A High-Efficiency Physics-Based Differentiable Renderer for Smoke Reconstruction

Yunchi Cen*
Hangzhou Polytechnic University
Hangzhou

Hanchen Deng*
Beihang University
Beijing

Qifan Zhang
Beihang University
Beijing

Frederick W. B. Li
University of Durham
Durham

Bailin Yang
Zhejiang Gongshang University
Hangzhou

Xiaohui Liang†
Beihang University
Beijing

Abstract

Reconstructing volumetric smoke from 2D images remains challenging in computer graphics. Although physics-based differentiable rendering offers a promising solution, existing methods are computationally expensive and impractical for interactive use. We introduce an efficient physics-based differentiable renderer tailored for smoke reconstruction, based on two insights: (1) for the optically thin-to-moderately dense smoke targeted in this work, appearance is well approximated by single scattering, allowing us to avoid costly multiple-scattering simulation; and (2) direction-independent terms can be precomputed to avoid redundant forward and backward passes. Our technical contributions include a pre-computation strategy that eliminates repetitive calculations, and an efficient ray marching method that computes density derivatives without costly Monte Carlo estimation or automatic differentiation. Comprehensive evaluation shows that our approach achieves real-time performance while maintaining high reconstruction quality. In contrast to state-of-the-art methods that require minutes per frame, our method enables interactive smoke reconstruction while maintaining competitive visual fidelity. We demonstrate its utility in applications such as augmented reality and fluid motion reconstruction.

Keywords: ***differentiable rendering, volumetric optimization, smoke reconstruction.***

1. Introduction

Smoke reconstruction from 2D observations is crucial for computer graphics applications, yet existing physics-based methods are often too slow for interactive use, with reconstruction times reaching tens of minutes per frame [4, 5]. Although recent advancements in differentiable rendering have improved the temporal coherence of reconstructions, they face significant performance bottlenecks.

State-of-the-art approaches, such as that of Franz et al. [7], rely on differentiable volumetric path tracing to optimize smoke density. This process is computationally expensive, primarily due to the Monte Carlo methods used to estimate radiance and its derivatives. Consequently, reconstructions can take tens of hours, with density optimization accounting for the majority of this time, rendering these methods impractical for real-time applications. Many physics-based differentiable rendering (PBDR) frameworks, as discussed in recent studies [22, 23, 26], rely on the path replay backpropagation approach to improve their performance. Despite reported improvements, with iteration times ranging between 200–400 milliseconds, these frameworks still fall short of meeting the demands of interactive applications.

To address this efficiency challenge, we propose a novel physics-based lightweight differentiable renderer based on the single scattering model. This assumption significantly reduces the computational complexity of light transport while maintaining visual fidelity for a wide range of common smoke phenomena. Our approach reduces per-iteration computation time from hundreds of milliseconds to approximately 5.6 milliseconds, achieving real-time performance for the first time in physics-based smoke reconstruction. To further optimize this process, we derive the differential form of the Radiative Transfer Equation (RTE) for single scatter-

* denotes that these authors contributed equally to this work.
† Corresponding author, E-mail: liang_xiaohui@buaa.edu.cn.

ing and introduce a pre-computation framework to enable the efficient calculation of smoke density derivatives.

We validate our method by demonstrating its superior efficiency compared to the state-of-the-art and show that it seamlessly integrates into existing smoke motion reconstruction frameworks [4]. Our main contributions are:

- A physics-based lightweight differentiable renderer that bridges the gap between the efficiency of rasterization-based methods and the quality of volumetric path tracing.
- The derivation of the differential Radiative Transfer Equation for single scattering, coupled with a pre-computation framework that facilitates efficient and accurate derivative computation for the density field.

2. Related Work

2.1. Fluid Reconstruction

The reconstruction of fluid from 2D observations is a well-studied problem. Early approaches often employed computed tomography (CT) [16], which is divided into direct and iterative methods. Direct methods, such as analytical inversion via Fourier transforms, require dense multi-view inputs that are often impractical for fluid scenes. In contrast, iterative methods frame the reconstruction as an energy minimization problem solved through optimization.

Ihrke and Magnor [13] reconstructed volumetric fire by minimizing a least-squares energy with the conjugate gradient method, later improving efficiency for thin smoke and flames via an adaptive grid [14]. Atcheson *et al.* [1] captured 3D gas flows using time-resolved schlieren tomography. To handle sparse inputs, Gregson *et al.* [9] introduced a stochastic reconstruction approach with regularizers for 8–16 videos, while Okabe *et al.* [25] used only 1–2 videos and augmented tomography with appearance transfer.

Recent approaches, such as Neural Radiance Fields (NeRF) [20] and its variants, have emerged as competitive alternatives. InstantNGP [21] accelerated training via multi-resolution hash encoding, and TensoRF [3] achieved compactness and speed through tensor decomposition. Although these methods reduce the need for dense views, they often struggle with physical accuracy and temporal consistency.

Hybrid neural-physics methods have emerged to enhance reconstruction quality and temporal coherence. PhysGaussian [28] combines physical simulation with 3D Gaussians, while ScalarFlow [5] integrates deep learning with physical priors for fluids. Foundational works jointly estimate density and velocity: Eckert et al. [4] couple density updates with flow motion from single views, and Franz et al. [7] connect physical priors to observations via a differentiable renderer.

However, balancing physical accuracy with computational efficiency remains challenging. Current PBDR methods [24] suffer from costly automatic differentiation and high memory usage. Recent advances [23, 26] improve efficiency but still require 200-400 milliseconds per iteration, falling short of real-time performance. Our work bridges this gap through a lightweight, physics-based design optimized for interactive smoke reconstruction.

2.2. Differentiable Rendering

Differentiable rendering enables the computation and propagation of derivatives of scene parameters through images, finding applications in analysis-by-synthesis problems and machine learning pipelines. While early works like OpenDR [19] and Neural Renderer [17] focused on mesh-based rendering, our work focuses on participating media, where light transport is governed by the Radiance Transfer Equation (RTE) [2, 12].

Computing derivatives of participating media, such as smoke, poses significant challenges due to the complex nature of light transport within volumetric materials. The RTE that governs this transport involves multiple integral terms accounting for emission, absorption, and scattering events. Obtaining derivatives requires carefully tracking how changes in media properties affect each of these terms and their interactions. While frameworks like those proposed by Gkioulekas *et al.* [8] and Zhang *et al.* [29, 30] have provided theoretical foundations for computing these derivatives through Monte Carlo estimation, the high-dimensional integration spaces and complex light paths involved make the computation extremely resource-intensive.

Recent research has focused on improving the efficiency of volumetric differentiable rendering through various innovative approaches. Nimier-David *et al.* [23] introduced radiative backpropagation (RB), which interprets the backward computation as the propagation of gradients from the camera to scene objects along light paths. This method eliminates the need to store a large transcript of intermediate states, thus reducing memory overhead. Vicini *et al.* [26] made a significant advancement with path replay backpropagation (PRB), which achieves linear computational complexity by cleverly reusing paths from forward rendering and leveraging the invertibility of local Jacobians. Jakob *et al.* [15] proposed *Dr.Jit*, a just-in-time compiler for differentiable rendering that compiles all program instructions into a megakernel to reduce memory usage and inter-kernel communication costs.

