


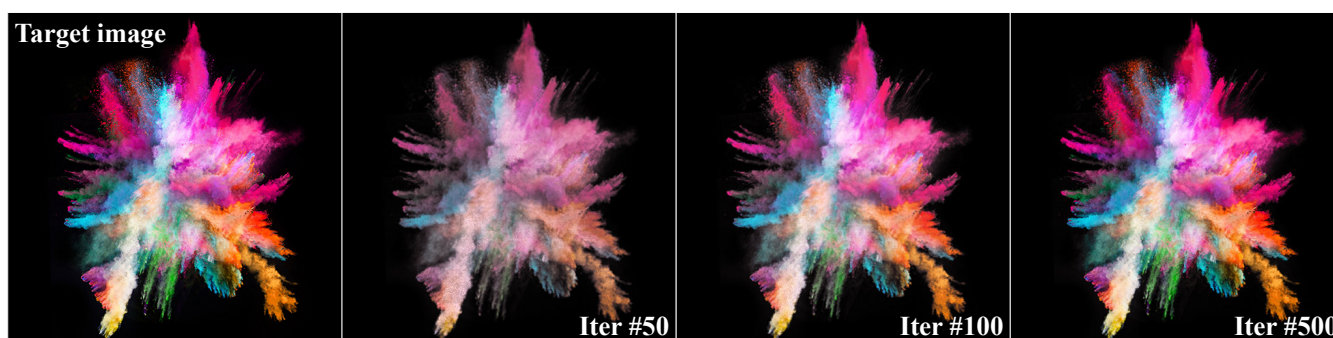


# A Differential Diffusion Theory for Participating Media

Yunchi Cen<sup>1</sup> , Chen Li<sup>1</sup>, Frederick W. B. Li<sup>2</sup> , Bailin Yang<sup>3</sup>, and Xiaohui Liang<sup>1</sup> 

<sup>1</sup>State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, China  
<sup>2</sup>University of Durham, United Kingdom, <sup>3</sup>Zhejiang Gongshang University, China



**Figure 1:** Single-view reconstruction of a complex colored smoke scene using our proposed method. Our method involves joint optimization of the density field, optical coefficient field, and illumination parameters, requiring the computation of over 49 million derivatives. Each iteration takes an average of 8 seconds to compute. After 500 iterations, the rendered image closely converges to the target image.

## Abstract

We present a novel approach to differentiable rendering for participating media, addressing the challenge of computing scene parameter derivatives. While existing methods focus on derivative computation within volumetric path tracing, they fail to significantly improve computational performance due to the expensive computation of multiply-scattered light. To overcome this limitation, we propose a differential diffusion theory inspired by the classical diffusion equation. Our theory enables real-time computation of arbitrary derivatives such as optical absorption, scattering coefficients, and anisotropic parameters of phase functions. By solving derivatives through the differential form of the diffusion equation, our approach achieves remarkable speed gains compared to Monte Carlo methods. This marks the first differentiable rendering framework to compute scene parameter derivatives based on diffusion approximation. Additionally, we derive the discrete form of diffusion equation derivatives, facilitating efficient numerical solutions. Our experimental results using synthetic and realistic images demonstrate the accurate and efficient estimation of arbitrary scene parameter derivatives. Our work represents a significant advancement in differentiable rendering for participating media, offering a practical and efficient solution to compute derivatives while addressing the limitations of existing approaches.

## CCS Concepts

• **Computing methodologies** → Volumetric models; • **Mathematics of computing** → Partial differential equations;

## 1. Introduction

Differentiable rendering (DR) is a pivotal area of research in computer graphics, playing a crucial role in computing derivatives for arbitrary scene parameters and finding applications in computer graphics, computer vision, and machine learning. The integration

of differentiable rendering in neural network pipelines for learning-based 3D estimation tasks has bridged the gap between 2D and 3D processing, empowering neural networks to optimize 3D entities using 2D projections. However, the computation of derivatives for scene parameters remains challenging due to the complex and nonlinear relationship between pixel intensities and scene parameters. Despite advancements in physics-based differentiable renderers that rely on ray tracing and Monte Carlo estimation for realistic

† Corresponding author: liang\_xiaohui@buaa.edu.cn

image optimization, the computation of derivatives remains highly expensive, particularly for scenes involving participating media. Resolving these challenges is essential to overcome current limitations and advance differentiable rendering techniques, unlocking new possibilities for realistic image generation and enhancing learning-based 3D estimation tasks.

Many natural environments contain participating media, such as fog, smoke, clouds, or dust, presenting challenges for accurately simulating light propagation due to the complexity of modeling scattering events. The Radiative Transfer Equation (RTE), widely utilized in fields like astrophysics, remote sensing, and biomedical imaging [Cha60, HvdH81], has been introduced to computer graphics [Bli82, KVH84]. In computer graphics, the RTE describes the forward rendering process and has been extended to handle light transport effects. Researchers have further expanded the RTE's capabilities to incorporate differentiable rendering techniques. Zhang et al. [ZWZ\*19] proposed a differential theory of radiative transfer, accompanied by the Monte Carlo algorithm, to compute derivatives of scene parameters, enabling differentiable rendering for participating media. However, this method involves computationally expensive processing, employing Monte Carlo estimation for the radiance of multiply-scattered light and RTE derivatives. Despite the existence of physically-based differentiable rendering methods reported in [ZJL20], their performance often falls short of meeting the interactivity requirements of many graphics applications.

Efforts have been made to improve the efficiency of derivative computation for participating media. Nimier-David et al. [ND-SRJ20] introduced the *Radiative Backpropagation (RB)* method, utilizing an adjoint approach that significantly reduces memory usage to a constant footprint and enhances efficiency. Building upon RB, Vicini et al. [VSJ21] proposed the *Path Replay Backpropagation (PRB)* method, capable of computing unbiased reverse-mode derivatives of volume transport with linear time complexity. However, these speed-up methods primarily focus on accelerating derivative computation within traditional differentiable rendering frameworks, such as path tracing and backpropagation, without addressing the fundamental challenges of derivative computation for participating media. The lack of a new theoretical framework that enables highly efficient derivative computation of multiple scattering lights hinders radical improvements in computation efficiency.

In the realm of participating media rendering, efficiently computing the radiance and its derivatives for multiply-scattered light poses a significant computational challenge, particularly in dense media where scattering paths proliferate. To address this problem, we propose a groundbreaking concept called *differential diffusion theory*, inspired by the examination of participating media rendering grounded in diffusion theory [Sta95, ZRL\*08, KPS\*14]. This novel framework enables the differentiable rendering of participating media, facilitating the efficient computation of various derivatives, including volumetric density fields, absorption and scattering coefficients, and asymmetry parameters of phase functions. In contrast to prior approaches, we leverage a partial differential equation solver to estimate the radiance and its derivatives for multiply-scattered light. This approach offers a substantial computational advantage compared to traditional volumetric path tracing with Monte Carlo sampling techniques. Notably, our frame-

work represents the first differentiable rendering framework capable of computing arbitrary derivatives of scene parameters based on the diffusion approximation. Leveraging the power of the *Taichi language* [HLY\*21] for implementation, our approach sets itself apart as the inaugural differentiable renderer capable of achieving real-time performance on density optimization tasks. We make our implementation available at [https://github.com/cenyc/differential\\_diffusion](https://github.com/cenyc/differential_diffusion) to promote further research into efficient rendering, especially for complex multiple scattering in participating media. Our key contributions are:

- Introducing the pioneering differentiable rendering framework for participating media, leveraging the diffusion approximation. Notably, our approach surpasses previous methods in terms of computational efficiency, enabling remarkable speed gains and even achieving real-time performance.
- Deriving the differential forms of the Radiative Transfer Equation (RTE) based on the diffusion approximation, thereby enabling differentiation with respect to arbitrary derivatives of scene parameters. This advancement extends the scope of differentiable rendering techniques, facilitating more comprehensive and accurate derivative computations.
- Deriving the discrete form of the diffusion equation derivatives, facilitating rapid computation through numerical methods. This development enhances the overall efficiency of our framework, enabling efficient and precise estimation of scene parameter derivatives.

## 2. Related Work

### 2.1. Diffusion Theory in Graphics

Diffusion theory serves as a fundamental framework in our research due to its ability to address the computational challenges associated with rendering multiply-scattered light in dense participating media. Stam [Sta95] introduced diffusion theory (DT) to computer graphics, enabling the efficient simulation of heterogeneous media by solving a discrete form of the diffusion equation on a grid. However, classical diffusion approximation faces limitations in transparent regions, leading to non-physical radiative fluxes and inaccuracies in light transport. To overcome this, Koerner et al. [KPS\*14] introduced *flux-limited diffusion*, which modulates the diffusion coefficient to improve accuracy in low-density or high-albedo regions. Inspired by these advancements, our method incorporates flux-limited diffusion for rendering thin smoke, considering it as a variant of classical diffusion approximation. In this paper, we establish classical diffusion approximation as the theoretical foundation of our approach, with the potential for easy extension to other existing variants. Moreover, diffusion theory has also been developed for fast simulation of subsurface scattering. Donner and Jensen [DJ05] extended previous work on light diffusion in translucent materials, presenting a multipole diffusion approximation for light scattering in thin slabs. D'Eon and Irving [dl11] introduced a modified diffusion theory that offers enhanced accuracy for highly absorbing materials and near the point of illumination. By leveraging diffusion theory, we establish a robust and versatile framework for efficient rendering, addressing the complexities of multiple scattering lights in participating media.

## 2.2. Differentiable Rendering

Prior work in differentiable rendering has made important contributions, yet limitations remain. Approaches such as OpenDR [LB14] and its subsequent works [CGL\*19, GCM\*18, HF18, KUH18, LCLL19, PBDCO19] approximate the backward pass or rasterization process of the graphics pipeline, focusing on primary visibility without accounting for complex indirect effects. Consequently, generating photorealistic images that capture the intricate interactions of light, geometry, and materials becomes challenging. On the other hand, physically based differentiable rendering techniques aim to simulate light transport and scattering using Monte Carlo estimation to compute pixel color and its derivatives. Li *et al.* [LADL18] introduce a differentiable ray tracing framework, enabling the computation of derivatives with respect to input parameters. However, their method lacks support for scenes containing participating media. Another physically-based renderer, Mitsuba 2 [NDVZ19], relies on differentiable volumetric path tracing and automatic differentiation (AD) for derivative evaluation. Nonetheless, the AD method requires substantial memory usage to record gradients, limiting its optimization capabilities for high-resolution scattering media. These deficiencies motivate the need for our proposed research, which addresses the challenges of differentiable rendering, particularly in scenes involving participating media, and provides a more efficient and comprehensive solution.

Derivative estimation for participating media poses significant challenges in differentiable rendering. Early works [GZB\*13, GLZ16] focused on inverse transport simulations to estimate the material properties of volumes. Zhang *et al.* [ZWZ\*19] introduced a differential theory of radiative transfer along with an unbiased Monte Carlo algorithm for differentiating surfaces and volumes, addressing the issue of discontinuities. They further proposed a method to estimate derivatives of the path integral formulation [Vea97] in [ZMY\*20]. However, physically-based differentiable rendering suffers from poor performance, particularly in rendering and differentiating participating media, making it unsuitable for interactive graphics applications. Nimier-David *et al.* [NDSRJ20] introduced the *radiative backpropagation* approach, treating differentiable rendering as a reversed light transport problem. Although this approach avoids intermediate state storage, the computation time for unbiased gradient estimation becomes quadratic in the number of scattering events along a light path, resulting in high computational costs. Vicini *et al.* [VSJ21] proposed the *Path Replay Backpropagation* approach to improve performance, achieving linear computation time in the number of scattering events. Recently, Jakob *et al.* [JSRV22] introduced *Mitsuba 3*, a new version of the Mitsuba renderer based on *Dr.Jit*. Although *Mitsuba 3* is one of the prominent differentiable renderers, its volumetric path tracing framework, which relies on Monte Carlo estimation for derivative computation, can be time-consuming and become a performance bottleneck in differentiable volume rendering.

Moreover, to handle complex geometries and light transport effects more efficiently, Zhang *et al.* [ZYZ21] proposed a generalized differential path integral approach that captures both interfacial and volumetric light transport. To solve the problem of computing derivative for parameters that affect visibility, Loubet *et al.* [LHJ19] proposed a new technique for differentiating path-traced images

with respect to scene parameters that affect visibility, such as light sources and the camera positions, allowing to product gradients at low sample counts with low bias. Nimier *et al.* [NDMKJ22] proposed differential ratio tracing for volumes, to achieve unbiased sampling of gradient computation. For a comprehensive review of differentiable rendering, please refer to [KBM\*20, ZJL20].

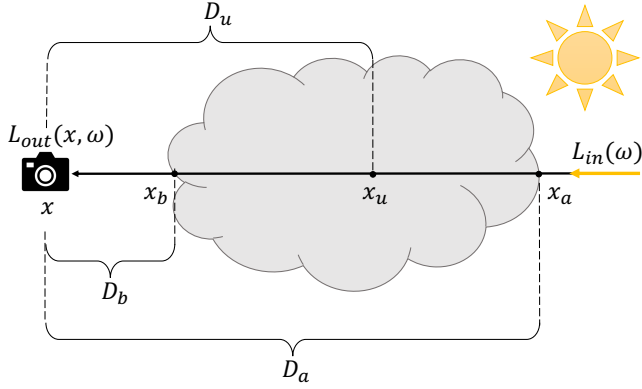
Our method stands out from previous approaches in both theory and implementation, offering significant novelty. The foundation of our approach lies in the novel concept of differential diffusion theory, which revolutionizes the computation of derivatives for multiple scattering radiance, enabling efficient rendering. In terms of implementation, we depart from the conventional Monte Carlo method for estimating radiance and its derivatives. Instead, we employ a ray marching technique combined with numerical methods, resulting in a highly efficient computational process.

Table 1: List of symbols commonly used in this work.

Symbol	Meaning
$L_{in}(\omega)$	the radiance of ambient light along direction $\omega$ .
$L_{out}$	outgoing radiance.
$L_d$	media radiance.
$J_{ms}, J_{ss}$	single scattering and multiple scattering terms.
$L_{ri}$	reduced incident radiance.
$L_{ms}, L_{ss}$	single scattering and multiple scattering radiance.
$L_d^0, \bar{L}_d^1$	the first two terms of the Taylor expansion of $L_d$ .
$Q_{ri}$	the radiance due to the first scatter of $L_{ri}$ .
$Q_{ri}^0, \bar{Q}_{ri}^1$	the first two terms of Taylor expansion of $Q_{ri}$ .
$x$	the position of the viewpoint.
$x_u$	the position inside the volume.
$x_a, x_b$	the farthest and the nearest intersections from view point along direction $-\omega$ .
$\kappa_a, \kappa_s, \kappa_t$	absorption, scattering, and extinction coefficients.
$\sigma_a, \sigma_s, \sigma_t$	absorption, scattering, and extinction cross sections.
$\Omega, \rho(x)$	scattering albedo and density of medium.
$\sigma_r$	transport cross section.
$\tau(x_i, x_j)$	transmittance from $x_i$ to $x_j$ .
$D_a, D_b, D_u$	the distance between $x$ and $x_a, x_b, x_u$ , respectively.
$\bar{\mu}$	the first moment of a phase function.
$p(\omega, \omega')$	a phase function with incident direction $\omega'$ and outgoing direction $\omega$ .
$\langle \cdot, \cdot \rangle$	indicates the inner product between two vectors.
$\pi$	an arbitrary scene parameter.

