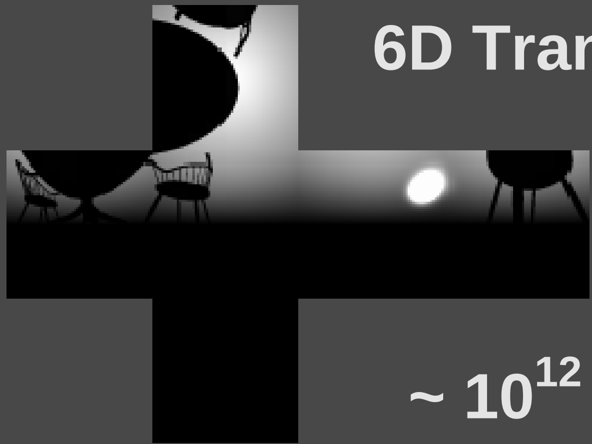
- **Distant Lighting (2D)**
- **Surface (2D)**
- **View (2D)**



With ~ 100 samples per dimension

- **Total of ~ 10^{12} samples**

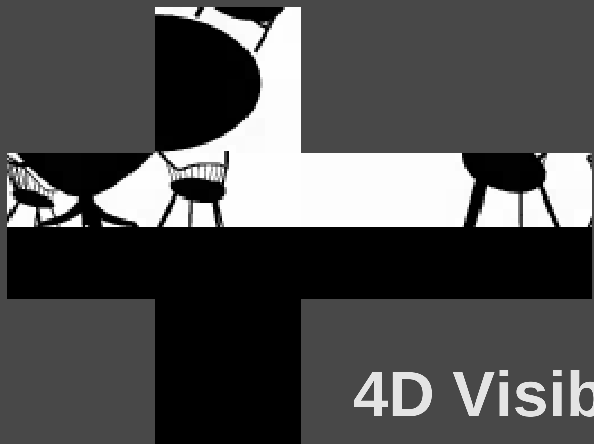
Factorization Approach



6D Transport

=

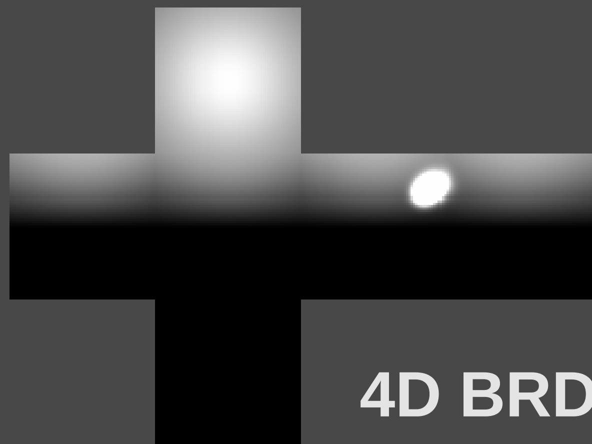
~ 10^{12} samples



4D Visibility

~ 10^{8} samples

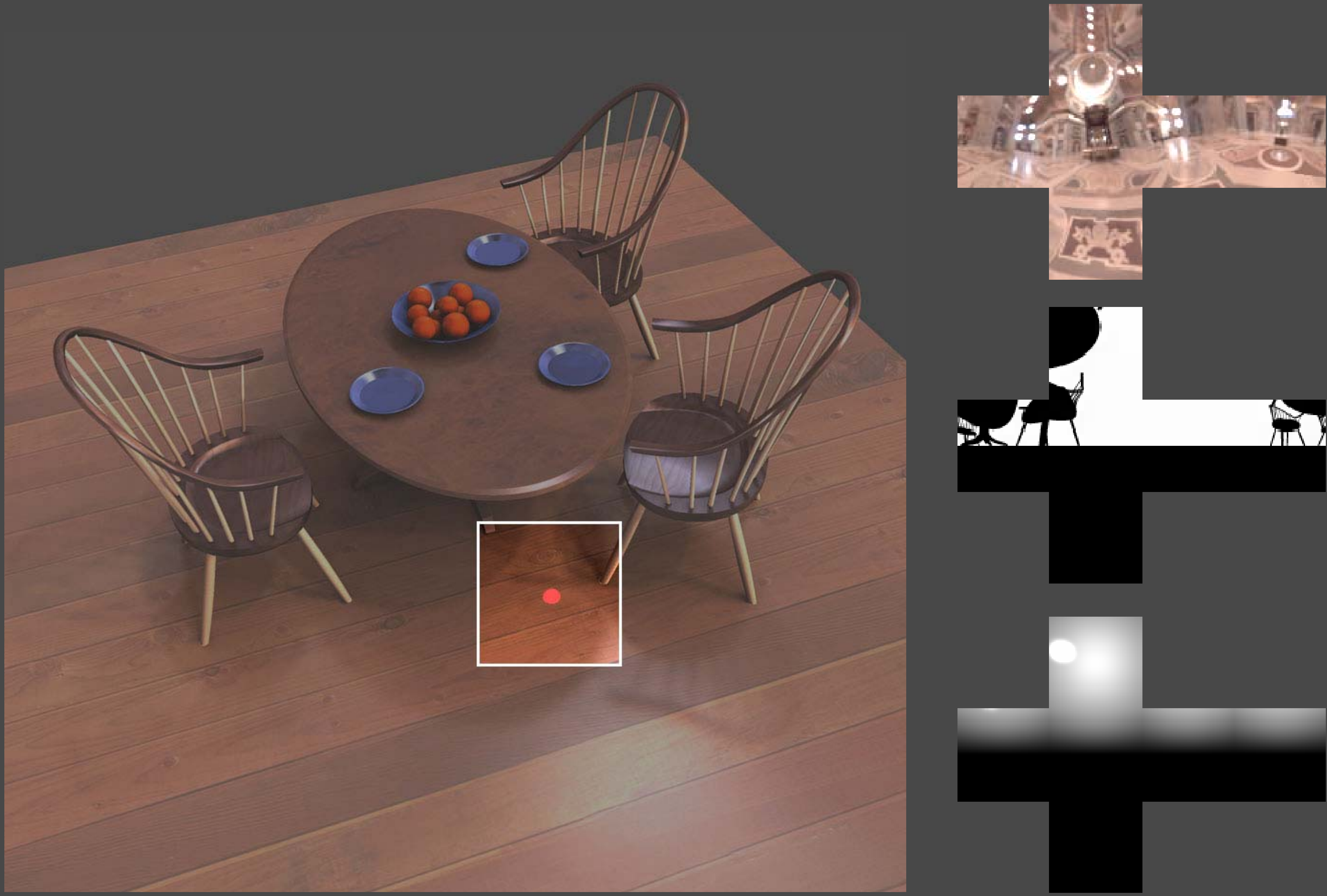
*



4D BRDF

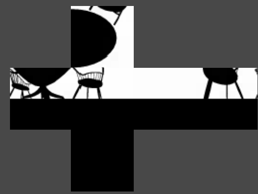
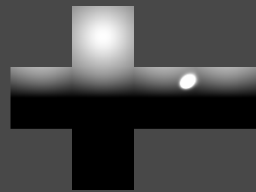
~ 10^{8} samples

Triple Product Integral Relighting



Triple Product Integrals

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

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

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

$$= \sum_i \sum_j \sum_k L_i\, V_j\, \tilde{\rho}_k\, C_{ijk}$$

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

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

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

Basis Requirements

$$B = \sum_i \sum_j \sum_k L_i \, V_j \, \tilde{\rho}_k \, C_{ijk}$$

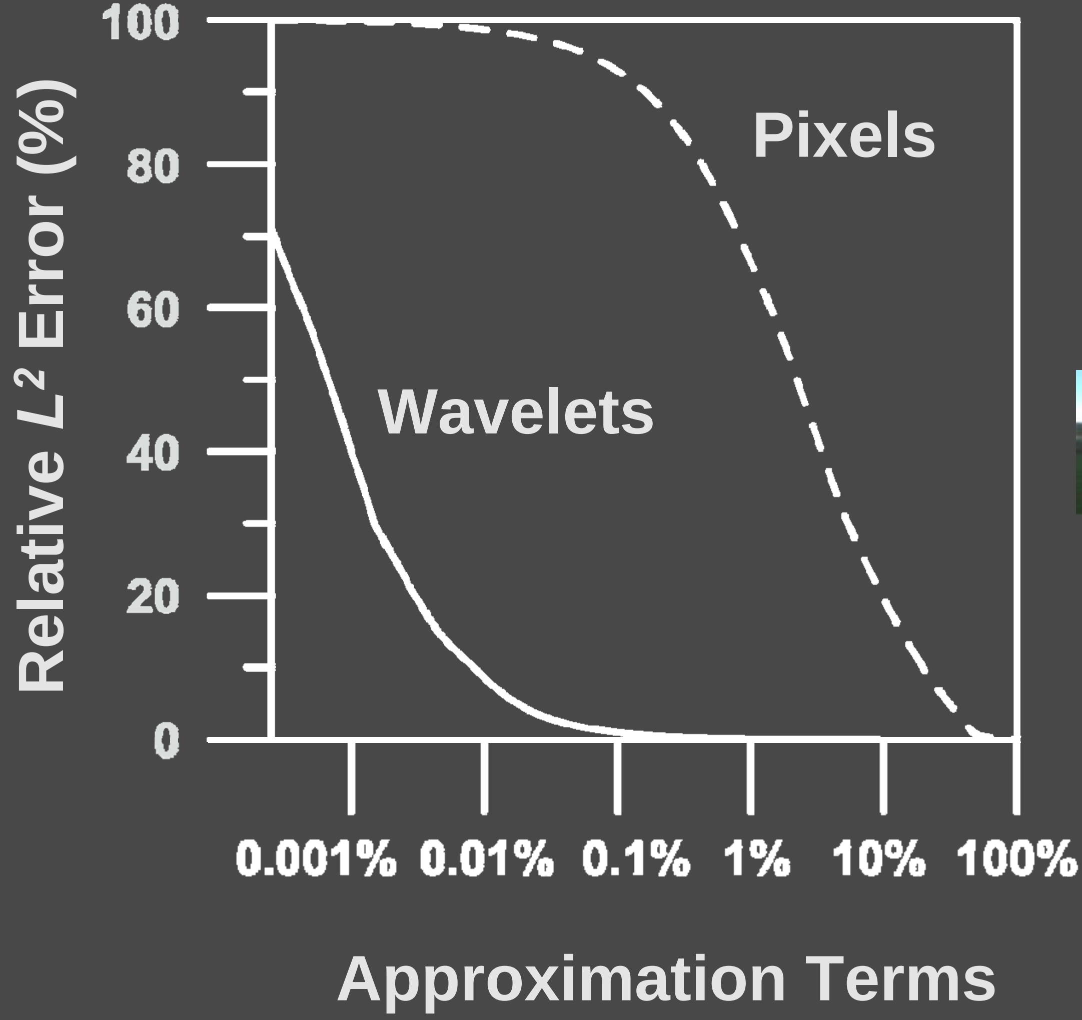
1. **Need few non-zero "tripling" coefficients**

