Radiance Caching for Participating Media

Wojciech Jarosz    Craig Donner
Matthias Zwicker    Henrik Wann Jensen

University of California, San Diego
Pixel Lab

Thursday, 6 September 12
In this talk, we are interested in rendering scene with participating media, or scenes where the volume or medium participates in the lighting interactions.

Participating media is actually all around us.

These are just a few example photographs of the type of striking effects that are caused by participating media.
* In order to render images:
  * We need to compute the radiance, \( L \), arriving at the eye along a ray in the presence of participating media.
  * This can be expressed using the volume rendering equation, which consists of two main terms:
    * The right term incorporates lighting arriving from surfaces
    * and the left term, scattering of light from the medium
In order to render images:

We need to compute the radiance, $L$, arriving at the eye along a ray in the presence of participating media.

This can be expressed using the volume rendering equation, which consists of two main terms:

* The right term incorporates lighting arriving from surfaces
* and the left term, scattering of light from the medium
**Volume Rendering Eq.**

\[ L(e, \overline{\omega}) = \int_0^s T_r(e \leftrightarrow x) \sigma_s(x) L_i(x, \overline{\omega}) \, dx + T_r(e \leftrightarrow x') L(x', \overline{\omega}) \]

Transmittance: \[ T_r(e \leftrightarrow x') = e^{-\int_e^x \sigma(t) \, dt} \]

*This light is then diminished by the transmittance as it travels through the medium towards the eye.*
Volume Rendering Eq.

\[ L(e, \bar{\omega}) = \int_0^s T_r(e \leftrightarrow x) \sigma_s(x) L_i(x, \bar{\omega}) \, dx + T_r(e \leftrightarrow x') L(x', \bar{\omega}) \]

all light arriving at \( x \) which scatters towards \( e \)
Volume Rendering Eq.

\[ L(e, \omega) = \int_0^s T_r(e \leftrightarrow x) \sigma_s(x) L_i(x, \omega) \, dx + T_r(e \leftrightarrow x') L(x', \omega) \]

scattering coefficient
\[ L(e, \omega) = \int_0^S T_r(e \leftrightarrow x) \sigma_s(x) L_i(x, \omega) \, dx + T_r(e \leftrightarrow x') L(x', \omega) \]
Volume Rendering Eq.

\[ L(e, \bar{\omega}) = \int_{0}^{S} T_r(e \leftrightarrow x) \sigma_s(x) L_i(x, \bar{\omega}) \, dx + T_r(e \leftrightarrow x') L(x', \bar{\omega}) \]

Very costly!
Previous Work

Participating Media

Path tracing
[Kajiya and Herzen 84, Kajiya 86, Lafortune and Willems 96]
- Slow convergence/noisy results.

Photon mapping
[Jensen and Christensen 1998.]
- Costly for high albedo
- Costly for scenes with large extent

Finite Element
[Rushmeier and Torrance 87]
- Requires discretization

* A number of methods have been developed to handle participating media, but they all have significant limitations.
* this motivates us to develop a new method.
**Related Work**

Global Illumination

Caching:

- “Radiance Caching for Efficient Global Illumination Computation.” Křivánek et al. ‘05

* we draw inspiration for our method from irradiance caching methods for surfaces.
Indirect Illumination

* direct illumination has sharp discontinuities
* Indirect illumination smooth in large regions
* compute irradiance accurately only at a sparse set of locations (shown in yellow) and interpolate whenever possible.
Irradiance Gradients

Ward and Heckbert ‘92

* follow-up work
Observations

Smooth in large portions of the image

* We make the observation that same property is true for participating media
* computationally very expensive, but very smooth and low frequency in large parts of the image
Goals

- Exploit this property by caching lighting within participating media.
- Develop an efficient but general rendering algorithm which can handle:
  - single, multiple, anisotropic scattering
  - heterogeneous media
  - production quality
at a high level, radiance caching gains efficiency by caching expensive lighting calculations within the medium.
for this ray, since a cache point overlaps with every part of the ray, we can compute the lighting by interpolating the cache points
* a neighboring ray can re-use many of the same cache points
**Challenges**

- What should the cache points store?
- Where to place cache points to minimize visible error?
- How to interpolate cache points accurately?