## 3. Diffusion Approximation of RTE

We provide a brief overview of light transport in participating media and introduce the diffusion approximation of multiple scattering radiance, which serves as the foundation for our novel *differential diffusion theory*. Notations used in this paper are summarized in Table 1. The schematic diagram in Figure 2 illustrates the process of radiance transport in a participating medium, where light travels along direction  $\omega$  and passes through the volume of the medium. The radiance  $L_{out}(x, \omega)$  represents the attenuated radiance of  $L_{in}(\omega)$  (the ambient light radiance), reaching the viewpoint  $x$  in the direction  $\omega$ . As light propagates through the par-



**Figure 2:** Schematic diagram illustrating the transport of radiance in a participating medium volume. The ambient light radiation  $L_{in}(\omega)$  propagates along a unit directional vector  $\omega$  (indicated by the yellow arrow), passing through point  $x_u$  inside the volume and intersecting the volume boundary at points  $x_a$  and  $x_b$ .  $L_{out}(x, \omega)$  represents the radiance of  $L_{in}(\omega)$ , attenuated by the medium, reaching the view point  $x$  along direction  $\omega$ . The distances between view-point  $x$  and the farthest intersection  $x_a$ , nearest intersection  $x_b$ , and position inside the volume  $x_u$  are denoted as  $D_a$ ,  $D_b$ , and  $D_u$  respectively. The relationships between  $x$  and  $x_a$ ,  $x_b$ ,  $x_u$  can be expressed as  $x = x_a + \omega \cdot D_a = x_b + \omega \cdot D_b = x_u + \omega \cdot D_u$ .

participating medium, it can undergo scattering or absorption, altering its path and reducing its contribution in the original direction. This phenomenon is described by the *Radiative Transport Equation (RTE)* [Cha60]. The integral form of RTE is expressed as follows:

$$L_{out}(x, \omega) = \underbrace{L_{ms}(x, \omega)}_{\text{Section 3}} + \underbrace{L_{ri}(x, \omega) + L_{ss}(x, \omega)}_{\text{Appendix A}}. \quad (1)$$

Here,  $L_{out}$  consists of the *multiple scattering radiance*  $L_{ms}$ , *reduced incident radiance*  $L_{ri}$ , and the *single scattering radiance*  $L_{ss}$ . To improve the efficiency of forward rendering, we adopt the *diffusion approximation* [Sta95] to estimate the multiple scattering radiance  $L_{ms}$ . In implementation, for efficient computation of the final rendering radiance  $L_{out}$ , we employ a precomputation strategy and a ray marching algorithm. Further details of the forward rendering algorithms are presented in Section 5. Before introducing our proposed method, we explicitly state the assumptions of our approach:

- For the sake of simplicity, in this work, our method is limited to volume scenes without surface models.
- To ensure high performance, the data types of participating media are restricted to regular volume data, which typically have a cube-like boundary.

Firstly, let's define the following notations. We denote the volume density at position  $x$  as  $\rho(x)$  and the optical coefficient as  $\kappa_i(x)$ , where  $\kappa_i \in \{\kappa_a, \kappa_s, \kappa_t\}$ . Here,  $\kappa_a$ ,  $\kappa_s$ , and  $\kappa_t$  indicate the absorption, scattering, and extinction coefficients, respectively. In this section, our main aim is to present a method for calculating radiance that arises from multiple scattering, using diffusion approximation. This method forms the basis for introducing our new theory called differential diffusion. For ease of understanding, we have moved

the detailed explanations about reduced incident radiance  $L_{ri}$ , single scattering radiance  $L_{ss}$ , and some of the fundamental formulas from the Radiative Transfer Equation (RTE) to Appendix A. The formula for calculating radiance due to multiple scattering,  $L_{ms}$ , is:

$$L_{ms}(x, \omega) = \int_{D_b}^{D_a} \tau(x_u, x_b) \kappa_t(x_u) J_{ms}(x_u, \omega) du. \quad (2)$$

This term represents the radiance of light that has undergone multiple scattering interactions before reaching the viewpoint  $x$ . The term  $J_{ms}$  corresponds to the multiple scattering term, which is computationally expensive to compute in the presence of dense participating media due to the proliferation of scattering paths. In our differentiable rendering framework, we utilize *diffusion theory* [Sta95] to approximate  $J_{ms}$  for forward rendering. Diffusion theory provides a good approximation for multiply-scattered light by modeling it as a simple partial differential equation, which can be efficiently solved using various analytical and numerical methods. The formulas and derivation of the diffusion approximation can be found in Stam's paper [Sta95], which we include here for completeness. The expression for  $J_{ms}$  is given as:

$$J_{ms}(x_u, \omega) = \Omega L_d^0(x_u) + \frac{\Omega \bar{\mu}}{3} \cdot \bar{L}_d^1(x_u) \cdot \omega, \quad (3)$$

where  $L_d^0$  and  $\bar{L}_d^1$  represent the first two Taylor expansions of the media radiance  $L_d$  in the directional component. The derivation of  $J_{ms}$  and  $\bar{\mu}$  (the first moment of the phase function) is presented in Appendix A. Specifically,  $L_d^0(x_u)$  denotes the average radiance over all angles and is determined by a diffusion equation:

$$\nabla \cdot (\kappa(x_u) \nabla L_d^0(x_u)) - \kappa_a(x_u) L_d^0(x_u) + S(x_u) = 0. \quad (4)$$

To enhance readability, the expressions for  $\kappa$  and  $S$  are also moved to Appendix A. The diffusion equation is a partial differential equation that can be efficiently solved using Jacobi or Gauss-Seidel iteration.  $\bar{L}_d^1$  can be obtained by substituting  $L_d^0$  into the following formula:

$$\bar{L}_d^1(x_u) = \kappa(x_u) (-\nabla L_d^0(x_u) + \sigma_s(x_u) \bar{Q}_{ri}^1(x_u)), \quad (5)$$

where  $\bar{Q}_{ri}^1$  is defined in Appendix A. Finally, substituting  $L_d^0$  and  $\bar{L}_d^1$  into Equation 3 yields  $J_{ms}$ .

All of the above describes the complete process of light transport in participating media. The most significant difference of our method compared to previous works is the introduction of diffusion theory into the differentiable rendering framework for forward rendering. In the next section, we will present the derivation of the differential form of RTE based on the diffusion approximation.

#### 4. Differentiable Diffusion Approximation of RTE

In section 3, we discussed the integral form of RTE and the use of the diffusion equation to approximate multiply-scattered radiance. While diffusion theory provides an effective approximation for multiply-scattered light, there is limited work on extending this theory to compute derivatives of multiply-scattered radiance.

In this section, we present a differential theory based on diffusion approximation to compute the derivatives of scene parameters  $\pi$ , which include camera pose, volume density, optical coefficients, and lighting parameters.



The differentiation of Equation 1 can be expanded into three terms: the differentiation of multiple scattering radiance  $\partial_\pi L_{ms}$ , the differentiation of reduced incident radiance  $\partial_\pi L_{ri}$ , and the differentiation of single scattering radiance  $\partial_\pi L_{ss}$ . An overview of our derivations is as follows:

$$\partial_\pi L_{out}(x, \omega) = \underbrace{\partial_\pi L_{ms}(x, \omega)}_{\text{Section 4.1}} + \underbrace{\partial_\pi L_{ri}(x, \omega) + \partial_\pi L_{ss}(x, \omega)}_{\text{Appendix B}}. \quad (6)$$

In Section 4.1, our primary aim is to introduce the derivatives of the multiple scattering radiance  $\partial_\pi L_{ms}$ . To ensure the main text remains clear and focused, we have included the derivatives of the reduced incident radiance  $\partial_\pi L_{ri}$  and the derivative of the single scattering radiance  $\partial_\pi L_{ss}$  in Appendix B.

#### 4.1. Differentiation of Diffuse Radiance

In this section, we focus on introducing the derivative of multiple scattering radiance based on our proposed differential diffusion equation. We will focus on the multiple scattering radiance  $L_{ms}$ , for which the derivative can be formulated as follows:

$$\begin{aligned} \partial_\pi L_{ms}(x, \omega) &= \int_{D_b}^{D_a} \tau(x_u, x_b) \kappa_t(x_u) \partial_\pi J_{ms}(x_u, \omega) du \\ &+ \int_{D_b}^{D_a} (\partial_\pi \tau(x_u, x_b)) \kappa_t(x_u) \\ &+ \tau(x_u, x_b) \partial_\pi \kappa_t(x_u) J_{ms}(x_u, \omega) du. \end{aligned} \quad (7)$$

This equation expresses a summation of two integrals involving the transmission  $\tau$ , extinction coefficient  $\kappa_t$ , multiple scattering radiance term  $J_{ms}$ , and their corresponding derivatives. The term  $\partial_\pi L_{ms}$  represents the derivative of the multiple scattering radiance, providing insights into the sensitivity of an arbitrary scene parameter to changes for differential rendering.

The *Reynolds transport theorem* is often used for computing derivatives of hydrodynamic integral equations. However, in our case, we are working with regular volume data that has a fixed boundary and constant normals. Therefore, we do not require the use of the Reynolds transport theorem to differentiate  $D_a$  and  $D_b$  in Equation 7. While Zhang et al. [ZWZ\*19] applied the Reynolds transport theorem to differentiate with respect to the shape of the boundary, our method is specifically developed for real-time differentiable volume rendering, and our goals and constraints are different from Zhang's general-purpose framework.

To evaluate the right-hand side (RHS) of Equation 7, we need to compute the derivatives of the transmittance  $\partial_\pi \tau(x_u, x_b)$ , the derivatives of the optical coefficient  $\partial_\pi \kappa_t(x_u)$ , and the derivatives of the multiple scattering term  $\partial_\pi J_{ms}(x_u, \omega)$ . The expressions for  $\partial_\pi \tau(x_u, x_b)$  and  $\partial_\pi \kappa_t(x_u)$  are provided in Appendix B.

Since differentiating the multiple scattering radiance with diffusion approximation is the primary contribution of our work, we present the derivation of these derivatives as follows.

*Derivation of  $\partial_\pi J_{ms}$ .* Differentiating Equation 3 yields:

$$\begin{aligned} \partial_\pi J_{ms}(x_u, \omega) &= \partial_\pi \Omega(L_d^0(x_u) + \frac{\bar{\mu}}{3} \bar{L}_d^1(x_u) \cdot \omega) \\ &+ \Omega(\partial_\pi L_d^0(x_u) + \frac{\bar{\mu}}{3} \partial_\pi \bar{L}_d^1(x_u) \cdot \omega + \frac{\partial_\pi \bar{\mu}}{3} \bar{L}_d^1(x_u) \cdot \omega), \end{aligned} \quad (8)$$

where the derivative of  $\Omega$  is expressed as:

$$\partial_\pi \Omega = \frac{\sigma_{tr} \partial_\pi \sigma_s - \sigma_s \partial_\pi \sigma_{tr}}{(\sigma_t)^2}. \quad (9)$$

Equation 8 represents the derivative of  $J_{ms}$  with respect to an arbitrary scene parameter  $\pi$ , where it is expressed as a combination of the derivative of the albedo term  $\Omega$  and the derivative of the first two Taylor expansions of the media radiance  $L_d$  with respect to  $\pi$ , accounting for the first moment of the phase function  $\bar{\mu}$  (please refer to Equation 34 of Appendix A), while Equation 9 defines the derivative of  $\Omega$ , involving the derivatives of the scattering and extinction coefficients.

Note that one can choose an appropriate phase function and substitute into Equation 34 for rendering. Differentiating Equation 34 yields:

$$\partial_\pi \bar{\mu} = \frac{3}{2} \int_{-1}^1 \partial_\pi \mu p(u) + \mu \partial_\pi p(u) du. \quad (10)$$

In practical scenarios, a phase function  $p(u)$  often has analytical expressions, making it possible to obtain  $\partial_\pi p(u)$  using symbolic differentiation. To evaluate the RHS of Equation 8, we also need to compute  $\partial_\pi L_d^0(x_u)$  and  $\partial_\pi \bar{L}_d^1(x_u)$ .  $L_d^0(x_u)$  represents the average radiance over all angles and is determined by the differential Equation 4. Differentiating Equation 4 yields:

$$\begin{aligned} \nabla \partial_\pi \kappa(x_u) \cdot \nabla L_d^0(x_u) + \nabla \kappa(x_u) \cdot \nabla \partial_\pi L_d^0(x_u) \\ + \partial_\pi \kappa(x_u) \nabla^2 L_d^0(x_u) + \kappa(x_u) \nabla^2 \partial_\pi L_d^0(x_u) \\ - \partial_\pi \kappa_a(x_u) L_d^0(x_u) - \kappa_a(x_u) \partial_\pi L_d^0(x_u) + \partial_\pi S(x_u) = 0. \end{aligned} \quad (11)$$

The above partial differential equation is referred to as the *differential diffusion equation*. The derivation of this equation and providing a solution for its solving (refer to Section 5.3) are the main contributions of this paper. These advancements form the core of our research and contribute to the field of differentiable rendering for participating media. Fortunately, the differential diffusion equation can be efficiently solved using numerical methods. Solving  $\partial_\pi L_d^0$  is similar to solving the radiance  $L_d^0$  in the diffusion equation. To solve Equation 11, we need the expressions for  $\partial_\pi \kappa(x_u)$  and  $\partial_\pi S(x_u)$ , which are given by:

$$\partial_\pi \kappa(x_u) = -\frac{\partial_\pi \rho(x_u)}{\sigma_{tr} \rho^2(x_u)} - \frac{\partial_\pi \sigma_{tr}}{\sigma_{tr}^2 \rho(x_u)}, \quad (12)$$

where the derivative of the transport cross section is:

$$\partial_\pi \sigma_{tr} = \partial_\pi \sigma_s \left(1 - \frac{\bar{\mu}}{3}\right) - \frac{\partial_\pi \bar{\mu} \sigma_s}{3} + \partial_\pi \sigma_a. \quad (13)$$

Additionally, the expression for  $\partial_\pi S(x_u)$  is given by:

$$\begin{aligned} \partial_\pi S(x_u) &= \partial_\pi \kappa_s(x_u) Q_{ri}^0(x_u) + \kappa_s(x_u) \partial_\pi Q_{ri}^0(x_u) \\ &- \frac{\sigma_s}{\sigma_{tr}} \nabla \cdot \partial_\pi \bar{Q}_{ri}^1(x_u) - \frac{\partial_\pi \sigma_s \sigma_{tr} - \partial_\pi \sigma_{tr} \sigma_s}{\sigma_{tr}^2} \nabla \cdot \bar{Q}_{ri}^1(x_u). \end{aligned} \quad (14)$$

Equation 12 and Equation 13 provide the expressions for  $\partial_\pi \kappa(x_u)$  and  $\partial_\pi \sigma_{tr}$ , respectively, which are needed to solve the differential Equation 11 for  $\partial_\pi L_d^0$ . Also, Equation 14 gives the expression for  $\partial_\pi S(x_u)$ , which is required to evaluate the source term in

the differential diffusion equation. These equations enable the efficient numerical solution of differential multiple scattering radiance.