However, these framework still face fundamental limitations: (1) the implementation of existing differential rendering frameworks is complex, requiring sophisticated system architectures and careful optimization strategies, (2) their comprehensive path tracing approach, while physically accurate, incurs significant computational overhead

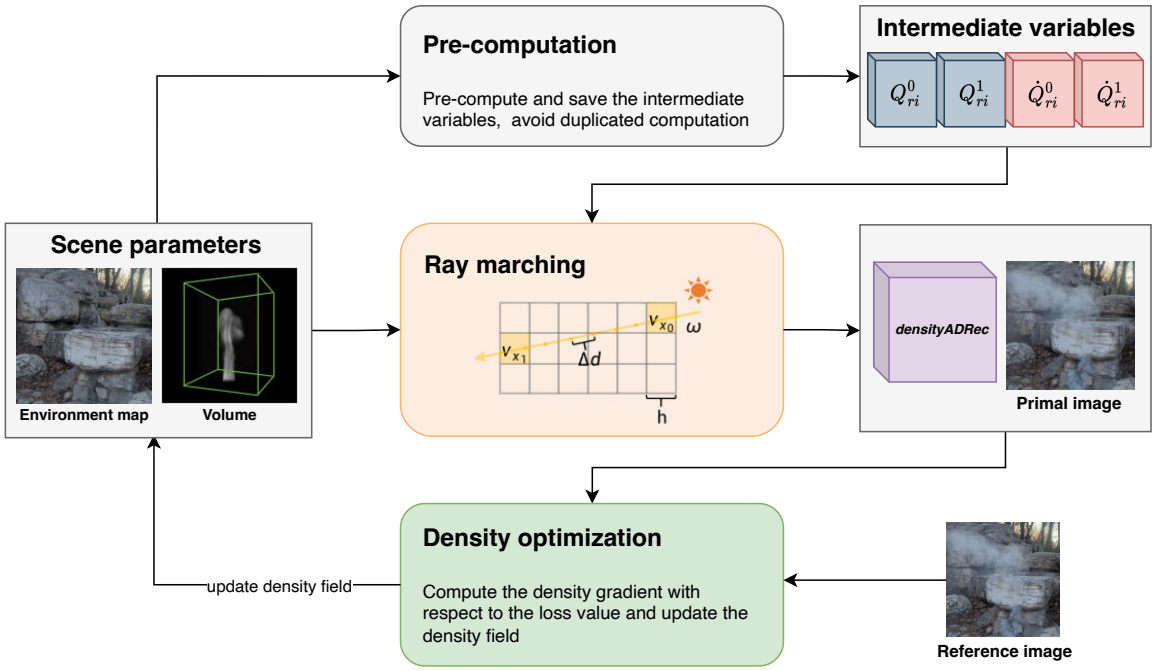


Figure 1. Our method proceeds in three stages: (1) Pre-computation of per-voxel radiance and its gradients; (2) Differentiable ray marching that renders an image and records density gradients; and (3) Density optimization, where a photometric loss is backpropagated through the recorded gradients to update the scene.

from tracking multiple scattering events, and (3) the need to sample and process multiple light paths for each pixel adds substantial computational cost. Our approach takes a different direction by adopting a lightweight, single-scattering model combined with efficient pre-computation strategies. While this simplification introduces some approximation in light transport simulation, it enables us to focus on the dominant light transport effects while eliminating redundant calculations. This design choice makes our method particularly well-suited for interactive applications and efficient fluid reconstruction tasks where speed is crucial.

3. Method

Our framework for smoke reconstruction, illustrated in Figure 1, consists of three key components designed to balance computational efficiency with physical accuracy:

- **Pre-computation Component**: Identifies and caches direction-independent variables to eliminate redundant computations in both forward and backward passes.
- **Ray Marching Component**: Renders the primal image and computes density gradients with respect to pixel values using our derived differential form of the radiative transfer equation.
- **Density Optimization Component**: Updates the density field ρ by minimizing the difference between rendered and reference images.

The proposed approach makes two key assumptions to achieve efficient reconstruction: **(1) Single Scattering Dominance:** We model only single scattering events, as they dominate light transport in smoke-like media with relatively low density. Multiple scattering and emission are ignored. **(2) Pure Volume Scenes:** We focus on reconstructing smoke volumes from images containing only smoke-like media, without considering scenes that contain surface-media interactions or other complex geometries.

Our framework processes smoke reconstruction through several sequential stages, which we detail in the following sections:

- Section 3.1 presents our **Pre-computation Component**'s theoretical foundation—an efficient single scattering model that eliminates Monte Carlo integration by analytically solving the radiative transfer equation.
- Section 3.2 develops the mathematical basis of our **Ray Marching Component** through a novel differential formulation, enabling direct gradient computation during ray marching without automatic differentiation.
- Section 3.3 details the implementation of both the **Pre-computation Component** and **Ray Marching Component**, showing how direction-independent terms are cached and utilized for accelerated rendering and gradient computation.

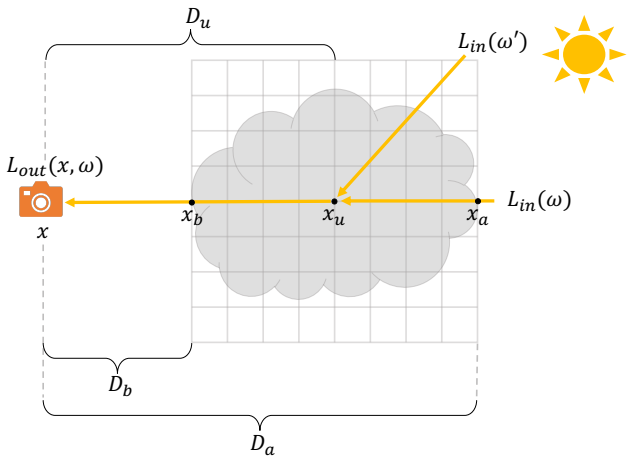


Figure 2. Light transport in a smoke volume. Ambient light $L_{in}(\omega)$ travels along direction ω, passing through position x_u inside the volume and intersects with the volume boundary at positions x_a and x_b. Under our single scattering assumption, the total radiance at viewpoint x ($L_{out}(x,\omega)$) consists of direct transmission and single scattering contributions. D_a, D_b, D_u represent the distances between viewpoint x and positions x_a, x_b, x_u, respectively.

- Section 3.4 presents our **Density Optimization Component**, demonstrating how the pre-computed terms and efficient gradients work together to achieve real-time volumetric reconstruction.

3.1. Single Scattering Radiance Transport

Our framework builds upon a simplified yet accurate model of light transport in participating media. For smoke-like media where the optical depth is relatively small, single scattering tends to dominate the radiance transport compared to multiple scattering events. This physical characteristic enables us to develop an efficient differentiable renderer for smoke reconstruction applications.

3.1.1 Optical Properties and Light Transport

We represent the smoke's density at any point x as $\rho(x)$. The smoke's optical properties are characterized by three coefficients: absorption κ_a, scattering κ_s, and extinction κ_t, where $\kappa_t(x) = \kappa_a(x) + \kappa_s(x)$. The scattering albedo is defined as $\Omega = \kappa_s(x)/\kappa_t(x)$, representing the probability of scattering versus absorption.