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

2. **Need sparse basis coefficients**

$$L_i, \; V_j, \; \tilde{\rho}_k$$

2. Sparsity in Light Approximation



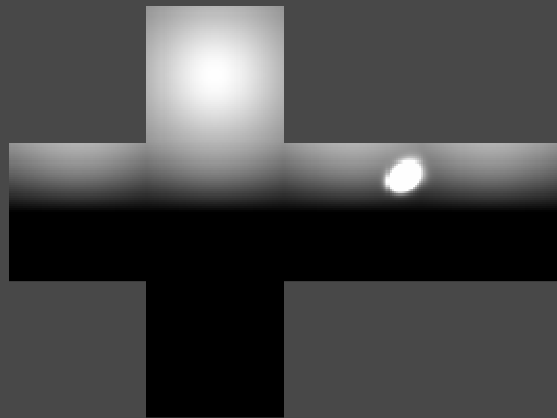
2. Sparsity in Visibility and BRDF

Visibility

- ~ 40% sparse in pixels
- 1 – 5 % sparse in wavelets

BRDF with diffuse component

- ~ 50% sparse in pixels
- 0.1 – 1% sparse in wavelets



Summary of Basis Analysis

Choose wavelet basis because

1. **Few non-zero tripling coefficients**

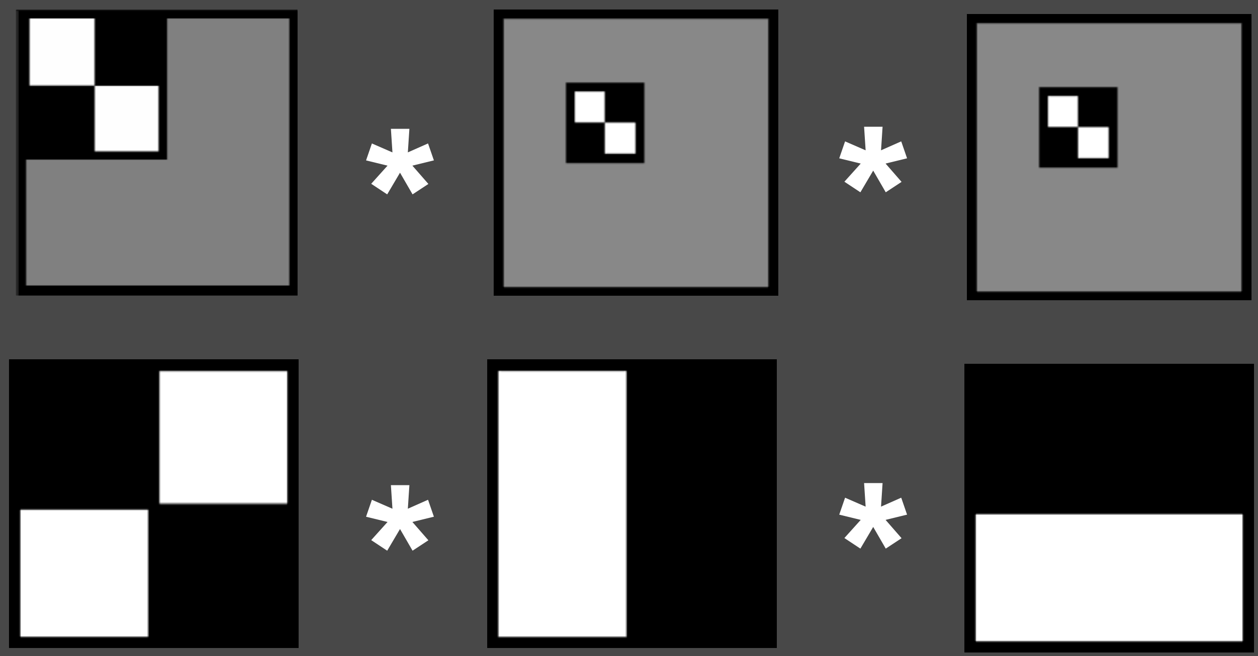
$$C_{ijk} = \int_{S^2} \Psi_i(\omega)\Psi_j(\omega)\Psi_k(\omega)\, d\omega$$

