5

Radiance Caching in Participating Media

"A good action is never lost; it is a treasure laid up and guarded for the doer's need."

—Edwin Markham, 1852–1940

In this chapter we develop a novel method for efficiently rendering participating media using Monte Carlo ray tracing. Our approach is inspired by the irradiance caching techniques described in Chapter 3 but applied to light transport within participating media as described in the previous chapter. Our method handles all types of light scattering, including anisotropic scattering, and it works in both homogeneous and heterogeneous media.

5.1 Contributions

A key contribution of this chapter is a technique for computing gradients of radiance in participating media. Computing illumination gradients is of interest for many applications and has been investigated by many researchers [Ward and Heckbert, 1992; Arvo, 1994; Holzschuch and Sillion, 1995, 1998; Igehy, 1999; Annen et al., 2004; Křivánek et al., 2005b; Durand et al., 2005; Ramamoorthi et al., 2007]. However, no previous work has performed a first-order analysis of light within participating media. We derive analytic expressions for the gradients of the radiative transport equation. These gradients take the full path of the scattered light into account, including the changing properties of the medium in the case of heterogeneous media. Moreover, the gradients can be estimated using Monte Carlo ray tracing simultaneously with the in-scattered radiance with negligible overhead.

Our second contribution is a new radiance caching scheme for participating media. We draw inspiration for our approach from irradiance caching methods for surfaces; however, we compute, cache, and reuse estimates of the radiance *within* participating media. This caching scheme uses the information in the radiance gradients to sparsely sample as well as interpolate radiance within the medium, utilizing a novel, perceptually-based error metric. We compute gradients for single scattering from lights and surfaces and for multiple scattering, and we use a spherical harmonics representation in anisotropic media. We further exploit the information in the gradients to improve the accuracy of interpolation. Volumetric radiance caching provides several orders of magnitude speedup compared to path tracing and produces higher quality results than volumetric photon mapping. Furthermore, it is view-driven and well suited for large scenes where methods such as photon mapping become costly.

5.2 Overview

To compute the in-scattered radiance within participating media we use Monte Carlo ray tracing based on a combination of ray marching and random-walk sampling. To simplify notation we ignore the radiance emitted by the volume as it is trivial to compute. We use ray marching to numerically integrate the radiance value seen directly by the observer. Ray marching solves the volume rendering equation by discretizing the 1D integral along a ray using small steps through the medium. Equation 4.27 is approximated as,

$$L(\mathbf{x}\leftarrow\vec{\omega}) \approx T_r(\mathbf{x}\leftrightarrow\mathbf{x}_s)L(\mathbf{x}_s\rightarrow-\vec{\omega}) + \left(\sum_{t=0}^{S-1} T_r(\mathbf{x}\leftrightarrow\mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_i(\mathbf{x}_t\rightarrow-\vec{\omega})\Delta_t\right), \tag{5.1}$$

where Δ_t is the length of each segment along the ray and $\mathbf{x}_0,\ldots,\mathbf{x}_s$ are the sample points for each segment ($\mathbf{x}_0$ is the point where the ray enters the medium and $\mathbf{x}_s$ is a point on a surface past the medium). This approach is illustrated in Figure 5.1.

The ray marching processes samples the in-scattered radiance at discrete points along the ray. The transmittance at each step can be computed incrementally by exploiting the multi-

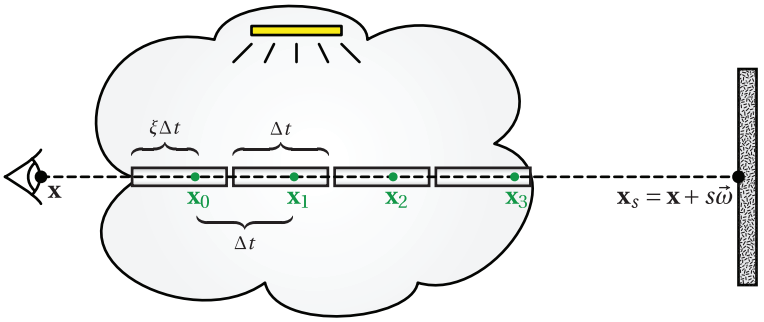

Figure 5.1: Ray marching computes lighting within participating media by dividing the ray into small discrete segments. The lighting and properties of the medium are computed at the sample locations $\mathbf{x}_i$ and assumed constant within each segment. To avoid aliasing, the collection of all samples is offset along the ray by a random amount ξ.

plicative property from Equation 4.10:

$$T_r(\mathbf{x} \leftrightarrow \mathbf{x}_t) = T_r(\mathbf{x} \leftrightarrow \mathbf{x}_{t-1}) T_r(\mathbf{x}_{t-1} \leftrightarrow \mathbf{x}_t). \tag{5.2}$$

To avoid structured aliasing artifacts, the sample points are randomly chosen within the extent of the segments. This can be accomplished using an independent random location within each segment or by randomly offsetting the collection of samples as a whole [Pauly et al., 2000].

The most expensive part to compute in Equation 5.1 is the in-scattered radiance L_i, because it involves integrating over all paths that carry radiance from any other point in the scene to the current sample point $\mathbf{x}_t$ along the eye ray. Our approach therefore focuses on the efficient computation of this term.

It is possible to recursively solve for in-scattered radiance using volumetric path tracing. This approach is, however, extremely costly since it typically requires tracing thousands of paths to obtain noise-free results. Fortunately, the distribution of in-scattered radiance is often smooth in large parts of the medium. We exploit this property by computing in-scattered radiance accurately only at a sparse set of locations and using interpolation whenever possible.

To make the computation more practical, we divide the in-scattered radiance into two components,

$$L_i(\mathbf{x} \rightarrow \vec{\omega}) = L_s(\mathbf{x} \rightarrow \vec{\omega}) + L_m(\mathbf{x} \rightarrow \vec{\omega}), \tag{5.3}$$

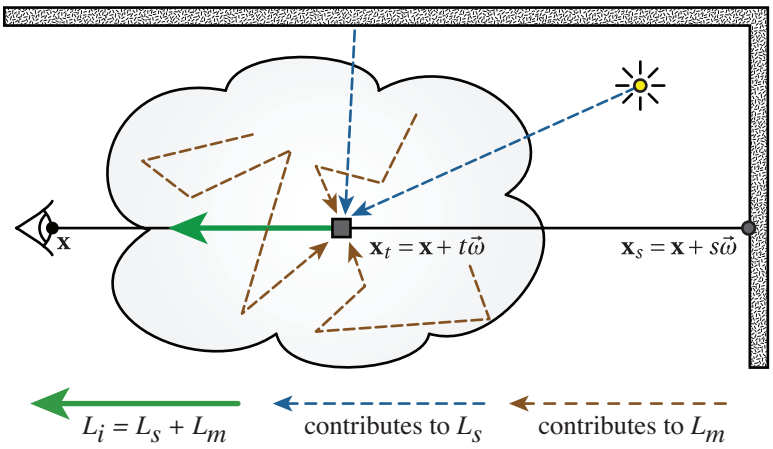

Figure 5.2: Radiance in participating media is computed by our method using a combination of ray marching and random-walk sampling. We split the computation of in-scattered radiance into single scattering, L_s, and multiple scattering, L_m, terms.

where L_s and L_m represent single and multiple scattering, respectively (see Figure 5.2).

The *single scattering term* represents radiance that undergoes a single scattering event along its path from a surface to the eye. Surfaces include light sources and any other surface that reflects light due to direct or indirect illumination. Single scattering may be formulated as an integration over the sphere, or over all surfaces.

The *multiple scattering term* incorporates radiance that scatters at least once in the medium before it reaches **x**. This implies that the path from the surface to the eye includes multiple scattering events. Incident radiance due to multiple scattering is more difficult to evaluate than single scattering, as it leads to a recursive integral.

The distribution of in-scattered radiance is often smooth in large parts of the participating medium. We take advantage of this property by *caching* radiance at a sparse set of locations, and using extrapolation to evaluate the radiance at nearby points in the medium. We show that results without visible errors are achievable with relatively few cached values, resulting in large speedups compared to Monte Carlo ray tracing without caching. Furthermore, the use of caching does not limit the type of media that can be rendered, and our method can easily be incorporated into existing Monte Carlo ray tracers.

We improve the efficiency of the cache by not only storing the value of scattered radiance at each cache point but also its gradient with respect to translation of the point. The gradient is

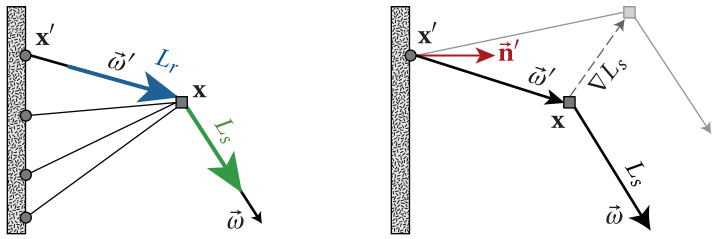

Figure 5.3: Computing the single scattering radiance (left) and gradient (right). For single scattering we distribute samples $\mathbf{x}'$ on the area of the light source. The gradient ∇L_s is computed with respect to translating the evaluation point $\mathbf{x}$ while keeping $\mathbf{x}'$ fixed.

used to estimate the local smoothness and to improve the accuracy of extrapolation of cached values. In the following sections, we explain two separate methods: one to compute the single scattering contribution and its gradient (Section 5.3), and another to compute the multiple scattering contribution and its gradient (Section 5.4). Finally, we detail how to use these quantities in a practical radiance caching algorithm for participating media in Section 5.5.