Thursday, 6 September 12

* Cannot re-use details from irradiance caching directly, since many underlying assumptions are different.
* What is a “good” valid radius?
* How do we interpolate the nearby cached values?
Approach

- Cache inscattered radiance:

\[
L(e, \vec{\omega}) = \int_0^s T_r(e \leftrightarrow x) \sigma_s(x) L_i(x, \vec{\omega}) \, dx + T_r(e \leftrightarrow x') L(x', \vec{\omega})
\]

* Since the gradient is a local measure of the smoothness of the radiance field, we use it to estimate a valid radius within which it’s OK to extrapolate each cache point.
Approach

• Cache inscattered radiance:

\[
L(e, \bar{\omega}) = \int_0^{s} T_r(e \leftrightarrow x) \sigma_s(x) L_i(x, \bar{\omega}) \, dx + T_r(e \leftrightarrow x') L(x', \bar{\omega})
\]

• Compute gradients due to translation

* Since the gradient is a local measure of the smoothness of the radiance field, we use it to estimate a valid radius within which it’s OK to extrapolate each cache point.
**Approach**

- Cache inscattered radiance:
  \[ L(e, \bar{\omega}) = \int_0^s T_r(e \leftrightarrow x) \sigma_s(x) L_i(x, \bar{\omega}) \, dx + T_r(e \leftrightarrow x') L(x', \bar{\omega}) \]

- Compute gradients due to translation

- Use gradients to:
  - Estimate valid radius within which it’s OK to extrapolate
  - Provide high quality interpolation

* Since the gradient is a local measure of the smoothness of the radiance field, we use it to estimate a valid radius within which it’s OK to extrapolate each cache point.
In order to make gradient derivations more convenient:

- Split computation into single and multiple scattering components:

\[ L_i = L_s + L_m \]

- How do we compute \( L_s \) and \( L_m \)?

- How do we compute \( \nabla L_s \) and \( \nabla L_m \)?
Single Scattering

\[ L_s(x, \bar{\omega}) = \int_A p(\bar{\omega}, x' \rightarrow x) L_r(x' \rightarrow x) V(x' \leftrightarrow x) H(x' \rightarrow x) \, dx' \]
Single Scattering

\[ L_s(x, \vec{\omega}) = \int_A p(\vec{\omega}, x' \to x) L_r(x' \to x) V(x' \leftrightarrow x) H(x' \to x) \, dx' \]

Integration over surface area

* over area of light sources and surfaces
* reduce radiance
* radiance from light, diminished through medium
Single Scattering

\[ L_s(x, \bar{\omega}) = \int_A p(\bar{\omega}, x' \rightarrow x) L_r(x' \rightarrow x) V(x' \leftrightarrow x) H(x' \rightarrow x) \, dx' \]

Reduced Radiance: \[ L_r(x' \rightarrow x) = L(x' \rightarrow x) T_r(x' \leftrightarrow x) \]

* over area of light sources and surfaces
* reduce radiance
* radiance from light, diminished through medium
Single Scattering

\[ L_s(x, \tilde{\omega}) = \int_A p(\tilde{\omega}, x' \rightarrow x) L_r(x' \rightarrow x) V(x' \leftrightarrow x) H(x' \rightarrow x) \, dx' \]

Phase function
Single Scattering

\[ L_s(x, \vec{\omega}) = \int_A p(\vec{\omega}, x' \rightarrow x) L_r(x' \rightarrow x) V(x' \leftrightarrow x) H(x' \rightarrow x) \, dx' \]

Visibility Function and Geometry Term

* there is also a visibility and geometry function
Gradient Computation

\[ L_s(x, \bar{\omega}) = \int_A p(\bar{\omega}, x' \to x) L_r(x' \to x)V(x' \leftrightarrow x) H(x' \to x) \, dx' \]

* In order to obtain the gradient, we analytically differentiate the terms in the integrand using the product rule.
* gradient of Lr is most significant:
  * accounts for change in transmission, even in heterogeneous media
* In order to obtain the gradient, we analytically differentiate the terms in the integrand using the product rule.
* gradient of \( L_r \) is most significant:
  * accounts for change in transmission, even in heterogeneous media
**Gradient Computation**

\[ L_s(x, \omega) = \int_A p(\omega, x' \rightarrow x) L_r(x' \rightarrow x) V(x' \leftrightarrow x) H(x' \rightarrow x) \, dx' \]

\[ \nabla L_s(x, \omega) = \int_A (\nabla p)L_r V H + p(\nabla L_r)V H + pL_r V (\nabla H) \, dx' \]