To compute the derivative of  $S(x_u)$ , we need  $\partial_\pi Q_{ri}^0$  and  $\partial_\pi \bar{Q}_{ri}^1$ . Since our method is limited to volume scenes without surface models, the complexity of derivative computation caused by visibility is reduced. Therefore, we assume that  $L_{ri}$  is continuous on  $S^2$ . By differentiating  $Q_{ri}^0(x_u)$  and  $\bar{Q}_{ri}^1(x_u)$ , we obtain:

$$\partial_\pi Q_{ri}^0(x_u) = \frac{1}{4\pi} \int_{S^2} \partial_\pi L_{ri}(x_u, \omega') d\omega', \quad (15)$$

$$\begin{aligned} \partial_\pi \bar{Q}_{ri}^1(x_u) &= \frac{\partial_\pi \bar{\mu}}{4\pi} \int_{S^2} L_{ri}(x_u, \omega') \cdot \omega' d\omega' \\ &+ \frac{\bar{\mu}}{4\pi} \int_{S^2} \partial_\pi L_{ri}(x_u, \omega') \cdot \omega' d\omega', \end{aligned} \quad (16)$$

where the derivative of the interface integral can be neglected. To compute  $\partial_\pi Q_{ri}^0$  and  $\partial_\pi \bar{Q}_{ri}^1$ , we require  $\partial_\pi L_{ri}$ . The derivative of  $L_{ri}$  can be found in Equation 41 of Appendix B.

Equation 15 and Equation 16 provide the expressions for  $\partial_\pi Q_{ri}^0$  and  $\partial_\pi \bar{Q}_{ri}^1$ , respectively, which are computed based on the derivative of  $L_{ri}$  given in Equation 41. These equations enable the computation of the derivatives of the radiance transfer integrals necessary for the differential rendering process, considering the assumption of continuity and neglecting the interface integral derivative.

Computing equations like Equation 15 and Equation 16 is costly, particularly with panoramic maps. We address this with a pre-computation strategy that stores intermediate variables for quick access during differential rendering. This optimizes efficiency and eliminates redundancy. Further details are in Section 5, which delves into the computational aspects and their implications for the differential diffusion approximation.

To compute the terms  $\partial_\pi \kappa$ ,  $\partial_\pi \kappa_a$ , and  $\partial_\pi S$ , direct calculations can be performed using Equations (12), (13), and (14). By substituting Equations (12), (42), and (14) into Equation (11),  $\partial_\pi L_d^0(x_u)$  can be solved using numerical methods such as Jacobi or Gauss-Seidel iterations [IK12].

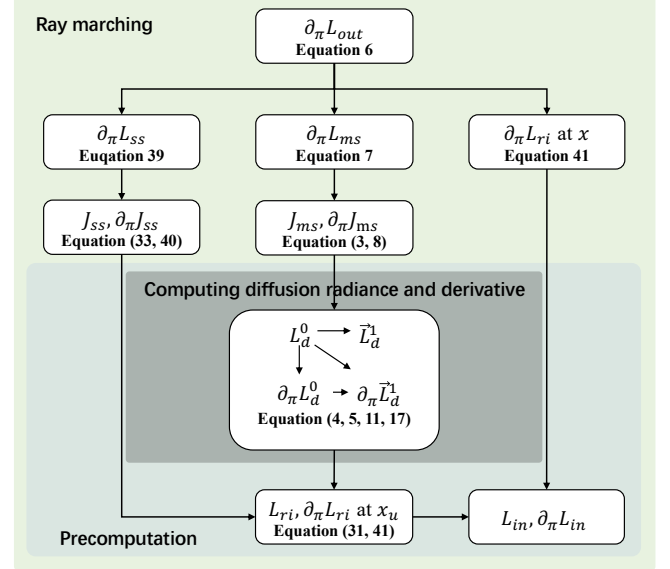
Once  $\partial_\pi L_d^0(x_u)$  has been calculated,  $\partial_\pi \bar{L}_d^1(x_u)$  can be obtained by substituting  $\partial_\pi L_d^0(x_u)$  into the following formula:

$$\begin{aligned} \partial_\pi \bar{L}_d^1(x_u) &= \partial_\pi \kappa(x_u) (-\nabla L_d^0(x_u) + \kappa_s(x_u) \bar{Q}_{ri}^1(x_u)) \\ &+ \kappa(x_u) (-\nabla \partial_\pi L_d^0(x_u) + \kappa_s(x_u) \partial_\pi \bar{Q}_{ri}^1(x_u)) \\ &+ \partial_\pi \kappa_s(x_u) \bar{Q}_{ri}^1(x_u). \end{aligned} \quad (17)$$

Hence, it computes  $\partial_\pi \bar{L}_d^1(x_u)$ , which is the derivative of the diffuse radiance vector at point  $x_u$  with respect to the change in the scene parameters  $\pi$ , based on  $\partial_\pi L_d^0(x_u)$ , using the given expressions for derivatives of  $\kappa$ ,  $\kappa_s$ , and  $\bar{Q}_{ri}^1$ .

Previous works often rely on Monte Carlo path tracing to estimate the derivative of multiple scattering radiance, which can be computationally demanding in the presence of dense participating media due to the proliferation of scattering paths. In this subsection, we derive the differential form of the diffusion equation (Equation 11), known as the *differential diffusion equation*. The derivative of  $L_d^0$  in the differential diffusion equation can be efficiently

solved using numerical methods. By substituting  $\partial_\pi L_d^0$  into Equation 17, we obtain  $\partial_\pi \bar{L}_d^1$ . Therefore, we provide a new approach to efficiently compute the derivative of multiple scattering radiance by solving the differential diffusion equation instead of relying on Monte Carlo methods.



**Figure 3:** Relations of algorithm components and the dependencies between radiance-related quantities. An arrow from A to B indicates that A depends on B.

## 4.2. Completing $\partial_\pi L_{out}$

Our objective is to compute the derivative of radiance  $L_{out}$  with respect to the scene parameters  $\pi$ . We decompose  $\partial_\pi L_{out}$  into three terms:  $\partial_\pi L_{ms}$ ,  $\partial_\pi L_{ri}$ , and  $\partial_\pi L_{ss}$ . Among these terms, the computation of  $\partial_\pi L_{ms}$  is the most computationally expensive. To address this, we introduced the differential diffusion equation in Section 4.1, which can be efficiently solved using numerical methods.

The derivatives of  $L_{ri}$  and  $L_{ss}$  are given by Equations (41) and (39), respectively. These derivatives can be computed through the ray marching process with relatively low computational cost. By combining Equations (39), (41), and (7), we obtain the expression for  $\partial_\pi L_{out}$  as shown in Equation 6. The dependencies between the various quantities involved in  $\partial_\pi L_{out}$  are illustrated in Figure 3.

Our main contribution lies in providing a novel and highly efficient method for computing the derivatives of scene parameters in the context of participating media. Since this paper primarily focuses on the differential rendering of volume models rather than surface models, our method is specifically designed for scenes that involve volumes without surface models. We solely consider the Radiative Transfer Equation (RTE) and omit the Rendering Equation (RE) in our approach.

## 5. Discrete Calculation

Our differential diffusion theory, as discussed above, is based on continuous forms of equations. However, for efficient computation,

it needs to derive discrete forms of these equations that can be effectively implemented. Now, we introduce the algorithm components of our method, which consist of three main components: 1) *Ray marching*, 2) *Pre-computation*, and 3) *Computing diffusion radiance and derivative*. The relationships and dependencies between these components and the radiance-related quantities are illustrated in Figure 3.

The *Ray marching* component (Section 5.1) serves as the entry point of our method. It is responsible for computing the final radiance  $L_{out}$  and its corresponding derivative  $\partial_{\pi}L_{out}$ . By utilizing the ray marching technique, this component traces the paths of light rays through the participating medium, considering scattering and absorption effects.

To initiate the ray marching process, the *Pre-computation* component (Section 5.2) is required. It performs pre-computation tasks, which involve the calculation of intermediate variables necessary for accelerating the overall computation. These pre-computed variables are then provided to the ray marching component.

Within the *Pre-computation* component, we include the *Computing diffusion radiance and derivative* component (Section 5.3). This subcomponent specifically focuses on solving the radiance and derivative of the diffusion terms. By employing suitable algorithms, it efficiently computes the diffusion-related radiance and its derivative, enhancing the accuracy of our method.

Once all the pre-computed intermediate variables are prepared by the *Pre-computation* component, the ray marching process is performed to obtain the final radiance  $L_{out}$  and its derivative  $\partial_{\pi}L_{out}$ . Before delving into the details of the algorithm components, let's establish the following assumption.

*Assumption:* We assume a uniform 3D grid comprised of voxels, where each voxel is characterized by a voxel position  $v_p$  (with the subscript of  $v$  indicating a point  $p$  inside the voxel) and a voxel edge length  $h$ . The voxel position  $v_p$  can be indexed using integer coordinates  $(i, j, k)$ . Additionally, we have a discrete density field  $\rho$  representing the volume of the participating medium, along with its corresponding optical coefficient field  $\kappa_i$ . These assumptions form the basis for our subsequent algorithm components.

## 5.1. Ray Marching

Ray marching (Algorithm 1) is utilized to compute the final radiance  $L_{out}$  and its derivative  $\partial_{\pi}L_{out}$  at the view point  $x$ . To improve efficiency, we employ pre-computation of necessary intermediate variables, which are stored as voxels (Line 2 of Algorithm 1). These pre-computed variables can then be directly utilized within the ray marching component (Lines 6-21 of Algorithm 1). Finally, the final radiance  $L_{out}$  and its derivative  $\partial_{\pi}L_{out}$  are obtained by summing up all the radiance and derivative terms separately, based on Equations (1, 6) (Lines 25-28 of Algorithm 1). In this subsection, we provide the discrete forms of  $L_{ss}$ ,  $\partial_{\pi}L_{ss}$ ,  $L_{ms}$ , and  $\partial_{\pi}L_{ms}$ .

$L_i$  and  $\partial_{\pi}L_i$  represent the generic discrete forms of scattering radiance and its corresponding derivative, respectively, which are:

$$L_i(x, \omega) = \sum_{p=x_a}^{x_b} \tau(v_p, v_{x_b}) \kappa_i(v_p) J_i(v_p, \omega) \Delta d, \quad (18)$$

### ALGORITHM 1: Ray Marching

```

1 Function RayMarching( $L_d^0, \bar{L}_d^1, L_{in}, S, \kappa, \kappa_a, \kappa_t$ ):
   /* Pre-computation function is shown in Alg. 2 */
2  $L_{ri}, L_d^0, \bar{L}_d^1, \partial_{\pi}L_{ri}, \partial_{\pi}L_d^0, \partial_{\pi}\bar{L}_d^1 \leftarrow$  Precomputation( $L_d^0, \bar{L}_d^1, S, \kappa, \kappa_a, \kappa_t$ )
3  $x_u \leftarrow x_a$ 
4  $D_b = \|x_b - x\|$  // Distance from  $x_b$  to the view point  $x$ 
5 repeat
6    $J_{ms}(v_{x_u}, \omega) \leftarrow$  Equation 3
7    $\partial_{\pi}J_{ms}(v_{x_u}, \omega) \leftarrow$  Equation 8
8    $J_{ss}(v_{x_u}, \omega) \leftarrow$  Equation 33
9    $\partial_{\pi}J_{ss}(v_{x_u}, \omega) \leftarrow$  Equation 40
10   $L_{ms}(x, \omega) \leftarrow L_{ms}(x, \omega) + \tau(v_{x_u}, v_{x_b}) \kappa_t(v_{x_u}) J_{ms}(v_{x_u}, \omega) \Delta d$ 
11   $\partial_{\pi}L_{ms}(x, \omega) \leftarrow \partial_{\pi}L_{ms}(x, \omega) + \tau(v_{x_u}, v_{x_b}) \kappa_t(v_{x_u}) \partial_{\pi}J_{ms}(v_{x_u}, \omega) \Delta d +$ 
    $(\partial_{\pi}\tau(v_{x_u}, v_{x_b}) \kappa_t(v_{x_u}) + \tau(v_{x_u}, v_{x_b}) \partial_{\pi}\kappa_t(v_{x_u})) J_{ms}(v_{x_u}, \omega) \Delta d$ 
12
13   $L_{ss}(x, \omega) \leftarrow L_{ss}(x, \omega) + \tau(v_{x_u}, v_{x_b}) \kappa_t(v_{x_u}) J_{ss}(v_{x_u}, \omega) \Delta d$ 
14   $\partial_{\pi}L_{ss}(x, \omega) \leftarrow \partial_{\pi}L_{ss}(x, \omega) + \tau(v_{x_u}, v_{x_b}) \kappa_t(v_{x_u}) \partial_{\pi}J_{ss}(v_{x_u}, \omega) \Delta d +$ 
    $(\partial_{\pi}\tau(v_{x_u}, v_{x_b}) \kappa_t(v_{x_u}) + \tau(v_{x_u}, v_{x_b}) \partial_{\pi}\kappa_t(v_{x_u})) J_{ss}(v_{x_u}, \omega) \Delta d$ 
15
16   $x_u \leftarrow x_u + \omega \cdot \Delta d$ 
17   $D_u = \|x_u - x\|$  // Distance from  $x_u$  to the view point  $x$ 
18 until  $D_u < D_b$ 
19  $L_{ri}(x, \omega) \leftarrow$  Equation 31
20  $\partial_{\pi}L_{ri}(x, \omega) \leftarrow$  Equation 41
21  $L_{out}(x, \omega) \leftarrow L_{ms}(x, \omega) + L_{ri}(x, \omega) + L_{ss}(x, \omega)$ 
22  $\partial_{\pi}L_{out}(x, \omega) \leftarrow \partial_{\pi}L_{ms}(x, \omega) + \partial_{\pi}L_{ri}(x, \omega) + \partial_{\pi}L_{ss}(x, \omega)$ 
return  $L_{out}, \partial_{\pi}L_{out}$ 

```

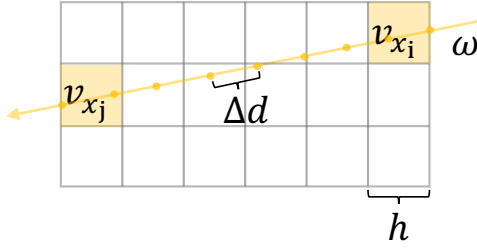
$$\begin{aligned} \partial_{\pi}L_i(x, \omega) = & \sum_{p=x_a}^{x_b} \tau(v_p, v_{x_b}) \kappa_t(v_p) \partial_{\pi}J_i(v_p, \omega) \Delta d + \\ & \sum_{p=x_a}^{x_b} (\partial_{\pi}\tau(v_p, v_{x_b}) \kappa_t(v_p) + \\ & \tau(v_p, v_{x_b}) \partial_{\pi}\kappa_t(v_p)) J_i(v_p, \omega) \Delta d, \end{aligned} \quad (19)$$

where  $i \in \{ms, ss\}$  with  $ms$  and  $ss$  indicating multiple scattering and single scattering, respectively.  $\Delta d$  is the sampling step length. In Equations (18, 19), the discrete form of transmittance  $\tau$  and its derivative  $\partial_{\pi}\tau$  are needed, which are expressed as:

$$\tau(v_{x_i}, v_{x_j}) = e^{-\sum_{p=x_i}^{x_j} \kappa_i(v_p) \Delta d}, \quad (20)$$

$$\partial_{\pi}\tau(v_{x_i}, v_{x_j}) = -\tau(v_{x_i}, v_{x_j}) \sum_{p=x_i}^{x_j} \partial_{\pi}\kappa_i(v_{x_u}) \Delta d. \quad (21)$$

To sum up, the equations provide the discrete forms of scattering radiance and its derivative for single scattering ( $L_{ss}$ ) and multiple scattering ( $L_{ms}$ ). The radiance values are obtained by summing up the contributions of transmittance, extinction coefficient, and scattering phase function along the ray path, while the derivatives incorporate additional terms involving the derivatives of transmittance and extinction coefficient. Furthermore, a pre-computation strategy is employed, which enables efficient computation of the final ra-



**Figure 4:** Illustration of ray marching in the volume of a participating medium. It demonstrates the sampling along the ray path with direction  $\omega$ , starting from voxel position  $v_{x_i}$  and advancing towards  $v_{x_j}$ . The sampling step length is  $\Delta d$ , and the voxel edge length is  $h$ .

