Interactive Rendering of Translucent Objects

Hendrik Lensch, Michael Goesele, Philippe Bekaert, Jan Kautz, Marcus Magnor, Jochen Lang, Hans-Peter Seidel

2003

Presented By: Mark Rubelmann

Outline

- Motivation
- Background
- Preprocessing
- Rendering
- Results

Motivation

- Translucent objects = subsurface scattering
- Calculating subsurface scattering is expensive
- Observation: multiple scattering blurs and smoothes radiance

Motivation

- Low frequency can be taken advantage of
  - Global response
    - Long distance
    - Lots of scattering
    - Radiance can be calculated sparsely and interpolated
  - Local response
    - Short distance
    - Little scattering
    - Need to maintain detail for small neighborhood

Background

- Full BSSRDF: 8 dimensions
  - $S(x_i, \omega_i, x_o, \omega_o)$
- Diffuse subsurface scattering reflectance function: 4 dimensions
  - $R_d(x_i, x_o)$

Background

- R_d relates incoming flux to outgoing diffuse radiance:

$$L^{\rightarrow}(x_o, \omega_o) = \frac{1}{\pi} F_t(\eta, \omega_o) B(x_o)$$

$$B(x_o) = \int_S E(x_i) R_d(x_i, x_o) dx_i$$

$$E(x_i) = \int_{\Omega_+(x_i)} L^{\leftarrow}(x_i, \omega_i) F_t(\eta, \omega_i) \left| N_i \cdot \omega_i \right| d\omega_i$$

Background

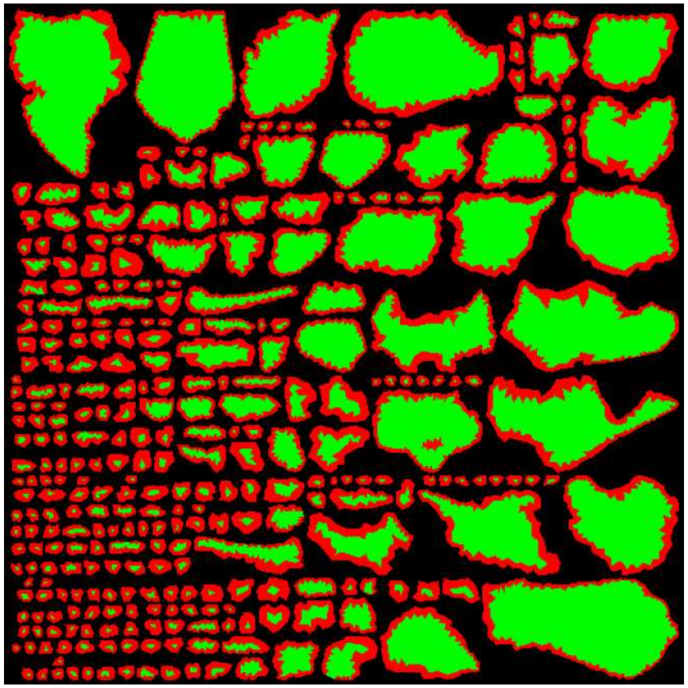
- R_d is very similar to G in radiosity
  - Both are *throughput factors* (discrete version in Galerkin radiosity is *form factor*)
- G only encodes geometric information; storage costs are too high for relighting
- R_d maintains light transport properties between any two points and can handle dynamic lighting

Preprocessing

- Need discrete formulation of $B(x_o)$
- Actually use 2 formulations with two sets of basis functions
  - Global basis: hat functions at object vertices
  - Local basis: Piecewise-constant functions corresponding to surface texels

Preprocessing - Geometry

- Split mesh up into chunks of nearly-planar triangles and build 2D texture atlas