5.3 Single Scattering

In this section, we describe how to compute the single scattering radiance L_s at a position $\mathbf{x}$ and its gradient ∇L_s with respect to $\mathbf{x}$. The derivation of the gradient is more convenient if we write L_s as an integral over the surfaces of the scene, instead of the standard integral over the sphere:

$$L_s(\mathbf{x}\rightarrow\vec{\omega}) = \int_A p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, L_r(\mathbf{x}\leftarrow\mathbf{x}')\, V(\mathbf{x}\leftrightarrow\mathbf{x}')\, H(\mathbf{x}\leftarrow\mathbf{x}')\, d\mathbf{x}', \tag{5.4}$$

where A denotes surface area and $\vec{\omega}'$ points from $\mathbf{x}$ towards $\mathbf{x}'$. We illustrate our notation in Figure 5.3. The term $L_r(\mathbf{x}\leftarrow\mathbf{x}')$ represents the *reduced radiance*. It is given by the product of the radiance leaving a surface at $\mathbf{x}'$ towards $\mathbf{x}$ and the transmittance through the medium:

$$L_r(\mathbf{x}\leftarrow\mathbf{x}') = T_r(\mathbf{x}\leftrightarrow\mathbf{x}')\, L(\mathbf{x}'\rightarrow\mathbf{x}). \tag{5.5}$$

We also include a *visibility function* $V(\mathbf{x} \leftrightarrow \mathbf{x}')$, whose value is unity if $\mathbf{x}'$ is visible from $\mathbf{x}$; otherwise it is zero. Finally, H is a *geometry term*,

$$H(\mathbf{x} \leftarrow \mathbf{x}') = \frac{\vec{\mathbf{n}}' \cdot \frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}}{||\mathbf{x}-\mathbf{x}'||^2}, \tag{5.6}$$

due to the change of the integration variable, where $\vec{\mathbf{n}}'$ is the surface normal at $\mathbf{x}'$. Note that negative values of the dot product must be clamped to zero, but we omit this in our notation.

The single scattering gradient follows directly from Equation 5.4,

$$\nabla L_s(\mathbf{x} \rightarrow \vec{\omega}) = \int_A \nabla\big(p\, L_r\, V\, H\big)\, d\mathbf{x}' \tag{5.7}$$

$$= \int_A (\nabla p)\, L_r\, V\, H + p\, (\nabla L_r)\, V\, H + p\, L_r\, (\nabla V)\, H + p\, L_r\, V\, (\nabla H)\, d\mathbf{x}', \tag{5.8}$$

where we omit the function arguments for brevity. The symbol ∇ always denotes a gradient with respect to $\mathbf{x}$. Intuitively, the gradient corresponds to the direction of maximum change of the scattered radiance at $\mathbf{x}$ under an infinitesimal motion of $\mathbf{x}$ while keeping $\mathbf{x}'$ fixed. This is illustrated in Figure 5.3.

In the remainder of this section we examine the single scattering computation in more detail. In Section 5.3.1 we discuss how to estimate Equation 5.4 and Equation 5.7 using Monte Carlo integration and describe how to incorporate point light sources in Section 5.3.2. Evaluating the radiance gradient relies on computing the gradient of each individual term in Equation 5.7. We derive the gradient of the geometry term ∇H in Section 5.3.3 and present gradients for common phase functions in Section 5.3.4. Finally, we discuss how to compute the gradient of the reduced radiance term in detail in Section 5.3.5.

5.3.1 Monte Carlo Integration

We compute single scattering radiance values (Equation 5.4) using Monte Carlo ray tracing, which leads to the following formula:

$$L_s(\mathbf{x} \rightarrow \vec{\omega}) \approx \frac{1}{N} \sum_{j=1}^{N} \frac{p\, L_r\, V\, H}{pdf(\mathbf{x}'_j)}. \tag{5.9}$$

Similarly, we use Monte Carlo estimation to evaluate the gradient of single scattering radiance from Equation 5.7:

$$\nabla L_s(\mathbf{x}\rightarrow\vec{\omega}) \approx \frac{1}{N}\sum_{j=1}^{N}\frac{(\nabla p)\,L_r\,V\,H + p\,(\nabla L_r)\,V\,H + p\,L_r\,V\,(\nabla H)}{pdf(\mathbf{x}'_j)}. \tag{5.10}$$

The radiance and the gradient share many of the same terms and are evaluated simultaneously.

Because the visibility function V is discontinuous, its "gradient" is not taken into account. For the purposes of gradient computation, we currently assume constant visibility and discuss the implications of this decision in Section 5.7.

Monte Carlo approximation relies on a set of samples $\mathbf{x}'_j$ that are distributed on the surfaces of the scene according to a probability density function $pdf(\mathbf{x}')$. Surfaces include both area light sources that emit light and any other surface in the scene that may reflect light.

To gather radiance emitted by area light sources, we place samples on their area. Our method allows the use of arbitrary light source sampling techniques to optimize the sample distribution. To account for radiance emitted by all other surfaces, we distribute samples uniformly over the sphere of directions. Using the standard solid angle to area conversion, uniform sampling on the sphere corresponds to a distribution

$$pdf(\mathbf{x}') = \frac{\vec{\mathbf{n}}'\cdot\frac{\mathbf{x}_0-\mathbf{x}'}{||\mathbf{x}_0-\mathbf{x}'||}}{||\mathbf{x}_0-\mathbf{x}'||^2}, \tag{5.11}$$

where $\vec{\mathbf{n}}'$ is the surface normal at $\mathbf{x}'$, and $\mathbf{x}_0$ is the location at which to compute the spherical distribution. Conceptually, $\mathbf{x}_0$ is independent of the location $\mathbf{x}$ where we evaluate in-scattering. Although in practice $\mathbf{x}_0$ and $\mathbf{x}$ usually coincide, $pdf(\mathbf{x}')$ does not depend on $\mathbf{x}$. Hence, it does not contribute to the gradient.

5.3.2 Point Lights

Even though Equation 5.4 is formulated with respect to surface area, it is straightforward to incorporate point lights. We account for the contribution of point lights to the Monte Carlo

estimate of L_s by adding the following summand to Equation 5.9:

$$L_s(\mathbf{x}\rightarrow\vec{\omega}) \mathrel{+}= p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\,\Phi_p\,T_r(\mathbf{x}\leftrightarrow\mathbf{x}')\,V(\mathbf{x}\leftrightarrow\mathbf{x}')\,H_p(\mathbf{x}'\rightarrow\mathbf{x}), \tag{5.12}$$

where Φ_p is the power of the light source and H_p describes the light's fall-off characteristics. For a uniform point light, we have

$$H_p^u(\mathbf{x}\leftarrow\mathbf{x}') = \frac{1}{||\mathbf{x}-\mathbf{x}'||^2}. \tag{5.13}$$

For a Phong light, the fall-off term is

$$H_p^p(\mathbf{x}\leftarrow\mathbf{x}') = \frac{\left(\vec{\mathbf{n}}'\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)^s}{||\mathbf{x}-\mathbf{x}'||^2}, \tag{5.14}$$

where s is the Phong exponent.

The contribution of a point light to the single scattering gradient is:

$$\nabla L_s(\mathbf{x}\rightarrow\vec{\omega}) \mathrel{+}= \Phi_p\left((\nabla p)\,T_r\,H_p + p\,(\nabla T_r)\,H_p + p\,T_r\,(\nabla H_p)\right). \tag{5.15}$$

5.3.3 Gradient of Geometry Terms

Computing the radiance gradient in Equations 5.10 and 5.15 relies on the gradients of several geometry terms. By differentiating the definitions in Equations 5.6, 5.13, and 5.14 we arrive at the following gradient terms:

$$\nabla H(\mathbf{x}\leftarrow\mathbf{x}') = \nabla\left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)\frac{1}{||\mathbf{x}-\mathbf{x}'||^2} + \vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\nabla\left(\frac{1}{||\mathbf{x}-\mathbf{x}'||^2}\right), \tag{5.16}$$

$$\nabla H_p^u(\mathbf{x}\leftarrow\mathbf{x}') = \nabla\left(\frac{1}{||\mathbf{x}-\mathbf{x}'||^2}\right), \tag{5.17}$$

$$\nabla H_p^p(\mathbf{x}\leftarrow\mathbf{x}') = s\left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)^{s-1}\nabla\left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)\frac{1}{||\mathbf{x}-\mathbf{x}'||^2} + \left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)^s\nabla\left(\frac{1}{||\mathbf{x}-\mathbf{x}'||^2}\right), \tag{5.18}$$

where

$$\nabla\left(\vec{\mathbf{n}}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}\right)=\frac{\vec{\mathbf{n}}}{||\mathbf{x}-\mathbf{x}'||}-\frac{(\vec{\mathbf{n}}\cdot(\mathbf{x}-\mathbf{x}'))(\mathbf{x}-\mathbf{x}')}{||\mathbf{x}-\mathbf{x}'||^3}, \tag{5.19}$$

$$\nabla\left(\frac{1}{||\mathbf{x}-\mathbf{x}'||^2}\right)=\frac{-2(\mathbf{x}-\mathbf{x}')}{||\mathbf{x}-\mathbf{x}'||^4}. \tag{5.20}$$

5.3.4 Gradient of Phase Function

In isotropic media, the phase function is a constant, and its gradient vanishes. For anisotropic media, we focus on the Henyey-Greenstein phase function p_{HG} and the Schlick phase function p_S defined in Chapter 4.