As shown in Figure 2, light transport in heterogeneous smoke volumes undergoes attenuation and scattering. Given viewpoint x and volume boundary points x_a and x_b at distances D_a and D_b, the transmittance $\tau(x_a, x_b)$ between any two points x_a and x_b is:

$$\tau(x_a, x_b) = e^{-\int_{D_b}^{D_a} \kappa_t(x_u)\mathrm{d}u}. \tag{1}$$

3.1.2 Radiative Transfer Equation Components

The radiance transport can be described by the Radiative Transfer Equation (RTE). Under our single scattering assumption, the outgoing radiance L_{out} at position x in direction ω can be decomposed into two components:

$$L_{out}(x,\omega) = L_{ri}(x,\omega) + L_{ss}(x,\omega), \tag{2}$$

where L_{ri} and L_{ss} are mathematically expressed as:

$$L_{ri}(x,\omega) = L_{in}(\omega)\tau(x_a, x), \tag{3}$$

$$L_{ss}(x,\omega) = \int_{D_b}^{D_a} \tau(x_u, x_b)\kappa_t(x_u)J_{ss}(x_u,\omega)\mathrm{d}u. \tag{4}$$

L_{ri} quantifies direct light transmission - ambient light L_{in} that reaches the viewpoint after attenuation through the medium. L_{ss} models accumulated single scattering contributions along the view ray, where each infinitesimal segment at position x_u contributes scattered radiance J_{ss} that is subsequently attenuated before reaching the sensor.

To compute the single scattering radiance L_{ss}, we first calculate the in-scattered radiance J_{ss} at each point x_u along the view ray. J_{ss} represents the radiance scattered from all incident directions into the view direction ω, and is given by:

$$J_{ss}(x_u,\omega) = \frac{\Omega}{4\pi}\int_{\mathbb{S}^2} L_{ri}(x_u,\omega')p(\omega,\omega')\mathrm{d}\omega', \tag{5}$$

where $\mathbb{S}^2$ represents the unit sphere of incident directions ω', and $p(\omega,\omega')$ is the phase function describing the angular distribution of scattered light. The phase function is normalized to 4π when integrated over all directions. The phase function can be expressed in terms of the cosine of the scattering angle $\mu = \omega \cdot \omega'$, writing it as $p(\mu)$.

3.1.3 Analytical Approximation

Traditional ray tracing-based frameworks for computing J_{ss} can be inefficient, particularly in the context of differentiable rendering where gradient computation requires tracing additional rays. Therefore, we propose an efficient analytical approximation that avoids repeated ray tracing by decomposing J_{ss} into direction-independent terms that can be pre-computed.

Starting from Equation 5, we approximate the phase function $p(\omega,\omega')$ using a Legendre polynomial expansion. This expansion is particularly suitable because phase functions in participating media are typically smooth and rotationally symmetric, making them ideal candidates for representation as a series of Legendre polynomials. The Legendre polynomials form a complete orthogonal basis for functions on $[-1,1]$, which matches our domain since the phase

function depends only on the cosine of the scattering angle $\mu = \omega \cdot \omega'$.

The phase function can be expressed in the following expanded form: $p(\omega, \omega') = \sum_{l=0}^{\infty} \frac{2l+1}{4\pi} \bar{\mu}_l P_l(\omega \cdot \omega')$, where P_l are Legendre polynomials and $\bar{\mu}_l$ are the expansion coefficients (moments). The zeroth moment $\bar{\mu}_0 = 1$ ensures normalization, while the first moment $\bar{\mu}_1 = \bar{\mu}$ characterizes the medium's scattering anisotropy through $\bar{\mu} = \frac{3}{2}\int_{-1}^{1} \mu p(\mu) d\mu$. For most common phase functions like Henyey-Greenstein, the higher-order moments ($l \geq 2$) have rapidly diminishing contributions. Therefore, we can truncate the expansion after the first moment:

$$p(\omega, \omega') \approx \frac{1}{4\pi}(1 + \bar{\mu}(\omega \cdot \omega')). \tag{6}$$

Similarly, we approximate the reduced incident radiance L_{ri} using a first-order Taylor expansion in direction ω':

$$L_{ri}(x_u, \omega') \approx L^0_{ri}(x_u) + L^1_{ri}(x_u) \cdot \omega'. \tag{7}$$

These linear approximations effectively capture both the average value and primary directional tendency of the phase function and reduced incident radiance while maintaining computational efficiency.

3.1.4 In-scattered Radiance Decomposition

Substituting these approximations (Equations 6 and 7) into Equation 5 yields:

$$\begin{aligned} J_{ss}(x_u, \omega) =& \frac{\Omega}{4\pi} \int_{\mathbb{S}^2} (1 + \bar{\mu}(\omega \cdot \omega'))(L^0_{ri}(x_u) + \\ & L^1_{ri}(x_u) \cdot \omega') d\omega' \\ =& \Omega(L^0_{ri}(x_u) + \frac{\bar{\mu}}{3} L^1_{ri}(x_u) \cdot \omega) \\ =& \Omega(Q^0_{ri}(x_u) + Q^1_{ri}(x_u) \cdot \omega). \end{aligned} \tag{8}$$

Here, Q^0_{ri} and Q^1_{ri} are direction-independent terms that capture the average radiance and principal directional component of single scattering, respectively. Specifically:

$$Q^0_{ri}(x_u) = L^0_{ri}(x_u), \tag{9}$$

$$Q^1_{ri}(x_u) = \frac{\bar{\mu}}{3} L^1_{ri}(x_u). \tag{10}$$

To compute L^0_{ri} and L^1_{ri}, we use importance sampling over the unit sphere $\mathbb{S}^2$ with N sample directions. For each sample direction ω'_i, we trace a ray to compute $L_{ri}(x_u, \omega'_i)$. Then:

$$L^0_{ri}(x_u) = \frac{1}{N} \sum_{i=1}^{N} L_{ri}(x_u, \omega'_i), \tag{11}$$

$$L^1_{ri}(x_u) = \frac{3}{N} \sum_{i=1}^{N} L_{ri}(x_u, \omega'_i)\omega'_i. \tag{12}$$

The factor of 3 in L^1_{ri} arises from the spherical integration of direction vectors. These calculations are executed once for each voxel in the smoke volume during preprocessing and subsequently stored. By pre-computing and storing Q^0_{ri} and Q^1_{ri} for each voxel, we can efficiently evaluate J_{ss} for any view direction without additional ray tracing during rendering.

Having introduced the necessary formulas for forward rendering, we will now derive the differential form of the RTE in the following section. This differential form can be integrated into the ray marching process to directly calculate the derivative of the RTE.

3.2. Differential Form of RTE

To compute density gradients efficiently, we derive the differential form of the Radiative Transfer Equation with respect to the density field ρ. The density field directly influences the optical properties through the relationship $\kappa_t(x) = \sigma_t \rho(x)$, where σ_t is the extinction cross-section. This linear relationship allows us to express derivatives with respect to density in terms of derivatives with respect to extinction.

The differential form of the RTE (Figure 2) can be expressed as:

$$\dot{L}_{out}(x, \omega) = \dot{L}_{ri}(x, \omega) + \dot{L}_{ss}(x, \omega), \tag{13}$$

where $\dot{L}_{out} = \frac{\partial L_{out}}{\partial \rho}$ represents the derivative with respect to density.