2. **Sparse basis coefficients**

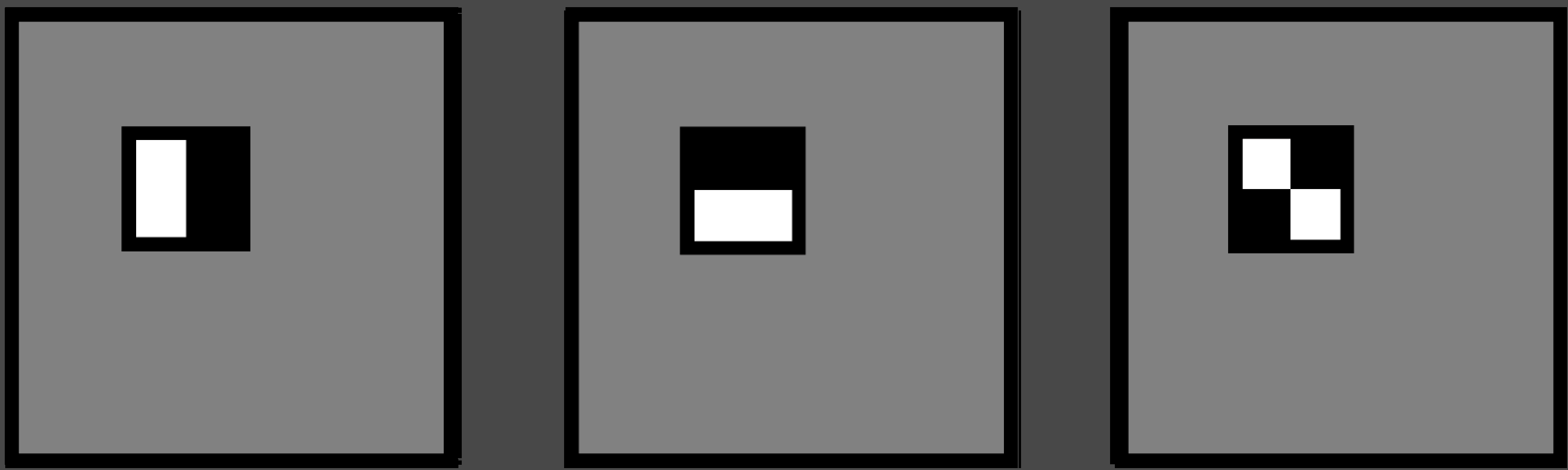
$$L_i,\ V_j,\ \tilde{\rho}_k$$

Haar Tripling Coefficient Analysis

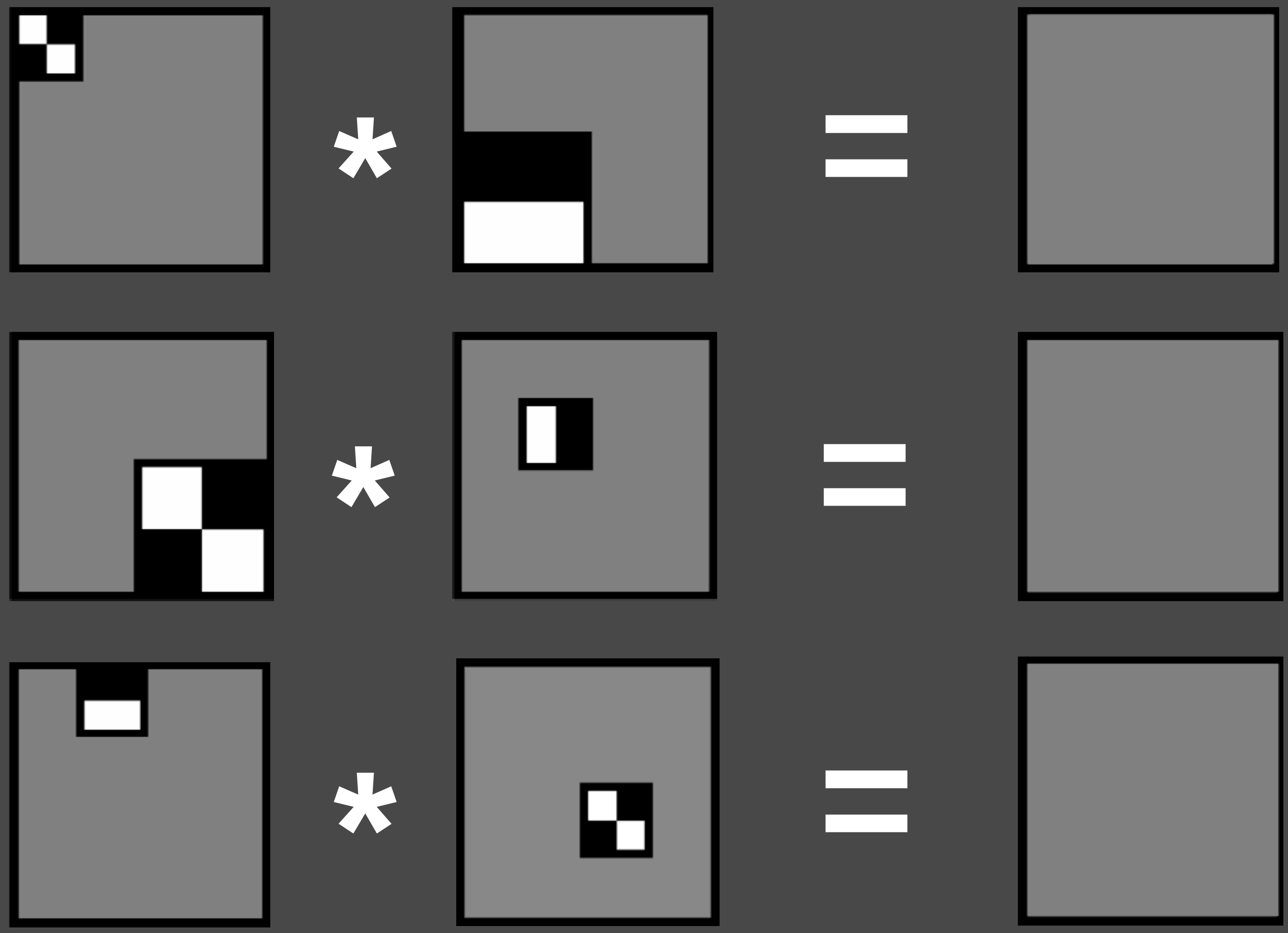
$$C_{ijk} = \int_{S^2} \Psi_i(\omega)\Psi_j(\omega)\Psi_k(\omega)\, d\omega$$