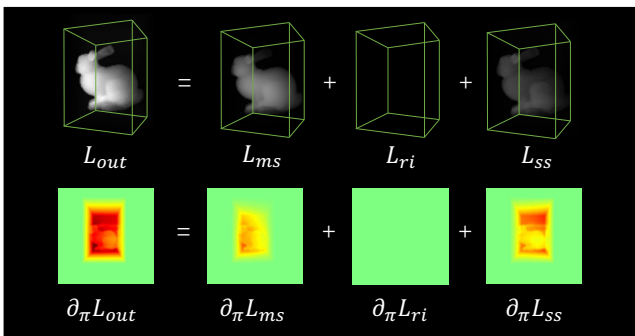
diance and its derivative in participating media by directly reading the intermediate variables.

The process of transmittance calculation is illustrated in Figure 4. A ray traverses the volume of a participating medium along direction  $\omega$ , intersecting the volume from voxel position  $v_{x_i}$  to  $v_{x_j}$ . This ray marching process involves sampling the ray path with a step size of  $\Delta d$ . The sampled values are accumulated to obtain the optical depth.

For the computation of single scattering terms, namely  $J_{ss}$  and  $\partial_{\pi}J_{ss}$ , we rely on  $L_{ri}$  and  $\partial_{\pi}L_{ri}$ . To avoid redundant calculations of  $L_{ri}$  and  $\partial_{\pi}L_{ri}$ , we propose a pre-computation component (discussed in Section 5.2) to compute these intermediate variables and store them as voxels. This allows for efficient reuse of the pre-computed values during the ray marching process.

Regarding the multiple scattering terms, the diffusion radiance  $L_d^0, \bar{L}_d^1$ , and their derivatives  $\partial_{\pi}L_d^0, \partial_{\pi}\bar{L}_d^1$  are obtained by solving the diffusion equation and the differential diffusion equation, which will be introduced in Section 5.3. Subsequently,  $J_{ms}$  and  $\partial_{\pi}J_{ms}$  can be computed using Equations (3) and (8). The ray marching technique is then employed to compute the scattering radiance and their derivatives (Lines 10-21 of Algorithm 1).

Finally, all of the radiance and derivative terms are accumulated separately to obtain the final radiance  $L_{out}$  and its derivative  $\partial_{\pi}L_{out}$ . An example of the computed results for  $L_{out}$  and  $\partial_{\pi}L_{out}$  is shown in Figure 5.



**Figure 5:** Visualization of the individual radiance components in the computation result of  $L_{out}$  and  $\partial_{\pi}L_{out}$  (with respect to density).

## 5.2. Pre-computation

In order to address the challenges involved in rendering a participating medium volume and computing radiance derivatives, our method incorporates a pre-computation step where certain intermediate variables are calculated and stored as voxels. These pre-computed values can then be efficiently reused during the ray marching process, eliminating the need for redundant calculations.

At the beginning of Algorithm 1, it is necessary to pre-compute the radiance fields  $L_{ri}, L_d^0, \bar{L}_d^1$ , and their corresponding derivative fields (Algorithm 2), which are then stored as voxels. The computation of  $L_{ri}$  and  $\partial_{\pi}L_{ri}$  involves estimating the integral of solid angle over the unit sphere  $S^2$  using the Monte Carlo Method (Line 3 of Algorithm 2). The intermediate variables  $Q_{ri}^0, \bar{Q}_{ri}^1, \kappa, S$ , and their derivatives are pre-computed as well (Lines 4-7 of Algorithm 2).

By obtaining these pre-computed intermediate variables, we are able to solve for the diffusion radiance and its derivatives (Line 9 of Algorithm 2). The details of the computation of diffusion radiance and its derivatives can be found in Section 5.3 (Algorithm 3).

---

### ALGORITHM 2: Precomputation

---

```

1 Function Precomputation ( $L_d^0, \bar{L}_d^1, S, \kappa, \kappa_d, \kappa_t$ ):
2   parfor each position  $v_{xu} \in \text{voxels}$  do
3      $L_{ri}(v_{xu}, \omega'), \partial_{\pi}L_{ri}(v_{xu}, \omega') \leftarrow$  Equation 31, 41
4      $Q_{ri}^0(v_{xu}), \partial_{\pi}Q_{ri}^0(v_{xu}) \leftarrow$  Equation 37, 15
5      $\bar{Q}_{ri}^1(v_{xu}), \partial_{\pi}\bar{Q}_{ri}^1(v_{xu}) \leftarrow$  Equation 38, 16
6      $\kappa(v_{xu}), \partial_{\pi}\kappa(v_{xu}) \leftarrow$  Equation 35, 12
7      $S(v_{xu}), \partial_{\pi}S(v_{xu}) \leftarrow$  Equation 36, 14
8   end
9   /* DiffusionSolver function is shown in Alg. 3 */
10   $L_d^0, \bar{L}_d^1, \partial_{\pi}L_d^0, \partial_{\pi}\bar{L}_d^1 \leftarrow$  DiffusionSolver( $L_d^0, \partial_{\pi}L_d^0, \bar{L}_d^1, \partial_{\pi}\bar{L}_d^1, S, \kappa, \kappa_d$ )
11  return  $L_{ri}, L_d^0, \bar{L}_d^1, \partial_{\pi}L_{ri}, \partial_{\pi}L_d^0, \partial_{\pi}\bar{L}_d^1$ 

```

---

## 5.3. Computing Diffusion Radiance and Derivative

In this section, we describe the computation of diffusion radiance and their corresponding derivatives. We first present the discrete forms of diffusion radiance, followed by an introduction to the solving process of the differential diffusion equation using Jacobi iterative method.

*Computing diffusion radiance.* To enhance the efficiency of rendering, we employ the diffusion theory to compute the multiply-scattering radiance. The diffusion equation (Equation 4) can be further expressed as:

$$\nabla \kappa \cdot \nabla L_d^0 + \kappa \cdot \nabla^2 L_d^0 - \kappa_d L_d^0 + S = 0. \quad (22)$$

In our implementation, we approximate the first and second order derivatives using the first-order central difference and the second-order central difference, respectively. The discrete approximations are given by:

$$\nabla F \approx \frac{F_{i+1,j,k} - F_{i-1,j,k}}{2h}$$



and

$$\nabla^2 F \approx \frac{F_{i+1,j,k} + F_{i-1,j,k} - 2F_{i,j,k}}{h^2}.$$

Using the discrete approximations, we express the discrete forms of  $\nabla \kappa \cdot \nabla L_d^0$  and  $\nabla^2 L_d^0$  as:

$$\nabla \kappa \cdot \nabla L_d^0 = \frac{1}{4h^2} \Sigma_{ijk}^4, \quad (23)$$

$$\nabla^2 L_d^0 = \frac{1}{h^2} \Sigma_{ijk}^3, \quad (24)$$

where the expressions of  $\Sigma_{ijk}^3$ ,  $\Sigma_{ijk}^4$ , and  $\Sigma_{ijk}^5$  (used in Equation 25) are provided in Appendix B. With these discrete expressions, we can derive the discrete form of  $L_d^0$  as follows:

$$L_d^0 = \frac{1}{h^2(\kappa_a)_{i,j,k} + 6\kappa_{i,j,k}} \cdot (h^2 S_{i,j,k} + \frac{1}{4} \Sigma_{ijk}^4 + \kappa_{i,j,k} \Sigma_{ijk}^5). \quad (25)$$

In our implementation, we utilize the Jacobi iterative method to solve for  $L_d^0$ . Once  $L_d^0$  is obtained, it can be easily substituted into Equation 5 for computing  $\bar{L}_d^1$ .

The above equations describe the computation of diffusion radiance using the diffusion theory. The diffusion equation (Equation 22) represents the balance between the gradients and Laplacian of the radiance, absorption, and a source term. By discretizing the derivatives using central difference approximations, the expressions for the discrete forms of the gradients and Laplacian (Equation 23, Equation 24) are derived. The discrete form of  $L_d^0$  (Equation 25) is then obtained using these expressions, and it can be solved iteratively using the Jacobi method to compute the diffusion radiance.

Algorithm 3 iteratively computes the diffusion radiance  $L_d^0$  and its derivative  $\partial_\pi L_d^0$ , as well as the diffusion radiance derivative  $\bar{L}_d^1$  and its derivative  $\partial_\pi \bar{L}_d^1$ , by solving the diffusion equation. It initializes the radiance values and then iteratively updates them until the error between the updated values and the previous values falls below a specified threshold. The final computed radiance and derivative values are returned as the output.

---

#### ALGORITHM 3: Computing Diffusion Radiance and Derivative

---

```

1 Function DiffusionSolver ( $L_d^0, \partial_\pi L_d^0, \bar{L}_d^1, \partial_\pi \bar{L}_d^1, S, \kappa, \kappa_a$ ):
2   if  $L_d^0, \bar{L}_d^1$  and  $\partial_\pi L_d^0, \partial_\pi \bar{L}_d^1$  are uninitialized then
3     random assignment of  $L_d^0, \bar{L}_d^1$  and  $\partial_\pi L_d^0, \partial_\pi \bar{L}_d^1$ 
4   end
5   repeat
6      $L_{new} \leftarrow$  Equation 25
7      $error \leftarrow MSE(L_{new}, L_d^0)$  /* mean-square error */
8      $L_d^0 \leftarrow L_{new}$ 
9   until  $error < error\_threshold$ 
10   $\bar{L}_d^1 \leftarrow$  via Equation 5
11  repeat
12     $\partial_\pi L_{new} \leftarrow$  Equation 29
13     $error \leftarrow MSE(\partial_\pi L_{new}, \partial_\pi L_d^0)$ 
14     $\partial_\pi L_d^0 \leftarrow \partial_\pi L_{new}$ 
15  until  $error < error\_threshold$ 
16   $\partial_\pi \bar{L}_d^1 \leftarrow$  via Equation 30
17  return  $L_d^0, \bar{L}_d^1, \partial_\pi L_d^0, \partial_\pi \bar{L}_d^1$ 

```

---

*Computing derivative of diffusion radiance.* The calculation of the derivative of diffusion radiance follows a similar process to solving the diffusion equation. The differential diffusion equation (Equation 11) needs to be solved using Jacobi iterative method. The discrete forms of the terms  $\nabla \partial_\pi \kappa \cdot \nabla L_d^0$ ,  $\nabla \kappa \cdot \nabla \partial_\pi L_d^0$ , and  $\partial_\pi S$  are presented as follows:

$$\nabla \partial_\pi \kappa \cdot \nabla L_d^0 = \frac{1}{4h^2} \Sigma_{ijk}^1, \quad (26)$$

$$\nabla \kappa \cdot \nabla \partial_\pi L_d^0 = \frac{1}{4h^2} \Sigma_{ijk}^2, \quad (27)$$

$$\partial_\pi S = \Sigma_{ijk}^6. \quad (28)$$

The expressions of  $\Sigma_{ijk}^1$ ,  $\Sigma_{ijk}^2$ ,  $\Sigma_{ijk}^6$ , and  $\Sigma_{ijk}^9$  (used in Equation 29 and Equation 30) are provided in Appendix B. Substituting these discrete forms into Equation 11, we obtain:

$$\partial_\pi L_d^0 = \frac{1}{h^2(\kappa_a)_{i,j,k} + 6\kappa_{i,j,k}} \left( \frac{1}{4} (\Sigma_{ijk}^1 + \Sigma_{ijk}^2) + \kappa_{ijk} \Sigma_{ijk}^3 - h^2 (\kappa_a)_{ijk} (L_d^0)_{ijk} + h^2 \Sigma_{ijk}^6 \right). \quad (29)$$

The above equations describe the computation of the derivative of media radiance's Tylor series expansion term of zero order. It is an important component of calculating the derivative of media radiance  $L_d$ . By discretizing the derivatives using central difference approximations, the expressions for the discrete forms of the gradients and the derivative of diffusion radiance (Equation 26, Equation 27, Equation 28) are derived. These expressions are then used to solve the differential diffusion equation (Equation 11) iteratively using the Jacobi method, resulting in the discrete form of the derivative of diffusion radiance (Equation 29).

Next, we compute  $\partial_\pi \bar{L}_d^1$  using the following discrete equation:

$$\partial_\pi \bar{L}_d^1 = \Sigma_{ijk}^9, \quad (30)$$

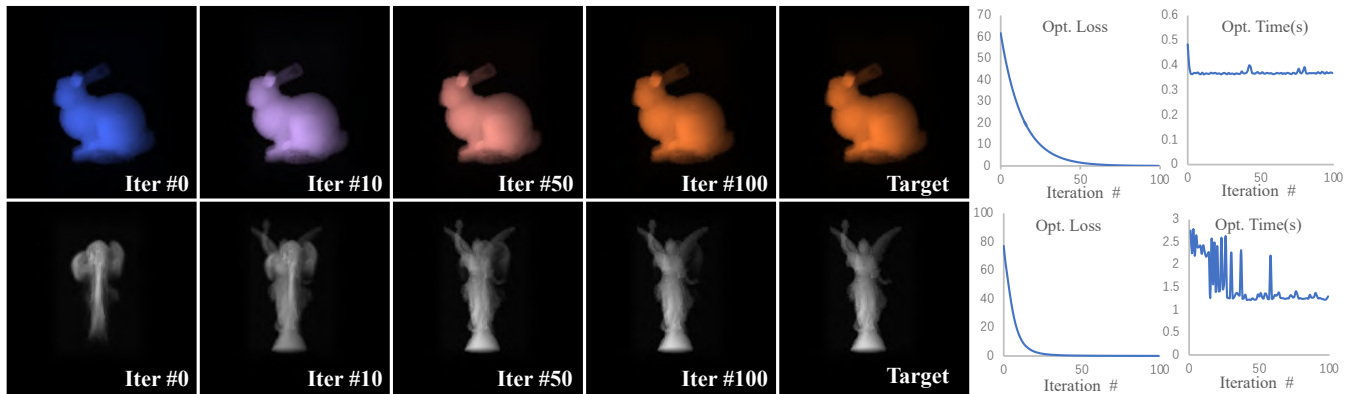
which calculates the derivative of the media radiance's Tylor series expansion term of first order using the discrete form  $\Sigma_{ijk}^9$ . The overall process of solving for  $L_d^0, \bar{L}_d^1$ , and their derivatives is outlined in algorithm 3, and the computed data is stored as voxels.

## 6. Results

Our experimental evaluations were conducted on a high-performance workstation consisting of two Intel Xeon E5-2620 V4 processors with 128 GB of memory, and an NVIDIA GeForce RTX 2080 Ti graphics card. All square images generated for our experiments had a resolution of  $512 \times 512$  pixels, unless stated otherwise. We implemented our approach on both CPU and GPU backends.

For the CPU backend, we wrote the implementation code in C++ and utilized the *Enoki* library [Jak19] for numerical computation. We also employed *OpenMP* for parallel programming to leverage the multicore architecture of the CPU. For the GPU backend, we utilized the Taichi framework [HLY\*21] specifically designed for high-performance parallel numerical computation on GPUs.

