Acquisition of Point-Sampled Geometry and Appearance

Hanspeter Pfister, MERL
pfister@merl.com

Wojciech Matusik, MIT
Addy Ngan, MIT
Paul Beardsley, MERL
Remo Ziegler, MERL
Leonard McMillan, MIT

The Goal: To Capture Reality

- Fully-automated 3D model creation of real objects.
- Faithful representation of appearance for these objects.

Image-Based 3D Photography

- An image-based 3D scanning system.
  - Handles fuzzy, refractive, transparent objects.
  - Robust, automatic
  - Point-sampled geometry based on the *visual hull*.
  - Objects can be rendered in novel environments.

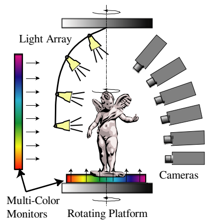
Previous Work

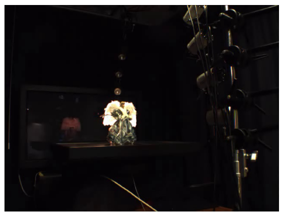
- Active and passive 3D scanners
  - Work best for diffuse materials.
  - Fuzzy, transparent, and refractive objects are difficult.
- BRDF estimation, inverse rendering
- Image based modeling and rendering
  - Reflectance fields [Debevec et al. 00]
    - Light Stage system to capture reflectance fields
    - Fixed viewpoint, no geometry
  - Environment matting [Zongker et al. 99, Chuang et al. 00]
    - Capture reflections and refractions
    - Fixed viewpoint, no geometry

Outline

- Overview
- ➢ System
- Geometry
- Reflectance
- Rendering
- Results

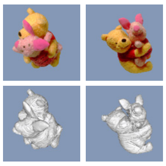
The System





Outline

- Overview
- System
- ➢ Geometry
- Reflectance
- Rendering
- Results

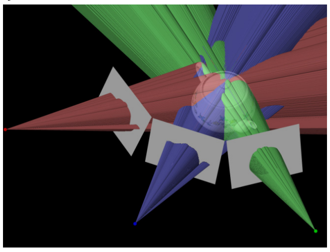


Acquisition

- For each viewpoint (6 cameras x 72 positions)
  - Alpha mattes
    - Use multiple backgrounds [Smith and Blinn 96]
  - Reflectance images
    - Pictures of the object under different lighting (4 lights x 11 positions)
  - Environment mattes
    - Use similar techniques as [Chuang et al. 2000]

Geometry – Opacity Hull

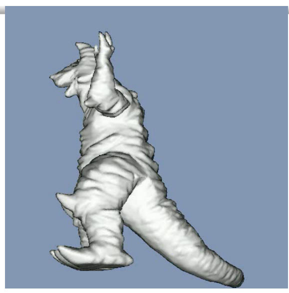
- Visual hull augmented with view-dependent opacity.



Approximate Geometry

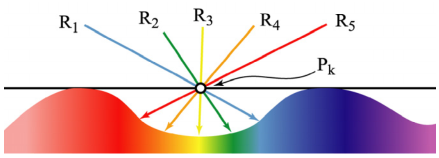
- The approximate visual hull is augmented by radiance data to render concavities, reflections, and transparency.

Geometry Example



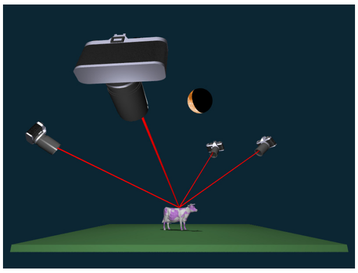
Surface Light Fields

- A surface light field is a function that assigns a color to each ray originating on a surface. [Wood *et al.*, 2000]



Shading Algorithm

- A view-dependent strategy.



Color Blending

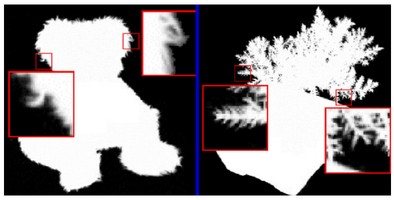
- Blend colors based on angle between virtual camera and stored colors.
- Unstructured Lumigraph Rendering
  [Buehler et al., SIGGRAPH 2001]
- View-Dependent Texture Mapping
  [Debevec, EGRW 98]

Point-Based Rendering

- Point-based rendering using LDC tree, visibility splatting, and view-dependent shading.

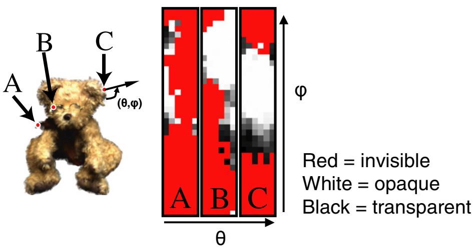
Geometry – Opacity Hull

- Store the opacity of each observation at each point on the visual hull [Matusik et al. SIG2002].