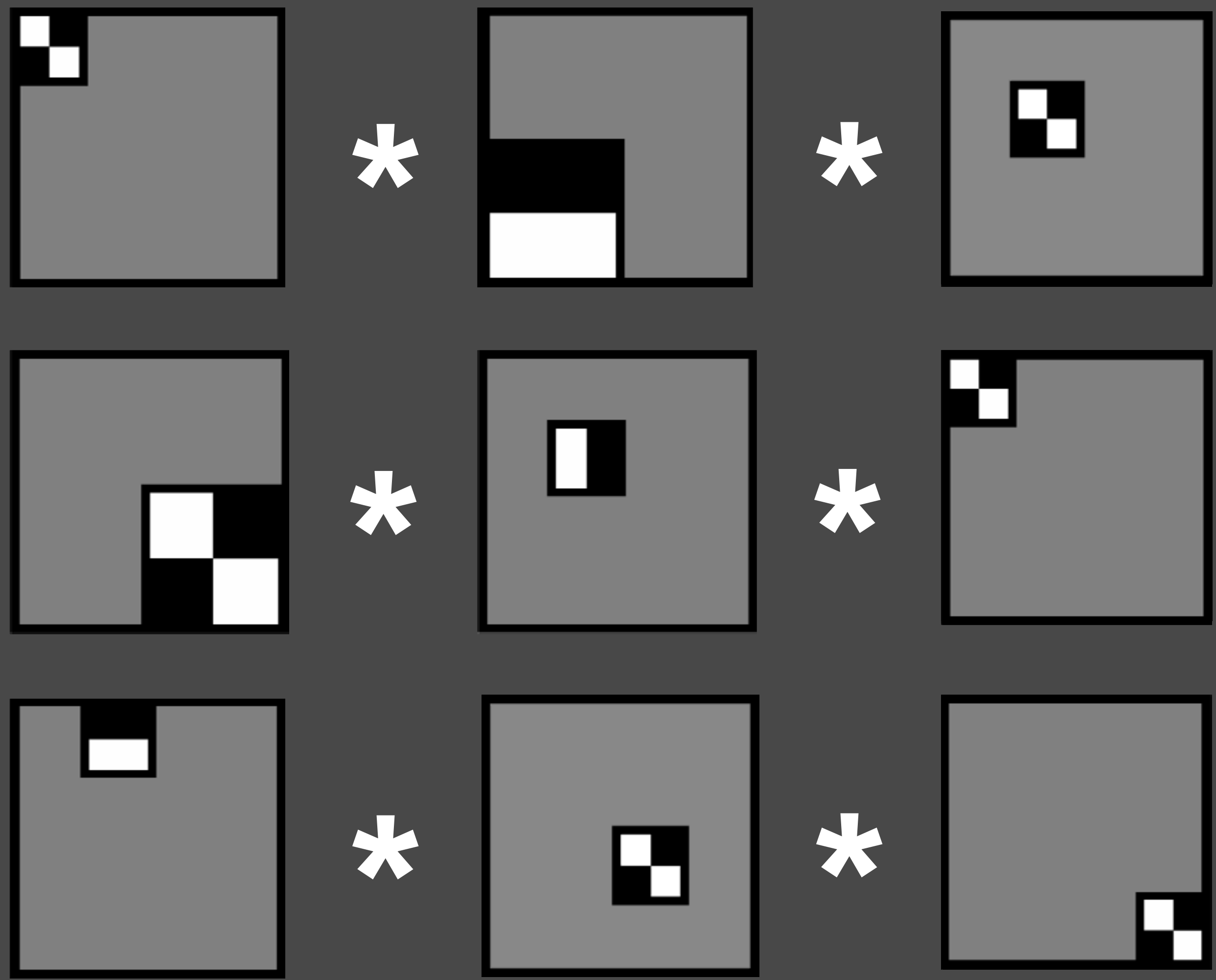
Visual Review of 2D Haar Basis



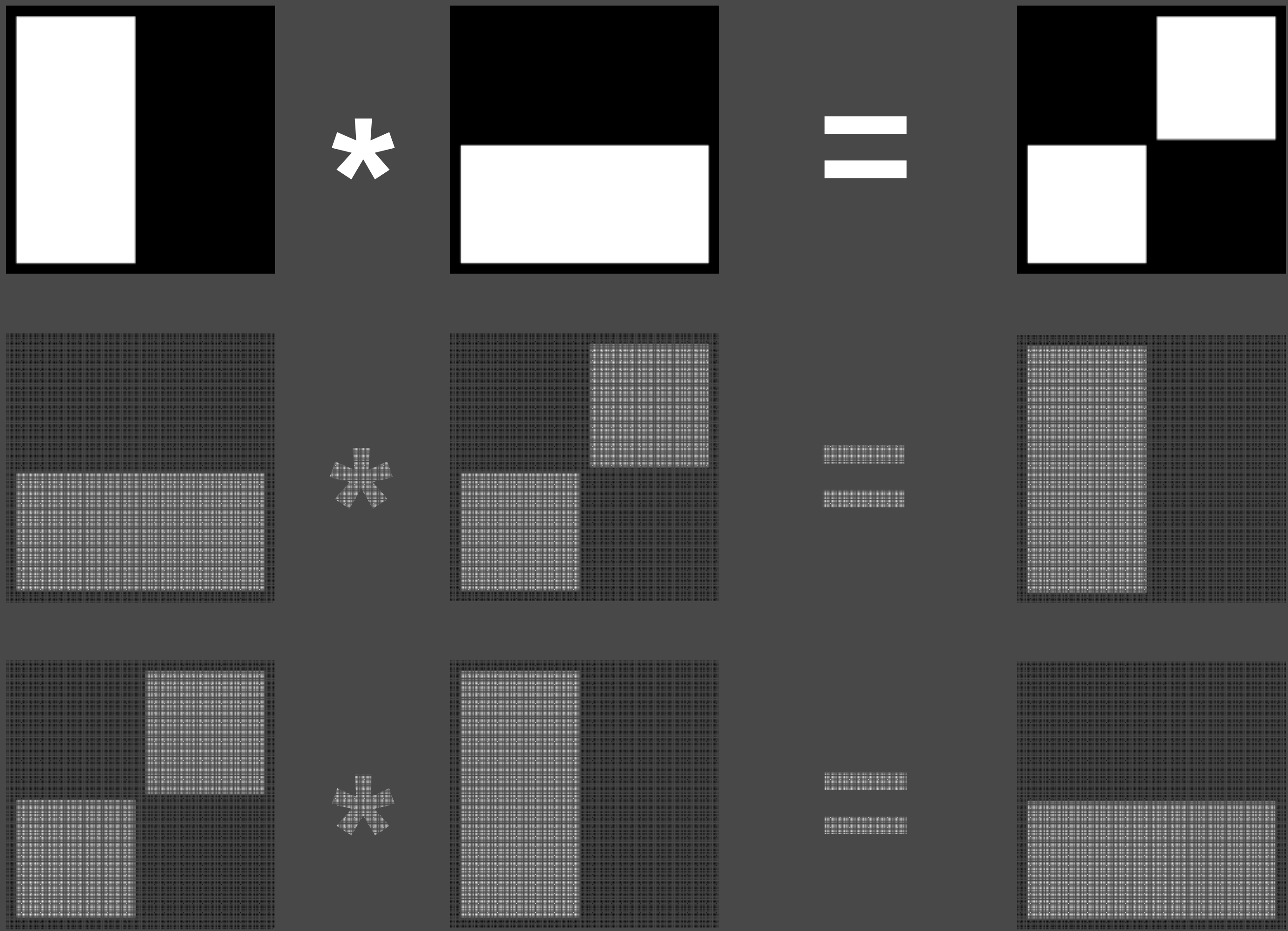
1. Non-Overlapping Haar Multiplication



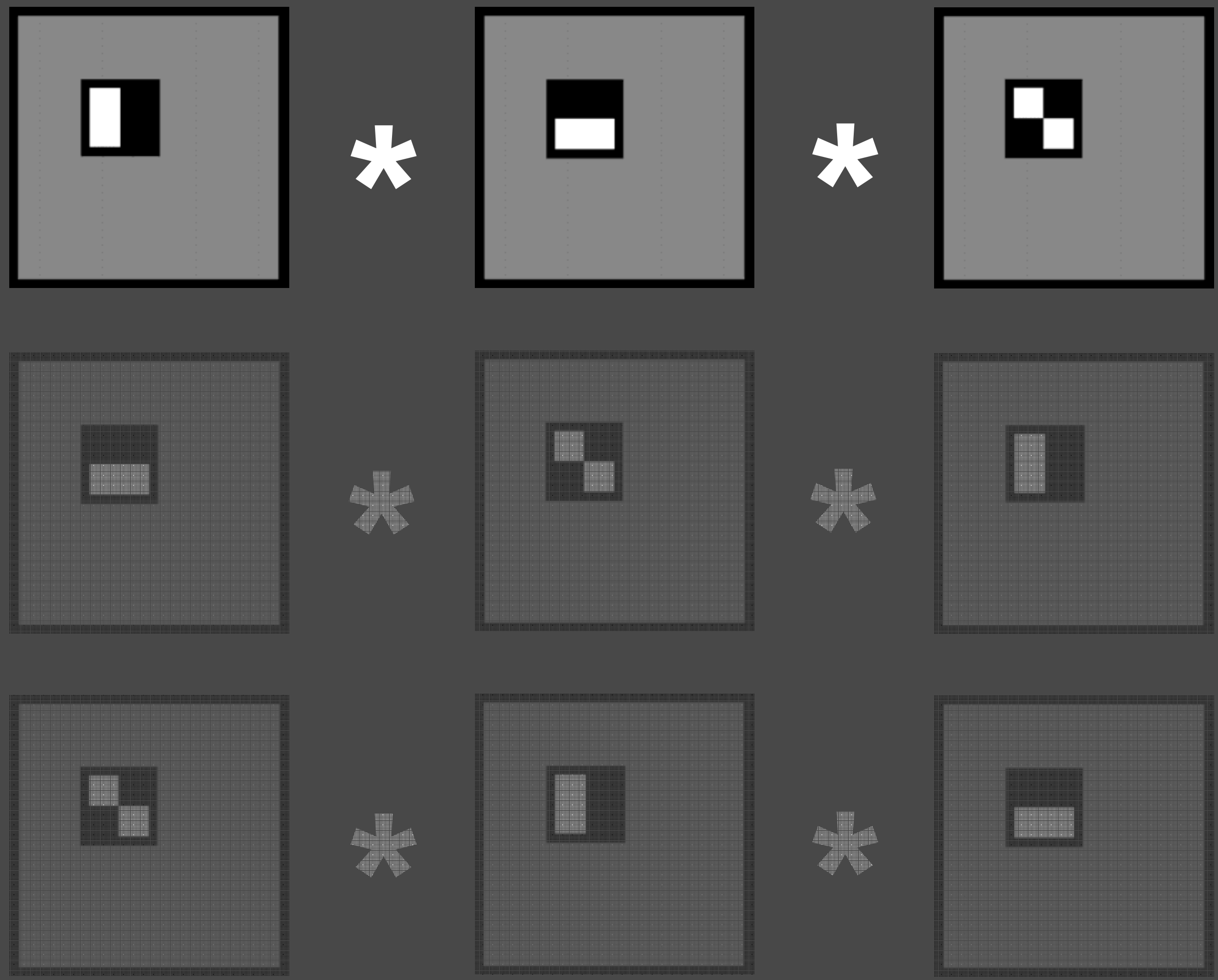
1. Zero Triple Product Integral



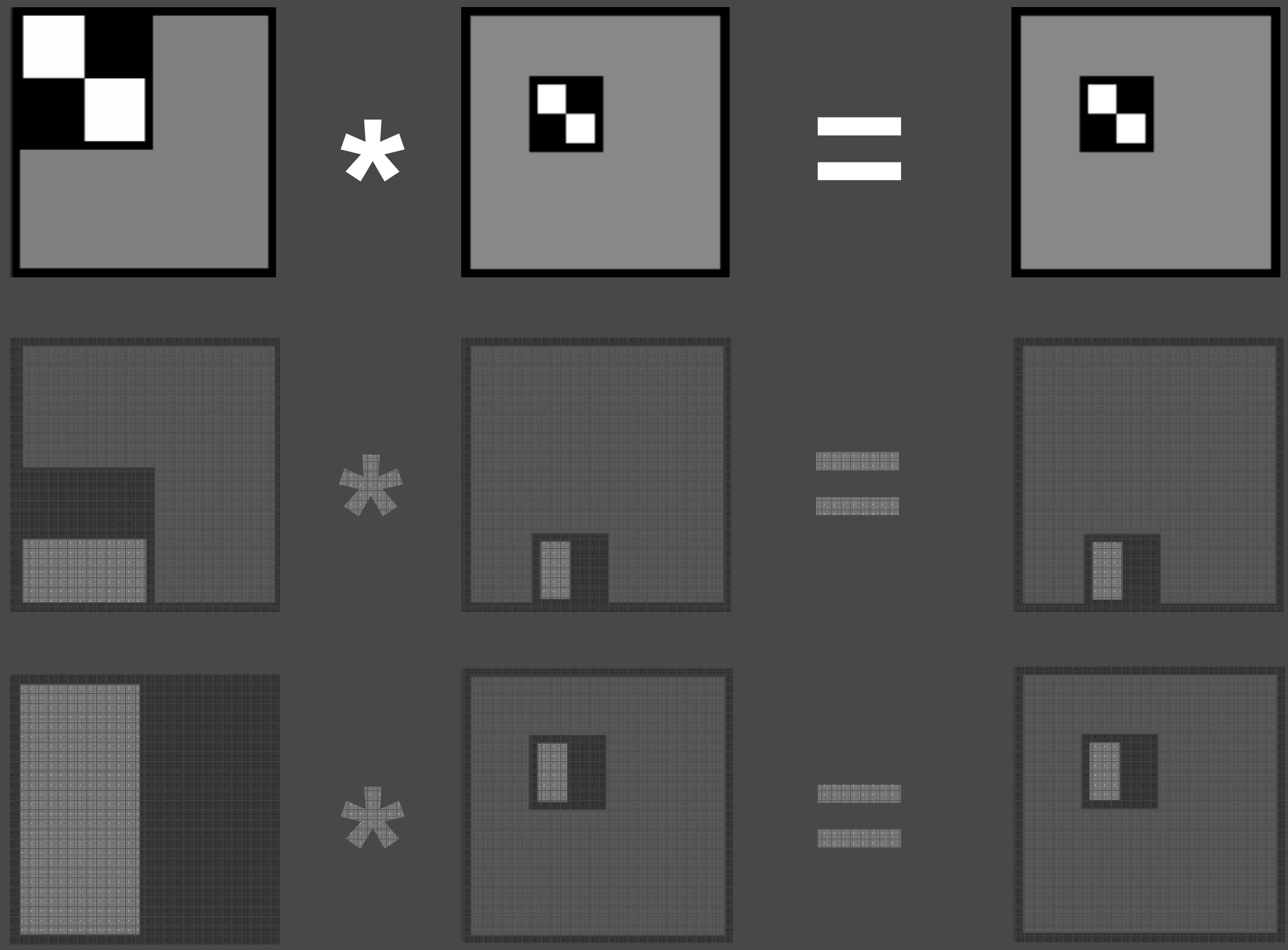
2. Co-Square Haar Multiplication



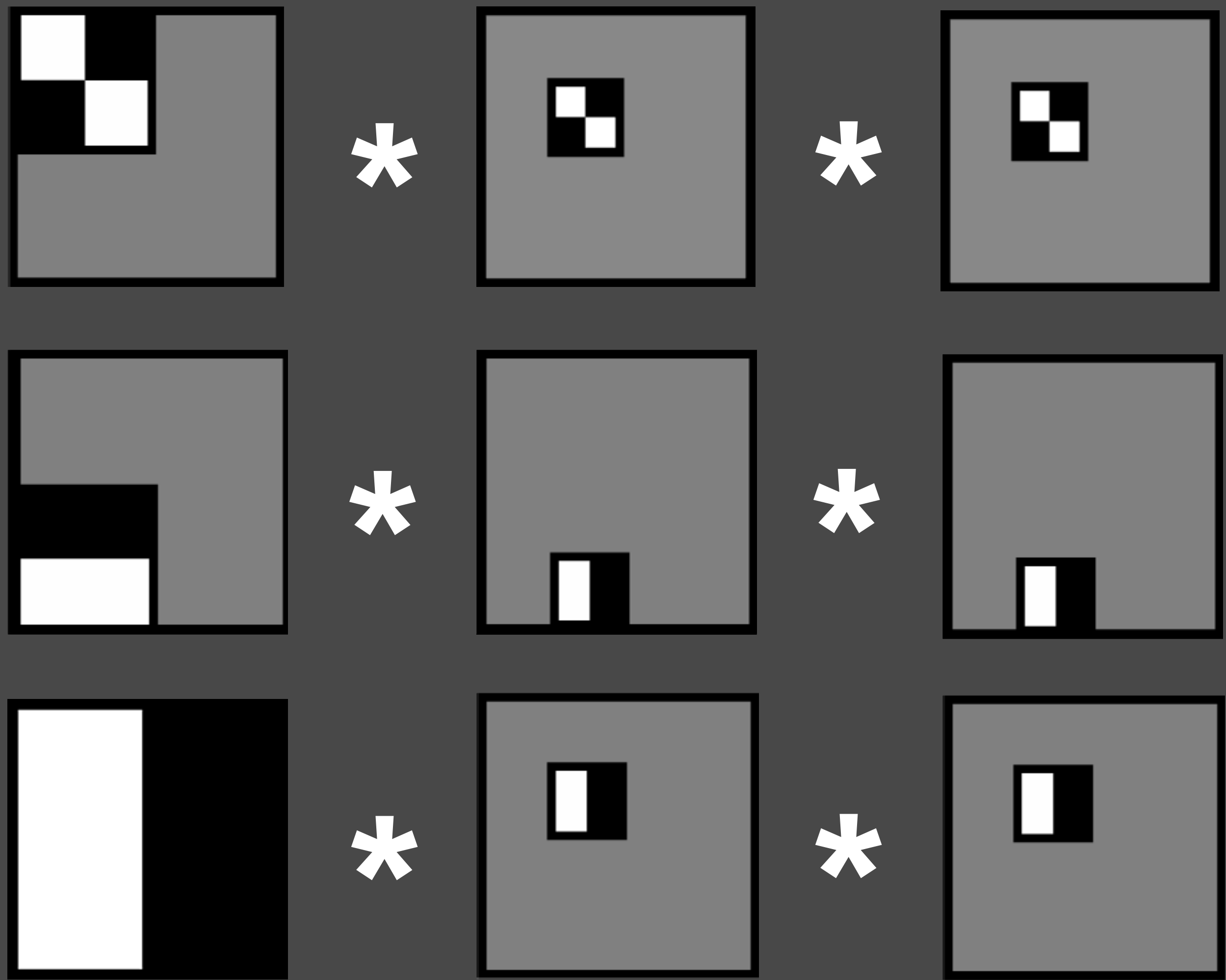
2. Non-Zero Triple Product Integrals



3. Overlapping Haar Multiplication



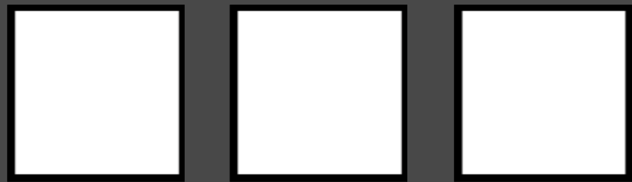
3. More Non-Zero Triple Integrals



Haar Tripling Coefficient Theorem

The integral of three Haar wavelets is non-zero iff

- All three are the scaling function



- All three are co-square and different

- Two are identical, and the third overlaps at a coarser level

Theorem Consequences

1. Prove O(*N* log *N*) Haar sparsity

2. Derive O(*N* log *N*) triple product integral algorithm
   - Dynamic programming eliminates log *N* term
   - Final complexity is linear in number of retained basis coefficients