To ensure an objective evaluation of our performance, we conducted comprehensive validations on both CPU and GPU backends. Through these evaluations, we assessed the effectiveness of



**Figure 6: Efficiency Evaluation of Different Scene Parameters.** To evaluate the effectiveness and efficiency of our method, we conducted separate optimizations on the lighting color and density field for two different scenes. These validations were performed on CPU backends, and the volume size for both scenes was  $64 \times 96 \times 64$ . The first row shows the results of iterative optimization for the lighting color, involving 3 optimized parameters. The second row demonstrates the results of iterative optimization for the volume density, which requires optimizing 39,216 parameters. In each iteration, the initial values for the current iteration were set as  $L_d^0$  and  $\partial_{\pi} L_d^0$  obtained from the previous calculation. This initialization strategy significantly reduces the number of iterations required to solve the differential diffusion equations.

our approach and compared its performance across different scenarios. Additionally, we provide a supplementary video that includes live demonstrations showcasing the results of iterative optimization with respect to density, illumination, and optical coefficients. We encourage you to watch the video for a more visual understanding of our approach.

### 6.1. Performance and Validation

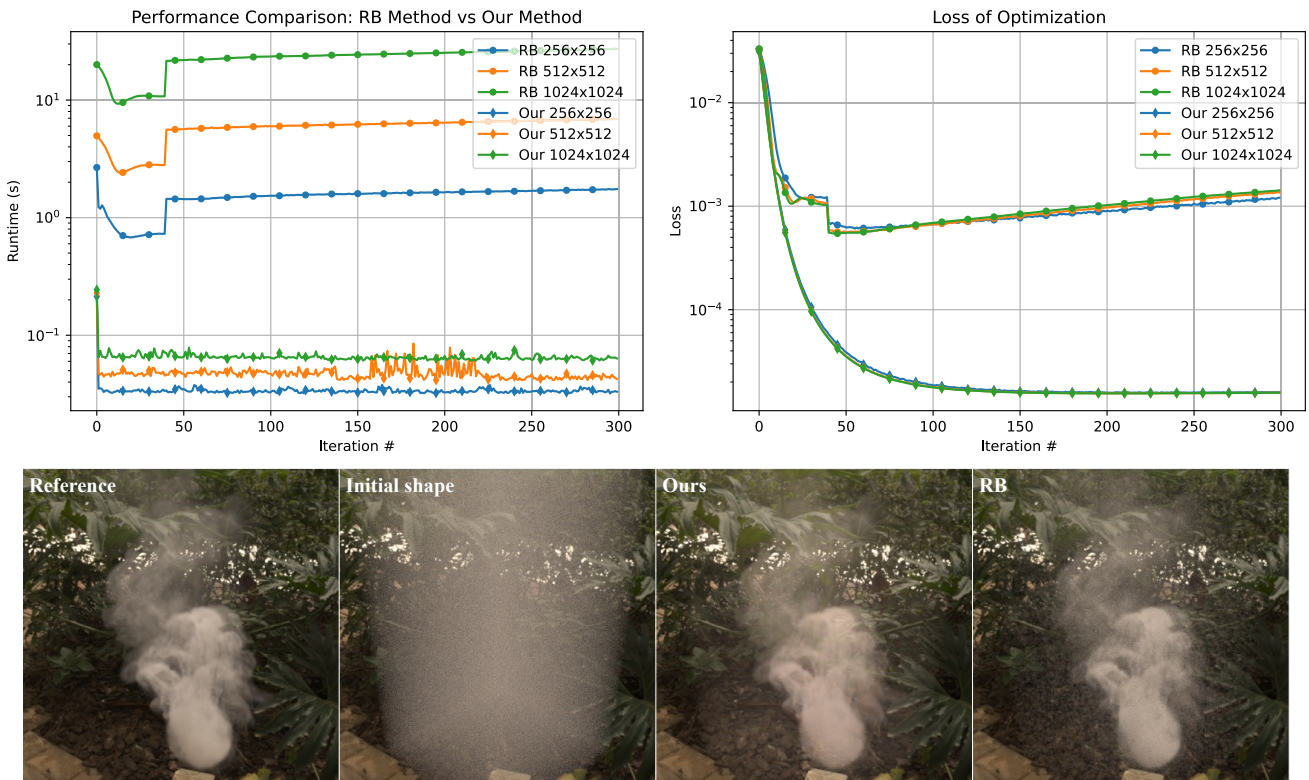
We conducted comprehensive and rigorous experiments to evaluate the performance and effectiveness of our proposed method. Key indicators for evaluating a differentiable rendering framework include runtime efficiency and the accuracy of computed gradients. Additionally, we assessed the effectiveness of the diffusion approximation. It is worth mentioning that the target images used in this paper were generated using standard volumetric path tracing.

*Efficiency evaluation.* We further evaluated the efficiency of our method through validation experiments on CPU backends. We conducted experiments on two different scenes: the optimization of lighting color in a bunny scene (with 3 optimized parameters) and the optimization of a smoke density field (with 39,216 optimized parameters for a volume size of  $64 \times 96 \times 64$ ). As shown in Figure 6, the time spent per iteration ranged between 1 and 3 seconds before the loss converged to the globally optimal solution. With more iterations, the time per iteration decreased significantly as the rendered image approached similarity to the target image. This reduction in time can be attributed to the utilization of the last pre-computation results as the initial condition for solving the partial differential equations during pre-computation, leading to a sharp decrease in the number of iterations as image similarity increased. Eventually, the time per iteration reached approximately 0.37 seconds and 1.35 seconds for the two optimized tasks, respectively. As expected, tasks involving a larger number of optimized parameters required longer runtimes compared to tasks with fewer parameters. However, the runtime did not exhibit an exponential increase

with the number of parameters. These results demonstrate that our method maintains high performance even when faced with the demanding task of computing numerous optimized parameters. The density optimization effects of our method are particularly impressive, as shown in the second row of Figure 6. When considering the target image as the Lucy model with intricate details rendered from a specific camera angle, our rendered image of the reconstructed participating medium volume closely converged to the target image after 100 iterations of optimization. The above comprehensive evaluations and validations provide strong evidence for the efficiency and effectiveness of our method in differentiable rendering.

*Comparing with backpropagation methods.* Recently, backpropagation methods [NDSRJ20, VSJ21] have been proposed for participating media, offering improved performance compared to traditional automatic differentiation (AD) methods. Among these methods, the *radiative backpropagation* (RB) method by Nimier-David et al. [NDSRJ20] has gained attention and provides an open-source implementation. In order to establish a fair comparison, we selected the RB method as a representative backpropagation method and evaluated it alongside our method on GPU backends.

To evaluate the performance of the two methods, we conducted experiments using the same smoke scene rendered at three different resolutions:  $256 \times 256$ ,  $512 \times 512$ , and  $1024 \times 1024$  pixels. In the RB method, we employed 64 samples per pixel (SPP) for both forward and backward rendering. Figure 7 presents a comparison of the efficiency between the two methods when optimizing the density of a heterogeneous volume. The charts depict the loss and optimization time for each iteration. Although both methods eventually converge, our method exhibited a significant advantage in terms of efficiency, being approximately a hundred times faster than the RB method. Specifically, our method completes 300 iterations within total elapsed times of 10.2 seconds, 13.3 seconds, and 18.2 seconds for resolutions of  $256 \times 256$ ,  $512 \times 512$ , and  $1024 \times 1024$  pixels, respectively. This corresponds to average frame rates of 29.4



**Figure 7: Efficiency Comparison Against Radiative Backpropagation (RB) Method [NDSRJ20].** This comparison is conducted on GPU backends. We optimize the density of a heterogeneous volume using both our method and the RB method while employing the same smoke scene at three distinct rendering resolutions:  $256 \times 256$ ,  $512 \times 512$ , and  $1024 \times 1024$  pixels. The RB method uses 64 samples per pixel (SPP) for both forward and backward rendering. The experiments run for 300 iterations. The bottom row of this figure displays the reference image, the initial scene configuration, and the reconstructed results of our method and the RB method, respectively. Furthermore, the loss and runtime for each iteration are depicted in the accompanying charts. Both methods achieve convergence within a low loss interval, typically ranging between  $10^{-2}$  and  $10^{-4}$ , after just a few dozen iterations. Notably, our method showcases a remarkable hundredfold increase in efficiency compared to the RB method, underlining its computational advantage.

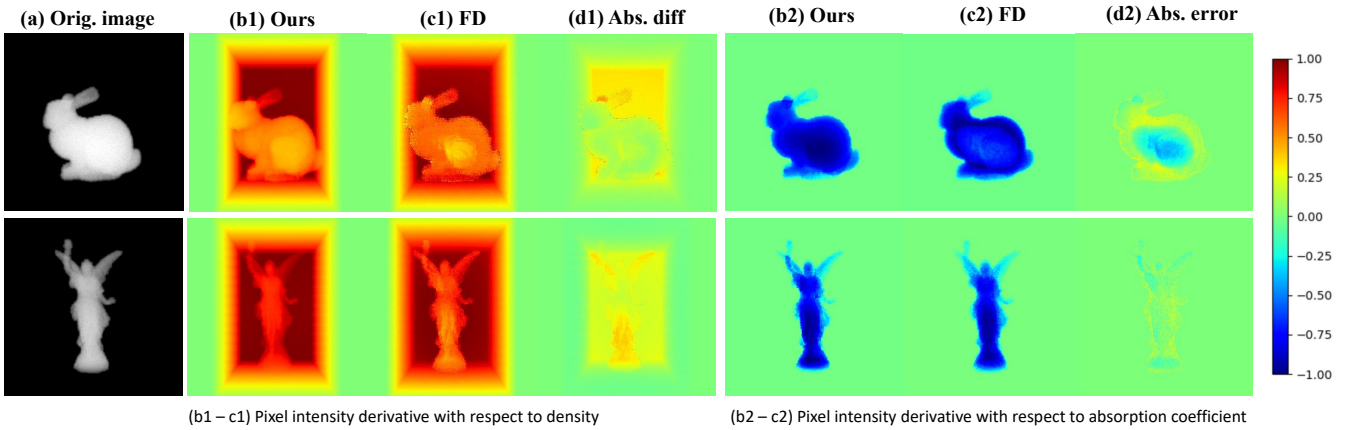
FPS, 22.5 FPS, and 16.4 FPS. In contrast, the RB method requires significantly more time to achieve convergence.

Furthermore, we conducted a comprehensive performance comparison between the CPU and GPU backends for the same optimization task. Notably, the GPU implementation showcases remarkable speed, running approximately 140 times faster than its CPU counterpart. This achievement is particularly significant as our framework is the pioneering physically-based differentiable rendering framework for participating media, capable of achieving real-time performance.

*Derivative validation.* To ensure the accuracy of the gradients computed by our method, we conducted a comprehensive comparison with finite differences (FD). For the FD computation, we employed standard volumetric path tracing with 120 samples per pixel (SPP). Figure 8 illustrates the computed gradients for density and absorption coefficient across various scenes. The volumetric data in these scenes has a size of  $64 \times 94 \times 64$ . The gradient images shown in this comparison are the cumulative gradients projected from a specific viewpoint. We used the FD results as the ground truth for

comparison. The absolute error of the gradients obtained from our method is displayed in **d1** and **d2**, representing the discrepancy between our method and the FD results.

Analyzing these absolute error images, we can observe that our method exhibits cumulative errors in the gradient computation at the surface of the volume data or in regions with a vacuum. These errors can be attributed to two factors. Firstly, our method utilizes ray marching to estimate the radiance and its derivatives for participating media. However, ray marching is a biased sampling method, which introduces errors in the computed results. Secondly, diffusion theory, being an approximation method, may have limitations, especially in regions of vacuum, thin regions, and surfaces with complex lighting conditions. Consequently, the computed results from differential diffusion theory only provide approximate values of the ground truth, leading to errors in the computed gradients. Nevertheless, despite the introduced bias by our approximation, it significantly enhances the efficiency of differentiable rendering. Furthermore, our method produces satisfactory reconstruction re-



**Figure 8: Comparison of Gradients: Differential Diffusion vs. Finite Differences (FD).** We compare the derivatives of pixel intensity with respect to density and absorption coefficient obtained from our method with those computed using finite differences (FD). The FD results are generated through computational brute force, providing reference gradient values. However, due to the prohibitive runtime, the FD method cannot be used for optimization tasks. For the FD computation, we employ a standard volumetric path tracing approach with a free flight sampling based estimator. The gradient images presented in this comparison are the accumulated gradients obtained by projecting them from a specific viewpoint. These images provide measurements of the accumulated gradients along the light path, enabling an assessment of the accuracy of the computed gradients within thin and thick regions of the volume data. Finally, we use the absolute error to assess the discrepancy between our method and the FD results. The absolute error images (**d1** and **d2**) indicate that our method exhibits minimal biases in the thin and vacuum regions compared to the FD results.

sults, indicating that the biases incurred by these approximations remain within an acceptable range.

*Validation of the ability of diffusion approximation derivative.* To assess the effectiveness of the diffusion approximation derivative in the optimization task, we compared the reconstructed effectiveness of using diffusion radiance derivatives, single scattering radiance derivatives, and both derivatives combined to optimize the density field. The same target image was used for all scenarios.

In this comparison, we focus on optimized tasks starting with a bunny-shaped volume. The target images for optimization are single-view images of a dragon, generated using standard volumetric path tracing with free flight sampling. The scene is illuminated by a directional light source, and the volume size is  $64 \times 96 \times 64$ . The objective is to evaluate the impact of different derivative components on the optimization results. To accomplish this, we performed 100 iterations of optimization using diffusion radiance derivatives, single scattering radiance derivatives, and a combination of both. Throughout these iterations, we recorded the loss value and runtime for each iteration, enabling a comprehensive analysis of the effectiveness of each derivative component.

As shown in Figure 9, we observed challenges when using single scattering radiance derivatives to accurately model light transport within participating media, particularly in areas with complex structures. This led to improper computation of derivatives. In contrast, our diffusion approximation method provided valuable and precise derivatives, enabling accurate reconstruction of model details, especially for complex structures. The third row of Figure 9 displays the results obtained by combining diffusion and single scattering derivatives. After 70 iterations, the reconstruction converged to the target image, indicating that the joint use of diffusion

derivatives and single scattering derivatives enhances the efficiency of reconstruction compared to using either derivative alone.