- Assumes constant visibility

* In order to obtain the gradient, we analytically differentiate the terms in the integrand using the product rule.
* Gradient of \( L_r \) is most significant:
  * accounts for change in transmission, even in heterogeneous media
Gradient Computation

\[ L_s(x, \omega) = \int_A p(\omega, x' \rightarrow x) L_r(x' \rightarrow x) V(x' \leftrightarrow x) H(x' \rightarrow x) \, dx' \]

\[ \nabla L_s(x, \omega) = \int_A (\nabla p) L_r V H + p(\nabla L_r) V H + p L_r V (\nabla H) \, dx' \]

- Assumes constant visibility
- Evaluated together using Monte Carlo integration and ray marching

* In order to obtain the gradient, we analytically differentiate the terms in the integrand using the product rule.
* Gradient of \( L_r \) is most significant:
  * accounts for change in transmission, even in heterogeneous media
**Gradient Computation**

\[ L_s(x, \omega) = \int_A p(\omega, x' \rightarrow x) L_r(x' \rightarrow x) V(x' \leftarrow x) H(x' \rightarrow x) \, dx' \]

\[ \nabla L_s(x, \omega) = \int_A (\nabla p)L_r V H + p(\nabla L_r)V H + pL_r V (\nabla H) \, dx' \]

- Assumes constant visibility
- Evaluated **together** using Monte Carlo integration and ray marching
- Gradients take into account changing properties of medium **along the whole ray**

* In order to obtain the gradient, we analytically differentiate the terms in the integrand using the product rule.
  * gradient of \( L_r \) is most significant:
    * accounts for change in transmission, even in heterogeneous media
Reduced Radiance: $L_r(x' \rightarrow x) = L(x' \rightarrow x)T_r(x' \leftrightarrow x)$
Ray Marching
Transmission Gradient
* take into account how the extinction coefficients change along the whole ray segment
* these changes would induce a different overall transmission when $x$ is translated
* gradients contain meaningful information about how $T_r$ changes as we move $x$ in any direction, even out of the line connecting $x$ to $x'$
Multiple Scattering

$L_m$ X
Multiple Scattering Gradient
Cache Storage

• Cached points store:
  • 3D position
  • Value (in scattered radiance)
  • Gradient
  • Valid Radius

* distinct caches for single, surface, and multiple scattering
**Cache Storage**

- Cached points store:
  - 3D position
  - Value
  - Gradient
  - Valid Radius

Isotropic Media

\[ \text{inscattered radiance is a scalar} \]
Cache Storage

• Cached points store:
  • 3D position
  • Value
  • Gradient
  • Valid Radius

Anisotropic Media

- inscattered radiance is a spherical function
- projected onto SH
Valid Radius

Want density of cache points to adapt to the local variation of illumination:
* smooth radiance $\rightarrow$ large radius, few cache points
* sharp radiance $\rightarrow$ small radius, many cache points
**Optimal Radius**

\[ A(x') = \text{actual radiance at } x' \]
\[ E_x(x') = \text{radiance extrapolated from } x \text{ to } x' \]

* Maximum radius such that the total relative error between the extrapolated and actual radiance within the cached region is below some error threshold \( t \).
* using relative error because human vision is sensitive to contrast, not absolute errors
**Optimal Radius**

\[
A(x') = \text{actual radiance at } x'
\]

\[
E_x(x') = \text{radiance extrapolated from } x \text{ to } x'
\]

\[
R_{opt}(x) = \max_r \left( \frac{\int_{x' \in \Omega_r} |E_x(x') - A(x')| \, dx'}{\int_{x' \in \Omega_r} A(x') \, dx'} < t \right)
\]

* Maximum radius such that the total relative error between the extrapolated and actual radiance within the cached region is below some error threshold t.
* using relative error because human vision is sensitive to contrast, not absolute errors
Valid Radius

1D Scene with 2 Point Lights

Valid radius given error threshold
Optimal radius
Position

Location of Light 1
Location of Light 2

* numerically computed optimal radius
Valid Radius

1D Scene with 2 Point Lights

Valid radius given error threshold

Position

Location of Light 1

Location of Light 2

Optimal radius

measure of local contrast:

\[ \nabla \ln(L) = \frac{\nabla L}{L} \]

* log-space gradient. perceptually motivated: contrast
measure of local contrast:

\[ \nabla \ln(L) = \frac{\nabla L}{L} \]

\[ r \propto \frac{1}{\|\nabla \ln(L)\|} = \frac{\sum_j L_j}{\|\sum_j \nabla L_j\|} \]
Valid Radius

1D Scene with 2 Point Lights

Avoid singularities:

$$r \propto \frac{\sum_j L_j}{\| \sum_j \nabla L_j \|} \quad \rightarrow \quad r \propto \frac{\sum_j L_j}{\sum_j \| \nabla L_j \|}$$
• Perform a weighted interpolation from nearby cache points.