3.2.1 Derivative of Reduced Incident Radiance

The derivative of reduced incident radiance $\dot{L}_{ri}$ is obtained by differentiating Equation 3:

$$\dot{L}_{ri}(x, \omega) = L_{in}(\omega)\dot{\tau}(x_a, x), \tag{14}$$

where $\dot{\tau}$ is the derivative of transmittance. From Equation 1, we can derive:

$$\dot{\tau}(x_0, x_1) = -\tau(x_0, x_1)\sigma_t \int_{x_0}^{x_1} \mathrm{d}x. \tag{15}$$

During ray marching, we discretize this integral using a fixed step size Δd:

$$\dot{\tau}(v_{x_0}, v_{x_1}) = -\tau(v_{x_0}, v_{x_1})\sigma_t(x_1 - x_0), \tag{16}$$

where $(x_1 - x_0)$ represents the total distance between points x_0 and x_1.

3.2.2 Derivative of Single Scattering Radiance

The derivative of single scattering radiance $\dot{L}_{ss}$ requires differentiating both the transmittance and the in-scattered ra-

diance terms from Equation 4:

$$\dot{L}_{ss}(x,\omega) = \int_{D_b}^{D_a} [\dot{\tau}(x_u,x_b)\kappa_t(x_u) + \tau(x_u,x_b)\sigma_t]J_{ss}(x_u,\omega)\mathrm{d}u + \int_{D_b}^{D_a} \tau(x_u,x_b)\kappa_t(x_u)\dot{J}_{ss}(x_u,\omega)\mathrm{d}u, \tag{17}$$

where we have used $\dot{\kappa}_t(x_u) = \sigma_t$ from our linear relationship between density and extinction.

The derivative of in-scattered radiance $\dot{J}_{ss}$ follows from Equation 8:

$$\dot{J}_{ss}(x_u,\omega) = \Omega(\dot{Q}^0_{ri}(x_u) + \dot{Q}^1_{ri}(x_u)\cdot\omega), \tag{18}$$

where $\dot{Q}^0_{ri}$ and $\dot{Q}^1_{ri}$ are pre-computed direction-independent terms:

$$\dot{Q}^0_{ri}(x_u) = \frac{1}{N}\sum_{i=1}^{N}\dot{L}_{ri}(x_u,\omega'_i), \tag{19}$$

$$\dot{Q}^1_{ri}(x_u) = \frac{3}{N}\sum_{i=1}^{N}\dot{L}_{ri}(x_u,\omega'_i)\omega'_i. \tag{20}$$

During implementation, we use trilinear interpolation to sample density values at arbitrary positions within the volume. The computed gradients are scattered back to the eight nearest voxels using the same interpolation weights to maintain consistency between forward and backward passes.

The complete derivative computation is integrated into our ray marching process (Algorithm 2), where we accumulate both primal values and derivatives simultaneously while stepping through the volume. This approach eliminates the need for separate backward passes and enables efficient gradient computation without storing intermediate results.

3.3. Pre-computation and Ray Marching

Our method utilizes ray marching to compute the final radiance L_{out} and its derivatives with respect to density ρ. To accelerate this process, particularly for multi-view scenarios, we introduce a pre-computation step that calculates and stores viewpoint-independent quantities.

3.3.1 Pre-computation

The pre-computation component, outlined in Algorithm 1, calculates and stores direction-independent terms Q^0_{ri}, Q^1_{ri} and their derivatives $\dot{Q}^0_{ri}$, $\dot{Q}^1_{ri}$. These pre-computed quantities are later used to efficiently evaluate the in-scattered radiance J_{ss} and its derivative $\dot{J}_{ss}$ during ray marching. The inputs to the pre-computation are: the density field ρ, the extinction cross-section σ_t (which defines the extinction coefficient field via $\kappa_t(x) = \sigma_t\rho(x)$), and the incident ambient light distribution L_{in}. By pre-computing and storing these direction-independent terms, we avoid redundant calculations during the forward and inverse rendering phase.

The algorithm iterates through each voxel v_p in the grid. For each voxel, it computes the direction-independent terms $Q^0_{ri}(v_p)$ and $Q^1_{ri}(v_p)$ using Equations 9, 11 and 10, 12. This involves importance sampling N directions ω'_i over the unit sphere and, for each sample, tracing a ray from the volume boundary to v_p to compute the reduced incident radiance $L_{ri}(v_p,\omega'_i)$ using Equation 3, which accounts for attenuation via τ. Similarly, the corresponding derivative terms $\dot{Q}^0_{ri}(v_p)$ and $\dot{Q}^1_{ri}(v_p)$ are computed using Equations 19 and 20, which rely on $\dot{L}_{ri}(v_p,\omega'_i)$ calculated via Equation 14.

These pre-computed voxel grids $Q^0_{ri}, \dot{Q}^0_{ri}, Q^1_{ri}, \dot{Q}^1_{ri}$ store the average and first moment approximations of the incident radiance and its density derivative at each voxel. Since they do not depend on the camera's view direction ω, they only need to be computed once per density field update, significantly reducing redundant calculations during the subsequent ray marching stage, especially when reconstructing from multiple viewpoints.

Algorithm 1: Pre-computation

Input : Density field ρ, extinction cross-section σ_t, ambient light L_{in}
Output: Pre-computed terms $Q^0_{ri}, \dot{Q}^0_{ri}, Q^1_{ri}, \dot{Q}^1_{ri}$

Function `PreComputation`:
- $\kappa_t(x) \leftarrow \sigma_t\rho(x)$ for all x
- **parfor** *each voxel position* v_p **do**
  - `/* Sample N directions over unit sphere */`
  - Sample ω'_i for $i = 1$ to N
  - `/* Compute reduced incident radiance for each direction */`
  - **for** $i = 1$ *to* N **do**
    - $L_{ri}(v_p,\omega'_i) \leftarrow L_{in}\tau(v_p,\omega'_i)$
  - **end**
  - `/* Compute direction-independent terms */`
  - $Q^0_{ri}(v_p) \leftarrow \frac{1}{N}\sum_{i=1}^{N} L_{ri}(v_p,\omega'_i)$
  - $Q^1_{ri}(v_p) \leftarrow \frac{3}{N}\sum_{i=1}^{N} L_{ri}(v_p,\omega'_i)\omega'_i$
  - `/* Compute derivative terms */`
  - $\dot{Q}^0_{ri}(v_p) \leftarrow \frac{1}{N}\sum_{i=1}^{N} \dot{L}_{ri}(v_p,\omega'_i)$
  - $\dot{Q}^1_{ri}(v_p) \leftarrow \frac{3}{N}\sum_{i=1}^{N} \dot{L}_{ri}(v_p,\omega'_i)\omega'_i$
- **end**

3.3.2 Ray Marching for Rendering and Gradient Accumulation

The ray marching component, detailed in Algorithm 2, computes the final outgoing radiance L_{out} for each pixel

and simultaneously accumulates the necessary information to compute the gradient $\frac{\partial L_{out}}{\partial \rho}$. Inputs include the density field ρ, extinction coefficient κ_t, ambient light L_{in}, and the pre-computed terms $Q^0_{ri}, \dot{Q}^0_{ri}, Q^1_{ri}, \dot{Q}^1_{ri}$ from Algorithm 1.

