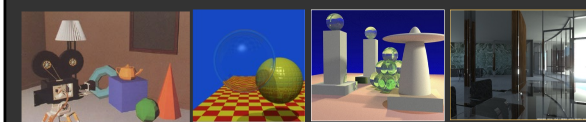
Computer Graphics II: Rendering

CSE 168 [Spr 26], Lecture 16: Precomputed Rendering
Ravi Ramamoorthi

http://viscomp.ucsd.edu/classes/cse168/sp26



1

To Do

- Final Project Milestones due on May 27
  - 1-2 page PDF or website with at least one image
  - Brief description of project, proposal for final version
  - [1-2 para proposal of what you hope to accomplish]
  - Must include image milestone of what's done so far
  - We may say ok or schedule time to meet, discuss
  - Talk to us if any difficulty finding project (Assignment gives some well specified, loose, other options, you can do anything else related to the course too).

2

Motivation

- Next lecture: Image-Based Rendering. 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

3

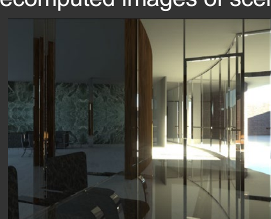
My General Philosophy

- 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; signal processing

4

Precomputation-Based Relighting

- Analyze precomputed images of scene

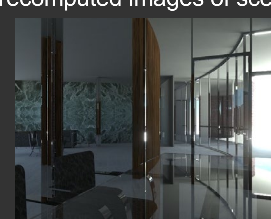


Jensen 2000

5

Precomputation-Based Relighting

- Analyze precomputed images of scene

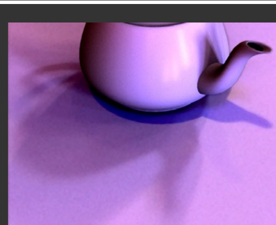


Jensen 2000

6

Assumptions

- Static geometry
- Precomputation



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

7

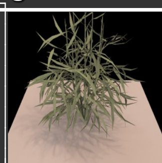
Why is This Hard?

- 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?

8

Relighting as a Matrix-Vector Multiply

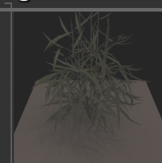
$$\begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \vdots \\ P_N \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1M} \\ T_{21} & T_{22} & \cdots & T_{2M} \\ T_{31} & T_{32} & \cdots & T_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ T_{N1} & T_{N2} & \cdots & T_{NM} \end{bmatrix} \begin{bmatrix} L_1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



9

Relighting as a Matrix-Vector Multiply

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



Output Image (Pixel Vector)

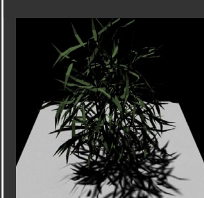
Input Lighting (Cubemap Vector)

Precomputed Transport Matrix

10

Matrix Columns (Images)

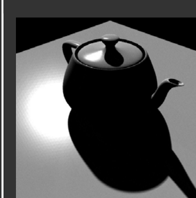
$$\begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1M} \\ T_{21} & T_{22} & \cdots & T_{2M} \\ T_{31} & T_{32} & \cdots & T_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ T_{N1} & T_{N2} & \cdots & T_{NM} \end{bmatrix}$$



11

Precompute: Ray-Trace Image Cols

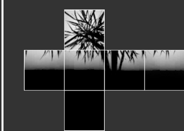
$$\begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1M} \\ T_{21} & T_{22} & \cdots & T_{2M} \\ T_{31} & T_{32} & \cdots & T_{3M} \\ \vdots & \vdots & \ddots & \vdots \\ T_{N1} & T_{N2} & \cdots & T_{NM} \end{bmatrix}$$



12

Precompute 2: Rasterize Matrix Rows

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



13

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?