These phase functions are 1D, only depending on the cosine of the angle between the incident and outgoing directions. We express $\cos(\theta)$ in terms of $\vec{\omega}$, $\mathbf{x}$, and $\mathbf{x}'$ as:

$$\cos(\theta)=\vec{\omega}\cdot\frac{\mathbf{x}-\mathbf{x}'}{||\mathbf{x}-\mathbf{x}'||}. \tag{5.21}$$

The gradients of the phase functions with respect to $\mathbf{x}$ are then

$$\nabla p_{\mathrm{HG}}(\mathbf{x},\theta)=\frac{3g(1-g^2)}{4\pi(1+g^2-2g\cos\theta)^{2.5}}\nabla\cos\theta \tag{5.22}$$

$$\nabla p_{\mathrm{S}}(\mathbf{x},\theta)=\frac{k(1-k^2)}{2\pi(1+k\cos\theta)^3}\nabla\cos\theta, \tag{5.23}$$

where

$$\nabla\cos(\theta)=\cos\theta\frac{\mathbf{x}-\mathbf{x}'}{\|\mathbf{x}-\mathbf{x}'\|^2}-\frac{\vec{\omega}}{\|\mathbf{x}-\mathbf{x}'\|}. \tag{5.24}$$

5.3.5 Reduced Radiance and Transmittance Gradient

For the sake of computing the gradient of the reduced radiance, we assume diffuse surfaces and light sources. In this case, the gradient of the surface radiance $L(\mathbf{x}' \rightarrow \mathbf{x})$ with respect to $\mathbf{x}$ vanishes. However, it is straightforward to account for general radiance distributions by adding a term for ∇L. With this simplification, the gradient of the reduced radiance is

$$\nabla L_r(\mathbf{x}' \rightarrow \mathbf{x}) = L(\mathbf{x}' \rightarrow \mathbf{x}) \nabla T_r(\mathbf{x}' \leftrightarrow \mathbf{x}),$$

and the gradient of the transmittance is

$$\nabla T_r(\mathbf{x}' \leftrightarrow \mathbf{x}) = -\nabla \tau(\mathbf{x}' \leftrightarrow \mathbf{x}) T_r(\mathbf{x}' \leftrightarrow \mathbf{x}). \tag{5.25}$$

Homogeneous Media. If the medium is homogeneous with constant extinction coefficient σ_t, then the optical thickness is

$$\tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \sigma_t ||\mathbf{x} - \mathbf{x}'||, \tag{5.26}$$

and its gradient is

$$\nabla \tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \sigma_t \nabla (||\mathbf{x} - \mathbf{x}'||). \tag{5.27}$$

Heterogeneous Media. In a heterogeneous medium, to compute optical thickness the extinction needs to be integrated as

$$\tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \int_0^1 \sigma_t(\mathbf{y}(t)) ||\mathbf{x}' - \mathbf{x}|| \, dt, \tag{5.28}$$

where we have parametrized the line segment between $\mathbf{x}'$ and $\mathbf{x}$ as $\mathbf{y}(t) = \mathbf{x} + t(\mathbf{x}' - \mathbf{x})$.

The corresponding gradient is

$$\nabla \tau(\mathbf{x}' \leftrightarrow \mathbf{x}) = \int_0^1 \nabla \left(\sigma_t(\mathbf{y}(t)) ||\mathbf{x}' - \mathbf{x}|| \right) dt. \tag{5.29}$$

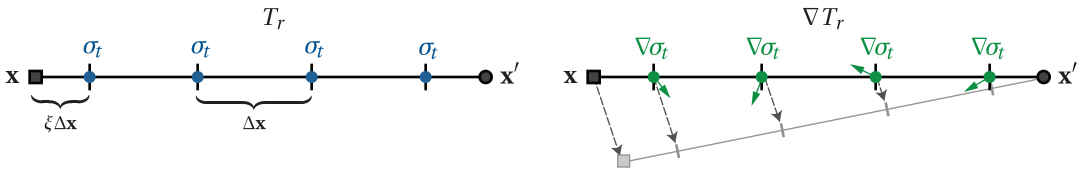

Figure 5.4: In heterogeneous media, the transmittance (left) between two points $\mathbf{x}$ and $\mathbf{x}'$ is evaluated using ray marching with a step size $\Delta\mathbf{x}$ and a random offset ξ. Within each step the extinction coefficient of the medium is assumed to be constant. Our gradient computation not only takes into account the local properties of the medium at $\mathbf{x}$, but also the changing properties along the whole segment between $\mathbf{x}$ and $\mathbf{x}'$. For single scattering, we compute the gradient with respect to translation of $\mathbf{x}$ as $\mathbf{x}'$ is kept fixed.

The optical thickness and its gradient can be expressed using Monte Carlo integration as

$$\tau(\mathbf{x}'\leftrightarrow\mathbf{x}) \approx \frac{1}{N}\sum_j \frac{\sigma_t(\mathbf{y}(t_j))||\mathbf{x}'-\mathbf{x}||}{pdf(t_j)}, \tag{5.30}$$

$$\nabla\tau(\mathbf{x}'\leftrightarrow\mathbf{x}) \approx \frac{1}{N}\sum_j \frac{\nabla\left(\sigma_t(\mathbf{y}(t_j))||\mathbf{x}'-\mathbf{x}||\right)}{pdf(t_j)}. \tag{5.31}$$

We evaluate the two Monte Carlo integrals simultaneously using ray marching with a fixed step size and a single random offset ξ (see Figure 5.4).

At a high level, the evaluation of transmittance aggregates σ_t along a line segment, while its gradient aggregates the contribution of $\nabla\sigma_t$ along the line segment. An important characteristic of this gradient formulation is that it not only takes into account the local properties at $\mathbf{x}$, it incorporates how the properties change along the whole segment between $\mathbf{x}$ and $\mathbf{x}'$. This implies that our gradient computation contains meaningful information about how the transmittance changes even when moving $\mathbf{x}$ out of the line connecting $\mathbf{x}$ and $\mathbf{x}'$.

5.3.6 Isotropic and Anisotropic Scattering

Isotropic Media. In isotropic media, the phase function is a constant $p = 1/4\pi$. Therefore, the scattered radiance and its gradient are independent of the outgoing direction $\vec{\omega}$, and all terms that include the gradient of the phase function ∇p in Equations 5.10 and 5.15 vanish.

Anisotropic Media. For anisotropic media, the scattered radiance is a function of the outgoing direction $\vec{\omega}$, depending on the shape of the phase function. As phase functions are smooth, we can compute a compact representation of the directional radiance distribution. We use a spherical

harmonic expansion, although other spherical functions could be used. For each sample $\mathbf{x}'_j$ with a fixed incident direction $\vec{\omega}'_j$, we compute the spherical harmonic expansion of the phase function as the dot product

$$p(\mathbf{x}, \vec{\omega}'_j \leftrightarrow \vec{\omega}) \approx \sum_{k=1}^{M} y_k(\vec{\omega}) p_{j,k}, \tag{5.32}$$

where y_k are the spherical harmonic functions, and $p_{j,k}$ are the corresponding coefficients. We use a single index for the spherical harmonics to simplify notation.

Applying this formulation to Equation 5.9 gives the following expression for scattered radiance in anisotropic media:

$$L_s(\mathbf{x} \rightarrow \vec{\omega}) \approx \frac{1}{N} \sum_{k=1}^{M} y_k(\vec{\omega}) \sum_{j=1}^{N} \frac{p_{j,k} L_r V H}{pdf(\mathbf{x}'_j)}. \tag{5.33}$$

In order to efficiently evaluate the lighting for arbitrary outgoing directions, we project the entire in-scattered radiance in the above expression onto spherical harmonics. In anisotropic media, we represent the single scattering radiance L_s using a vector of spherical harmonic coefficients, $\Lambda_s = \{\lambda_{s,k}\}$, computed as:

$$\lambda_{s,k} = \frac{1}{N} \sum_{j=1}^{N} \frac{p_{j,k} L_r V H}{pdf(\mathbf{x}'_j)}, \tag{5.34}$$

which allows us to evaluate Equation 5.33 for any direction $\vec{\omega}$ using a simple dot product

$$L_s(\mathbf{x} \rightarrow \vec{\omega}) \approx \sum_{k=1}^{M} y_k(\vec{\omega}) \lambda_{s,k}. \tag{5.35}$$

We apply the same projection mechanism to the gradient computation. We denote the spherical harmonic expansion of the gradient of the phase function as a function over outgoing angle $\vec{\omega}$, by

$$\nabla p(\mathbf{x}, \vec{\omega}'_j \leftrightarrow \vec{\omega}) \approx \sum_{k=1}^{M} p'_{j,k} y_k(\vec{\omega}). \tag{5.36}$$

This leads to the following expression for the gradient:

$$\nabla L_s(\mathbf{x}\rightarrow\vec{\omega}) \approx \frac{1}{N}\sum_{k=1}^{M} y_k(\vec{\omega}) \sum_{j=1}^{N} \frac{p'_{j,k}\, L_r\, V\, H + p_{j,k}\,(\nabla L_r)\, V\, H + p_{j,k}\, L_r\, V\,(\nabla H)}{pdf(\mathbf{x}'_j)}. \tag{5.37}$$

As with radiance, the entire gradient computation is projected onto spherical harmonics, resulting in a vector of gradient coefficients, $\Lambda'_s = \{\lambda'_{s,k}\}$:

$$\lambda'_{s,k} = \frac{1}{N}\sum_{j=1}^{N} \frac{p'_{j,k}\, L_r\, V\, H + p_{j,k}\,(\nabla L_r)\, V\, H + p_{j,k}\, L_r\, V\,(\nabla H)}{pdf(\mathbf{x}'_j)}. \tag{5.38}$$

The radiance gradient can be evaluated in any direction $\vec{\omega}$ by taking the dot product of the coefficient vector Λ'_s with the basis functions, as in Equation 5.35.

5.4 Multiple Scattering

In this section we describe how we calculate the multiple scattering radiance L_m and its gradient ∇L_m. We evaluate multiple scattering using standard Monte Carlo path tracing and compute a Monte Carlo estimate of the gradient of the path tracing integral.