For each pixel, a camera ray is generated. The algorithm marches along the ray segment intersecting the volume, from the back intersection point x_b to the front x_a, using a fixed step size Δd. At each sample point x_u along the ray:

1. **Fetch Pre-computed Data:** The density value $\rho(x_u)$ and extinction coefficient $\kappa_t(x_u) = \sigma_t \rho(x_u)$ are sampled using trilinear interpolation from the density grid. Similarly, the pre-computed values $Q^0_{ri}(x_u)$, $Q^1_{ri}(x_u)$, $\dot{Q}^0_{ri}(x_u)$, and $\dot{Q}^1_{ri}(x_u)$ are obtained from their corresponding voxel grids using trilinear interpolation. The corresponding voxel position v_{x_u} (e.g., the integer coordinates of the voxel containing x_u) is also identified.

2. **Compute Local Scattering and Derivatives:** The in-scattered radiance $J_{ss}(x_u, \omega)$ and its derivative $\dot{J}_{ss}(x_u, \omega)$ are computed using the interpolated pre-computed terms and Equations 8 and 18.

3. **Accumulate Radiance:** The contribution of $J_{ss}(x_u, \omega)$ to the total single scattering radiance L_{ss} is accumulated according to the discretized version of Equation 4, considering the transmittance $\tau(x_u, x_b)$ back to the starting point.

4. **Compute and Store Gradient Components:** The local contribution to the total derivative $\dot{L}_{out}$ at x_u is computed. This involves terms from the derivative of single scattering $\dot{L}_{ss}$ (Equation 17, considering $\dot{\tau}$, σ_t, and $\dot{J}_{ss}$) and potentially the derivative of direct transmission $\dot{L}_{ri}$ if needed for specific gradient formulations (though $\dot{L}_{ri}$ for the whole ray is computed separately after the loop based on $\tau(x_a, x_b)$). This computed local gradient contribution, $\dot{L}_{out}(x_u, \omega)$, is stored along with the associated voxel position v_{x_u} in the 'densityADRec' structure for the current pixel.

5. **Update Position:** The ray position is advanced by Δd.

After marching through the volume, the final direct transmission term $L_{ri}(x, \omega)$ is computed using Equation 3, and the total outgoing radiance $L_{out}(x, \omega) = L_{ri}(x, \omega) + L_{ss}(x, \omega)$ is obtained.

The algorithm outputs the rendered image L_{out} and the $densityADRec$ structure. This structure contains, for each pixel, a list of voxel positions v_{x_u} visited along the ray and their corresponding gradient contributions $\dot{L}_{out}(x_u, \omega)$. This allows the density optimization step (Section 3.4) to efficiently compute the total gradient $\frac{\partial \text{Loss}}{\partial \rho}$ by combining these stored contributions with the image-space loss gradient, without needing a separate backward pass or recomputing values. Note that storing $densityADRec$ increases memory usage compared to methods that recompute gradients on-the-fly, as discussed further in Section 3.5. The gradient computed at x_u is associated with v_{x_u}; during density updates (Algorithm 3), this gradient is implicitly scattered to the 8 neighbors of v_{x_u} when trilinear updates are employed, mirroring the forward interpolation.

Algorithm 2: Ray Marching for Rendering and Gradient Accumulation

Input : $\rho, \kappa_t, L_{in}, Q^0_{ri}, \dot{Q}^0_{ri}, Q^1_{ri}, \dot{Q}^1_{ri}$
Output: Rendered image L_{out}, Gradient records $densityADRec$

Function `RayMarching`:
  foreach *pixel position* (i, j) **do**
    $ray \leftarrow GenerateRay(i, j)$
    $\omega \leftarrow -ray.direction$
    $x_a, x_b \leftarrow Hit(ray)$
    $x_u \leftarrow x_b$
    $D_{max} = \|x_b - x_a\|$
    $L_{ss}, \dot{L}_{ss} \leftarrow 0$
    repeat
      `/* Compute single scattering and its derivative */`
      $J_{ss} \leftarrow \Omega(Q^0_{ri} + Q^1_{ri} \cdot \omega)$
      $L_{ss} \leftarrow L_{ss} + \tau \kappa_t J_{ss}$
      $\dot{J}_{ss} \leftarrow \Omega(\dot{Q}^0_{ri} + \dot{Q}^1_{ri} \cdot \omega)$
      $\dot{L}_{ri} \leftarrow -L_{in} \tau \dot{\kappa}_t$
      $\dot{L}_{ss} \leftarrow \tau \kappa_t \dot{J}_{ss} \Delta d + (\dot{\tau} \kappa_t + \tau \dot{\kappa}_t) J_{ss} \Delta d$
      $\dot{L}_{out} \leftarrow \dot{L}_{ri} + \dot{L}_{ss}$
      `/* Record gradient */`
      $densityADRec[i, j].position.push\_back(x_u)$
      $densityADRec[i, j].grad.push\_back(\dot{L}_{out})$
      $x_u \leftarrow x_u + \omega \cdot \Delta d$
      $D_u = \|x_b - x_u\|$
    until $D_u > D_{max}$
    $L_{ri} \leftarrow$ Equation 3
    $L_{out} \leftarrow L_{ri} + L_{ss}$
  end

3.4. Density Reconstruction

Building upon our differentiable rendering framework, we introduce a density field reconstruction methodology outlined in Algorithm 3. Our method optimizes the density field ρ by exploiting the linear relationship $\kappa_t(x) = \sigma_t \rho(x)$, where σ_t is the extinction cross-section. This relationship

allows us to compute derivatives with respect to ρ directly from the extinction coefficient derivatives obtained during ray marching.

The reconstruction process takes as input the target image L_{tag}, current density field ρ, and the gradient records $densityADRec$ from Algorithm 2. We employ mean squared error (MSE) to measure the discrepancy between rendered and target images:

$$\text{Loss} = \frac{1}{N} \sum_{i,j} ||L_{out}[i,j] - L_{tag}[i,j]||^2 \tag{21}$$

For gradient accumulation, we iterate through each pixel's recorded gradient contributions. At each sampling position x_u along a ray, the stored gradient $\dot{L}_{out}(x_u, \omega)$ represents the local contribution to the total derivative. Since x_u typically falls between voxel centers, we use trilinear interpolation during forward rendering to sample density values . Correspondingly, during gradient scattering, we distribute the gradient $\dot{L}_{out}(x_u, \omega)$ to the eight neighboring voxels using the same interpolation weights, ensuring consistency between forward and backward passes.

The final density gradient with respect to the loss is computed using the chain rule:

$$\begin{aligned}\frac{\partial \text{Loss}}{\partial \rho[v_p]} = &\sum_{(i,j)} \sum_{k} 2(L_{out}[i,j] - L_{tag}[i,j]) \\ &\cdot \dot{L}_{out}(x_{u,k}, \omega) \cdot w_{v_p}(x_{u,k})\end{aligned} \tag{22}$$

where $w_{v_p}(x_{u,k})$ represents the trilinear interpolation weight of voxel v_p at position $x_{u,k}$, and the summation is over all pixels (i,j) and sampling positions k along each ray.

The density field is updated using gradient descent: $\rho[v_p] \leftarrow \rho[v_p] - \alpha \frac{\partial \text{Loss}}{\partial \rho[v_p]}$, where α is the learning rate. This approach directly optimizes the physical density field while maintaining the linear relationship with optical properties, enabling efficient and physically meaningful reconstruction.

3.5. Discussion

Our method achieves computational efficiency through three key design decisions: (1) single scattering approximation, (2) pre-computing direction-independent radiance components, and (3) deterministic ray marching instead of Monte Carlo sampling.