Preprocessing – Global Response

- Scattering over long distances is smooth
- Vertex-to-vertex throughput factors are used

$$F_{ij} \approx R_d(v_i, v_j) \cdot \int_S \psi_i(x)dx \cdot \int_S \tilde{\psi}(y)dy = \frac{A_i}{3} R_d(v_i, v_j)$$

$$B_j^g = \sum_i E_i F_{ij}$$

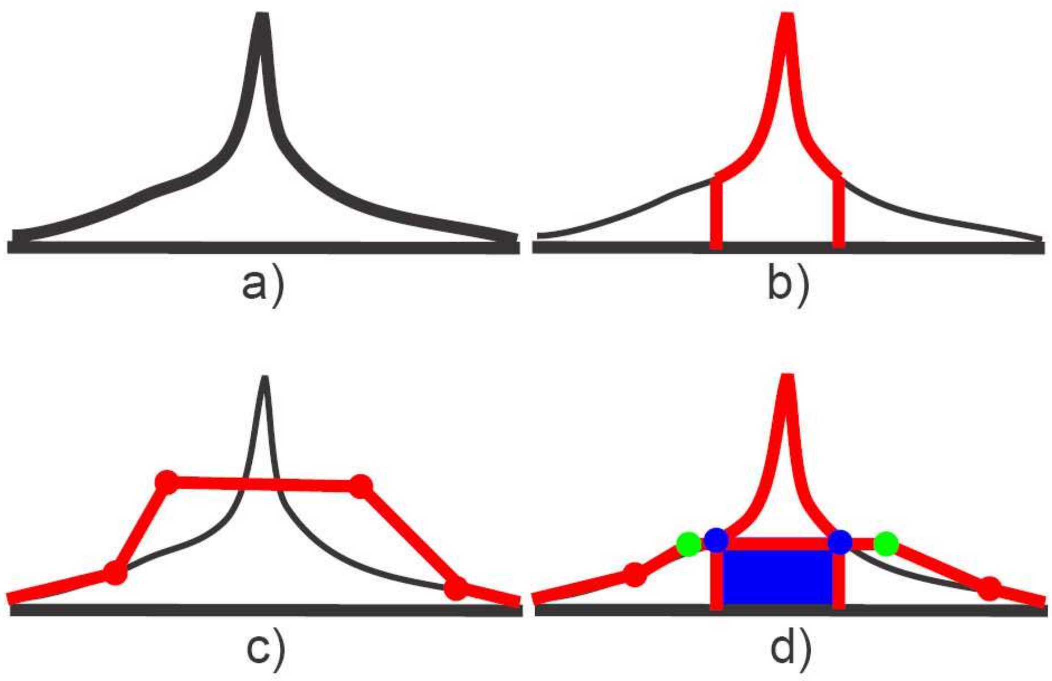
Preprocessing – Local Response

- Use texel-to-texel throughput factors to preserve details
- Modeled as 7 x 7 filter kernel

$$K_{(u,v)}(s,t) = A(u,v) R_d(x_c(u,v), x_c(s,t))$$

Preprocessing – Blending Local and Global

- Adding local and global results in twice the correct amount in direct illumination areas



Preprocessing – Blending Local and Global

- Direct illumination found along diagonal of form factor matrix *F*
- F^0 is *F* without direct illumination
- *B(x)* found by introducing B^d

