Advanced Computer Graphics (Fall 2009)

CS 294, Rendering Lecture 10

Precomputation-Based Real-Time Rendering

Ravi Ramamoorthi

http://inst.eecs.berkeley.edu/~cs294-13/fa09

Motivation

- Previously: seen IBR. Use measured data (real photographs) and interpolate for realistic real-time
- Why not apply to real-time rendering?
  - Precompute (offline) some information (images) of interest
  - Must assume something about scene is constant to do so
  - Thereafter real-time rendering. Often accelerate hardware
- Easier and harder than conventional IBR
  - Easier because synthetic scenes give info re geometry, reflectance (but CG rendering often longer than nature)
  - Harder because of more complex effects (lighting from all directions for instance, not just changing view)
- Representations and Signal-Processing crucial

My General Philosophy

- This general line of work is a large data management and signal-processing problem
- Precompute high-dimensional complex data
- Store efficiently (find right mathematical represent.)
- Render in real-time
  - Worry about systems issues like caching
  - Good signal-processing: use only small amount of data but guarantee high fidelity
- Many insights into structure of lighting, BRDFs, …
  - Not just blind interpolation and IBR, signal processing

Precomputation-Based Relighting

- Analyze precomputed images of scene

Jensen 2000

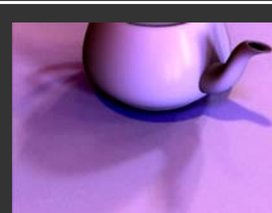
Precomputation-Based Relighting

- Analyze precomputed images of scene

Jensen 2000

Assumptions

- Static geometry
- Precomputation



- Real-Time Rendering (relight all-frequency effects)
  - Exploit linearity of light transport for this
  - Later, change viewpoint as well

Why is This Hard?

- Plain graphics hardware supports only simple (point) lights, BRDFs (Phong) without any shadows
- Shadow maps can handle point lights (hard shadows)
- Environment maps complex lighting, BRDFs but no shadows
- IBR can often do changing view, fixed lighting

- How to do complex shadows in complex lighting?
- With dynamically changing illumination and view?

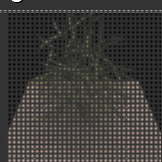
Relighting as a Matrix-Vector Multiply

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

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

Relighting as a Matrix-Vector Multiply

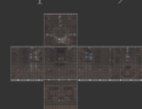
$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \vdots \\ P_N \end{bmatrix}$$



Output Image
(Pixel Vector)

Input Lighting
(Cubemap Vector)

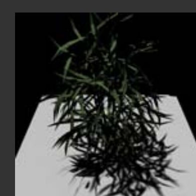
$$= \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1M} \\ T_{21} & T_{22} & \cdots & T_{2M} \\ T_{31} & T_{32} & \cdots & T_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ T_{N1} & T_{N2} & \cdots & T_{NM} \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_M \end{bmatrix}$$



Precomputed
Transport
Matrix

Matrix Columns (Images)

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

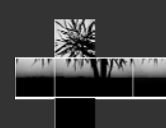


Precompute: Ray-Trace Image Cols

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

Precompute 2: Rasterize Matrix Rows

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



Problem Definition

Matrix is Enormous
- 512 x 512 pixel images
- 6 x 64 x 64 cubemap environments

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

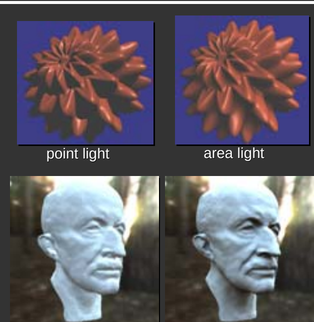
How to relight quickly?