* whenever possible, interpolate from nearby cache points
* given a valid radius for each cache point, in order to compute radiance: interpolate
* weighted average
* smooth weighting function
Interpolation

- Perform a weighted interpolation from nearby cache points.

\[
L(x) \approx \exp \left( \sum_{k \in C} \left( \ln(L_k) + \frac{\nabla L_k}{L_k} \cdot (x - x_k) \right) w(||x - x_k||) \right) \frac{\sum_{k \in C} w(||x - x_k||)}{\sum_{k \in C} w(||x - x_k||)}
\]

* whenever possible, interpolate from nearby cache points
* given a valid radius for each cache point, in order to compute radiance: interpolate
* weighted average
* smooth weighting function
* given a valid radius for each cache point, in order to compute radiance: interpolate
* weighted average
* smooth weighting function
Find overlapping cache points.

\[
L(x) \approx \exp \left( \sum_{k \in C} \left( \ln(L_k) + \frac{ \nabla L_k}{L_k} \cdot (x - x_k) \right) w(||x - x_k||) \right) / \sum_{k \in C} w(||x - x_k||)
\]

* given a valid radius for each cache point, in order to compute radiance: interpolate
* weighted average
* smooth weighting function
Extrapolate along gradients in log-space.

\[
L(x) \approx \exp \left( \sum_{k \in C} \left( \ln(L_k) + \frac{\nabla L_k}{L_k} \cdot (x - x_k) \right) w(||x - x_k||) \right) / \sum_{k \in C} w(||x - x_k||)
\]

* given a valid radius for each cache point, in order to compute radiance: interpolate
* weighted average
* smooth weighting function
Weight contributions using smooth kernel

\[
L(x) \approx \exp \left( \sum_{k \in C} \left( \ln(L_k) + \frac{\nabla L_k}{L_k} \cdot (x - x_k) \right) w(\|x - x_k\|) \right) \frac{\sum_{k \in C} w(\|x - x_k\|)}{\sum_{k \in C}}
\]

* given a valid radius for each cache point, in order to compute radiance: interpolate
* weighted average
* smooth weighting function
Exponentiate result to obtain interpolated radiance.

\[ L(x) \approx \exp \left( \sum_{k \in C} \left( \ln(L_k) + \frac{\nabla L_k}{L_k} \cdot (x - x_k) \right) \frac{w(||x - x_k||)}{\sum_{k \in C} w(||x - x_k||)} \right) \]

* given a valid radius for each cache point, in order to compute radiance: interpolate
* weighted average
* smooth weighting function
Results

• All results rendered:
  • at 1K horizontal resolution
  • with up to 16 samples per pixel
  • on a Core 2 Duo 2.4 GHz
Results

1.4 minutes
Results

Thursday, 6 September 12

* can handle anisotropic media
* project radiance and gradient onto SH
* photon mapping works quite well in contained scenes like this, however...
Results

19 minutes

* very difficult for photon mapping
* reuse for walk-through animations
Results
5.8 minutes

* can handle heterogeneous media
Results

Single, Surface, and Multiple Scattering
**Results**

20 minutes

- can handle scenes with large extent
- difficult for photon mapping
Results

contrast enhanced

* 8M photons
Contributions
Contributions

- Radiance caching scheme for Part. media:
  - complementary to photon mapping
  - perceptually motivated error metric
Contributions

- Radiance caching scheme for Part. media:
  - complementary to photon mapping
  - perceptually motivated error metric

- Analytic gradient derivations for inscattered radiance:
  - efficient to compute
  - take into account changing properties of medium
Limitations

- Gradient ignores visibility/occlusion changes
- Multiple scattering still costly
Future Work

• Gradient ignores visibility/occlusion changes

• Multiple scattering still costly
  • Terminate recursion using volumetric photon mapping
Thank you