$$B(x) = B^l(x) + B^d(x) + B^{g_0}(x)$$

$$B_j^{g_0} = \sum_i E_i^g F_{ij}^0$$

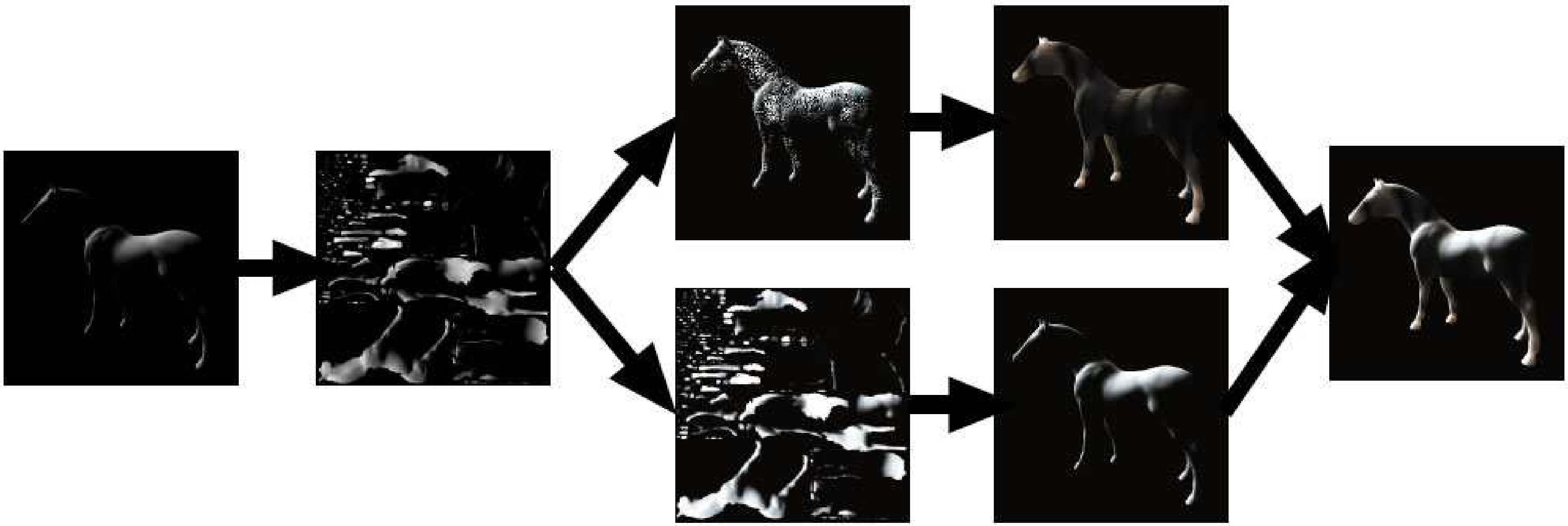
Preprocessing – Blending Local and Global

- Also need to blend border between local and global
- Calculate “correct” radiosity by generating 9 x 9 kernel
- Adjust weighting of global radiosity to minimize difference

Rendering

- Compute direct illumination map
  - Implemented with vertex shader
- Split processing into two branches: global and local
- Global and local responses combined by multi-texturing in hardware

Rendering



Rendering – Global Response

- Find irradiance at each vertex
- $B^g(y)$ at intermediate surface point y is calculated by linear interpolation
- Surface radiosity can be modulated by texture, T_p

$$B_i^T = \frac{B_i^g}{T_p(v_i)}$$

Rendering – Local Response

- Convolve illumination map with filter kernel of every texel

$$B^l(x) = K_{(u,v)}(s,t) \otimes E(s,t) = \sum_{(s,t)\in 7\times 7} K_{(u,v)}(s,t) E(s,t)$$