The radiance due to multiple scattering is an integral of radiance arriving from all other points within the participating medium. We express this as

$$L_m(\mathbf{x}\rightarrow\vec{\omega}) = \int_{\Omega_{4\pi}}\int_0^{\infty} p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, T_r(\mathbf{x}'\leftrightarrow\mathbf{x})\, \sigma_s(\mathbf{x}')\, L_i(\mathbf{x}'\rightarrow\vec{\omega}')\, V(\mathbf{x}'\leftrightarrow\mathbf{x})\, dr' d\vec{\omega}', \tag{5.39}$$

where we parametrize the integration domain using polar coordinates $(r',\vec{\omega}')$ centered at $\mathbf{x}$ so $\mathbf{x}' = \mathbf{x} + r'\vec{\omega}'$. The $L_i(\mathbf{x}'\rightarrow\vec{\omega}')$ term is the in-scattered radiance at $\mathbf{x}'$ towards $\mathbf{x}$ and needs to be evaluated recursively. See Figure 5.5 for an illustration.

The gradient of multiple scattering is computed by differentiating Equation 5.39 with respect to $\mathbf{x}$:

$$\nabla L_m(\mathbf{x}\rightarrow\vec{\omega}) = \int_{\Omega_{4\pi}}\int_0^{\infty} p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, \nabla\big(T_r(\mathbf{x}'\leftrightarrow\mathbf{x})\, \sigma_s(\mathbf{x}')\, L_i(\mathbf{x}'\rightarrow\vec{\omega}')\, V(\mathbf{x}'\leftrightarrow\mathbf{x})\big)\, dr' d\vec{\omega}'.$$

Note that with the chosen parametrization the incoming and outgoing directions do not change

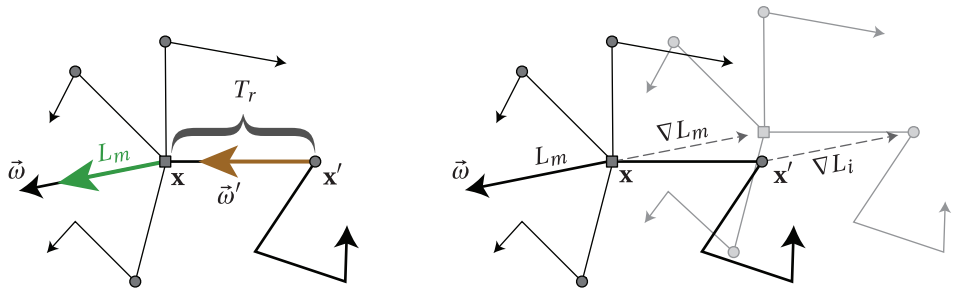

Figure 5.5: Computing the multiple scattering radiance (left) and gradient (right). For multiple scattering we distribute samples $\mathbf{x}'$ by generating random-walk paths starting at $\mathbf{x}$. The gradient ∇L_m is computed by considering the translation of $\mathbf{x}$ and the paths *as a whole*.

under translation of $\mathbf{x}$, so there is no gradient term for the phase function. In homogeneous media, the gradient of the transmittance vanishes. For the general case of heterogeneous media, the gradients of transmittance and optical thickness are computed as in Equation 5.25 and 5.29. However, in the multiple scattering case, $\|\mathbf{x}-\mathbf{x}'\|$ does not change under a translation, and so the $\nabla(\|\mathbf{x}-\mathbf{x}'\|)$ term vanishes. This is illustrated in Figure 5.6.

5.4.1 Monte Carlo Integration

Just as with single scattering, we evaluate the radiance due to multiple scattering and its gradient using Monte Carlo integration. The radiance is estimated by

$$L_m(\mathbf{x}\rightarrow\vec{\omega}) \approx \frac{1}{N}\sum_{j=1}^{N}\frac{p\,T_r\,\sigma_s\,L_i\,V}{pdf(r'_j)\,pdf(\vec{\omega}'_j)}. \tag{5.40}$$

In practice, at the first scattering event we evaluate this expression using N samples. In order to prevent exponential growth, all subsequent scattering events use only one random sample. Hence, Equation 5.40 forms the start of N random-walk paths through the medium, with the polar coordinates $(r'_j,\vec{\omega}'_j)$ as random variables. We sample the radius according to

$$pdf(r'_j) = \sigma_t(\mathbf{x})e^{-\sigma_t(\mathbf{x})r'_j}. \tag{5.41}$$

We distinguish between isotropic and anisotropic media when sampling directions, discussed in more detail below.

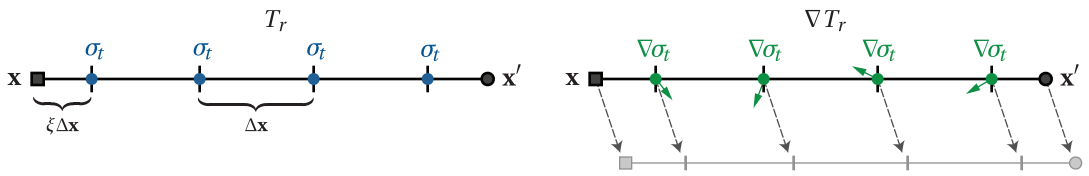

Figure 5.6: For multiple scattering we compute the transmittance (left) using ray marching just like in the single scattering case. The gradient, however, is computed as both endpoints are translated together (right).

The gradient $\nabla L_m(\mathbf{x}\rightarrow\vec{\omega})$ is also found using Monte Carlo sampling:

$$\nabla L_m(\mathbf{x}\rightarrow\vec{\omega}) \approx \frac{1}{N}\sum_{j=1}^{N} \frac{p\left((\nabla T_r)\sigma_s L_i V + T_r(\nabla\sigma_s)L_i V + T_r\sigma_s(\nabla L_i)V\right)}{pdf(r'_j)\,pdf(\vec{\omega}'_j)}. \tag{5.42}$$

Intuitively, the gradient captures the change in radiance as each random-walk path is translated, as a whole, by an infinitesimal amount. This is illustrated in Figure 5.5. As with single scattering, we currently ignore the change in the visibility term.

5.4.2 Isotropic and Anisotropic Scattering

Isotropic Media. For isotropic media, we distribute the directions $\vec{\omega}'_j$ uniformly, i.e., $pdf(\vec{\omega}'_j) = 1/4\pi$.

Anisotropic Media. For anisotropic media, the multiple scattered radiance depends on the outgoing direction $\vec{\omega}$. As with the single scattering contribution, we use spherical harmonics to represent the phase function $p(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})$, which leads to expressions analogous to Equations 5.33 and 5.37. However, there is no term for the gradient of the phase function in multiple scattering, as the angle between path vertices does not change under a translation of the whole path. We project the multiple scattering radiance function and its gradient as vectors of spherical harmonic coefficients $\Lambda_m = \{\lambda_{m,k}\}$ and $\Lambda'_m = \{\lambda'_{m,k}\}$.

We apply importance sampling according to the phase function at all but the first scattering event, since the outgoing direction is given by the direction to the previous path vertex. However, we always sample the direction at the first scattering event uniformly. The reason is that the outgoing direction $\vec{\omega}$ is not known a priori. It will be determined only when the cache point is

evaluated for a certain view direction.

5.5 Algorithm

Using the groundwork laid out in the previous sections, we now describe a radiance caching method for participating media. Our algorithm efficiently computes lighting within participating media by sparsely sampling L_s and L_m and caching them for extrapolation. In addition to computing single and multiple scattering separately as described in Section 5.2, we separate our cache based on the expected frequency content of the radiance values. Specifically, we maintain distinct cache points for single scattering from lights, single scattering from surfaces, and multiple scattering. We call these *single scattering*, *surface scattering*, and *multiple scattering* cache points. This allows our system to sample low-frequency components more sparsely.

The core algorithm for radiance caching in participating media is as follows. For each point in the volume where we need to evaluate in-scattered radiance, we first query the cache for a set of usable cache points. If we find any points, then we use the cached values to extrapolate the radiance to this point, as described in Section 5.5.3. If there are no usable points, then we compute the radiance and its gradient, as described in Sections 5.3 and 5.4, and store these values in the cache. The procedure is performed for all three caches. The total in-scattered radiance is the sum of the three contributions.

5.5.1 Cache Entry Storage

For efficient access to cache points, we use an octree. The data structure and lookup procedure is similar to previous work on surface radiance caching techniques [Ward et al., 1988].

All cache entries store the in-scattered radiance and its gradients at a location in space, as well as a valid radius to later determine if a cache entry is valid for extrapolation at a given query location. Note that the in-scattered radiance is the amount of light that *leaves* a point in a particular direction; thus, we must store the radiance over the full sphere of directions.

Isotropic Media. Since the in-scattered radiance does not depend on direction, each cache entry stores it using three scalar values, one for each color channel. In addition, we compute and

store the gradient for each channel separately for greater accuracy. The total storage requirement for each isotropic entry is 16 floating point numbers. In total, the size of an isotropic cache entry is 48 bytes, though this could easily be reduced using an RGBE [Ward, 1991] or 16-bit floating point representation [Kainz et al., 2006].

Anisotropic Media. In anisotropic media, each Monte Carlo sample of the radiance has a different directional contribution that depends on the shape of the phase function. Because phase functions are generally smooth, we capture this directional dependence using spherical harmonics by caching the spherical harmonic coefficient vectors Λ_s and Λ_m, as well as their corresponding gradient coefficient vectors Λ'_s and Λ'_m.