Limitations: The single scattering assumption limits applicability to scenarios where multiple scattering effects are negligible, such as smoke with relatively low optical density. For denser media like cumulus clouds or highly turbid liquids, our method may underestimate multiple scattering contributions. Our first-order Legendre polynomial approximation of the phase function works well for common cases but may require higher-order terms for highly anisotropic scattering.

Algorithm 3: Density Reconstruction

Input : Target image L_{tag}, rendered image L_{out}, density field ρ, gradient records $densityADRec$

Output: Updated density field ρ, $loss$ value

Function `Reconstruction`**:**
- $loss \leftarrow MSE(L_{tag}, L_{out})$
- Initialize $densityGrad$ to zero
- **foreach** *pixel position* (i,j) **do**
  - $diff \leftarrow 2(L_{out}[i,j] - L_{tag}[i,j])$
  - **for** $k = 0$ *to* $densityADRec[i,j].size()$ **do**
    - $x_u \leftarrow densityADRec[i,j].position[k]$
    - $grad \leftarrow densityADRec[i,j].grad[k]$
    - `/* Scatter gradient to 8 neighboring voxels using trilinear weights */`
    - ScatterTrilinearGradient(x_u, $diff \cdot grad$, $densityGrad$)
  - **end**
- **end**
- `/* Update density field using gradient descent */`
- **parfor** *each voxel position* v_p **do**
  - $\rho[v_p] \leftarrow \rho[v_p] - LR \cdot densityGrad[v_p]$
- **end**

Memory Trade-offs: Our pre-computation trades space for speed, scaling as $O(N^3)$. A 256^3 volume uses 2.73 GB (Table 3); while a 512^3 volume would require $\approx$ 22 GB, this remains within the capacity of high-end GPUs. For production-scale resolutions exceeding this, our method would require integration with sparse data structures (e.g., VDB) to decouple memory usage from grid resolution.

Performance Characteristics: Our method shows notable performance advantages over traditional DRT methods, particularly in multi-view scenarios where speedups range from 5.3× to 27.6×. Even in single-view reconstruction, we observe speedups between 1.2× and 6.3× (Table 4). These performance improvements primarily result from our analytical approximation of the phase function and precomputation of direction-independent terms, which significantly reduce the computational overhead compared to traditional ray tracing approaches.

Integration: Our renderer can serve as a drop-in replacement for traditional volumetric path tracing in existing reconstruction pipelines, as demonstrated with Eckert's framework [4], provided the forward rendering model is compatible with single scattering assumptions.

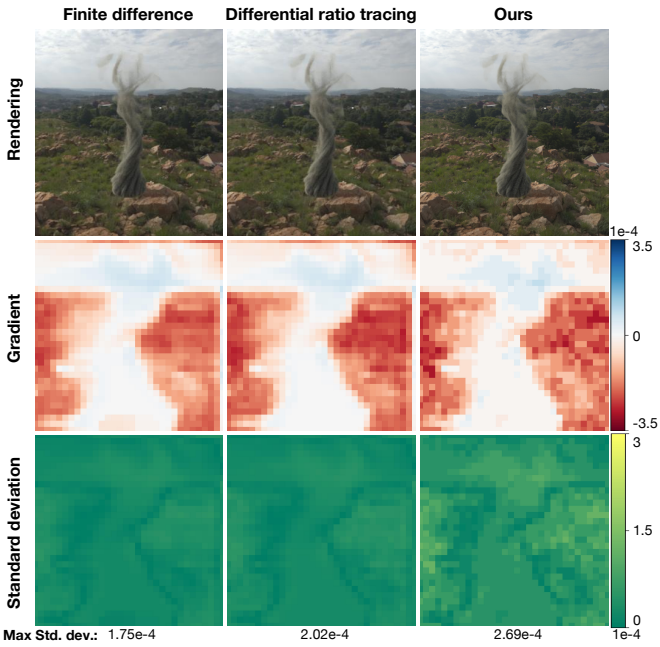


Figure 3. **A comparison and validation of our gradients.** We validate our mean gradient values (with respect to medium density) using slice $z = 128$ of the density 3D grid, comparing against the the FD method (left column) and the differential ratio tracing estimator [22] (middle column), both implemented in *Mitsuba 3*.

4. Evaluation and Results

We evaluate our proposed differentiable renderer on a workstation featuring an *Intel Xeon E5-2620 V4* processor and an *NVIDIA GeForce RTX 2080 Ti* graphics card. The renderer is implemented in Python and leverages *Taichi* [10, 11] for efficient parallel programming. All experiments in this section are conducted on GPUs. For live demonstrations of our approach, please refer to the supplementary video.

4.1. Evaluation

Our evaluation focuses on three key aspects: (1) the accuracy and reliability of our gradient computation method compared to established techniques, (2) the reconstruction quality and limitations of our approach in both single-view and multi-view scenarios, and (3) the computational efficiency and scalability of our method across different volume resolutions. We provide quantitative comparisons with state-of-the-art methods and analyze the trade-offs between accuracy and performance.

4.1.1 Validation of Gradient Computation

Our gradient computation is based on the differential form of the RTE derived in Section 3.2, where we compute derivatives with respect to the density field $\rho(x)$. The density directly influences optical properties through $\kappa_t(x) = \sigma_t \rho(x)$, enabling us to express $\frac{\partial L_{out}}{\partial \rho}$ analytically. This section validates both the accuracy and reliability of our gradient computation approach.

Technical Implementation Details: Our gradient computation involves three key components: (1) derivatives of transmittance $\dot{\tau}$ (Equation 16), (2) derivatives of precomputed radiance terms $\dot{Q}^0_{ri}$ and $\dot{Q}^1_{ri}$ (Equations 19–20), and (3) derivatives of in-scattered radiance $\dot{J}_{ss}$ (Equation 18). During ray marching, we use trilinear interpolation to sample density values at arbitrary positions, and correspondingly distribute the resulting gradients to the eight neighboring voxels using the same interpolation weights to ensure consistency between forward and backward passes.

Experimental Setup: We evaluate gradient accuracy using a heterogeneous smoke volume with resolution $256 \times 256 \times 256$ under high dynamic range environment map lighting. To ensure fair comparison, we configure *Mitsuba 3* to use single scattering only (maximum depth=1) since our method assumes single scattering dominance.

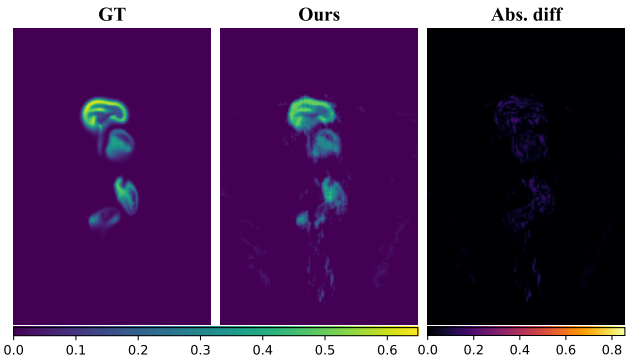
Baseline Comparisons: We compare against two established methods: (1) finite differences (FD) to establish ground truth reference gradients, and (2) *Mitsuba 3*'s differential ratio tracing (DRT) [22] with 2048 samples per pixel as a state-of-the-art unbiased estimator. Both baseline methods are configured for single scattering to ensure fair comparison with our approach.