Outline

- Motivation and Background
- *Compression methods*
  - *Low frequency linear spherical harmonic approximation*
  - Factorization and PCA
  - Local factorization and clustered PCA
  - Non-linear wavelet approximation
- Changing view as well as lighting
  - Clustered PCA
  - Factored BRDFs
  - Triple Product Integrals

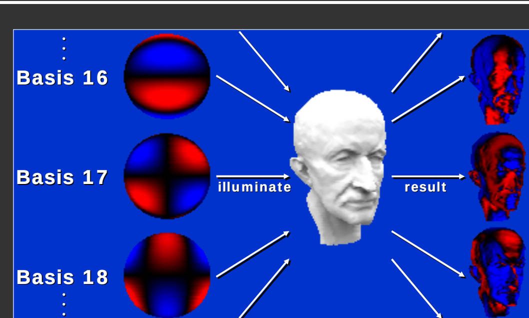
Precomputed Radiance Transfer

- Better light integration and transport
  - dynamic, area lights
  - self-shadowing
  - interreflections
- For diffuse and glossy surfaces
- At real-time rates
- Sloan et al. 02



point light, area light, area lighting, no shadows, area lighting, shadows

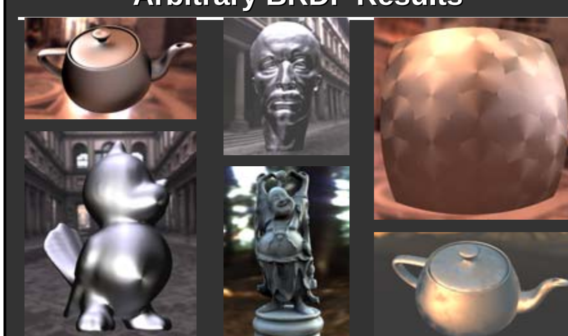
Precomputation: Spherical Harmonics



Diffuse Transfer Results

No Shadows/Inter Shadows Shadows+Inter

Arbitrary BRDF Results



Anisotropic BRDFs Other BRDFs Spatially Varying

Relighting as a Matrix-Vector Multiply

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

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

Idea of Compression

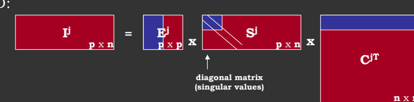
- The vector is projected onto low-frequency components (say 25). Size greatly reduced.
- Hence, only 25 matrix columns
- But each pixel still treated separately (still have 300000 matrix rows for 512 x 512 image)
- Actually, for each pixel, dot product of matrix row (25 elems) and lighting vector (25 elems) in hardware
- Good technique (becoming common in games) but useful only for broad low-frequency lighting

Outline

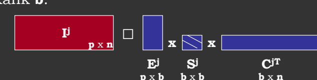
- Motivation and Background
- *Compression methods*
  - Low frequency linear spherical harmonic approximation
  - *Factorization and PCA*
  - *Local factorization and clustered PCA*
  - Non-linear wavelet approximation
- Changing view as well as lighting
  - Clustered PCA
  - Factored BRDFs
  - Triple Product Integrals

PCA or SVD factorization

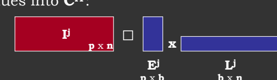
- SVD:



- Applying Rank **b**:



- Absorbing **S^j** values into **C^jT**:



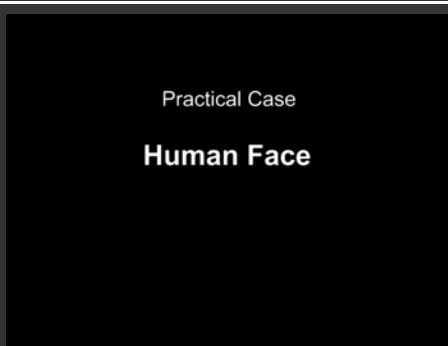
Idea of Compression

- Represent matrix (rather than light vector) compactly
- Can be (and is) combined with low frequency vector
- Useful in broad contexts.
  - BRDF factorization for real-time rendering (reduce 4D BRDF to 2D texture maps) McCool et al. 01 etc
  - Surface Light field factorization for real-time rendering (4D to 2D maps) Chen et al. 02, Nishino et al. 01
  - Factorization of Orientation Light field for complex lighting and BRDFs (4D to 2D) Latta et al. 02
- **Not too useful for general precomput. relighting**
  - **Transport matrix not low-dimensional!!**