Our system supports storing spherical harmonic expansions to an arbitrary number of coefficients. However, for scenes with area lights and moderately anisotropic media (e.g., a Henyey-Greenstein phase function with $|g| < 0.6$), we have found that using $M = 9$ coefficients in Equation 5.32 is sufficient to accurately represent the in-scattered radiance. Using nine coefficients, the full RGB representation of the in-scattered radiance requires 27 coefficients. Including the cache sample location and valid radius, this gives 28 floating point values, or 112 bytes per entry, assuming 32-bit values.

5.5.2 Valid Radius and Error Tolerance

Each cache point has a valid radius within which it is a candidate for extrapolation. To compute this radius we have developed a perceptually-based error metric, which adapts to the local smoothness of the radiance field.

To estimate a valid radius for cache points we define the notion of an "optimal radius." The optimal radius for a cache point is the largest radius such that the total relative L_2 error over the extrapolated region is below some error threshold ε. More formally, if $\hat{L}_{x_0}(x)$ is the radiance extrapolated from location x_0 to x, then the optimal radius is given by

$$R_{opt}(\mathbf{x}) = \max_r \left(\frac{\int_{\mathbf{x}\in\Omega_r} \left| L(\mathbf{x}) - \hat{L}_{\mathbf{x}_0}(\mathbf{x}) \right| d\mathbf{x}}{\int_{\mathbf{x}\in\Omega_r} L(\mathbf{x})\, d\mathbf{x}} \leq \varepsilon \right), \tag{5.43}$$

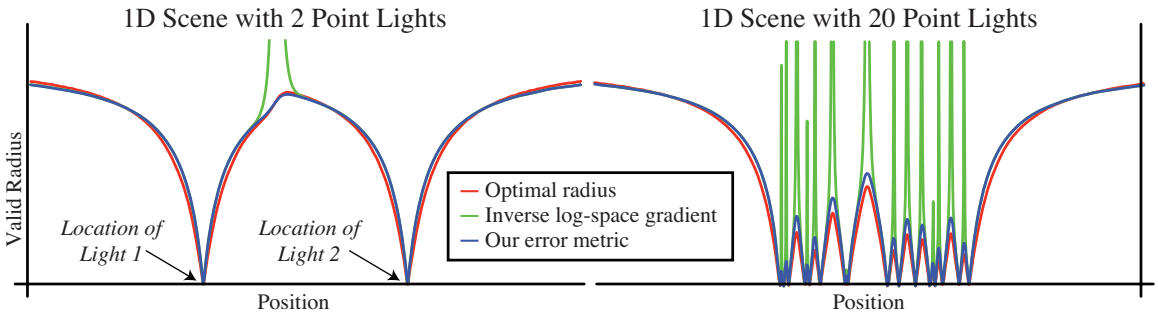

Figure 5.7: Experimental validation of our error metric as compared to a numerically computed optimal radius for a 1D scene with two point lights of differing strengths (left) and 20 equal strength point lights (right). Though the log-space gradient matches the optimal radius in most situations, it produces infinite radii where the gradient vanishes. These singularities become even more prominent with many light sources. In both cases, our perceptually-based error metric robustly approximates the optimal radius even in the presence of such saddle points in the radiance field.

where Ω_r is the spherical domain of radius r around $\mathbf{x}_0$. We use the relative error because the human visual system is more sensitive to contrast than absolute differences in intensity [Glassner, 1995]. Solving for $R_{opt}(\mathbf{x})$ during rendering would be impractical since it requires knowledge of the true radiance within the valid radius of any cache point. Our goal is therefore to devise an error metric which approximates the optimal radius as closely as possible using only information already available from the Monte Carlo samples at $\mathbf{x}$.

The ratio of the radiance gradient to the radiance is a local measure of the contrast. It is also equivalent to the log-space gradient of the radiance:

$$\nabla \ln(L) = \frac{\nabla L}{L}. \tag{5.44}$$

Making the valid radius inversely proportional to the magnitude of the log-space gradient at $\mathbf{x}$ makes sense at an intuitive level: areas where the radiance field has high contrast will be sampled more densely than smoothly varying regions. Furthermore, using the log-space gradient instead of the regular gradient means that the valid radius computation is independent of the absolute intensity of the lighting in the scene.

In our 1D test cases this approach matches the behavior of the optimal radius well at many locations (see Figure 5.7). Unfortunately, the log-space gradient is a completely local quantity and can be a bad predictor of the contrast in the surrounding area. Specifically, the magnitude of the

gradient can drop to zero, causing an infinite valid radius. A zero gradient, or saddle-point, can naturally occur in many situations, e.g., between two lights.

Using the Monte Carlo samples, the reciprocal magnitude of the log-space gradient can be expressed as:

$$\frac{L(\mathbf{x})}{\|\nabla L(\mathbf{x})\|} = \frac{\sum_j L_j(\mathbf{x})}{\left\|\sum_j \nabla L_j(\mathbf{x})\right\|}, \tag{5.45}$$

where j represents the Monte Carlo samples over light sources (for single scattering), surfaces (for surface scattering), and random-walk paths (for multiple scattering). However, instead of using the log-space gradient directly, we use a modified metric designed to eliminate saddle-points. The valid radius of cache points is computed as:

$$R(\mathbf{x}) = \varepsilon \frac{\sum_j L_j(\mathbf{x})}{\sum_j \left\|\nabla L_j(\mathbf{x})\right\|}, \tag{5.46}$$

where ε is a user controlled global error tolerance parameter. By summing over gradient magnitudes instead of gradient directions in the denominator, we prevent the introduction of infinite radii. In our 1D test cases this error metric matches the optimal radius remarkably well (see Figure 5.7) and in practice produces very good cache point distributions.

For colored media, we compute the valid radius for the three color channels independently and use the minimum. In anisotropic media, we use the zero-order spherical harmonic coefficients to compute the radius.

In our implementation, the amount of error in the extrapolation is controlled by the tolerance ε. A tolerance of 0 indicates that values should never be cached or reused. Setting $\varepsilon = 1$ indicates that the extrapolated radiance will be usable until it differs by about a factor of two from the cached radiance value.

In areas that are occluded from all light sources, the radiance due to direct illumination and its gradient are zero. This means that the valid radius of the direct cache is undefined. To remedy this problem, we also compute an *unshadowed* gradient for the single scattering cache by assuming that all light sources are visible. We then choose the smaller of the regular and the unshadowed radii as the valid radius.

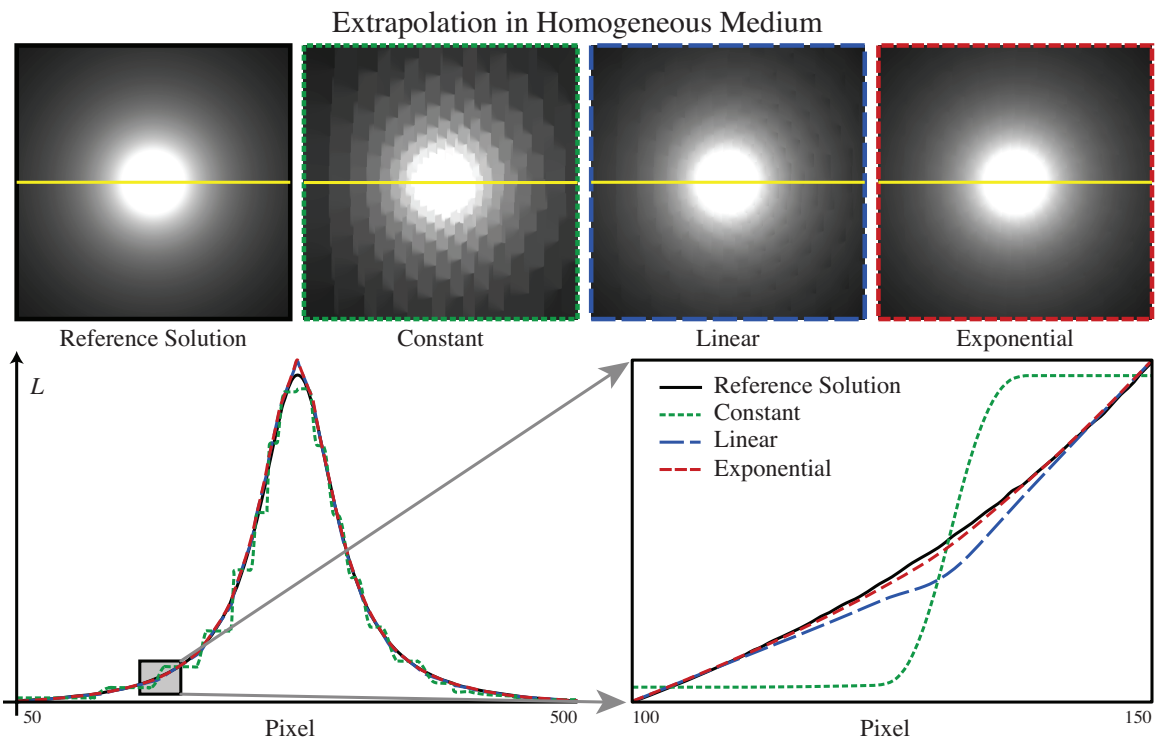

Figure 5.8: A comparison of extrapolation methods for a point-light scene in a homogeneous medium. The top row shows renderings using three different extrapolation methods. The scanlines highlighted in yellow are plotted in the graphs below. Our proposed exponential extrapolation scheme reconstructs the sampled function more accurately than the other methods.

Finally, we set minimum and maximum radii for the cache points in both world and screen space, and clamp gradients according to standard techniques when using radiance caching methods [Křivánek et al., 2006].