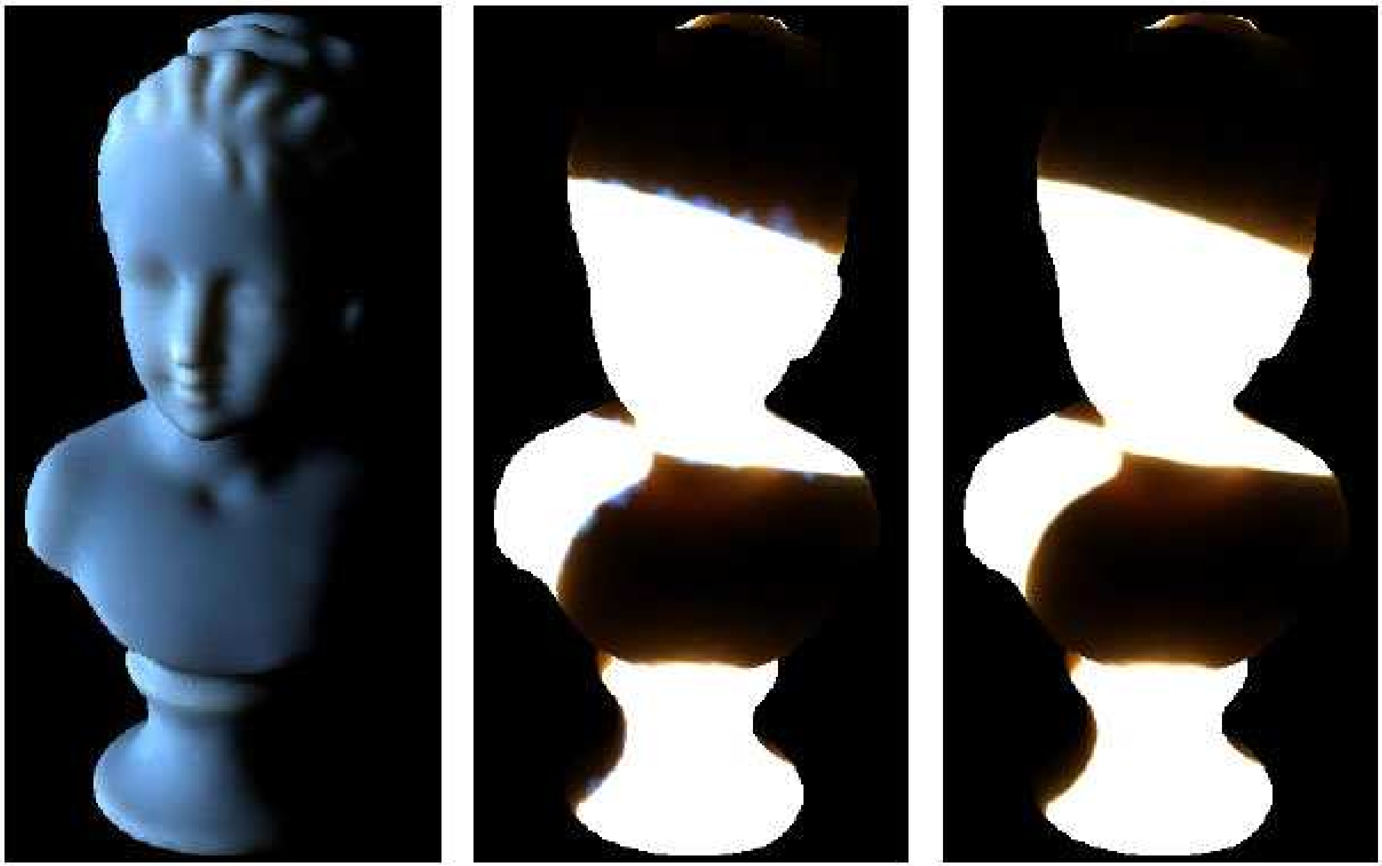
- Initial implementation done in software

Results

- Renderings done on dual 1.7 GHz Xeon with 1 GB RAM and GeForce3 video card

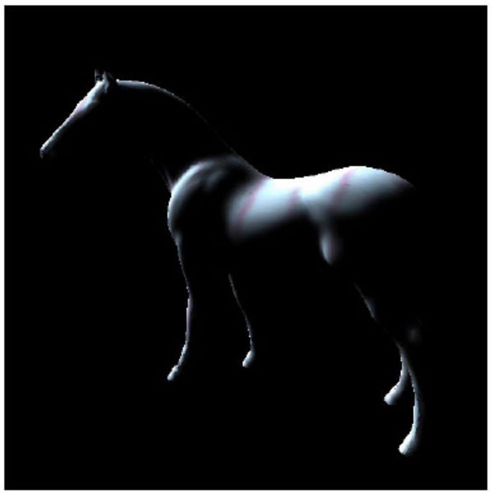
model	# vertices	#form factors	fps	illummap	local	global	display	total
horse	10000	16441460	2.3	29	149	33	371	431
horse textured	10000	12409116	2.7	29	145	302	33	364
bust	8574	4946764	5	24	147	144	28	199
bird	4000	1750862	5.6	16	139	86	26	180