Results and Analysis: Figure 3 shows gradient comparisons across methods. Visual inspection of the primal images reveals minimal discrepancies between *Mitsuba 3* and our method, indicating accurate forward rendering. The computed gradients show some noise compared to FD and DRT methods, attributed to our ray marching approach being a biased sampling technique. However, the gradient standard deviations remain within acceptable ranges: 0.000175 (FD), 0.000202 (DRT), and 0.000269 (ours), indicating that our gradients are reliable for optimization purposes.

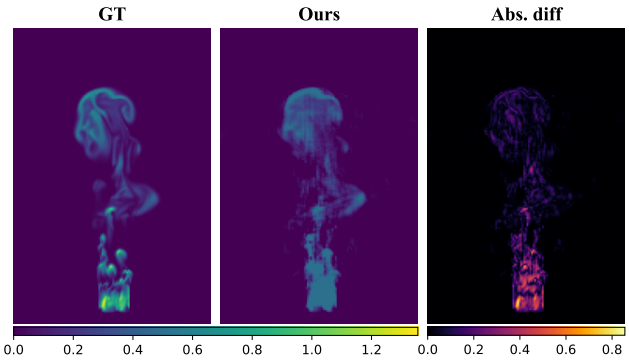
4.1.2 Validation of Reconstruction Accuracy

To validate our single-scattering assumption, we conduct an ablation study utilizing reference images rendered with high-order scattering (depth=64). We reconstruct these using our single-scattering renderer. This setup isolates the model mismatch error, allowing us to quantify the method's robustness when ignoring the multiple scattering effects present in the input data. The results are shown in Figure 4.

Experimental Setup: We generated reference images from 10 viewpoints uniformly distributed around the volume using directional lighting. The reconstruction was performed over 200 iterations using gradient descent with learning rate $\alpha = 0.01$. No additional regularization terms were applied to isolate the performance of our core differ-



(a) Density slices at $z = 84$



(b) Density slices at $z = 76$

Figure 4. **Evaluation of Reconstruction Accuracy.** Multi-view reconstruction results are compared against ground truth by showing density field slices at positions $z = 84, 76$, with error measured as absolute difference. Slices at the volume's outer positions **(a)** closely match the ground truth, whereas slices from the inner slices **(b)** exhibit larger deviations.

entiable rendering approach.

Spatial Accuracy Variation: Outer volume regions ($z = 84$) demonstrate higher reconstruction fidelity, while inner regions ($z = 76$) show increased deviation. This spatial bias occurs because image-based reconstruction inherently provides stronger constraints for volume regions that more directly influence pixel appearances.

Grid Artifacts: The reconstructed density exhibits subtle grid-aligned artifacts visible as regular patterns in certain regions. These artifacts arise from our discrete voxel representation combined with trilinear interpolation during ray marching. The gradient scattering process tends to preserve voxel-grid structure rather than promoting smooth transitions, particularly in regions with weak image constraints.

Flat Density Regions: Some areas display unnaturally flat density distributions rather than expected smooth variations. This occurs due to two factors: (1) our single scattering model provides insufficient gradient information in optically thin regions where multiple scattering would contribute meaningful detail, and (2) the under-constrained nature of the inverse problem allows multiple density configurations to produce similar rendered appearances.

Analysis of Reconstruction Quality: Figure 4 presents density slice comparisons at two representative depths: $z = 84$ (outer layer) and $z = 76$ (middle region). The results reveal several important characteristics of our method:

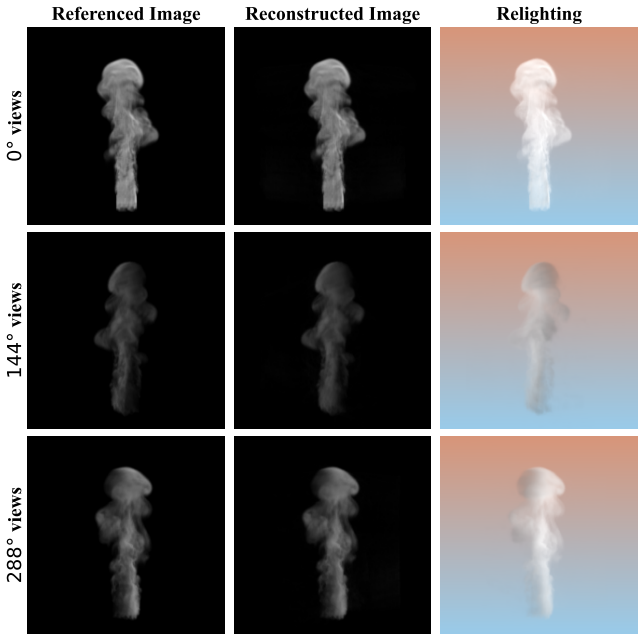


Figure 5. **Visual evaluation of reconstruction effects.** We present side-by-side comparisons of the reference images, reconstructed images, and relighting images from three different perspectives. Visually, our reconstructed images exhibit a high degree of similarity with the references. Furthermore, we relight the reconstructed density field under a variety of lighting conditions using a standard volumetric path tracer. This demonstrates that our reconstructed results are adaptable and can be practically applied across diverse scenarios.

Trade-off Analysis: Our method represents a deliberate trade-off between reconstruction accuracy and computational efficiency. While absolute accuracy decreases compared to full multiple scattering methods, the 50-100× speedup enables interactive reconstruction workflows. The method is most effective for: (1) moderately dense smoke, (2) scenarios prioritizing reconstruction speed over absolute accuracy, and (3) applications where approximate density distributions are sufficient for downstream processing.

To further substantiate the effectiveness of our method, we present both visual and quantitative comparisons of renderings produced from our reconstructed density field from diverse viewpoints. Figure 5 showcases side-by-side comparisons of reference images, reconstructed images, and relighting images, providing a comprehensive perspective on our method's performance. The reference images and the reconstructed images show a high degree of visual similar-

Table 1. **Quantitative Comparison of MSE and SSIM for different viewpoints.**

View Angle (Degree)	0°	36°	72°	108°	144°	180°	216°	252°	288°	324°
MSE	8.88	6.63	6.49	3.18	3.31	7.26	4.80	3.10	5.17	6.01
SSIM	0.83	0.87	0.90	0.92	0.95	0.95	0.95	0.93	0.90	0.87

ity. This is a testament to the accuracy of our reconstructed appearance of the smoke, which is capable of capturing intricate details. Relighting images are rendered using the standard volumetric path tracker and exhibit remarkable fidelity and consistency with the reference images. The relighting images also display a high degree of photorealism, which is a crucial factor in practical applications.

Quantitative measures also attest to the quality of reconstruction. The Mean Squared Error (MSE) and Structural Similarity Index (SSIM) between the reconstructed and reference images are computed for each view, with results detailed in Table 1. The table indicates that the MSE values range from 3.1 to 8.88. Given that pixel intensities in our images range from 0 to 255, this relatively low MSE underscores that our reconstruction method can consistently produce images that bear a close resemblance to the ground truth on average. SSIM is a perception-based metric that compares the structural similarity between two images, making it a more comprehensive and accurate measure of image quality compared to pixel-based metrics like MSE. For all views, the SSIM values are above 0.8, indicating a high degree of structural similarity between the reconstructed and reference images. In particular, the SSIM values for some views are close to 0.95, suggesting that our differentiable renderer is able to preserve the structural information of the density field very effectively.