14

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 (glossy objects)

15

Precomputed Radiance Transfer

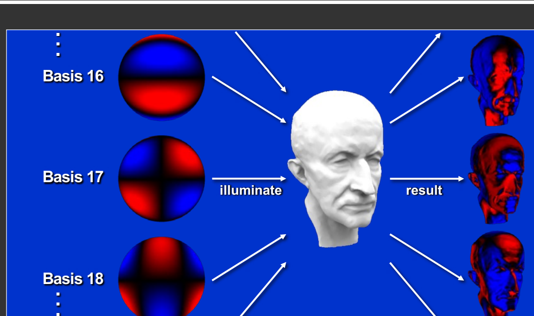
- Better light integration and transport
  - dynamic, area lights
  - self-shadowing
  - interreflections
- For diffuse and glossy surfaces
- At real-time rates
- Sloan et al. 02 (most cited rendering paper in last 20 years 1000+, widely used in games, movie production: Spherical Harmonic Lighting)

point light area light

area lighting, no shadows area lighting, shadows

16

Precomputation: Spherical Harmonics



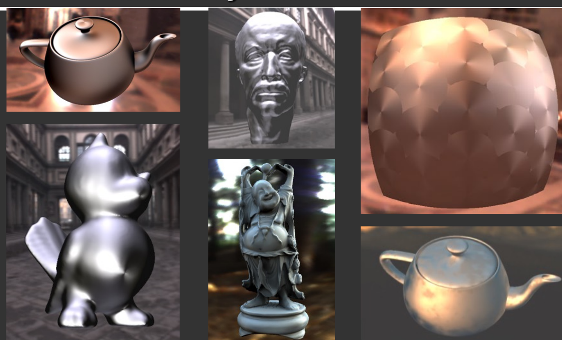
17

Diffuse Transfer Results

No Shadows/Inter Shadows Shadows+Inter

18

Arbitrary BRDF Results



Anisotropic BRDFs | Other BRDFs | Spatially Varying

19

Relighting as a Matrix-Vector Multiply

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

20

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 (common in games, movies) but useful only for broad low-frequency lighting

21

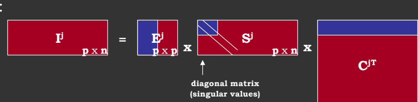
Outline

- 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 (glossy objects)

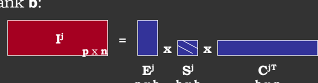
22

PCA or SVD factorization

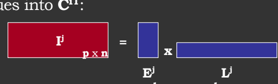
• SVD:



• Applying Rank **b**:



• Absorbing S^j values into C^{jT}:



23

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!!**

24

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

25

Image-Based Rendering

Practical Case

Human Face

Zickler, Enrique, Ramamoorthi, Belhumeur 05, 06

26

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 (glossy objects)

27

Sparse Matrix-Vector Multiplication

Choose data representations with mostly zeroes

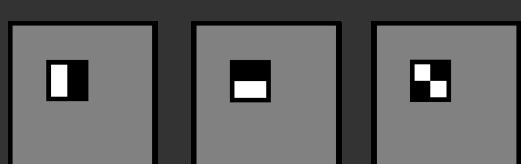
Vector: Use *non-linear wavelet approximation* on lighting

Matrix: Wavelet-encode transport rows

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

28

Haar Wavelet Basis



29

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

30

Non-linear Wavelet Light Approximation

Wavelet Transform

31

Non-linear Wavelet Light Approximation

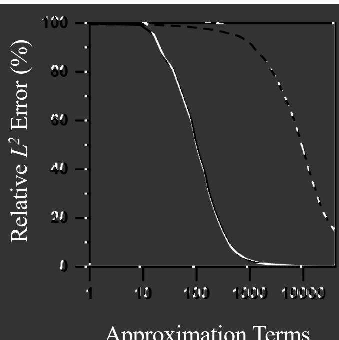
$$\begin{bmatrix} 0 \\ L_2 \\ 0 \\ 0 \\ 0 \\ L_6 \\ \vdots \\ 0 \end{bmatrix}$$

Non-linear Approximation

Retain 0.1% – 1% terms

32

Error in Lighting: St Peter's Basilica



Sph. Harmonics

Non-linear Wavelets

Ng, Ramamoorthi, Hanrahan 03

33

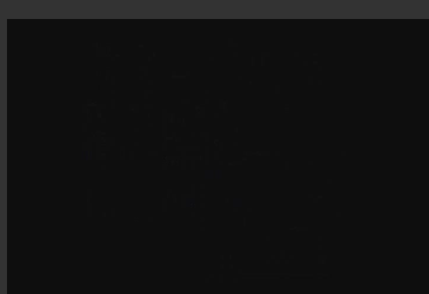
Output Image Comparison

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

25 200 2,000 20,000

34

Video: Real Time Relighting



35

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?