5.5.3 The Extrapolated Radiance Estimate

Previous caching techniques for surfaces use constant or linear extrapolation to approximate the radiance at a given location from a set of cached points. Constant extrapolation reuses the value at a cache entry for the new location, while linear extrapolation uses a constant slope to extrapolate the radiance with distance. In participating media, however, light intensity falls off exponentially with distance. We exploit this property in extrapolating the in-scattered radiance from cache points by exponentially extrapolating the cached values. We have found that exponential extrapolation provides a smoother reconstruction of the radiance field than linear extrapolation.

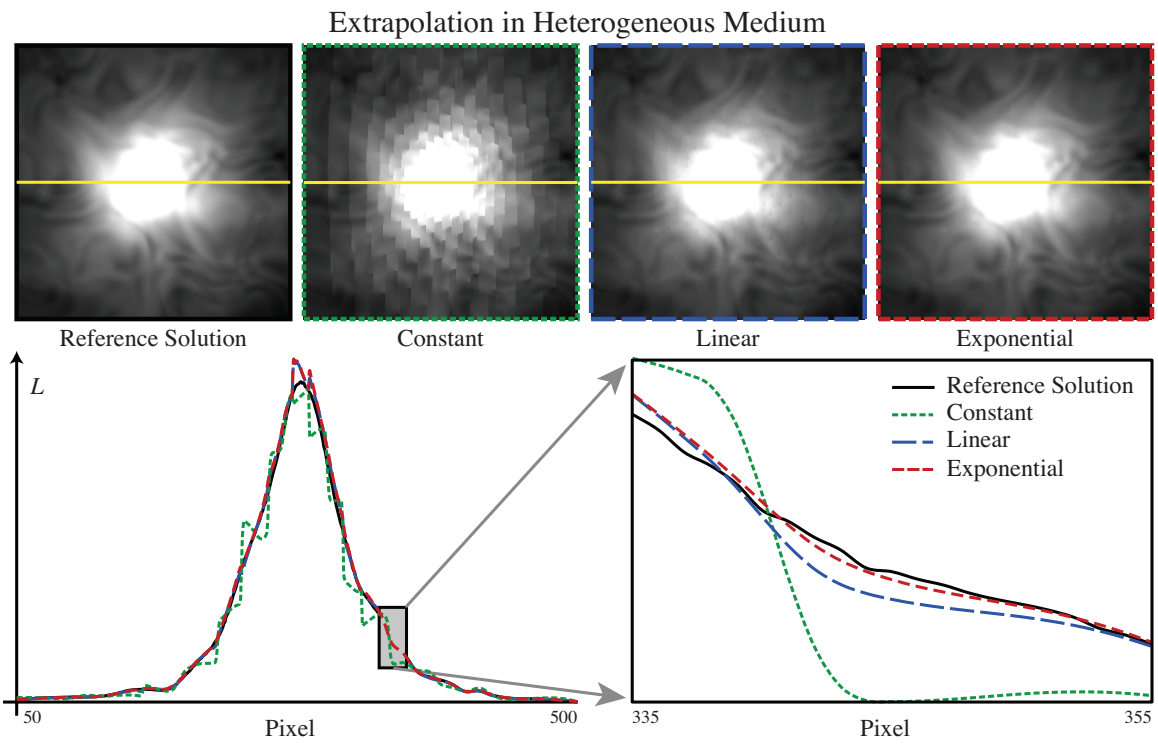

Figure 5.9: A comparison of extrapolation methods for a point-light scene in a heterogeneous medium. The top row shows renderings using three different extrapolation methods. The scanlines highlighted in yellow are plotted in the graphs below. Our proposed exponential extrapolation scheme reconstructs the sampled function more accurately than the other methods.

Exponential Extrapolation. Exponential extrapolation is equivalent to linear extrapolation in log-space. The radiance at a point $\mathbf{x}$ is approximated from a set of cache points C whose valid radii contain our query location by a weighted sum of extrapolated radiance values:

$$L(\mathbf{x}) \approx \exp\left(\frac{\sum_{k\in C}\left(\ln(L_k)+\frac{\nabla L_k}{L_k}\cdot(\mathbf{x}-\mathbf{x}_k)\right)w(d_k)}{\sum_{k\in C} w(d_k)}\right) \tag{5.47}$$

where L_k, ∇L_k, $\mathbf{x}_k$, and r_k are the radiance, gradient, position, and valid radius of cache point k, respectively. The weighting function w is a smooth cubic, $w(d) = 3d^2 - 2d^3$. We define $d_k = 1 - ||\mathbf{x}_k - \mathbf{x}||/r_k$ so that the weighting function has the most influence at the cache location, and this influence falls off smoothly to zero at the valid radius.

Figures 5.8 and 5.9 compare constant, linear, and exponential extrapolation in a 2D slice of a scene containing a point source in a participating medium. The images were rendered using

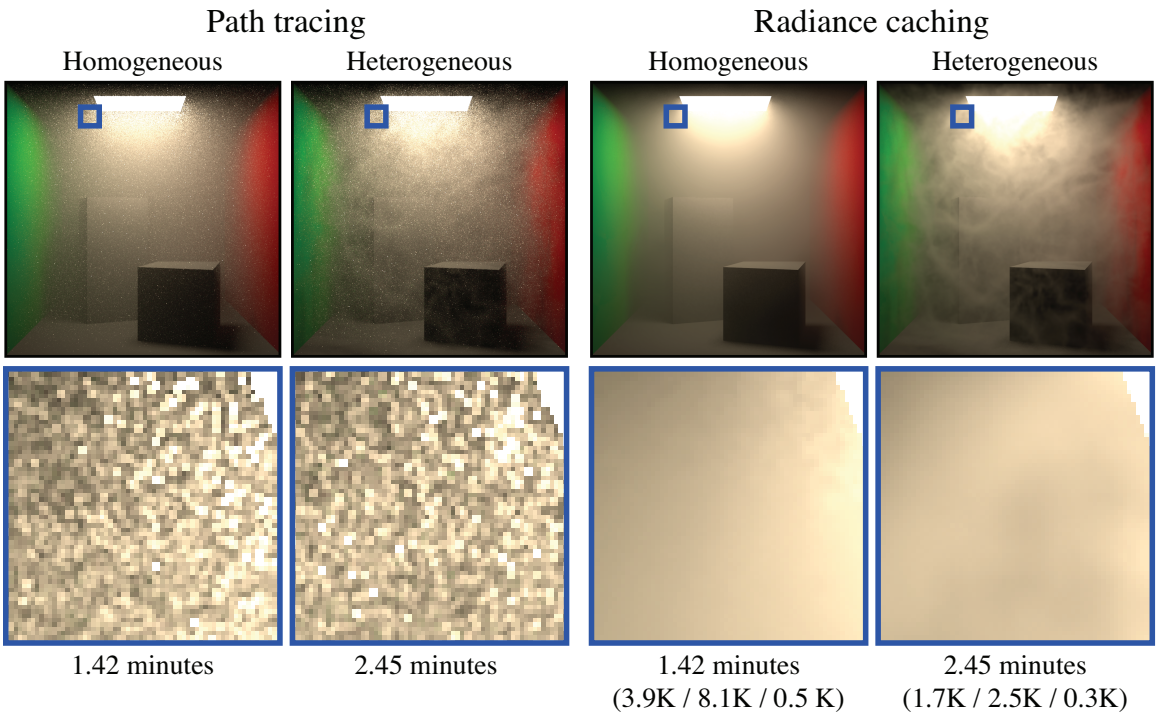

Figure 5.10: A Cornell box filled with isotropic smoke. We perform an equal-time comparison of our method to path tracing and perform a full lighting simulation including single scattering from lights, single scattering from surfaces, and multiple scattering. Render times are reported underneath each image, with the number of cache points in the single, surface, and multiple scattering caches in parentheses. A visualization of the cache points used to render the far left image is shown in Figure 5.12.

the same cache point locations; the only difference is the method used to extrapolate the radiance. Constant extrapolation is clearly a poor choice for extrapolation in participating media, as it gives strong discontinuity artifacts from the jumps between points, while linear extrapolation does a somewhat better job at approximating the change in radiance. Exponential extrapolation most precisely matches a reference solution with almost no visible artifacts, as it more closely approximates the real change in radiance over distance.

5.6 Results

We implemented the volumetric radiance caching algorithm in a Monte Carlo ray tracer, and all our timings were made on an Intel Core 2 Duo 2.4 GHz machine using only one core. We performed comparisons between path tracing, radiance caching, and photon mapping. Since each rendering technique has its own set of parameters, we performed equal-time comparisons