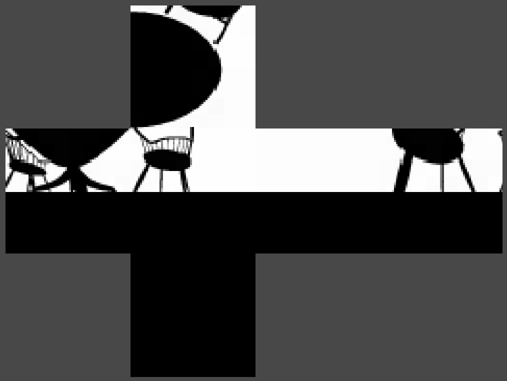
High-Level Algorithm: Precomputation

for each vertex

compute the visibility cubemap

wavelet encode

store

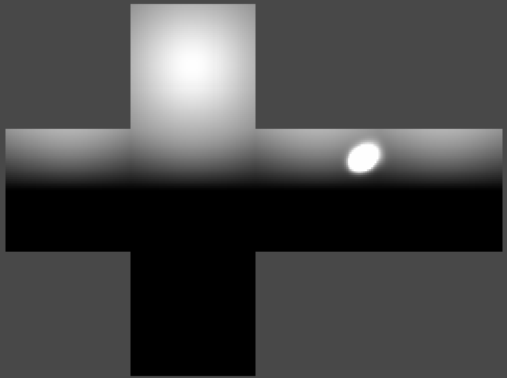


for each viewing direction

compute BRDF for that view

nonlinear wavelet approximation

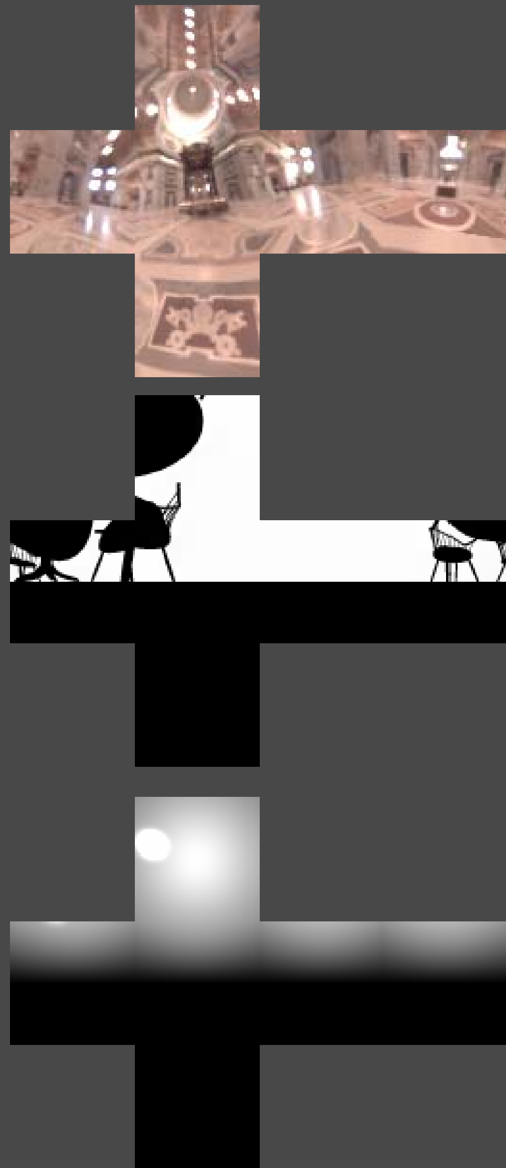
store



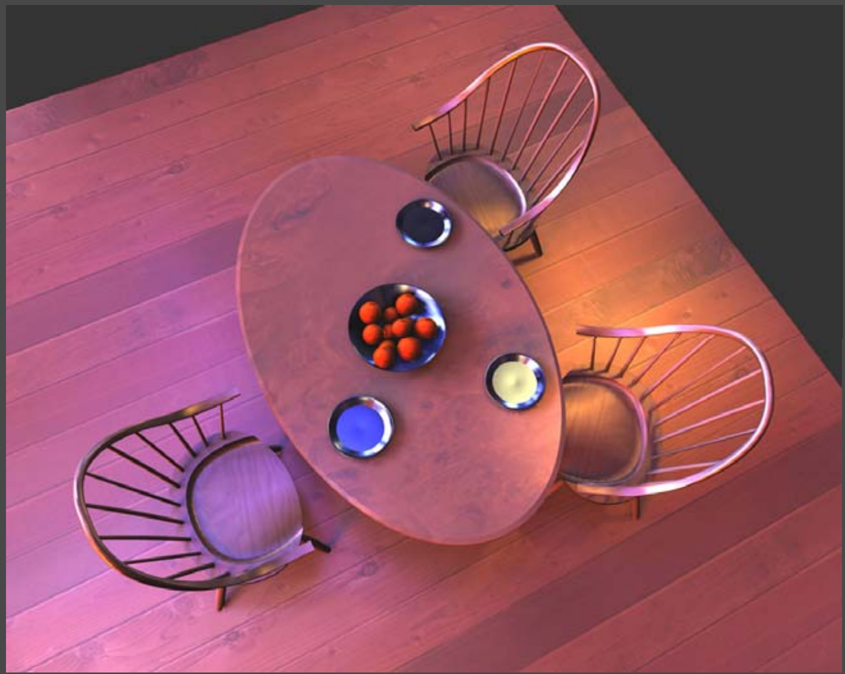
High-Level Algorithm: Relighting

for each frame

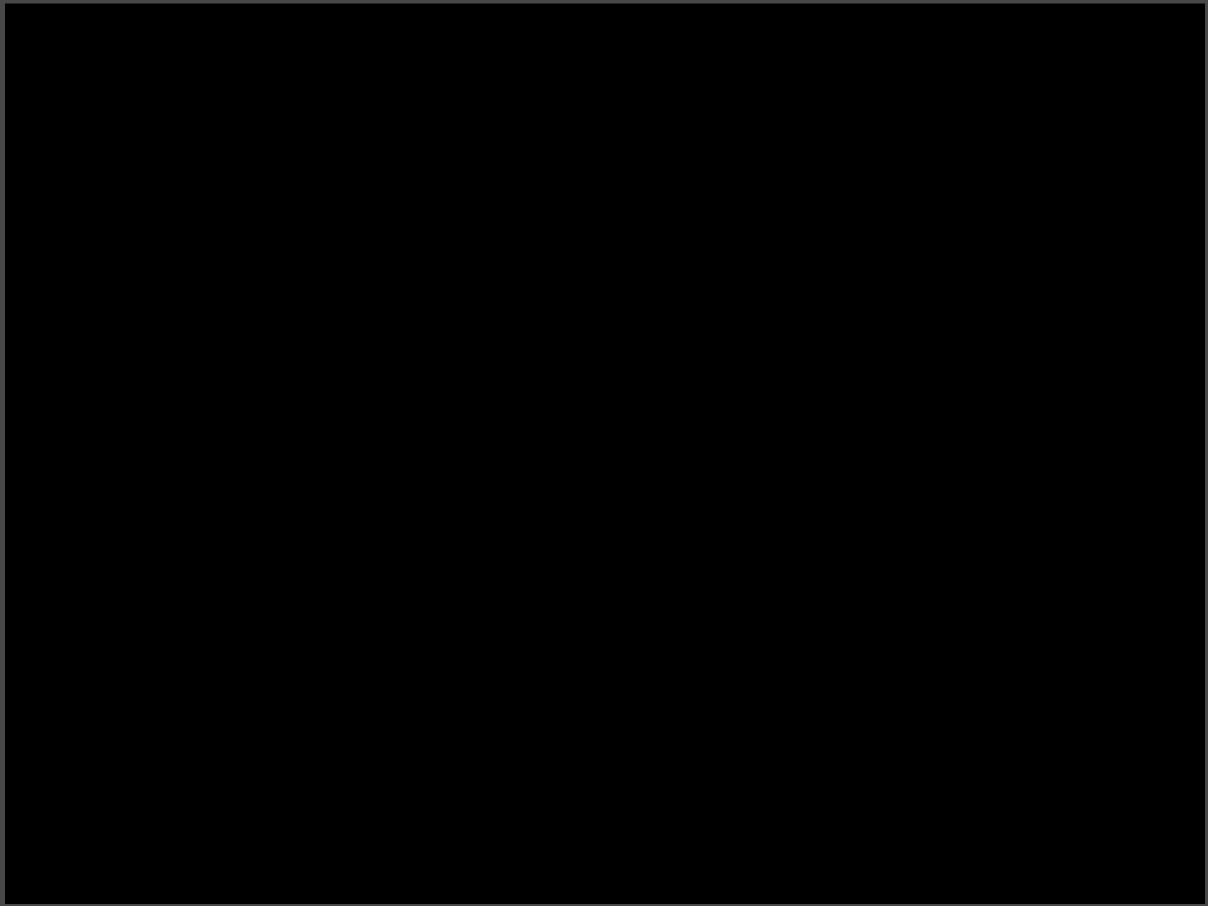
- wavelet encode lighting, *L*
- for each vertex in mesh
  - look up visibility, *V*
  - look up BRDF for view, *ρ*
  - integrate product of *L*, *V*, *ρ*
  - set color of vertex
- draw colored mesh



Relit Images

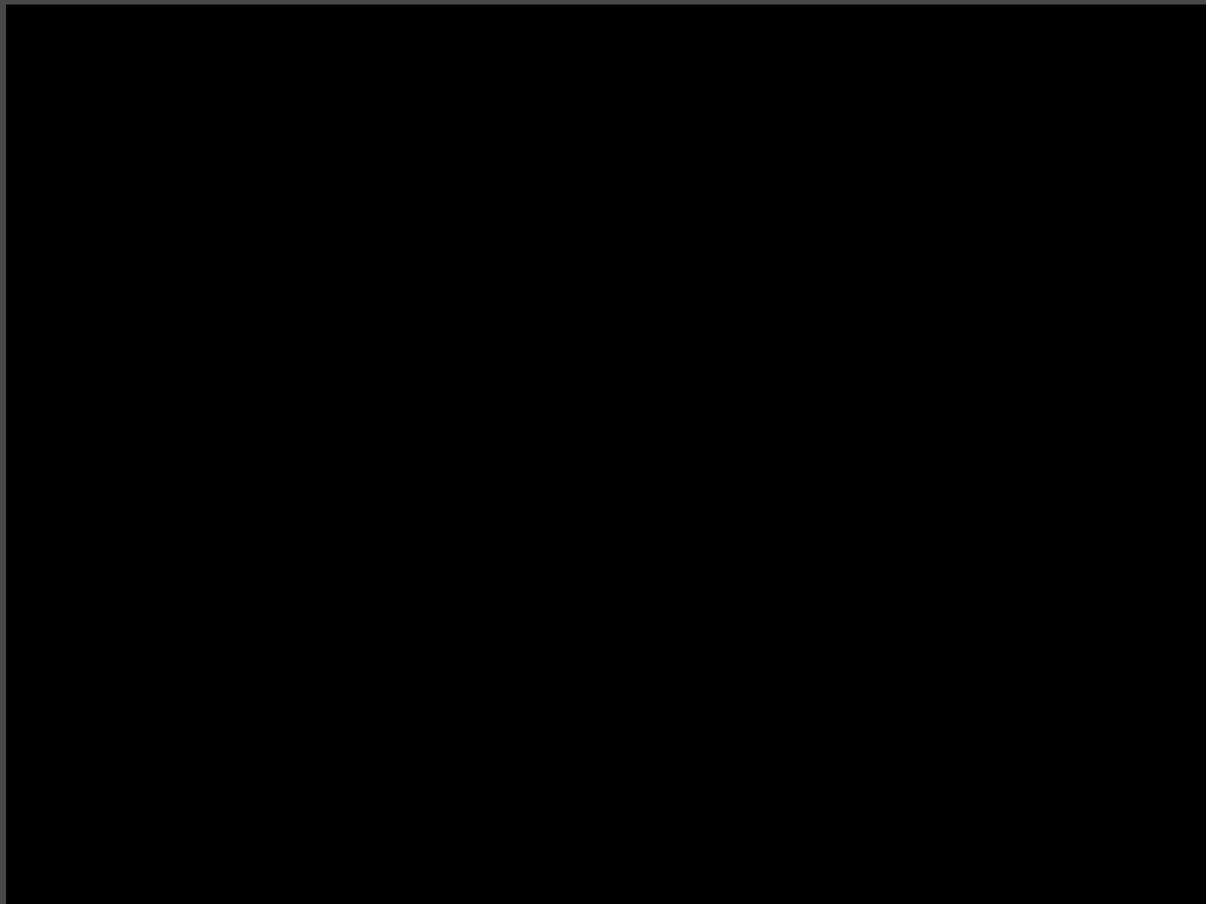


Changing Lighting, Fixed View



Changing View, Fixed Lighting

Dynamic Lighting and View



Results

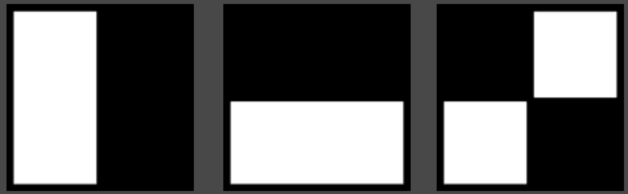
SIGGRAPH 2004 video

Summary

All-frequency relighting, changing view

- Factor into visibility and BRDF
- Fast relight in Haar basis

Triple product analysis and algorithms



- Analysis of several bases
- Haar tripling theorem
- Efficient triple product integral estimation

Efficient Wavelet Rotation for Environment Map Rendering

Rui Wang   Ren Ng   David Luebke   Greg Humphreys

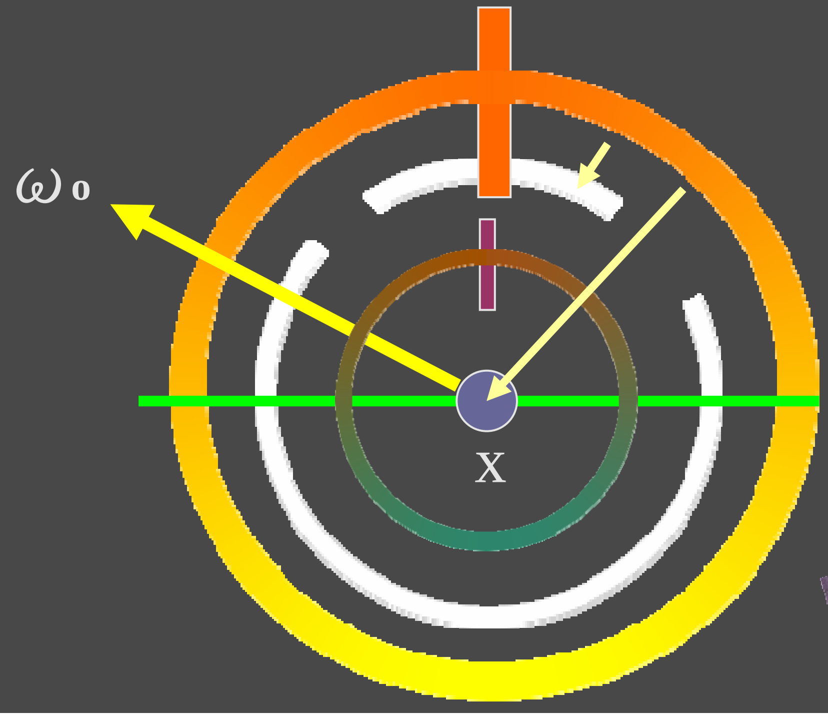
EGSR 2006

Triple product

[illegible]

[illegible]

