Global Illumination Overview

Tom Funkhouser
Princeton University
COS 526, Fall 2012

Overview

- Rendering equation
  - Rendering = integration
- Solution methods
  - Direct illumination
  - Recursive ray tracing
  - Distribution ray tracing
  - Path tracing
  - Photon mapping
  - Radiosity
  - etc.

Rendering Equation

\[ L(x', \omega') = L_v(x', \omega') + \int \frac{f_i(x', \omega, \omega') L_v(x', \omega)(\omega \cdot \hat{n})}{\cos \theta_i} d\omega \]

Rendering Equation (2)

\[ L(x' \rightarrow x) = L_v(x' \rightarrow x') + \int f_i(x \rightarrow x' \rightarrow x) L_v(x \rightarrow x') \Gamma(x, x') G(x, x') dA \]

Rendering Equation

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics
Rendering Equation

• Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics

\[ L_e = \int \Delta x \rightarrow e \Delta t \]

Herf

\[ L(x, \vec{w}) = L(x, x \rightarrow e) + \int_{\Delta x(x', x \rightarrow e)} L(x' \rightarrow e) \rho(x, x') \Delta x \Delta t \]

Debevec

\[ L(x, \vec{w}) = L(x, \vec{w}) + \int_{\Delta x(x', \vec{w}, \vec{v})} L(x, x' \rightarrow \vec{w}) \rho(x, x', \vec{w}, \vec{v}) \Delta x \Delta \vec{w} \Delta \vec{v} \]
Rendering Equation

- Rendering = integration
  - Antialiasing
  - Soft shadows
  - Indirect illumination
  - Caustics

\[ L(x, \omega) = L(x, \omega) + \iiint f_s(x', \omega') \Psi(x', \omega') \omega' \times \omega' d\omega' \]

Challenge

- Rendering integrals are difficult to evaluate
  - Multiple dimensions
  - Discontinuities
    - Partial occluders
    - Highlights
    - Caustics

\[ L(x, \omega) = L(x, \omega) + \iiint f_s(x', \omega') \Psi(x', \omega') \omega' \times \omega' d\omega' \]

Overview

- Rendering equation
  - Rendering is integration

Solution methods
  - Direct illumination
  - Recursive ray tracing
  - Distribution ray tracing
  - Path tracing
  - Photon mapping
  - Radiosity
  - etc.

Direct Illumination

\[ L_s(x', \omega') = L_s(x', \omega') + \int_{\partial \Omega} f_s(x', \omega, \omega') L_s(x', \omega') \omega \times \bar{n} d\omega \]
**OpenGL**

\[
L_o(x', \omega') = L_i(x', \omega') + \int_\Omega f_s(x', \omega, \omega') L_i(x', \omega) (\omega \cdot \bar{n}) d\omega
\]

Assume
direct illumination
from point lights
and ignore visibility

**Overview**

- Rendering equation
  - Rendering is integration
- Solution methods
  - Direct illumination
  - Recursive ray tracing
  - Distribution ray tracing
  - Path tracing
  - Photon mapping
  - Radiosity
  - etc.

**Recursive Ray Tracing**

\[
L_o(x', \omega') = L_i(x', \omega') + \sum_{i=1}^{n} f_s(x', \omega, \omega') L_i(x', \omega) (\omega \cdot \bar{n}) (\omega' \cdot \bar{n})
\]

Assume only significant
indirect illumination due
to perfect specular
reflection and refraction

**Recursive Ray Tracing?**

\[
L_o(x', \omega') = L_i(x', \omega') + \sum_{i=1}^{n} f_s(x', \omega, \omega') L_i(x', \omega) (\omega \cdot \bar{n}) + \text{specular}
\]

Paul Debevec

**Overview**

- Rendering equation
  - Rendering is integration
- Solution methods
  - Direct illumination
  - Recursive ray tracing
  - Distribution ray tracing
  - Path tracing
  - Photon mapping
  - Radiosity
  - etc.
Distribution Ray Tracing

\[ L_o(x', \omega') = L_e(x', \omega') + \int_{\omega} f_s(x', \omega, \omega') L_e(x', \omega)(\omega \cdot \hat{n}) d\omega \]

Estimate integral for each reflection by random sampling

Also:
- Depth of field
- Motion blur
- etc.

Overview

- Rendering equation
  - Rendering is integration
- Solution methods
  - Direct illumination
  - Recursive ray tracing
  - Distribution ray tracing
  - Path tracing
  - Photon mapping
  - Radiosity
  - etc.

Path Tracing

\[ L_o(x', \omega') = L_e(x', \omega') + \int_{\omega} f_s(x', \omega, \omega') L_e(x', \omega)(\omega \cdot \hat{n}) d\omega \]

Estimate integral for each pixel by random sampling

Path Tracing vs. Path Tracing
Overview

- Rendering equation
  - Rendering is integration
- Solution methods
  - Direct illumination
  - Recursive ray tracing
  - Distribution ray tracing
  - Path tracing
  - Photon mapping
  - Radiosity
  - etc.

Photon Mapping

- Two pass method:
  1. Build photon map by tracing paths from lights
  2. Render image by tracing paths from camera

Photon Mapping?

- Two pass method:
  1. Build photon map by tracing paths from lights
  2. Render image by tracing paths from camera
Radiosity

Discretize surfaces into small patches

Baum

Radiosity

Assume simple function (constant) is good approximation for radiosity (sum of energy leaving point in all directions) within patch

Thadani

Radiosity

Leads to sparse system of equations

\[ B_i A_i = E_i A_i + \rho \sum_{j=1}^{N} F_{ij} B_j A_j \]

\[
\begin{bmatrix}
1 - \rho_1 F_{i1} & \cdots & - \rho_i F_{i1} \\
- \rho_2 F_{i2} & 1 - \rho_2 F_{i2} & \cdots & - \rho_i F_{i2} \\
\vdots & \vdots & \ddots & \cdots \\
- \rho_m F_{im} & \cdots & - \rho_m F_{im} & 1 - \rho_i F_{im}
\end{bmatrix}
\begin{bmatrix}
B_1 \\
B_2 \\
\vdots \\
B_m
\end{bmatrix}
= 
\begin{bmatrix}
E_1 \\
E_2 \\
\vdots \\
E_m
\end{bmatrix}
\]

Radiosity

Overview

- Rendering equation
  - Rendering is integration
- Solution methods
  - Direct illumination
  - Recursive ray tracing
  - Distribution ray tracing
  - Path tracing
  - Photon mapping
  - Radiosity
  - etc.

Which method is best?

Path Types
Path Types

- OpenGL
  - LDE
- Ray tracing
  - LDS*E
- Path tracing
  - L(D|S)*E
- Radiosity
  - LD*E

Summary

- Rendering is integration
  - Rendering equation
- Different solution methods are best when different path types are important
  - OpenGL - LDE
  - Recursive ray tracing – LDS*E
  - Distribution ray tracing – L(SD)*E
  - Path tracing – L(SD)*E
  - Photon mapping – L(SD)*E (biased)
  - Radiosity - LD*E