36

Subsequent Work

- My survey 2009 (lecture only covers 2002-2004)
- Varied lighting/view. What about dynamic scenes, BRDFs
  - Much subsequent 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

37

Analytic SH Gradients

Analytic Spherical Harmonic Gradients for Real-Time Rendering with Many Polygonal Area Lights

Lifan Wu[1], Guangyan Cai[1], Shuang Zhao[2], Ravi Ramamoorthi[1]
[1] UC San Diego, [2] UC Irvine

NO AUDIO

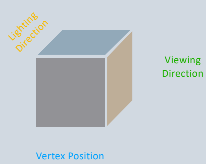
38

Glossy Precomputed Radiance Transfer

Discretize over viewing direction and store a different vector for each one.

This stores a *matrix* instead of a vector at each point.

When rotating the camera, *discretize* the direction.



Sloan, P.-P., et al. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. ACM Trans. Graph. 21-3 (2002)

39

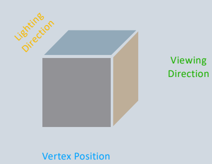
Glossy Precomputed Radiance Transfer

Problem: this is huge!

Six-dimensional tensor

For 512x512 image, with 128x128x6 cubemap, with 128x128 view discretization, the T matrix is 105 terabytes!

How can we compress this?



Sloan, P.-P., et al. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. ACM Trans. Graph. 21-3 (2002)

40

Direct Illumination with Haar Wavelets

Triple Product Wavelet Integrals For All-Frequency Relighting

Ren Ng Stanford University
Ravi Ramamoorthi Columbia University
Pat Hanrahan Stanford University

41

Clustered PCA

VQ PCA CPCA

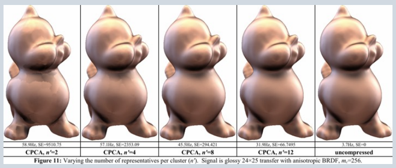
Slides borrowed from Peter-Pike Sloan's CPCA slides (2003)

42

Glossy Precomputed Radiance Transfer

Just apply this to light transport matrices instead.

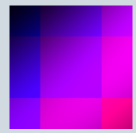
1D -> 625D (25 x 25)



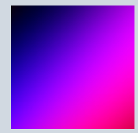
Slides borrowed from Peter-Pike Sloan's CPCA slides (2003)

43

Cluster Discontinuity Problem



Blockwise PCA [NNJ05]



$x \in \mathbb{R}^2;\ y \in \mathbb{R}^3;\ y = f(x)$

Clustered PCA [SHH03]

Slides borrowed from Ari Silvennoinen's SIGGRAPH slides (2021)

44

Approaching Cluster Discontinuity

Back to basics: any function can be expanded in terms of basis functions

PRT hinges on this to store each projection coefficient per vertex

But this basis decomposition may not be the best to compress.

$$f(x) = \sum_i c_i \Psi_i(x)$$

45

Solution: Moving Basis Decomposition

Just represent the projection coefficients themselves as functions of *x!*

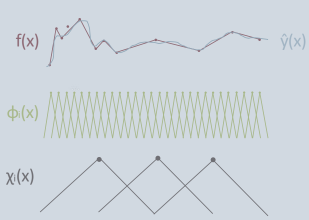
$$f(x) = \sum_i c_i(x) \Psi_i(x)$$

These are both separately represented as a *bilinear interpolation* over a texture (so we maintain piecewise-continuity)

$$c_i(x) = \sum_{m=1}^{M} \phi_m(x) \mathbf{c}_{m,i} \qquad \Psi_i(x) = \sum_{m=1}^{M} \chi_m(x) \mathbf{\Psi}_{m,i}$$

46

Solution: Moving Basis Decomposition



Slides borrowed from Ari Silvennoinen's SIGGRAPH slides (2021)

47

Moving Basis Decomposition for PRT

Densely sample the PRT matrices at each vertex

Use gradient descent or another optimization algorithm to *learn* two differentiable textures: one for coefficient and one for basis

Optimize over the reconstruction loss $r(x) = f(x) - \hat{f}(x)$

End up with two piecewise-linear textures that can be smoothly interpolated anywhere.