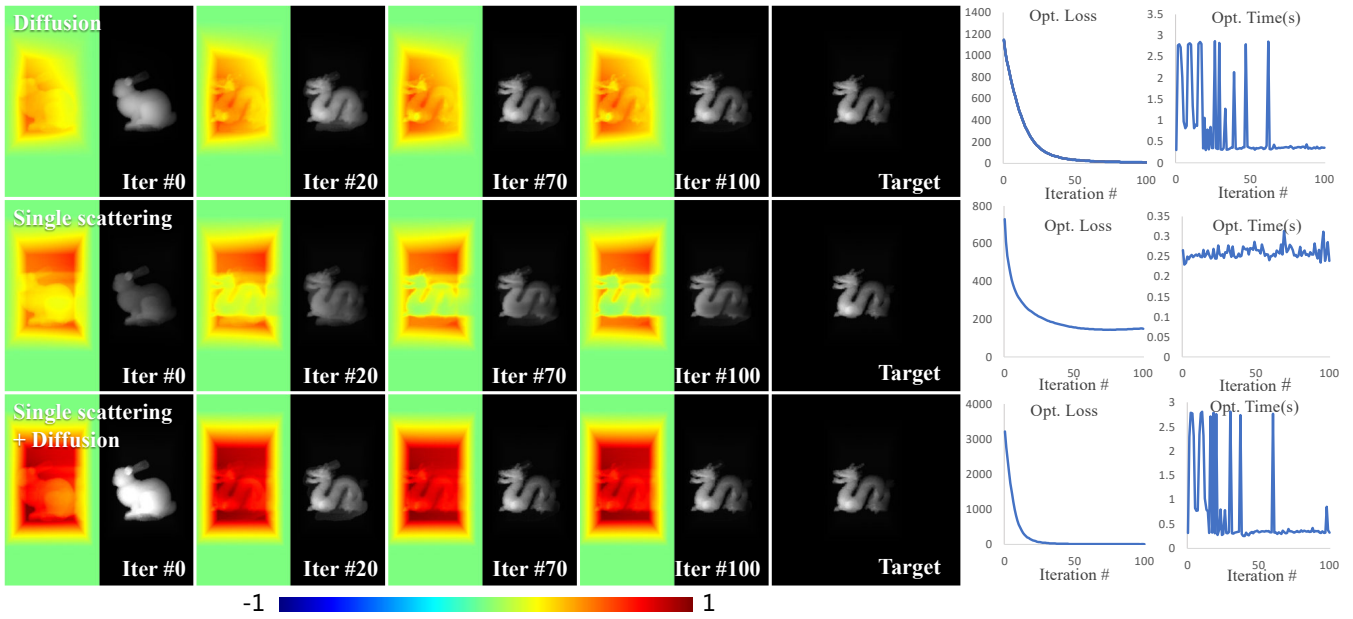
Moreover, it is worth noting that the runtime of solving the differential diffusion equation and diffusion equation decreases as (performing in the pre-computation process) the rendered image approaches similarity with the target image. This property is highly beneficial for the fine-tuning process in reconstruction tasks. Achieving high-quality results during reconstruction often requires an extensive fine-tuning process as the reconstructed results approach the target. However, existing methods start the differential computation from scratch at each iteration and cannot effectively leverage the results of the previous iteration to expedite the solving process. In contrast, our proposed method effectively utilizes the previous computation results during the fine-tuning process, significantly reducing redundant computations at each iteration. This capability improves the efficiency of the reconstruction process, resulting in faster convergence to high-quality results.

## 6.2. Optical Coefficient Optimization

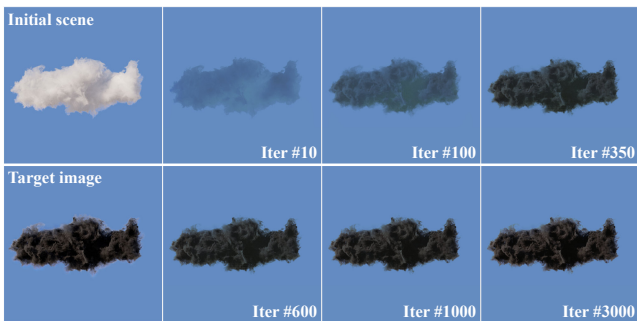
The optical coefficients, including the absorption coefficient, scattering coefficient, and parameters of the phase function, play a crucial role in determining the color and intensity of participating media. These parameters significantly affect the rendered appearance. To validate the effectiveness of optimizing the optical coefficients using our method, we conducted an optimization experiment on the optical coefficient field, which involves jointly optimizing the absorption coefficient field, scattering coefficient field, and asymmetry parameter of the Henyey-Greenstein (HG) phase function.

We present the optimization processing in Figure 10, the scene is



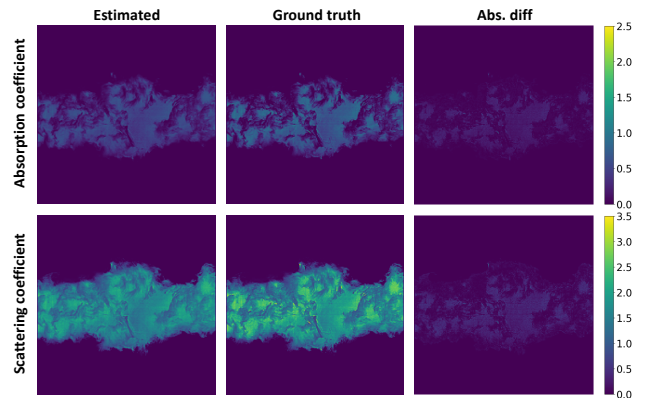


**Figure 9: Comparison of Diffusion Approximation and Single Scattering Derivatives in Density Reconstruction.** Experiment performed on CPU backends. A bunny model scene with directional lighting is used, and the same target image and scene parameters are applied in three experiments. The derivative images (pixel intensity derivative with respect to density) and corresponding rendered (reconstruction) images are shown for each selected iterative optimization step. The first row presents results from our diffusion approximation method, which consistently generates reliable derivatives throughout iterations, effectively assisting in reconstructing model details. In contrast, the second row displays single scattering results, which primarily capture derivatives along the model shape boundary but struggle to properly generate many model details, particularly in regions farther from the light source (e.g., dragon tail). Finally, the third row demonstrates the improved efficiency and result quality achieved by combining both derivatives.

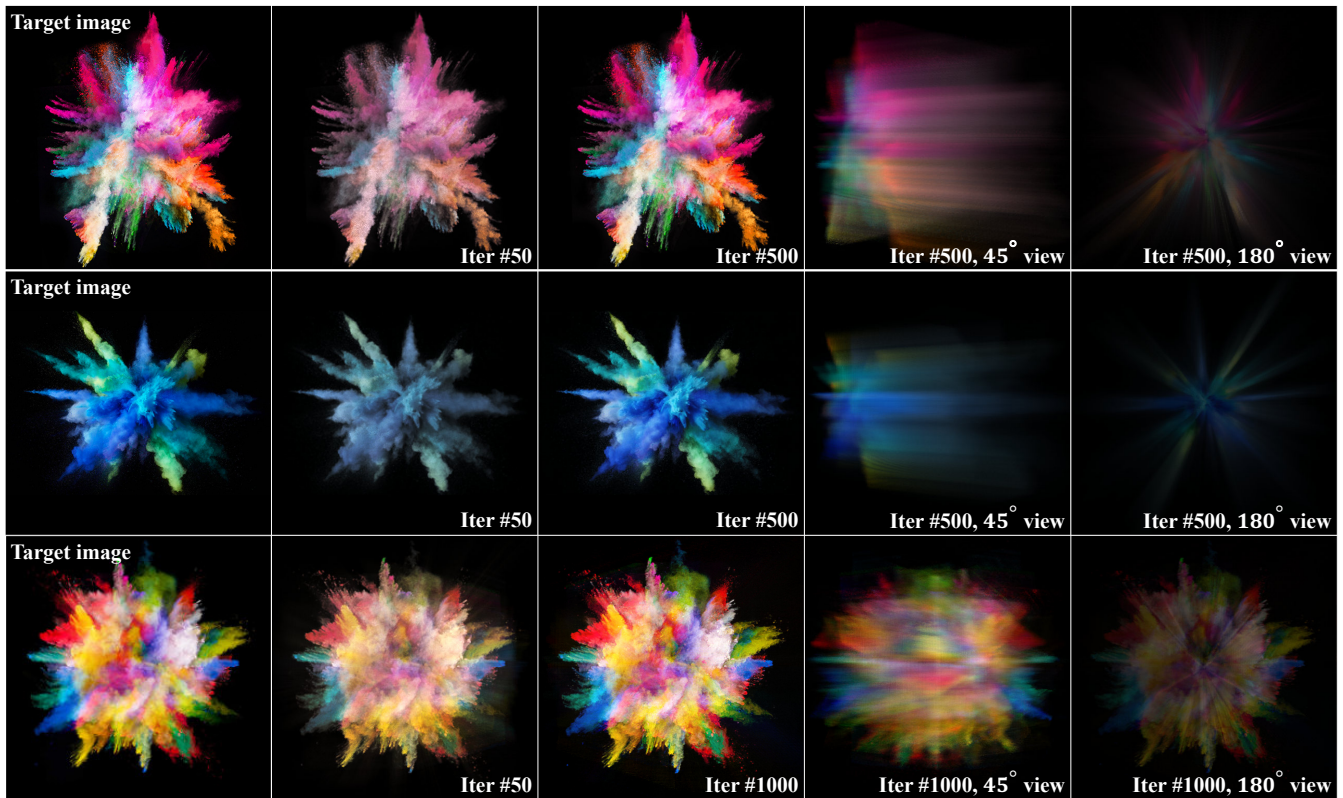


**Figure 10: Optical Coefficient Optimization.** The initial scene features a white cloud, while a rendered black cloud scene serves as the target image. This optimization task involves jointly optimizing the absorption coefficient field, scattering coefficient field, and asymmetry parameters of the Henyey-Greenstein (HG) phase function. Our results showcase the effectiveness of our proposed method in providing reliable derivatives of the optical coefficient, enabling complex optical optimization tasks.

initialized with a white cloud and kept the illumination parameters and density field fixed while jointly optimizing the optical coefficients. After 1000 iterations, the rendered image closely converged



**Figure 11: Validation of Estimated Optical Coefficient Fields.** Generating the slices involves using the red channel of the absorption coefficient field (the first row) and the green channel of the scattering coefficient field (the second row). These slices are produced by summing up values along the z-axis between  $z = 94$  and  $z = 98$ . The small absolute errors indicate that our method accurately estimates optical coefficient fields with a high level of precision, highlighting its advantageous performance.



**Figure 12: Complex Colored Smoke Reconstruction.** The first and second rows depict the reconstructed colored smoke scene from a single-view image. In contrast, the third row presents the reconstruction using ortho-symmetric four-view images. This means that the front and back views use the same image, while the left and right views are obtained by vertically flipping this image. This method ensures symmetry and consistency in the reconstruction process, yielding reliable results. In both scenarios, we jointly optimize the density field and optical coefficients. The optimization involves over 49 million parameters, posing considerable challenges. Despite slightly blurred results in the novel view images, the reference view images demonstrate our method’s ability to accurately reconstruct the intricate visual effects of complex colored smoke. For live demonstrations of our reconstructions, please refer to the supplementary video.

to the target image, then the optimization transitioned into a fine-tuning phase, where further refinements were made to enhance the similarity between the rendered and target images. We continued the optimization for an additional 2000 iterations, during which the loss decreased slightly, but the reconstructed details were significantly enhanced.

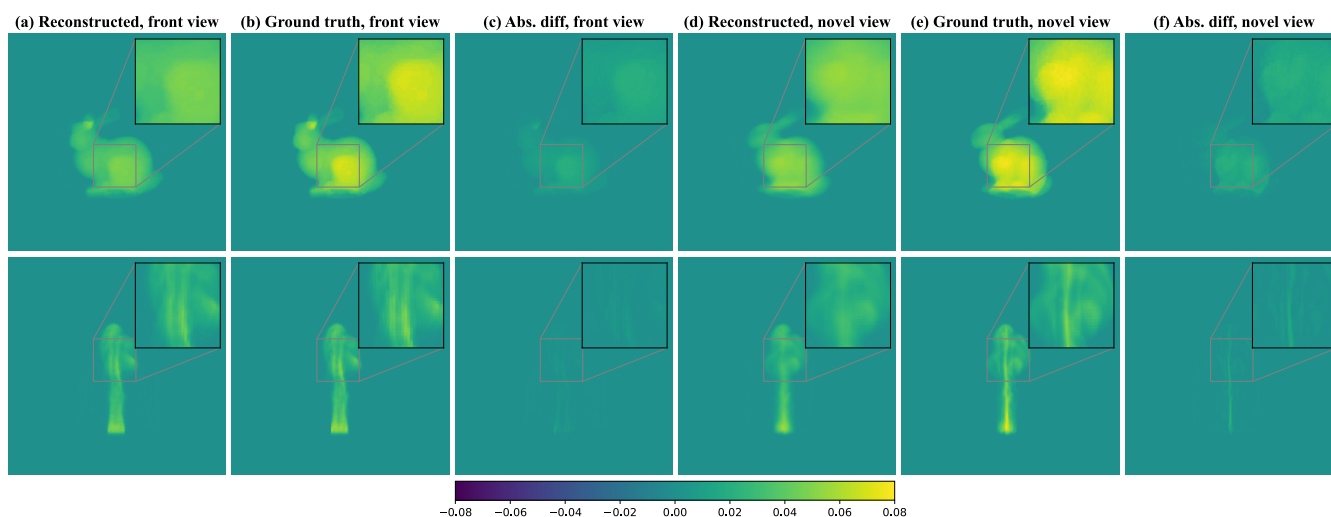
To validate the precision of our estimated optical coefficient fields compared to the ground truth, we conduct a comprehensive analysis by comparing slices from both the estimated fields and the ground truth, along with their corresponding absolute errors. These results are illustrated in Figure 11. In particular, the optical fields have a volumetric size of  $192 \times 192 \times 192 \times 3$  (with the last dimension representing RGB channels). The slices are generated by summing values along the z-axis within a range from  $z = 94$  to  $z = 98$ , situated in the middle of the volumetric data. The red channel of absorption coefficient fields and the green channel of scattering coefficient fields are presented in the first and second rows, respectively. The final column displays the absolute errors between our estimations and the ground truth. Notably, our estimated results closely resemble the ground truth, with the minimal absolute error underlining the high accuracy of our estimates.

The asymmetry parameter of the Henyey-Greenstein (HG) phase function is a scalar ranging from -1 to 1, which governs the direction of scattering: positive for forward, negative for backward, and zero for isotropic scattering. In our experiment, we utilize a consistent asymmetry parameter for the HG phase function throughout the optimization process. We initialize the asymmetry parameter at 0.67, while the actual value for the black cloud is  $-0.802$ . Over the course of 3,000 iterations, the asymmetry parameter converges to  $-0.813$ , remarkably close to the actual value. This outcome highlights our method’s exceptional ability to accurately estimate phase function parameters, reinforcing its advantage.

### 6.3. Density Reconstruction

In order to assess the reconstructed ability of our proposed method, we conducted two sets of three-dimensional reconstruction experiments: 1) single-view density field reconstruction and 2) multi-view density field reconstruction.

In the following experiments, we tackled highly complex tasks with volume resolutions of  $192 \times 192 \times 192$  (Figure 12) and  $64 \times 96 \times 64$  (Figure 13). These tasks involved a large number of param-



**Figure 13: Validation of Reconstructed Density Field.** The reference images were generated using volumetric path tracing from ten viewpoints. The cumulative density images, presented in columns (a), (b), and (d), (e), depict the density fields projected onto the image plane from frontal and novel viewpoints. These images provide insights into the volume’s optical depth, revealing regions of varying thickness. Columns (c) and (f) show the absolute difference between the reconstructed and ground truth volumes. Minor deviations may occur, particularly in the leg region of the bunny, due to our method’s use of a diffusion approximation for differentiable rendering, which does not fully simulate light scattering in regions with significant changes in optical thickness. However, overall, our method achieves a high level of accuracy in reconstructing the density field.

eters, ranging from  $10^6$  to  $10^7$ . Traditional methods heavily rely on Monte Carlo estimation to compute the required derivatives, which becomes challenging when dealing with a large number of parameters. However, our theory and method enable rapid and accurate density field reconstruction. We utilized three high-definition wallpapers featuring color smoke as target images for the reconstruction process. As demonstrated in Figure 12, reconstruction results consistently capture the visual effects of complex participating media from the same perspective.