Results



Middle: simple blending   Right: optimized blending

Results

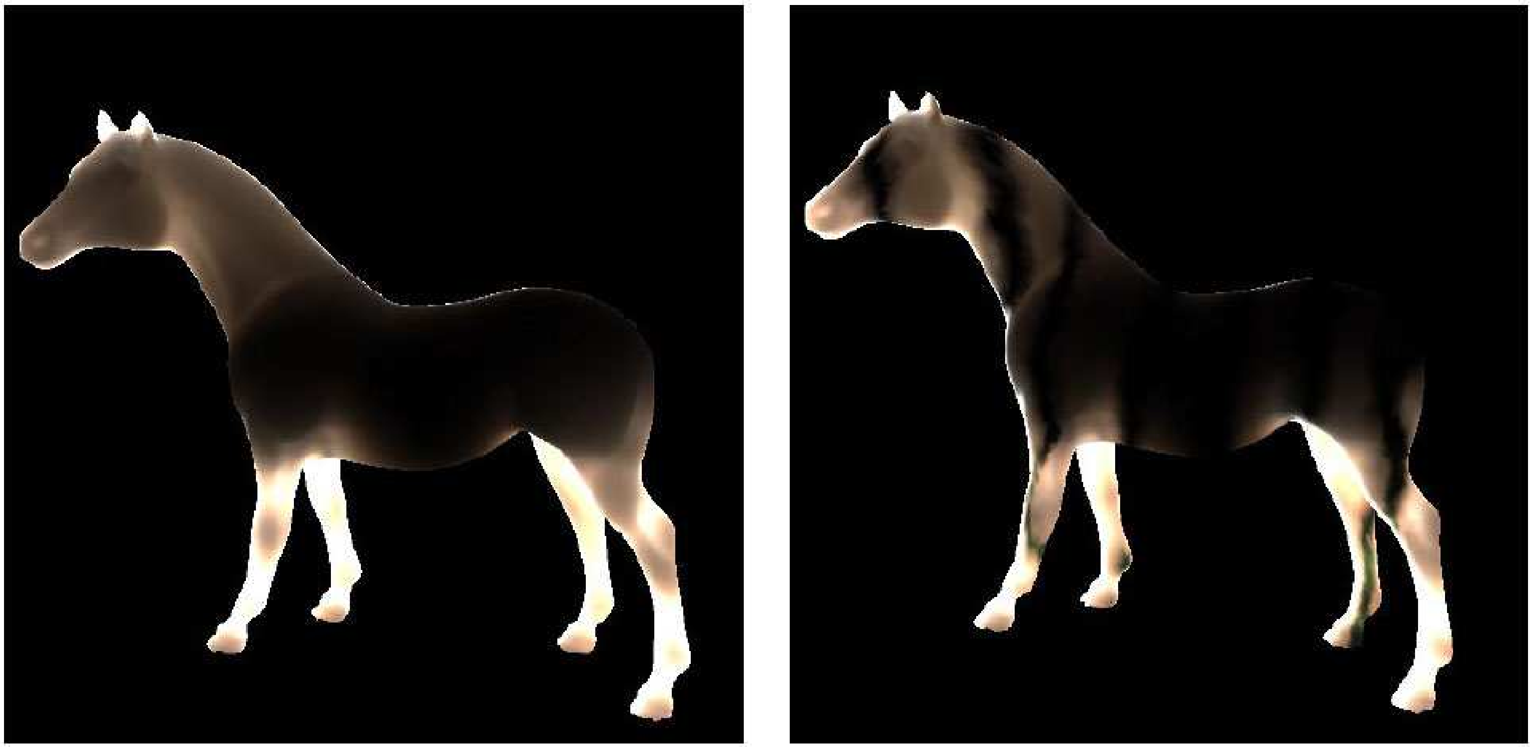


Local response

Global response

Combined

Results



With and without modulating texture

Results

Skim milk?