Local or Clustered PCA

- Exploit local coherence (in say 16x16 pixel blocks)
  - Idea: light transport is locally low-dimensional. Why?
  - Even though globally complex
  - See Mahajan et al. 07 for theoretical analysis
- Original idea: Each triangle separately
  - Example: Surface Light Fields 3D subspace works well
  - Vague analysis of size of triangles
  - Instead of triangle, 16x16 image blocks [Nayar et al. 04]
- Clustered PCA [Sloan et al. 2003]
  - Combines two widely used compression techniques: Vector Quantization or VQ and Principal Component Analysis
  - For complex geometry, no need for parameterization / topology

Demo (block PCA relighting)



Outline

- Motivation and Background
- *Compression methods*
  - Low frequency linear spherical harmonic approximation
  - Factorization and PCA
  - Local factorization and clustered PCA
  - *Non-linear wavelet approximation*
- Changing view as well as lighting
  - Clustered PCA
  - Factored BRDFs
  - Triple Product Integrals

Sparse Matrix-Vector Multiplication

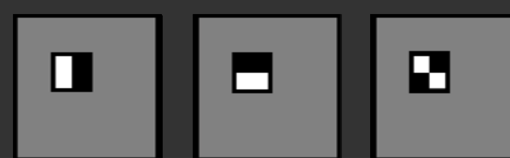
Choose data representations with mostly zeroes

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

Matrix: Wavelet-encode transport rows

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

Haar Wavelet Basis



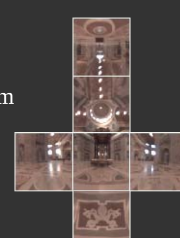
Non-linear Wavelet Approximation

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

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

Non-linear Wavelet Light Approximation

Wavelet Transform



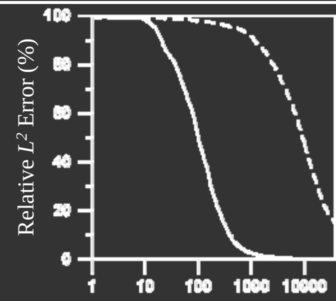
Non-linear Wavelet Light Approximation

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

Non-linear Approximation

Retain 0.1% – 1% terms

Error in Lighting: St Peter's Basilica



Sph. Harmonics

Non-linear Wavelets

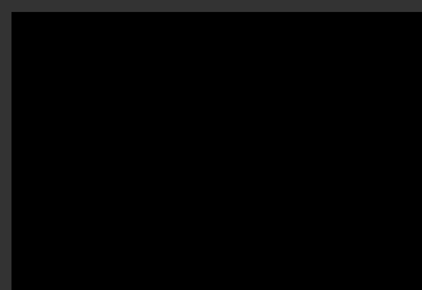
Ng, Ramamoorthi, Hanrahan 03

Output Image Comparison

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

25 200 2,000 20,000

Video



Outline

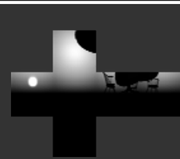
- Motivation and Background
- Compression methods
  - Low frequency linear spherical harmonic approximation
  - Factorization and PCA
  - Local factorization and clustered PCA
  - Non-linear wavelet approximation
- *Changing view as well as lighting*
  - *Clustered PCA*
  - *Factored BRDFs*
  - Triple Product Integrals

Changing Only The View

Problem Characterization

6D Precomputation Space

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



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

Clustered PCA

- Use low-frequency light and view variation (Order 4 spherical harmonic = 25 for both; total = 25*25=625)
- 625 element vector for each vertex
- Apply CPCA directly (Sloan et al. 2003)
- Does not easily scale to high frequencies
  - Really cubic complexity (number of vertices, illumination directions or harmonics, and view directions or harmonics)
- Practical real-time method on GPU

Factored BRDFs

- Sloan et al. 04, Wang et al. 04: All-frequency effects
- Combines lots of things: BRDF factorization, CPCA, nonlinear approx. with wavelets
- Idea: Factor BRDF to depend on incident, outgoing
  - Incident part handled with view-independent relighting
  - Then linearly combine based on outgoing factor
- Effectively, break problem into a few subproblems that can be solved view-independently and added up
  - Can apply nonlinear wavelet approx. to each subproblem
  - And CPCA to the matrices for further compression