48

Large Scale PRT

Slides borrowed from Ari Silvennoinen's SIGGRAPH slides (2021)

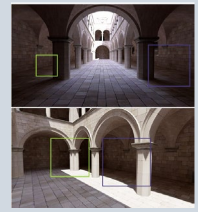
49

Hard to learn Glossy Elements

Moving Basis Decomposition is *independent* of the choice of PRT method used.

Traditional PRT technique; hard to implement glossy materials!

Other data-driven methods may have to be used.



50

Neural Networks

Neural networks are a class of differentiable functions

Defined as a composition of affine transformations and nonlinearities

Universal function approximation

$$g_\theta^{(1)}(x) = \sigma_1(W_\theta^{(1)}x + b_\theta^{(1)})$$

$$g_\theta^{(2)}(x) = \sigma_1(W_\theta^{(2)}g_\theta^{(1)}(x) + b_\theta^{(2)})$$

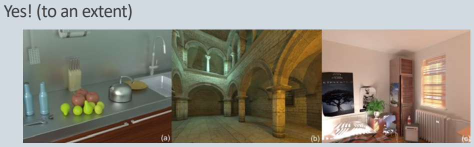
$$g_\theta^{(3)}(x) = \cdots$$

51

Neural Networks as PRT Regressors

Can we directly learn the view-dependent glossy transport by training a neural network directly?

Yes! (to an extent)



Peiran Ren, Jiaping Wang, Minmin Gong, Stephen Lin, Xin Tong, and Baining Guo. 2013. Global illumination with radiance regression functions. ACM Trans. Graph. 32, 4, Article 130 (July 2013), 12 pages. https://doi.org/10.1145/2461912.2462009

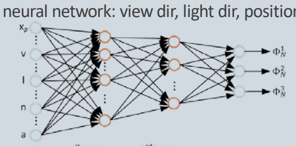
52

Radiance Regression Functions

How do we input an environment map to our network?

Hard: instead, just consider directional lights

Inputs to our neural network: view dir, light dir, position, normal, albedo



Peiran Ren, Jiaping Wang, Minmin Gong, Stephen Lin, Xin Tong, and Baining Guo. 2013. Global illumination with radiance regression functions. ACM Trans. Graph. 32, 4, Article 130 (July 2013), 12 pages. https://doi.org/10.1145/2461912.2462009

53

Radiance Regression Functions

Works well! Gets diffuse caustics and even gets glossy reflections too!

But it's too slow to render a full environment.

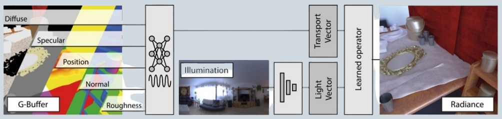
Peiran Ren, Jiaping Wang, Minmin Gong, Stephen Lin, Xin Tong, and Baining Guo. 2013. Global illumination with radiance regression functions. ACM Trans. Graph. 32, 4, Article 130 (July 2013), 12 pages. https://doi.org/10.1145/2461912.2462009

54

Neural Precomputed Radiance Transfer

Idea: encode the entire environment map as a learned neural feature vector.

Combine this with the G-buffer information via another *learned operator* to produce the final rendered color

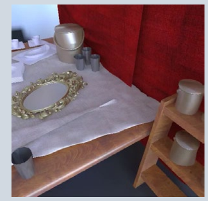


Rainer, G., Bousseau, A., Ritschel, T. and Drettakis, G. (2022), Neural Precomputed Radiance Transfer. Computer Graphics Forum, 41: 365-378. https://doi.org/10.1111/cgf.14480

55

Neural Precomputed Radiance Transfer

This also produces good results!



Rainer, G., Bousseau, A., Ritschel, T. and Drettakis, G. (2022), Neural Precomputed Radiance Transfer. Computer Graphics Forum, 41: 365-378. https://doi.org/10.1111/cgf.14480

56

Neural Precomputed Radiance Transfer

However, this doesn't generalize well to 1) novel environment maps, and 2) view directions far from training

Color shifts and other artifacts (lack of reflections) are seen

Raghavan, N., Xiao, Y., Lin, K.-E., Sun, T., Bi, S., Xu, Z., Li, T.-M. and Ramamoorthi, R. (2023), Neural Free-Viewpoint Relighting for Glossy Indirect Illumination. Computer Graphics Forum, 42: e14885. https://doi.org/10.1111/cgf.14885

57