Geometry – Opacity Hull

- Assign view-dependent opacity to each ray originating on a point of the visual hull.



Example

Results

- Point-based rendering using EWA splatting, A-buffer blending, and edge antialiasing.

Opacity Hull – Discussion

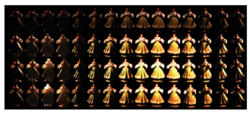
- View dependent opacity vs. geometry trade-off.
  - Similar to radiance vs. geometry trade-off.
- Sometimes acquiring the geometry is not possible (e.g. resolution of the acquisition device is not adequate).
- Sometimes representing true geometry would be very inefficient (e.g. hair, trees).
- Opacity hull stores the "macro" effect.

Point-Based Models

- No need to establish topology or connectivity.
- No need for a consistent surface parameterization for texture mapping.
- Represent organic models (feather, tree) much more readily than polygon models.
- Easy to represent view-dependent opacity and radiance per surface point.

Outline

- Overview
- Previous Works
- Geometry
- ➢ Reflectance
- Rendering
- Results



Light Transport Model

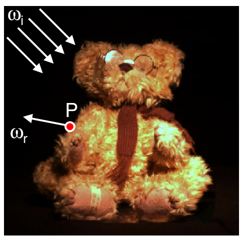
- Assume illumination originates from infinity.
- The light arriving at a camera pixel can be described as:

$$C(x, y) = \int_{\Omega} W(\omega) E(\omega) d\omega$$

C(x,y) - the pixel value
E - the environment
W - the *reflectance field*

Surface Reflectance Fields

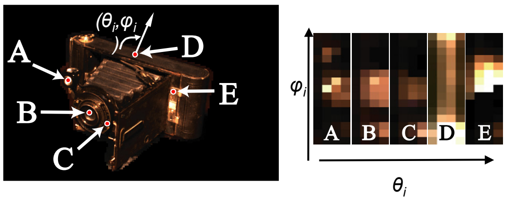
- 6D function: $W(P, \omega_i, \omega_r) = W(u_r, v_r; \theta_i, \Phi_i; \theta_r, \Phi_r)$



Reflectance Functions

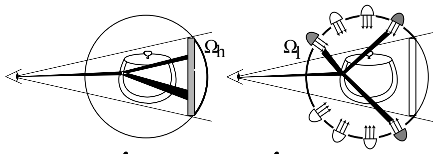
- For each viewpoint, 4D function:

$$W_{xy}(\omega_i) = W(x, y; \theta_i, \Phi_i)$$



Reflectance Field Acquisition

- We separate the hemisphere into high resolution Ω_h and low resolution Ω_l [Matusik et al., EGRW2002].



$$C(x,y) = \int_{\Omega_h} W_h(\xi) T(\xi) d\xi + \int_{\Omega_l} W_l(\omega_i) L(\omega_i) d\omega$$

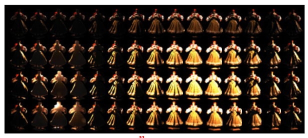
Acquisition

- For each viewpoint (6 cameras x 72 positions)
  - Alpha mattes
    - Use multiple backgrounds [Smith and Blinn 96]
  - Reflectance images ← Low resolution
    - Pictures of the object under different lighting (4 lights x 11 positions)
  - Environment mattes ← High resolution
    - Use similar techniques as [Chuang et al. 2000]

Low-Resolution Reflectance Field

$$C(x,y) = \int_{\Omega_h} W_h(\xi) T(\xi) d\xi + \int_{\Omega_l} W_l(\omega_i) L(\omega_i) d\omega$$

- W_l sampled by taking pictures with each light turned on at a time [Debevec et al 00].



$$\int_{\Omega_l} W_l(\omega_i) L(\omega_i) d\omega \approx \sum_{i=1}^{n} W_i L_i \quad \text{for n lights}$$

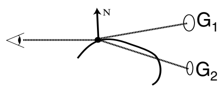
Compression

- Subdivide images into 8 x 8 pixel blocks.
- Keep blocks containing the object (avg. compression 1:7)
- PCA compression (avg. compression 1:10)

High-Resolution Reflectance Field

$$C(x,y) = \int_{\Omega_h} W_h(\xi) T(\xi) d\xi + \int_{\Omega_l} W_l(\omega_i) L(\omega_i) d\omega$$

- Use techniques of environment matting [Chuang et al., SIGGRAPH 00].
- Approximate W_h by a sum of up to two Gaussians:
  - Reflective G_1.
  - Refractive G_2.



$$W_h(\xi) = a_1 G_1 + a_2 G_2$$

Surface Reflectance Fields

- Work without accurate geometry.
- Surface normals are not necessary.
- Capture more than reflectance:
  - Inter-reflections
  - Subsurface scattering
  - Refraction
  - Dispersion
  - Non-uniform material variations
- Simplified version of the BSSRDF [Debevec et al., 00].