*Density optimization based on single-view images.* The target images in the first and second rows of Figure 12 depict complex colored smoke, necessitating joint optimization of the density field (with 7 million optimized parameters), absorption coefficient field (with 21 million optimized parameters), and scattering coefficient field (with 21 million optimized parameters). Consequently, more than 49 million relevant optimized parameters are involved, presenting highly challenging optimization tasks. Previous methods relying on the Monte Carlo approach struggle to directly address such tasks. In contrast, our method leverages the differential diffusion theory to approximate the derivative of multiple scattering radiance, circumventing the complexities and time-consuming nature of Monte Carlo path tracing computations.

*Density optimization based on multi-view images.* Reconstructing the density field based on a single view is often an ill-posed problem due to the multiple parameter combinations that can correspond to a single image. Although the experiments in Figure 12 showcase our method’s ability to consistently capture and reconstruct the visual effects of complex participating media from a single view, they do not fully demonstrate the multi-view reconstruc-

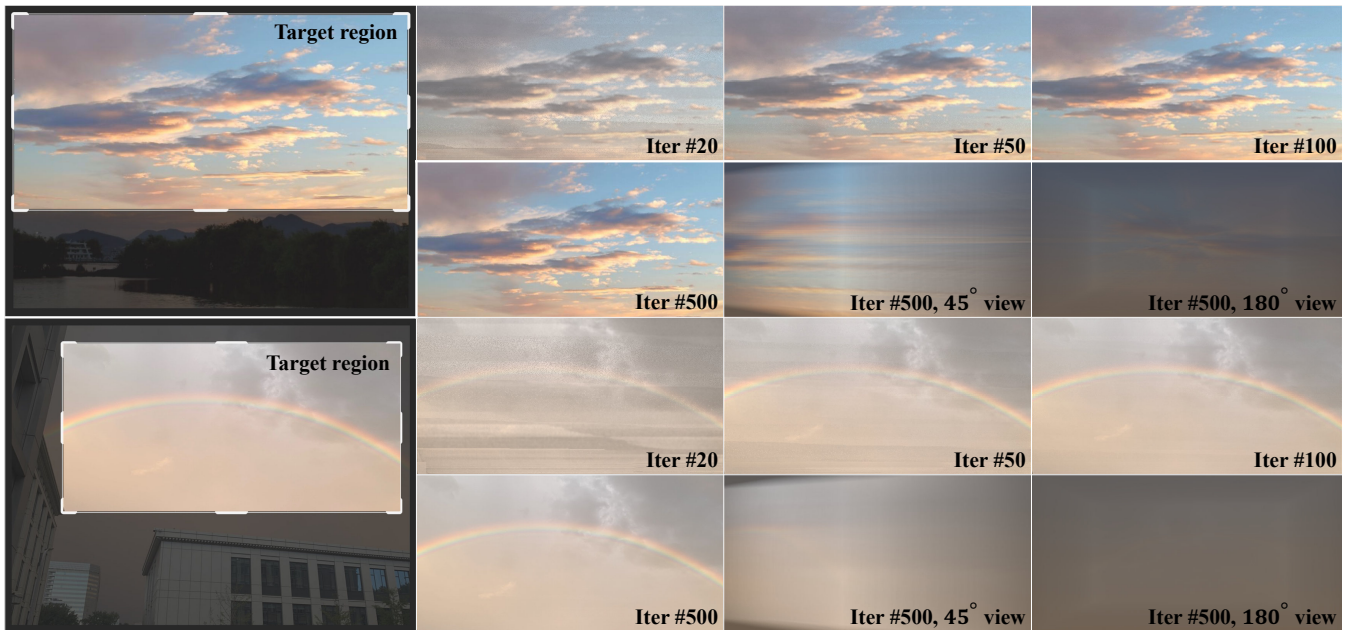
tion capability. To evaluate this aspect, we conducted experiments to reconstruct scenes from both sparse views and dense views.

In our approach to sparse view reconstruction, we employ an orthogonal perspective as a guiding framework, incorporating constraints into the reconstruction process. We rely on a high-definition wallpaper showcasing colored smoke, chosen as the reference image for reconstructing both front and back viewpoints. To ensure coherence and alignment, this image is vertically flipped to serve as the basis for both left and right perspectives, thus maintaining consistency throughout the reconstruction process. The outcomes are presented in the third row of Figure 12. These results highlight the capability of our method to approximate the primary shape even when data is scarce. However, it is important to note that our outcomes exhibit some blurriness. This blurring arises due to the inherent absence of robust visual cues in these new viewpoints. Despite this limitation, our method excels at generating valuable reconstructions from sparse data.

To prove that our method can create accurate reconstructions in dense views, we carry out volume reconstruction using 10 different views. To generate these target images, we position cameras all around the volume object, equally spaced at 36-degree intervals around its y-axis. The volume size we work with is  $64 \times 96 \times 64$ , starting off with an empty density setup. To bring our scene to life, we use directional lighting and rely on volumetric path tracing to create images from these 10 different camera viewpoints. This process gives us a series of target images from multiple views. The entire reconstruction process spans 1000 iterations and the reconstructed results are shown in Figure 13.

We evaluated the accuracy of our proposed method in recon-





**Figure 14: Reconstruction Based on Captured Natural Images.** Since our method focuses on reconstructing participating media volumes rather than surface models, we specifically reconstruct the sky regions in this experiment. The initial scene is empty, and we perform joint optimization of the density field, optical coefficient field, and illumination parameters. The rendered image has a size of  $592 \times 333$ , and the volume resolution is set to  $192 \times 192 \times 192$ .

structing the density field by comparing the reconstructed densities with the ground truth volumes from both frontal and novel viewpoints. The absolute difference between the reconstructed and ground truth volumes was used as a measurement. The density images were generated by projecting the volume data onto the image plane from a special viewpoint. In (c) and (f), some minor deviations are observed in certain regions, such as the leg region of the bunny. However, overall, our method achieved high accuracy. These deviations are a result of our method’s approximation approach, which may not fully capture the intricate details of light scattering in regions with significant changes in optical thickness. Thus, deviations can occur in both the forward and backward computation. Nevertheless, our method demonstrates superior performance in efficiently reconstructing the density field compared to Monte Carlo methods, with high overall reconstruction accuracy.

#### 6.4. Reconstruction based on Captured Images

While the previous optimization experiments focused on synthetic images, it is essential to evaluate the practical applicability of our method using natural images as targets. In real-world scenarios, optimizing different scene parameters based on these targets requires joint optimization. To validate the effectiveness of our method in practical applications, we designed an experiment using natural cloud images as input to jointly optimize the density field, optical coefficients, and lighting parameters. The results of this experiment are presented in Figure 14.

In summary, our proposed differential diffusion theory offers the advantages of accuracy and speed in optimizing and reconstructing the density field, overcoming the time-consuming and challeng-

ing nature of traditional Monte Carlo methods. This characteristic enables the integration of our theory into other complex gradient-based optimization algorithms, such as incorporating it as a layer within a neural network.

#### 7. Conclusion and Limitations

In this work, we have presented a novel differential diffusion theory for participating media, enabling the computation of arbitrary derivatives of scene parameters. Our theory leverages the differential form of the diffusion equation in the forward rendering process, resulting in significantly improved computational efficiency compared to Monte Carlo methods.

However, our method does have certain limitations. Firstly, it does not support surface model rendering, which restricts its applicability in certain scenarios. Nonetheless, our method can be integrated as a component within state-of-the-art general-purpose differentiable rendering frameworks, thereby expanding its range of applications. Also, our method is limited to regular volume data structures. While this limitation still allows for the representation of various types of participating media such as fluids, smoke, and clouds, it may not cover all possible irregular volume data.

For future research, it would be interesting to explore techniques for integrating surface model rendering into our diffusion approximation framework without compromising computational efficiency. This would enhance the versatility of our method and enable a broader range of applications.



## Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62272019), and the Internal Research Fund of The 15th Research Institute of China Electronics Technology Group Corporation (No. HTWB20233035).

## References

- [Bli82] BLINN J. F.: Light reflection functions for simulation of clouds and dusty surfaces. *Acm Siggraph Computer Graphics* 16, 3 (1982), 21–29. 2
- [CGL\*19] CHEN W., GAO J., LING H., SMITH E. J., LEHTINEN J., JACOBSON A., FIDLER S.: Learning to predict 3d objects with an interpolation-based differentiable renderer. *arXiv preprint arXiv:1908.01210* (2019). 3
- [Cha60] CHANDRASEKHAR S.: *Radiative Transfer*. Courier Corporation, 1960. 2, 4
- [dI11] D’EON E., IRVING G.: A quantized-diffusion model for rendering translucent materials. *ACM transactions on graphics (TOG)* 30, 4 (2011), 1–14. 2
- [DJ05] DONNER C., JENSEN H. W.: Light diffusion in multi-layered translucent materials. *ACM Transactions on Graphics (ToG)* 24, 3 (2005), 1032–1039. 2
- [GCM\*18] GENOVA K., COLE F., MASCHINOT A., SARNA A., VLASIC D., FREEMAN W. T.: Unsupervised training for 3d morphable model regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 8377–8386. 3
- [GLZ16] GKIOULEKAS I., LEVIN A., ZICKLER T.: An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision* (2016), Springer, pp. 685–701. 3
- [GZB\*13] GKIOULEKAS I., ZHAO S., BALA K., ZICKLER T., LEVIN A.: Inverse volume rendering with material dictionaries. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–13. 3
- [HF18] HENDERSON P., FERRARI V.: Learning to generate and reconstruct 3d meshes with only 2d supervision. *arXiv preprint arXiv:1807.09259* (2018). 3
- [HLY\*21] HU Y., LIU J., YANG X., XU M., KUANG Y., XU W., DAI Q., FREEMAN W. T., DURAND F.: Quantaichi: a compiler for quantized simulations. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–16. 2, 9
- [HvdH81] HULST H. C., VAN DE HULST H. C.: *Light scattering by small particles*. Courier Corporation, 1981. 2
- [IK12] ISAACSON E., KELLER H. B.: *Analysis of numerical methods*. Courier Corporation, 2012. 6
- [Jak19] JAKOB W.: Enoki: structured vectorization and differentiation on modern processor architectures, 2019. <https://github.com/mitsuba-renderer/enoki>. 9
- [JSRV22] JAKOB W., SPEIERER S., ROUSSEL N., VICINI D.: Dr. jit: a just-in-time compiler for differentiable rendering. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–19. 3
- [KBM\*20] KATO H., BEKER D., MORARIU M., ANDO T., MATSUOKA T., KEHL W., GAIDON A.: Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057* (2020). 3
- [KPS\*14] KOERNER D., PORTSMOUTH J., SADLO F., ERTL T., EBERHARDT B.: Flux-limited diffusion for multiple scattering in participating media. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 178–189. 2
- [KUH18] KATO H., USHIKU Y., HARADA T.: Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 3907–3916. 3
- [KVH84] KAJIYA J. T., VON HERZEN B. P.: Ray tracing volume densities. *ACM SIGGRAPH computer graphics* 18, 3 (1984), 165–174. 2
- [LADL18] LI T.-M., AITTALA M., DURAND F., LEHTINEN J.: Differentiable monte carlo ray tracing through edge sampling. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–11. 3
- [LB14] LOPER M. M., BLACK M. J.: Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision* (2014), Springer, pp. 154–169. 3
- [LCLL19] LIU S., CHEN W., LI T., LI H.: Soft rasterizer: Differentiable rendering for unsupervised single-view mesh reconstruction. *arXiv preprint arXiv:1901.05567* (2019). 3
- [LHJ19] LOUBET G., HOLZSCHUCH N., JAKOB W.: Reparameterizing discontinuous integrands for differentiable rendering. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–14. 3
- [NDMKJ22] NIMIER-DAVID M., MÜLLER T., KELLER A., JAKOB W.: Unbiased inverse volume rendering with differential trackers. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–20. 3
- [NDSRJ20] NIMIER-DAVID M., SPEIERER S., RUIZ B., JAKOB W.: Radiative backpropagation: an adjoint method for lightning-fast differentiable rendering. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 146–1. 2, 3, 10, 11
- [NDVZJ19] NIMIER-DAVID M., VICINI D., ZELTNER T., JAKOB W.: Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–17. 3
- [PBDCO19] PETERSEN F., BERMANO A. H., DEUSSEN O., COHEN-OR D.: Pix2vex: Image-to-geometry reconstruction using a smooth differentiable renderer. *arXiv preprint arXiv:1903.11149* (2019). 3
- [Sta95] STAM J.: Multiple scattering as a diffusion process. In *Eurographics Workshop on Rendering Techniques* (1995), Springer, pp. 41–50. 2, 4, 18
- [Vea97] VEACH E.: *Robust Monte Carlo methods for light transport simulation*, vol. 1610. Stanford University PhD thesis, 1997. 3
- [VSJ21] VICINI D., SPEIERER S., JAKOB W.: Path replay backpropagation: differentiating light paths using constant memory and linear time. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–14. 2, 3, 10
- [ZJL20] ZHAO S., JAKOB W., LI T.-M.: Physics-based differentiable rendering: From theory to implementation. In *ACM SIGGRAPH 2020 Courses* (New York, NY, USA, 2020), SIGGRAPH ’20, Association for Computing Machinery. 2, 3
- [ZMY\*20] ZHANG C., MILLER B., YAN K., GKIOULEKAS I., ZHAO S.: Path-space differentiable rendering. *ACM Trans. Graph.(Proc. SIGGRAPH)* 39, 6 (2020), 143. 3
- [ZRL\*08] ZHOU K., REN Z., LIN S., BAO H., GUO B., SHUM H.-Y.: Real-time smoke rendering using compensated ray marching. In *ACM SIGGRAPH 2008 papers*. 2008, pp. 1–12. 2
- [ZWZ\*19] ZHANG C., WU L., ZHENG C., GKIOULEKAS I., RAMAMOORTHY R., ZHAO S.: A differential theory of radiative transfer. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–16. 2, 3, 5
- [ZYZ21] ZHANG C., YU Z., ZHAO S.: Path-space differentiable rendering of participating media. *ACM Transactions on Graphics* 40, 4 (2021). 3

## Appendix A: Basic Formulas for Forward Rendering

We give some basic formulas of forward rendering here, which are mostly corresponding to section 3.