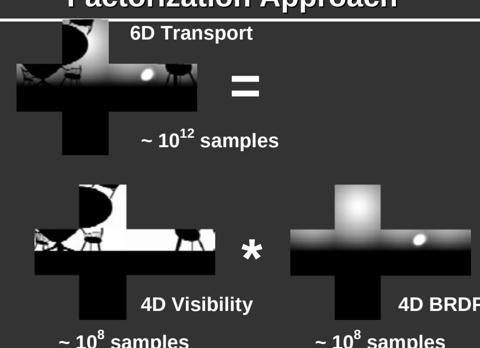
Factored BRDFs: Critique

- Simple, reasonably practical method
- Problem: Non-optimal factorization, few terms
  - Can only handle less glossy materials
  - Accuracy not properly investigated
- Very nice synthesis of many existing ideas
- Comparison to triple product integrals
  - Not as deep or cool, but simpler and real-time
  - Limits BRDF fidelity, glossiness much more
  - In a sense, they are different types of factorizations

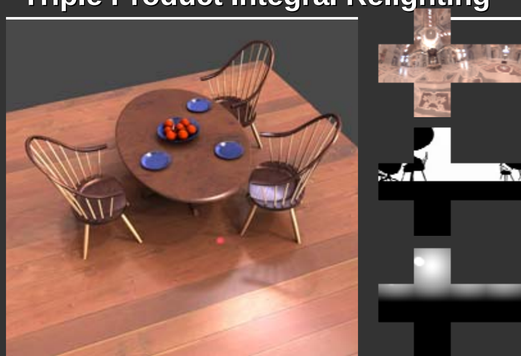
Outline

- Motivation and Background
- Compression methods
  - Low frequency linear spherical harmonic approximation
  - Factorization and PCA
  - Local factorization and clustered PCA
  - Non-linear wavelet approximation
- *Changing view as well as lighting*
  - Clustered PCA
  - Factored BRDFs
  - *Triple Product Integrals*

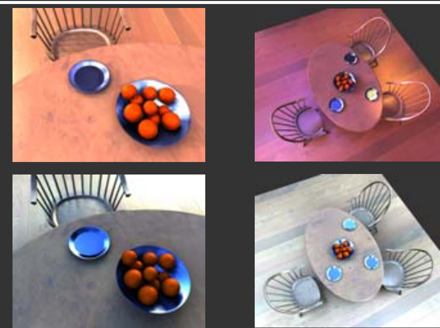
Factorization Approach



Triple Product Integral Relighting



Relit Images (3-5 sec/frame)



Triple Product Integrals

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

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

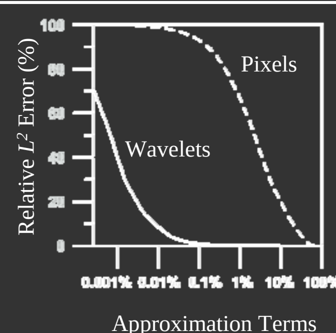
$$L_i,\ V_j,\ \tilde{\rho}_k$$

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	$O(N \log N)$

2. Sparsity in Light Approx.

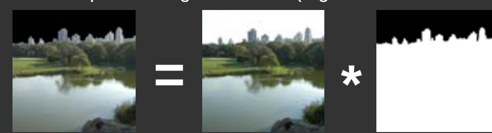


Summary of Wavelet Results

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

Broader Computational Relevance

- Clebsch-Gordan triple product series for spherical harmonics in quantum mechanics (but not focused on computation)
- Essentially no previous work graphics, applied math
- Same machinery applies to basic operation: multiplication
  - Signal multiplication for audio, image compositing,….
  - Compressed signals/videos (e.g. wavelets JPEG 2000)



Summary

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

Subsequent Work

- My survey linked from website (lecture only covers 2002-2004)
- Varied lighting/view. What about dynamic scenes, BRDFs
  - Much recent work [Zhou et al. 05, Ben-Artzi et al. 06]. But still limited for dynamic scenes
- Must work on GPU to be practical
- Sampling on object geometry remains a challenge
- Near-Field Lighting has had some work, remains a challenge
- Applications to lighting design, direct to indirect transfer
- New basis functions and theory
- Newer methods do not require precompute, various GPU tricks
- So far, low-frequency spherical harmonics used in games, all-frequency techniques have had limited applicability