Outline

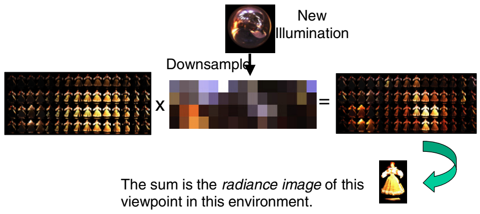
- Overview
- Previous Works
- Geometry
- Reflectance
- ➢ Rendering
- Results

Rendering

- Input: Opacity hull, reflectance data, new environment
- Create *radiance* images from environment and low-resolution reflectance field.
- Reparameterize environment mattes.
- Interpolate data to new viewpoint.

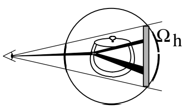
1st Step: Relighting Ω_l

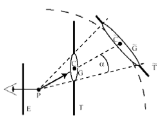
- Compute *radiance image* for each viewpoint.



2nd Step: Reproject Ω_h

- Project environment mattes onto the new environment.
  - Environment mattes acquired was parameterized on plane T (the plasma display).
  - We need to project the Gaussians to the new environment map, producing new Gaussians.





3rd Step: Interpolation

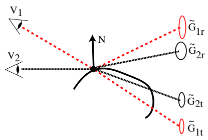
- From new viewpoint, for each surface point, find four nearest acquired viewpoints.
  - Store visibility vector per surface point.
- Interpolate using unstructured lumigraph interpolation [Buehler et al., SIGGRAPH 01] or view-dependent texture mapping [Debevec 96].
  - Opacity.
  - Contribution from low-res reflectance field (in the form of radiance images).
  - Contribution from high-res reflectance field.

3rd Step: Interpolation

- For low-res reflectance field, we interpolate the RGB color from the radiance images.

For high-resolution reflectance field:

Interpolate *direction* of reflection/refraction.
Interpolate other parameters of the Gaussians.
Convolve with the environment.



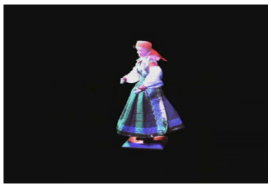
Outline

- Overview
- Previous Works
- Geometry
- Reflectance
- Rendering
- ➢ Results

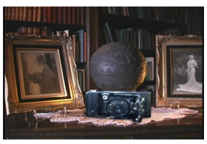
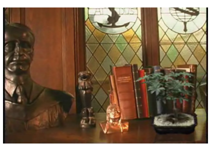
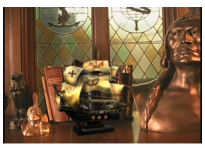
Results

- Performance for 6x72 = 432 viewpoints
- 337,824 images taken in total !!
  - Acquisition (47 hours)
    - Alpha mattes - 1 hour
    - Environment mattes - 18 hours
    - Reflectance images - 28 hours
  - Processing
    - Opacity hull ~ 30 minutes
    - PCA Compression ~ 20 hours (MATLAB, unoptimized)
  - Rendering ~ 5 minutes per frame
  - Size
    - Opacity hull ~ 30 - 50 MB
    - Environment mattes ~ 0.5 - 2 GB
    - Reflectance images ~ Raw 370 GB / Compressed 2 - 4 GB

Results

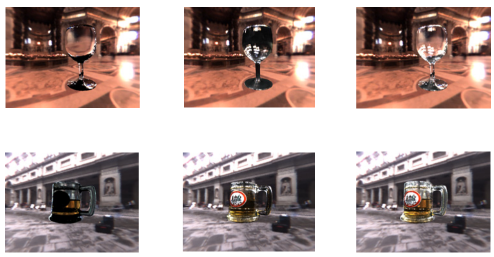


Results

Results

High-resolution Ω_h Low-resolution Ω_l Combined



Results

Results – Ω_h

Results – Ω_l

Results – Combined

Results

Results

Conclusions

- A fully automatic system that is able to capture and render any type of object.
- Opacity hulls combined with lightfields / surface reflectance fields provide realistic 3D graphics models.
- Point-based rendering offers easy surface parameterization of acquired models.
- Separation of surface reflectance fields into high- and low-resolution areas is practical.
- New rendering algorithm for environment matte interpolation.

Future Directions

- Use more than 2 Gaussians for the environment mattes.
- Better compression.
- Real-time rendering.

Acknowledgements

- Colleagues:
  - MIT: Chris Buehler, Tom Buehler.
  - MERL: Bill Yerazunis, Darren Leigh, Michael Stern.
- Thanks to:
  - David Tames, Jennifer Roderick Pfister.
- NSF grants CCR-9975859 and EIA-9802220.
- Papers available at:
  - http://www.merl.com/people/pfister/