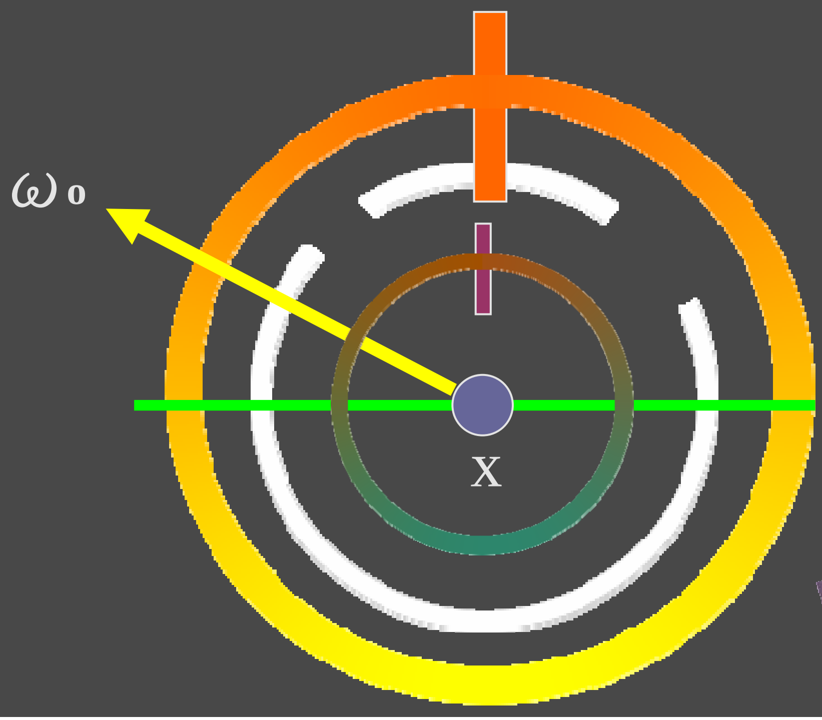
light direction ω to view direction ω_o.

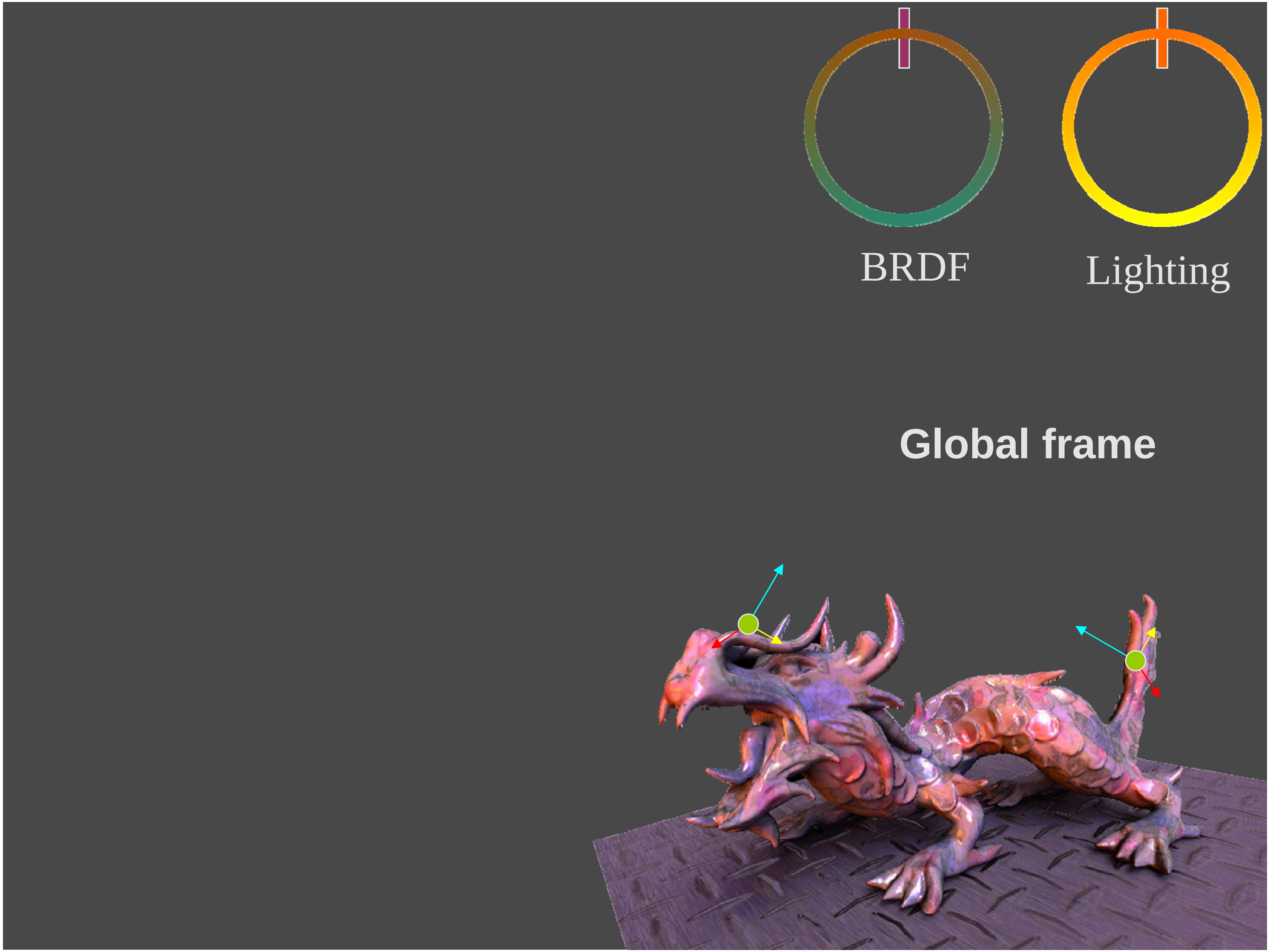


Triple product

$$B^{x,\omega_0} = \int_\Omega L^x(\omega) V^x(\omega) \tilde{\rho}^{x,\omega_0}(\omega) d\omega$$

The output radiance is the integral of these functions over all directions!