### 1) Reduced Incident Radiance $L_{ri}$

The term "reduced incident radiance" refers to the source illumination, denoted by  $L_{in}$ , that reaches a viewpoint  $x$  along a specific

direction  $\omega$  with some attenuation by the media:

$$L_{ri}(x, \omega) = L_{in}(\omega)\tau(x_a, x), \quad (31)$$

where  $\tau(x_a, x)$  denote the *transmittance* from point  $x_a$  to  $x$ , representing the fraction of radiant energy that passes through a participating medium without being absorbed or scattered. In this context, consider two points within the medium,  $x_i$  and  $x_j$ . The extinction coefficient,  $\kappa_r$ , is defined as the sum of the absorption coefficient  $\kappa_a$  and the scattering coefficient  $\kappa_s$ :  $\kappa_r(x) = \kappa_a(x) + \kappa_s(x)$ . Thus, the transmittance  $\tau$  between the points  $x_i$  and  $x_j$  can be expressed as:

$$\tau(x_i, x_j) = e^{-\int_{x_i}^{x_j} \kappa_r(x_u) du}.$$

## 2) Single Scattering Radiance $L_{ss}$

The term "single scattering radiance" refers to the integrated radiance that has been scattered only once before reaching the viewpoint  $x$  along the direction  $\omega$ , it is computed by integrating the radiance contributions along the corresponding view ray:

$$L_{ss}(x, \omega) = \int_{D_b}^{D_a} \tau(x_u, x_b) \kappa_r(x_u) J_{ss}(x_u, \omega) du. \quad (32)$$

Here,  $D_a$  and  $D_b$  represent the distances from the volume boundary points  $x_a$  and  $x_b$  to the viewpoint  $x$ , as depicted in Fig. 2. The variable  $x_u$  denotes a point inside the volume of the participating medium. The single scattering term  $J_{ss}$  represents reduced incident radiance, whose first scattering interaction in the participating medium occurs at  $x_u$ :

$$J_{ss}(x_u, \omega) = \frac{\Omega}{4\pi} \int_{S^2} L_{ri}(x_u, \omega') p(\omega, \omega') d\omega'. \quad (33)$$

This equation is an integral of solid angle over a unit sphere  $S^2$ .  $\Omega$  is termed albedo, representing the fraction of light that is scattered by the medium as opposed to being absorbed:  $\Omega = \kappa_s/\kappa_r$ . The phase function  $p(\omega, \omega')$  describes the directional scattering distribution of the participating media, which is usually normalized such that its integral over all directions is  $4\pi$  and can be parameterized by the angle between the incident and outgoing directions as  $p(\omega \cdot \omega')$ .

## 3) The First Moment of the Phase Function $\bar{\mu}$

A straightforward measure of the anisotropy within the participating media can be obtained through the first moment of the phase function, defined by:

$$\bar{\mu} = \frac{3}{2} \int_{-1}^1 \mu p(\mu) d\mu. \quad (34)$$

A negative value of  $\bar{\mu}$  indicates that the phase function favors back scattering over forward scattering. Conversely, a positive value of  $\bar{\mu}$  signifies a preference for forward scattering over back scattering.

## 4) Derivation of $J_{ms}$

$J_{ms}$  represents the multiple scattering term, signifying the radiance integral over all directions with the multiple scattering radiance  $L_d$  at position  $x_u$ . Although  $J_{ms}$  can be computed using the Monte Carlo estimation, this approach typically involves time-consuming calculation processes. Therefore, we employ the diffusion approximation for these calculations.

The core idea of the diffusion approximation is premised on the observation that as the number of scattering events increases, the angular dependence tends to be smoothed out. Consequently, the multiple scattering term can be approximated by expanding it into the first two terms of its Taylor expansion, considering only the directional component. The derivation process is as follows:

$$\begin{aligned} J_{ms}(x_u, \omega) &= \frac{\Omega}{4\pi} \int_{S^2} L_d(x_u, \omega) p(\omega, \omega') d\omega' \\ &= \frac{\Omega}{4\pi} \int_{S^2} (L_d^0(x_u) + \bar{L}_d^1(x_u) \cdot \omega) p(\omega, \omega') d\omega' \\ &= \Omega L_d^0(x_u) + \frac{\Omega \bar{\mu}}{3} \cdot \bar{L}_d^1(x_u) \cdot \omega. \end{aligned}$$

## 5) The Expression of $\kappa, S$

$$\kappa(x_u) = (\sigma_{tr} \rho(x_u))^{-1}, \quad (35)$$

$$S(x_u) = \kappa_s(x_u) Q_{ri}^0(x_u) - \frac{\sigma_s}{\sigma_{tr}} \nabla \cdot \bar{Q}_{ri}^1(x_u). \quad (36)$$

Assume that  $Q_{ri}$  is the radiance due to the first scatter of  $L_{ri}$ , while  $Q_{ri}^0$  and  $\bar{Q}_{ri}^1$  are the first two terms of the Taylor expansion of  $Q_{ri}$ , the expressions of  $Q_{ri}^0$  and  $\bar{Q}_{ri}^1$  are:

$$Q_{ri}^0(x_u) = L_{ri}^0(x_u), \quad (37)$$

$$\bar{Q}_{ri}^1(x_u) = \frac{\bar{\mu}}{3} \bar{L}_{ri}^1(x_u). \quad (38)$$

We provide the derivation of  $Q_{ri}^0$  and  $\bar{Q}_{ri}^1$  as follows:

$$\begin{aligned} Q_{ri}(x_u, \omega) &= \frac{1}{4\pi} \int_{4\pi} p(\omega \cdot \omega') L_{ri}(x_u, \omega') d\omega' \\ &= \frac{1}{4\pi} \int_{4\pi} (1 + \bar{\mu}(\omega \cdot \omega') + \dots) (L_{ri}^0(x_u) + \bar{L}_{ri}^1(x_u) \cdot \omega') d\omega' \\ &= L_{ri}^0(x_u) + \frac{\bar{\mu}}{3} \bar{L}_{ri}^1(x_u) \cdot \omega \\ &= Q_{ri}^0(x_u) + \bar{Q}_{ri}^1(x_u) \cdot \omega. \end{aligned}$$

Please refer to [Sta95] for more details about the derivation of the classical diffusion equation.

## Appendix B: Basic Formulas for Backward Rendering

We provide the derivative formulas of backward rendering here, which are mostly corresponding to section 4.

### 1) Differentiation of Single Scattering Radiance

Single scattering radiance  $L_{ss}$  represents the light that scatters only once before reaching the viewpoint at  $x$ . The derivative of single scattering radiance can be computed through the ray marching process with minimal calculations. The derivative of  $L_{ss}$  is:

$$\begin{aligned} \partial_\pi L_{ss}(x, \omega) &= \int_{D_b}^{D_a} (\partial_\pi \tau(x_u, x_b) \kappa_r(x_u) + \tau(x_u, x_b) \partial_\pi \kappa_r(x_u)) J_{ss}(x_u, \omega) du \\ &\quad + \int_{D_b}^{D_a} \tau(x_u, x_b) \kappa_r(x_u) \partial_\pi J_{ss}(x_u, \omega) du. \end{aligned} \quad (39)$$

The derivative of the single scattering radiance  $\partial_\pi L_{ss}$  is obtained

by performing ray marching and evaluating the integral in Equation 39 through the accumulation of partial derivatives of transmittance, extinction coefficient, and single scattering source term along the ray path.

To compute  $\partial_{\pi} L_{ss}(x, \omega)$ , we require  $\partial_{\pi} J_{ss}(x_u, \omega)$ . As mentioned in Section 4.1,  $L_{ri}$  is assumed to be continuous on  $\mathbb{S}^2$ , allowing us to neglect the derivative of the interface term. The expression for  $\partial_{\pi} J_{ss}(x_u, \omega)$  is then given by:

$$\begin{aligned} \partial_{\pi} J_{ss}(x_u, \omega) &= \frac{\partial_{\pi} \Omega}{4\pi} \int_{\mathbb{S}^2} L_{ri}(x_u, \omega') p(\omega, \omega') d\omega' \\ &+ \frac{\Omega}{4\pi} \int_{\mathbb{S}^2} (\partial_{\pi} L_{ri}(x_u, \omega') p(\omega, \omega') \\ &+ L_{ri}(x_u, \omega') \partial_{\pi} p(\omega, \omega')) d\omega'. \end{aligned} \quad (40)$$

The derivative of the single scattering radiance contribution  $\partial_{\pi} J_{ss}$  can be computed by integrating the partial derivatives of the incident radiance  $L_{ri}$  and the phase function  $p(\omega, \omega')$  over the sphere  $\mathbb{S}^2$  in Equation 40. The continuity assumption of  $L_{ri}$  on  $\mathbb{S}^2$  allows us to neglect the interface term.

## 2) The Derivative of Reduced Incident Radiance $L_{ri}$

The derivative of  $L_{ri}$  can be expressed as:

$$\partial_{\pi} L_{ri}(x_i, \omega') = \tau(x_a, x_i) \partial_{\pi} L_{in}(\omega') + \partial_{\pi} \tau(x_a, x_i) L_{in}(\omega'). \quad (41)$$

## 3) The Derivative of Optical Coefficient $\kappa_i$

Define  $\kappa_i(x_u)$  as the optical coefficient. Then, the expression of  $\partial_{\pi} \kappa_i(x_u)$  is:

$$\partial_{\pi} \kappa_i(x_u) = \partial_{\pi} \sigma_i \rho(x_u) + \sigma_i \partial_{\pi} \rho(x_u), \quad (42)$$

where  $\sigma_i$  is the corresponding optical cross section of  $\kappa_i$ .  $\sigma_i \in \{\sigma_a, \sigma_s, \sigma_t\}$  indicate the absorption, scattering and extinction cross section, respectively.

## 4) The Derivative of Transmittance $\tau$

$\tau(x_i, x_j)$  is the transmittance from position  $x_i$  to  $x_j$ , the derivative  $\partial_{\pi} \tau(x_i, x_j)$  can be expressed as:

$$\partial_{\pi} \tau(x_i, x_j) = -\tau(x_i, x_j) \int_{D_i}^{D_j} \partial_{\pi} \kappa_t(x_u) du. \quad (43)$$

## 5) The Expression of $\Sigma_{ijk}$

The expression of  $\Sigma_{ijk}^1$ :

$$\begin{aligned} \Sigma_{ijk}^1 &= (\partial_{\pi} \kappa_{i+1,j,k} - \partial_{\pi} \kappa_{i-1,j,k}) ((L_d^0)_{i+1,j,k} - (L_d^0)_{i-1,j,k}) \\ &+ (\partial_{\pi} \kappa_{i,j+1,k} - \partial_{\pi} \kappa_{i,j-1,k}) ((L_d^0)_{i,j+1,k} - (L_d^0)_{i,j-1,k}) \\ &+ (\partial_{\pi} \kappa_{i,j,k+1} - \partial_{\pi} \kappa_{i,j,k-1}) ((L_d^0)_{i,j,k+1} - (L_d^0)_{i,j,k-1}). \end{aligned}$$

The expression of  $\Sigma_{ijk}^2$ :

$$\begin{aligned} \Sigma_{ijk}^2 &= (\kappa_{i+1,j,k} - \kappa_{i-1,j,k}) ((\partial_{\pi} L_d^0)_{i+1,j,k} - (\partial_{\pi} L_d^0)_{i-1,j,k}) \\ &+ (\kappa_{i,j+1,k} - \kappa_{i,j-1,k}) ((\partial_{\pi} L_d^0)_{i,j+1,k} - (\partial_{\pi} L_d^0)_{i,j-1,k}) \\ &+ (\kappa_{i,j,k+1} - \kappa_{i,j,k-1}) ((\partial_{\pi} L_d^0)_{i,j,k+1} - (\partial_{\pi} L_d^0)_{i,j,k-1}). \end{aligned}$$

The expression of  $\Sigma_{ijk}^3$ :

$$\begin{aligned} \Sigma_{ijk}^3 &= (L_d^0)_{i+1,j,k} + (L_d^0)_{i-1,j,k} + (L_d^0)_{i,j+1,k} \\ &+ (L_d^0)_{i,j-1,k} + (L_d^0)_{i,j,k+1} + (L_d^0)_{i,j,k-1} - 6(L_d^0)_{i,j,k}. \end{aligned}$$

The expression of  $\Sigma_{ijk}^4$ :

$$\begin{aligned} \Sigma_{ijk}^4 &= (\kappa_{i+1,j,k} - \kappa_{i-1,j,k}) ((L_d^0)_{i+1,j,k} - (L_d^0)_{i-1,j,k}) \\ &+ (\kappa_{i,j+1,k} - \kappa_{i,j-1,k}) ((L_d^0)_{i,j+1,k} - (L_d^0)_{i,j-1,k}) \\ &+ (\kappa_{i,j,k+1} - \kappa_{i,j,k-1}) ((L_d^0)_{i,j,k+1} - (L_d^0)_{i,j,k-1}). \end{aligned}$$

The expression of  $\Sigma_{ijk}^5$ :

$$\Sigma_{ijk}^5 = \Sigma_{ijk}^3 + 6(L_d^0)_{i,j,k}.$$

The expression of  $\Sigma_{ijk}^6$ . Let  $Q$  be a vector function of  $\vec{Q}_{ri}^1$ , the vector of a variable point. As usual,  $Q$  can be expressed as

$$Q = Q_i i + Q_j j + Q_k k.$$

$i, j, k$  are the components of  $Q$  in the directions of the axes. The discrete form of  $\nabla \cdot \vec{Q}_{ri}^1$  is

$$\begin{aligned} \Sigma_{ijk}^7 &= \nabla \cdot \vec{Q}_{ri}^1 \\ &= \frac{(Q_i)_{i+1,j,k} - (Q_i)_{i-1,j,k}}{2h} + \frac{(Q_j)_{i,j+1,k} - (Q_j)_{i,j-1,k}}{2h} \\ &+ \frac{(Q_k)_{i,j,k+1} - (Q_k)_{i,j,k-1}}{2h}. \end{aligned} \quad (44)$$

Similarly, let  $\partial_{\pi} Q$  be a vector function of  $\partial_{\pi} \vec{Q}_{ri}^1$ , the discrete form of  $\partial_{\pi} \vec{Q}_{ri}^1$  is

$$\begin{aligned} \Sigma_{ijk}^8 &= \nabla \cdot \partial_{\pi} \vec{Q}_{ri}^1 \\ &= \frac{(\partial_{\pi} Q_i)_{i+1,j,k} - (\partial_{\pi} Q_i)_{i-1,j,k}}{2h} + \frac{(\partial_{\pi} Q_j)_{i,j+1,k} - (\partial_{\pi} Q_j)_{i,j-1,k}}{2h} \\ &+ \frac{(\partial_{\pi} Q_k)_{i,j,k+1} - (\partial_{\pi} Q_k)_{i,j,k-1}}{2h}. \end{aligned} \quad (45)$$

The discrete form of  $\partial_{\pi} S$  is

$$\begin{aligned} \Sigma_{ijk}^6 &= \partial_{\pi} S \\ &= (\partial_{\pi} \kappa_s)_{ijk} (Q^0)_{ijk} + (\kappa_s)_{ijk} (\partial_{\pi} Q_{ri}^0)_{ijk} \\ &- \frac{\sigma_s}{\sigma_{tr}} \Sigma_{ijk}^8 - \frac{\partial_{\pi} \sigma_s \sigma_{tr} - \sigma_s \partial_{\pi} \sigma_{tr}}{\sigma_{tr}^2} \Sigma_{ijk}^7. \end{aligned}$$

The expression of  $\Sigma_{ijk}^9$ :

$$\begin{aligned} \Sigma_{ijk}^9 &= \partial_{\pi} \kappa_{ijk} \cdot \left( \sigma_s \rho_{ijk} (\vec{Q}_{ri}^1)_{ijk} - (\nabla L_d^0)_{ijk} \right) \\ &+ \kappa_{ijk} \cdot \left( -(\nabla \partial_{\pi} L_d^0)_{ijk} + \partial_{\pi} \sigma_s \rho_{ijk} (\vec{Q}_{ri}^1)_{ijk} + \sigma_s \rho_{ijk} (\partial_{\pi} \vec{Q}_{ri}^1)_{ijk} \right) \\ &+ \sigma_s (\vec{Q}_{ri}^1)_{ijk} \left( \partial_{\pi} i^{\frac{\rho_{i+1,j,k} - \rho_{i-1,j,k}}{2h}} + \partial_{\pi} j^{\frac{\rho_{i,j+1,k} - \rho_{i,j-1,k}}{2h}} + \partial_{\pi} k^{\frac{\rho_{i,j,k+1} - \rho_{i,j,k-1}}{2h}} \right). \end{aligned}$$