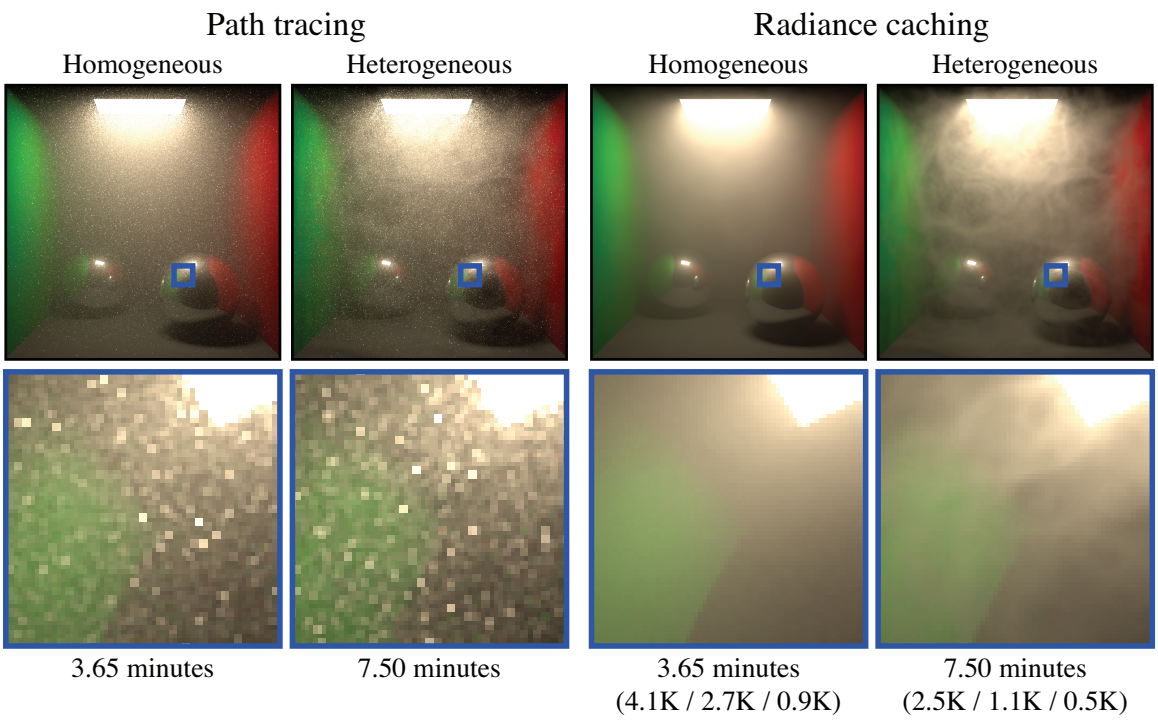

Figure 5.11: A Cornell box filled with anisotropic smoke. We perform an equal-time comparison of our method to path tracing and perform a full lighting simulation including single scattering from lights, single scattering from surfaces, and multiple scattering. We use a Henyey-Greenstein phase function with $g = 0.4$. Render times are reported underneath each image, with the number of cache points in the single, surface, and multiple scattering caches in parentheses.

while hand-picking the best parameters for each algorithm. All three methods make full use of standard optimization techniques including Russian roulette, stratification, and importance sampling, where applicable. All images were rendered at 1K horizontal resolution with up to 16 samples per pixel.

Figures 5.10 and 5.11 show a series of images of a Cornell box filled with smoke. We compare our method to path tracing for a variety of participating media types. Render times range from around 1.5 minutes for homogeneous media with isotropic scattering, to 7.5 minutes in heterogeneous media with anisotropic scattering. In all four examples, path tracing exhibits significant noise for the same render time. In the heterogeneous images, the scattering properties are defined procedurally using several octaves of noise [Ebert et al., 2002]. For anisotropic scattering, by projecting the full directional radiance field onto spherical harmonics, we are able to correctly reuse cached values even for reflection rays on the mirror spheres. A visualization of the radiance cache point locations from the homogeneous/isotropic image is shown in Figure 5.12.

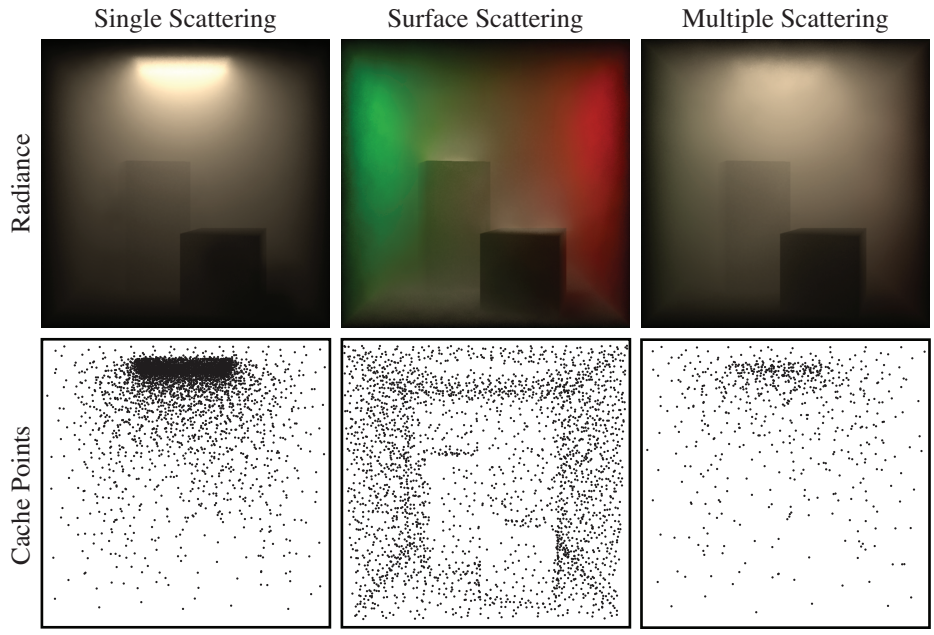

Figure 5.12: The bottom row shows a visualization of the single, surface, and multiple scattering cache points using the radiance cache from the homogeneous image in Figure 5.10. The top row shows the corresponding extrapolated radiance components (the radiance from surfaces past the medium is not included). For display purposes, the multiple and surface scattering images have been raised by one and three f-stops respectively.

Figure 5.13 shows a frame from an animation moving through light beams in the Sponza atrium. In this scene, all lighting is provided using a single point light, so we do not cache single-scattering but compute it directly. Computing multiple scattering is particularly challenging in this scene and requires tracing thousands of paths to estimate the in-scattered radiance. Our caching method is able to produce a noise-free result in 19 minutes. For walk-through style animations where only the camera position changes, we are able to reuse the radiance cache for subsequent frames, achieving an even greater speedup. Sponza is an extremely difficult scene for photon mapping. We tried performing an equal-time comparison to photon mapping. However, even by using projection maps, tracing photons for over 40 minutes produced less than 1K usable photons.

Figure 5.14 shows a frame from an animation of evolving smoke, which took less than six minutes to generate. In this scene, each frame is rendered using a separate cache. Our method is able to produce flicker-free animations even when the cache is recomputed at each frame. With the same render time the path tracing solution exhibits visible noise, which flickers significantly

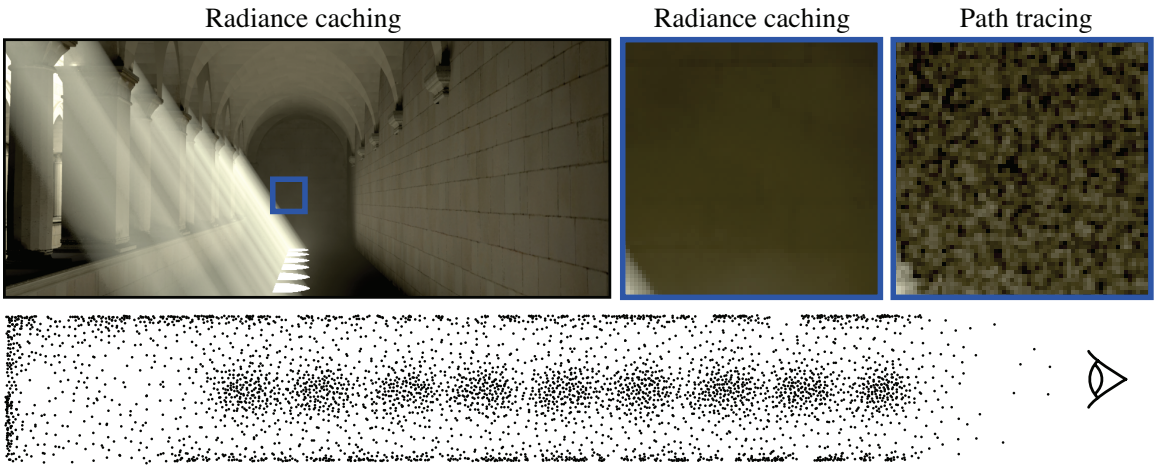

Figure 5.13: The Sponza atrium with beams of light and multiple scattering. With volumetric radiance caching, this scene renders in 19 minutes using about 46K cache points and 79M cache queries. The zoomed insets on the right provide an equal-time comparison to path tracing. The bottom image visualizes the surface cache using a horizontal cross section near the floor of the hallway. The camera is located on the right looking towards the left. Note the concentration of cache points near the brightly illuminated regions of the floor. The radiance caching image was rendered with error tolerances set at 0.5 and 0.01 for the surface and multiple scattering caches, respectively. For multiple scattering, our method traces 512 random-walk paths per cache point.

when animated. For this scene, we store about 9K cache points.

Figures 5.15 and 5.16 show a scene of two cars on a foggy road. This scene has a large extent and is costly to render with methods such as photon mapping. Here, our caching method fully exploits the smoothness in the radiance across the scene. In the same render time, we are able to attain significant noise reduction compared to path tracing and a smoother reconstruction than photon mapping using over eight million photons.

For all example scenes in this chapter, a large portion of render time is spent querying for cache points within the octree (typically between one third and one half of total render time). This is due to the large number of queries performed while ray marching through the medium (in the millions) compared to the number of cache points created (in the thousands). However, for the multiple scattering computation in our example scenes, we trace between 2K and 8K random-walk paths. Due to this, it is important to note that computing a new cache point takes about 100 to 1,000 times longer than querying within the octree.