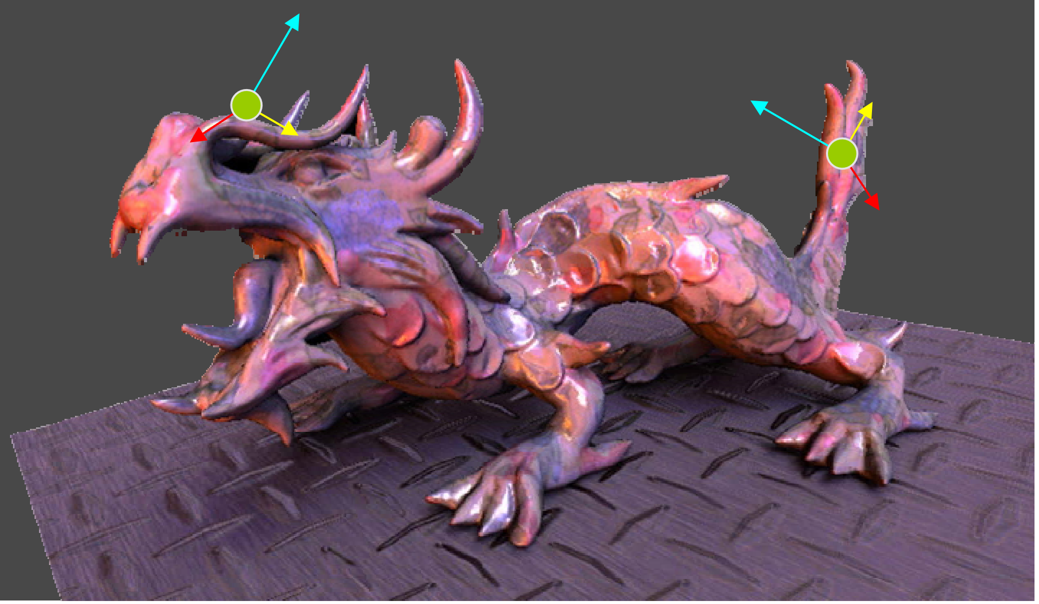



BRDF
Lighting
Global frame

BRDF

Lighting

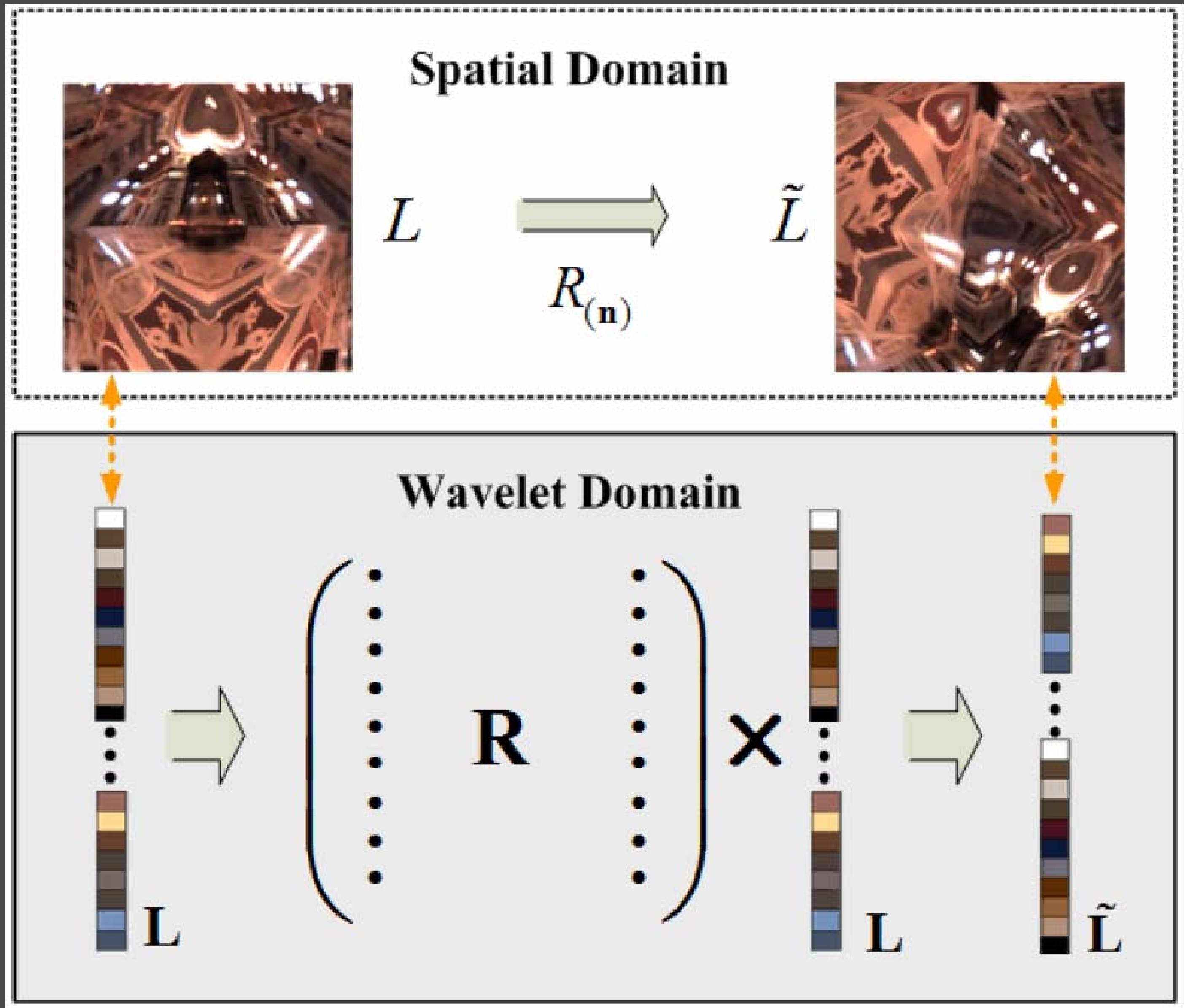
Local frame



Rotation

- Because we can't rotate wavelet basis easily, some functions must be sampled with different rotations in advance and use lookup at runtime.
- Spherical harmonics does not have this problem because it can be rotated easily.

Rotation of spherical functions



Rotation of spherical basis

source basis

$$L(R \cdot \omega) = \sum_i L_i \Psi_i (R \cdot \omega)$$

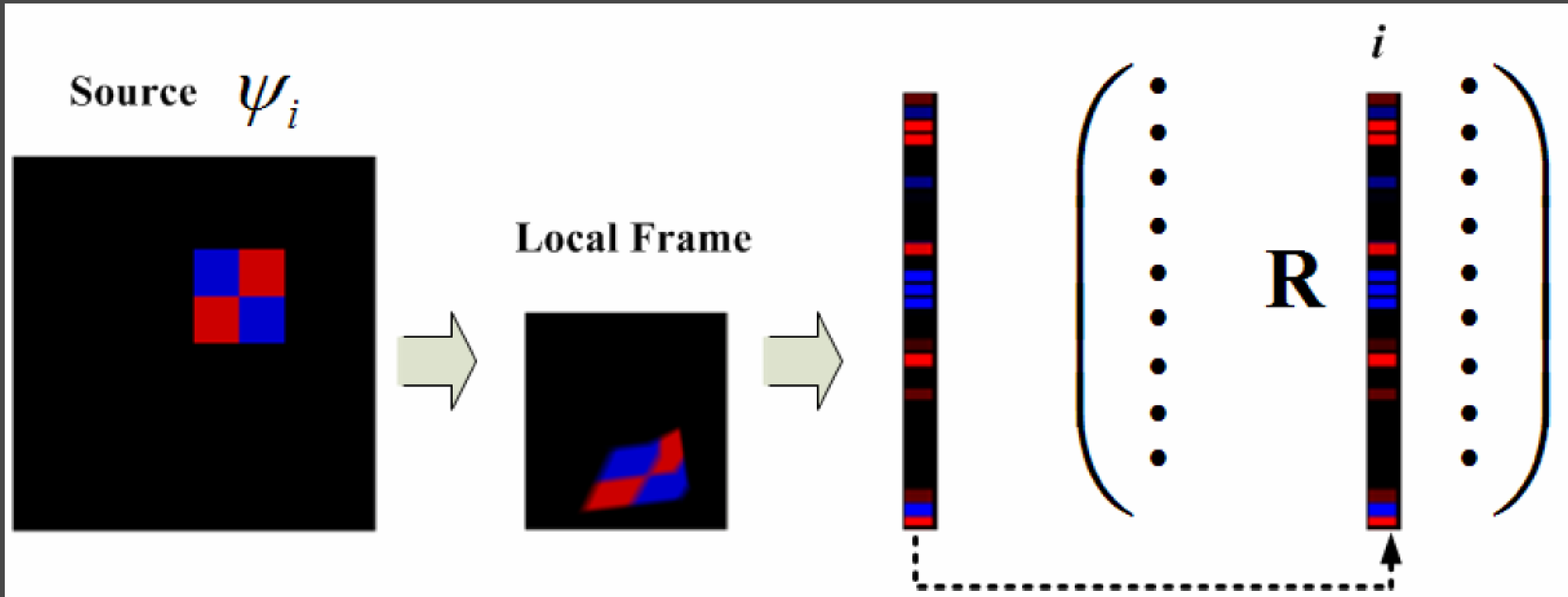
target basis

$$\Psi_i (R \cdot \omega) = \sum_j R_{ij} \varphi_j (\omega)$$

Basis transform matrix

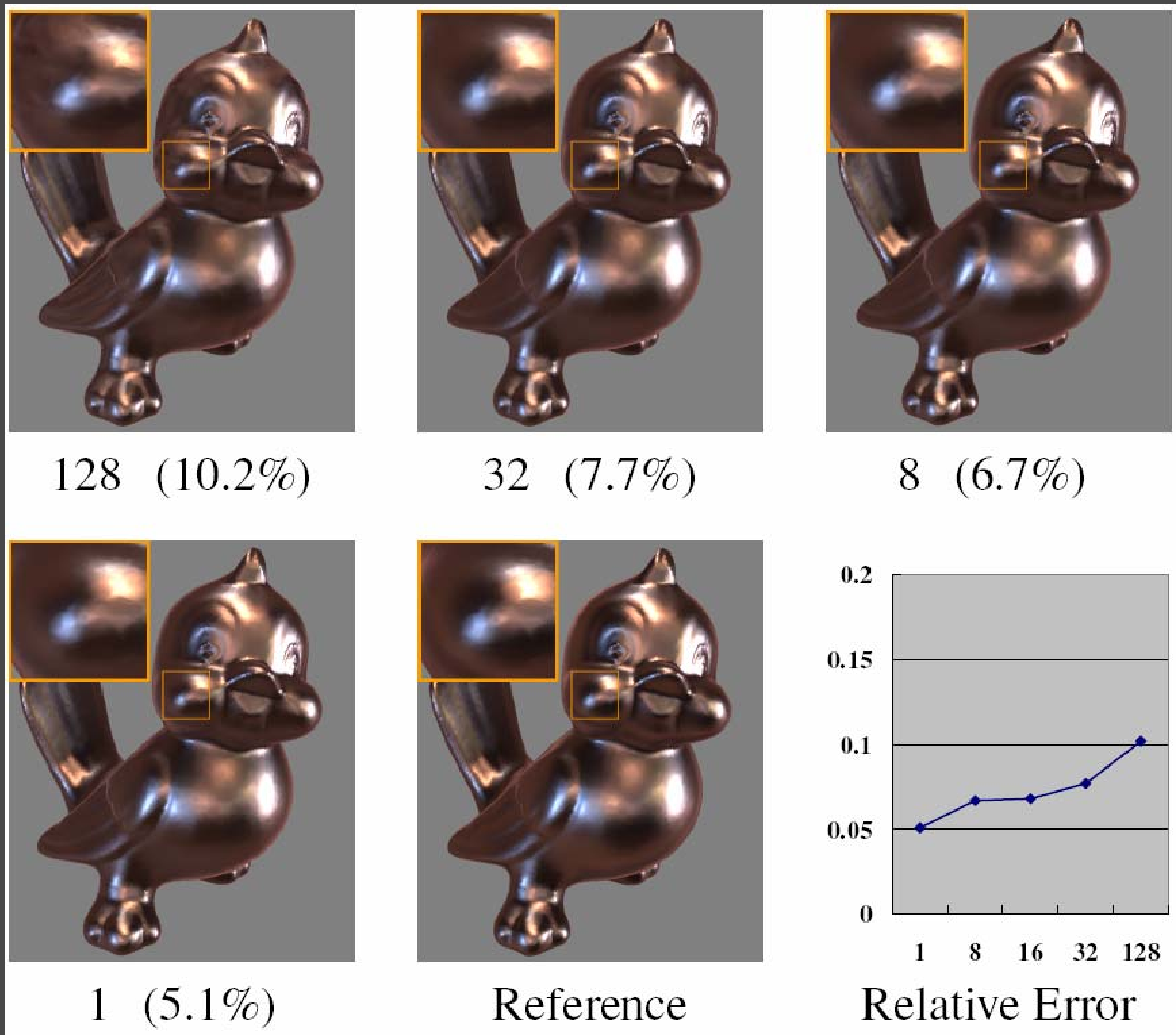
$$L(R \cdot \omega) = \sum_i L_i \sum_j R_{ij} \varphi_j (\omega) = \sum_j \left(\sum_i R_{ij} L_i \right) \varphi_j (\omega)$$

Basis transform matrix



- Transform matrices have to be pre-computed at several rotations

Varying truncation threshold



Varying rotation sampling rate