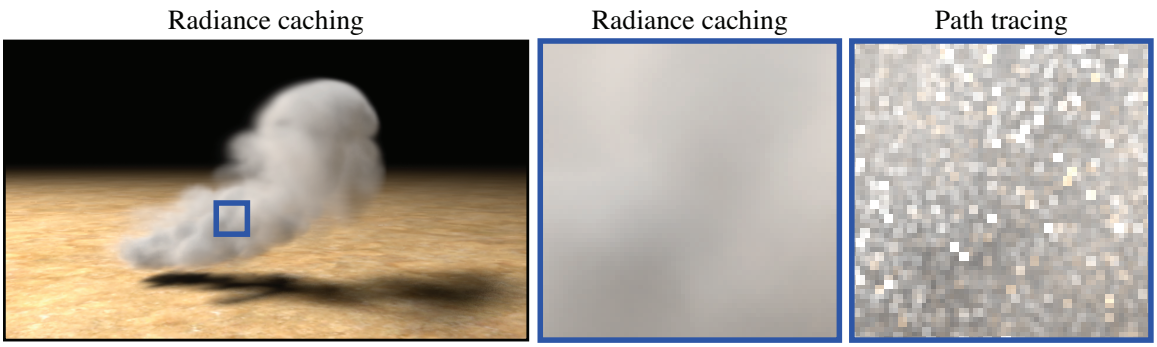

Figure 5.14: This still frame from an animation of heterogeneous smoke renders in 5.8 minutes using 9K cache points and 8.2M cache queries. Our method is able to produce flicker-free animation even when recomputing a new cache for each frame. The highlighted region is shown zoomed-in on the right, providing an equal-time comparison to path tracing. Error tolerances of 0.05, 1.0, and 1.0 were used for the single, surface, and multiple scattering caches with 2K multiple scattering paths per cache point.

5.7 Summary and Discussion

Many promising avenues remain for combining the strengths of photon mapping and radiance caching. Since photon mapping does not attempt to accelerate the computation of single scattering, radiance caching could effectively accelerate this computation within a photon mapping renderer. Another promising extension is to use our caching technique as a final gather pass for photon mapping. Currently, in highly scattering scenes with many occluders, a large number of multiple scattering rays are necessary using our method. The density estimation in photon mapping provides a natural smoothing, which could reduce the number of multiple scattering rays necessary and improve the cached result. Furthermore, photon mapping is able to efficiently simulate multiple scattering in a variety of scenes because it is well suited for detecting light paths difficult to sample from the eye. However, for smooth reconstruction a large number of photons are required. Performing a final gather would also reduce the necessary number of stored photons and improve the reconstruction quality as compared to regular photon mapping. For isotropic media, final gather optimizations developed for surfaces [Christensen, 1999] could be extended for volumetric final gather. Information from the photon map could also be used in other ways – for instance, to provide an importance sampling function for computing the direct and indirect in-scattered illumination.

One of the main challenges for our technique stems from ignoring the visibility gradient.

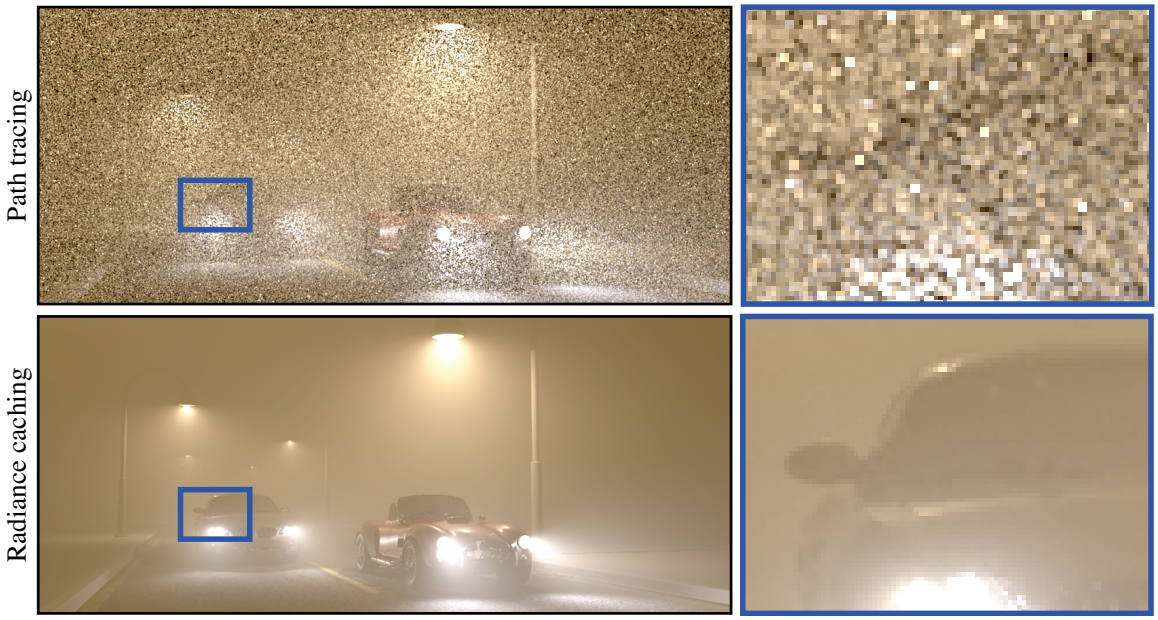

Figure 5.15: Two cars in a dense fog on a road illuminated by 60 lights. Our radiance caching method renders the cars scene in 20.35 minutes using only 175K cache points and 27M cache queries. The multiple scattering computation traces 4K random-walk paths per cache point. The highlighted region is shown zoomed-in on the right, providing an equal-time quality comparison between radiance caching and path tracing. The error tolerances were set at 0.25 and 0.75 for the single and multiple scattering caches, respectively.

Due to this simplification, our error metric is unable to adapt sample density at volumetric shadow boundaries, forcing the user to set a lower global error tolerance to produce artifact-free renderings. Incorporating the visibility gradient, both for extrapolation and in the error metric, could significantly increase quality while reducing render times. We plan to address this limitation in future work.

Caching the radiance and gradient computation using spherical harmonics is effective for moderately anisotropic scattering. For highly directional scattering, this approach becomes less beneficial. Since the cache points can potentially be reused from any viewing direction, our error metric does not account for the outgoing direction when computing the valid radius. An alternate approach could be to store radiance values in a 5D cache with associated outgoing directions. Since, in this case, cache points would only be reused for similar outgoing directions, importance sampling could be used even at the first bounce for multiple and surface scattering. Such an approach would benefit significantly from a full 5D error metric that computes a valid radius independently for the spatial and angular dimensions.

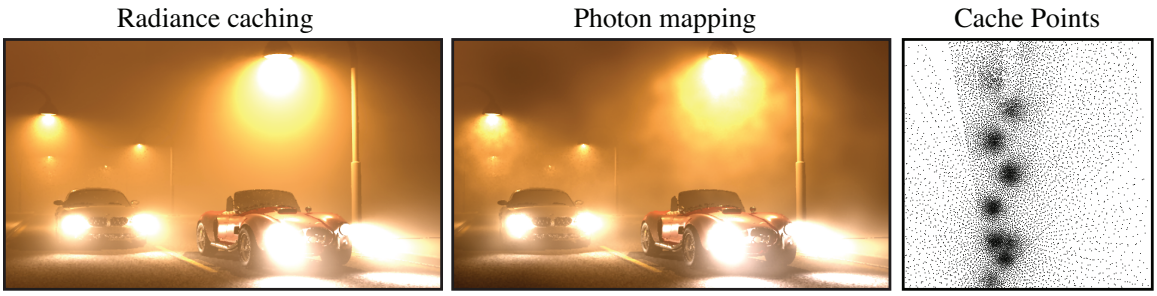

Figure 5.16: A equal-time, contrast-enhanced comparison between radiance caching and photon mapping in the cars scene from Figure 5.15. The scene has a large extent, making it difficult to render with methods such as photon mapping that compute scattering in the entire medium. Artifacts remain in the *homogeneous* medium even after storing 8M photons. On the right we show a horizontal cross section of the single scattering cache points as seen from above. In this visualization, the camera is located at the bottom looking up. Note the concentration of cache points around the street lights and headlights near the camera. The error tolerances were set at 0.25 and 0.75 for the single and multiple scattering caches, respectively.

Although our method caches the results of integrating single and multiple scattering, other expensive quantities could also be cached to further improve performance. For scenes with a large visible extent and large image resolutions, cache queries dominate the render time due to the repeated lookups necessary for integrating radiance along eye rays. Caching the radiance integration along eye rays could provide for efficient reuse of the *whole* radiance computation from neighboring pixels, completely eliminating the eye integration for a majority of the image.

Finally, the original surface irradiance caching algorithm could be extended to account for the presence of participating media and coupled with our method. Since, however, there is no distance to surfaces associated with volumetric cache points, this would be a challenging undertaking. Also, the irradiance gradients would have to account for the change in optical properties along a path as the point moves on the surface, similar to our scattering gradients.

5.8 Conclusion

In this chapter we presented a new method for caching radiance in the presence of participating media. The method is general and adapts to scenes with arbitrary scattering and absorbing properties. It works directly in both homogeneous and heterogeneous media, and can handle both isotropic or anisotropic scattering. When rendering anisotropic media, we compactly store the directional distribution of the radiance using spherical harmonics. Our method caches

both single and multiple scattered radiance from light sources and surfaces while providing significant gains in speed over traditional Monte Carlo ray tracing techniques.

Volumetric radiance caching is founded on using the gradients of a Monte Carlo formulation of radiative transport to accurately predict the change in radiance throughout the scene. We have shown how to compute these gradients and also how to use them to extrapolate the radiance to new locations. This, combined with an exponential extrapolation technique that exploits the properties of light in a medium, allow us to render high-quality images of scenes with a relatively small number of cache points. Caches produced by our method handle complex illumination and can be later reused for animations.

5.9 Acknowledgements

The material in this chapter is, in part, a reproduction of the material published in Wojciech Jarosz, Craig Donner, Matthias Zwicker, and Henrik Wann Jensen. “Radiance Caching for Participating Media.” In *ACM Transactions on Graphics*, 27(1):1–11, 2008. The dissertation author was the primary investigator and author of this paper. This work was supported in part by NSF grant CPA 0701992.