4.1.3 Runtime Performance Analysis

We conduct comprehensive performance evaluations to analyze the computational efficiency of our differentiable renderer across multiple dimensions. Our experimental setup uses the heterogeneous smoke volume from Figure 3 (resolution $256 \times 256 \times 256$) under high dynamic range environment map lighting. The evaluation examines four key aspects: component-wise performance breakdown, optimal parameter selection, volume resolution scaling, and comparative performance against state-of-the-art methods.

Table 2. **Performance breakdown by directional sampling count**

Directional sampling count	Computational time (ms)		
	Primal	Inverse	Pre-computation
10	21	95	131
50	28	95	732
100	25	93	1,328
500	22	91	3,975

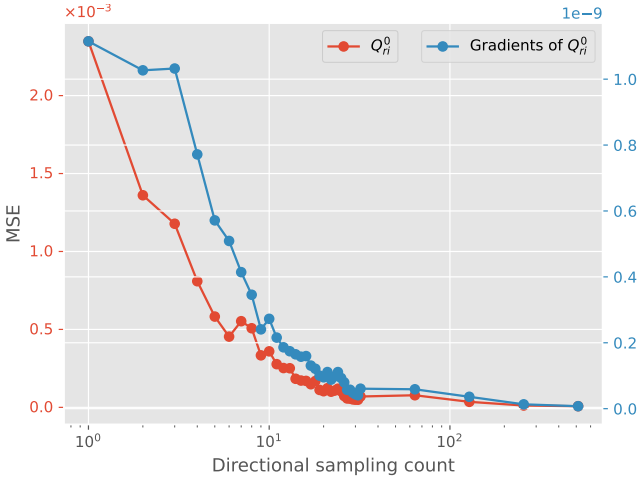


Figure 6. Findings showing the convergence of the number of directional sampling at approximately 30 samples, with a corresponding pre-computation time of 0.455 seconds.

Component Performance Breakdown: Table 2 shows the performance breakdown for each component. Key observations include: (1) Primal rendering remains consistently fast (21-28 ms) regardless of directional sampling count, (2) Inverse rendering time is stable (91-95 ms) due to our direct gradient computation approach, and (3) Pre-computation time scales linearly with directional samples, which is expected since we compute radiance for each sample direction independently.

Optimal Parameter Selection: To determine the optimal number of directional samples, we generated ground truth using 2048 samples and computed MSE between our approximated terms and ground truth. Figure 6 demonstrates convergence at approximately 30 directional samples, with pre-computation time of 455 ms. This provides an excellent balance between accuracy and computational efficiency.

Table 3. **Scalability analysis across volume resolutions**

Volume resolution	Primal (ms)	Inverse (ms)	Pre-comp (ms)	Memory (GB)
32^3	0.96	5.7	2.8	0.58
64^3	1.5	7.9	8.2	0.62
128^3	3.4	29	50	0.83
256^3	22.1	95	431	2.73

Volume Resolution Scaling: Table 3 shows performance scaling with volume resolution using 30 directional samples. The cubic scaling of pre-computation time is ex-

pected due to the $O(n^3)$ voxel count, while primal and inverse rendering scale more favorably. Memory usage remains reasonable even for high-resolution volumes due to our efficient data structures.

Table 4. **Performance comparison with Differential Ratio Tracing**

Samples per pixel	Single view speedup	Multi-view speedup
32	1.5×	6.4×
128	2.4×	10.5×
512	6.3×	27.6×

Baseline Method Comparison: We compared against Mitsuba 3's Differential Ratio Tracing (DRT) [22] using single scattering only for fair comparison. Table 4 shows speedup factors for different sampling densities. The comparison used volume resolution 128^3 with 30 directional samples for our method. Multi-view scenarios show dramatically higher speedups because our pre-computed terms are view-independent, requiring computation only once regardless of viewpoint count, while DRT must recompute for each view.

Performance Advantage Analysis: The increasing speedup with higher SPP occurs because: (1) our method's runtime is largely independent of sampling density after pre-computation, (2) DRT's Monte Carlo estimation becomes more expensive with higher sample counts, and (3) our deterministic ray marching avoids the variance inherent in stochastic sampling methods. For practical applications, we recommend 30 directional samples as the optimal balance between accuracy and performance.

4.2. Comparisons

We conduct comprehensive comparisons against representative methods from different paradigms to validate our approach's effectiveness. Our evaluation includes reconstruction methods (3D Gaussian Splatting), physics-based methods (Laplacian projection), and demonstrates our method's competitive performance across multiple evaluation criteria.

4.2.1 Baseline Methods and Justification

We compare against two fundamentally different approaches to volumetric reconstruction:

3D Gaussian Splatting (3DGS) [18]: A state-of-the-art reconstruction approach that represents scenes using 3D Gaussians. We include 3DGS as it represents the current frontier in scene reconstruction, providing a strong baseline for reconstruction quality despite not being specifically designed for volumetric media.

Laplacian Projection [27]: A physics-based tomographic reconstruction method that employs Laplacian regularization for sparse view reconstruction. This method represents the current state-of-the-art in physics-based smoke reconstruction and provides a direct comparison within the same domain as our method. We evaluate both single-view and multi-view variants to demonstrate the effectiveness of our approach across different input configurations.

4.2.2 Experimental Setup

Hardware and Dataset: All experiments are conducted on an NVIDIA GeForce RTX 2080 Ti GPU using the open-source ScalarFlow dataset [6], which provides multi-view smoke image sequences with a resolution of 300×531 pixels.

Configuration Parameters: To ensure fair comparison, all methods (3DGS, Laplacian projection, and ours) operate on the same volume resolution of $128 \times 196 \times 128$. Our method uses 30 directional samples for pre-computation, which was determined as optimal from the analysis in Section 4.1.3. We investigate three iteration limits (250, 1000, 5000) for our method to analyze the quality-efficiency trade-off. For the Laplacian projection method, we use 16 views in the multi-view reconstruction configuration, while the single-view variant uses only one input view.

Evaluation Protocol: Each method reconstructs smoke density from 3 temporal frames captured at different timestamps. Results are averaged over three independent trials to ensure statistical reliability. We evaluate reconstruction quality using Mean Squared Error (MSE) between rendered and reference images, and measure computational efficiency through per-frame reconstruction time.

4.2.3 Results and Analysis

The quantitative results presented in Figure 7 reveal distinct performance characteristics across all compared methods:

Quality Analysis: As shown in the reconstruction loss values (second text row of Figure 7), 3DGS achieves the highest reconstruction fidelity for complex scene reconstruction. Laplacian projection shows significant quality variation between single-view and multi-view configurations, with the single-view variant yielding the poorest results among all methods. Our method demonstrates competitive quality across all iteration settings, with higher iteration counts approaching 3DGS quality levels while maintaining substantially better computational efficiency.

Efficiency Analysis: The timing analysis (third text row of Figure 7) reveals our method's significant computational advantages. Our approach achieves substantial speedups compared to baseline methods. The 250-iteration configuration achieves the fastest reconstruction speed among all methods while maintaining reasonable quality. Even our highest-quality configuration maintains competitive efficiency compared to 3DGS.

Ground Truth	3DGS	Laplacian Projection		Ours		
		single view	multi views	max 250 iters	max 1k iters	max 5k iters
Average Loss	0.0001	29.1546	5.9165	2.5600	0.0712	0.0058
Average Time	217.32s	19.99s	34.55s	2.35s	12.59s	68.56s

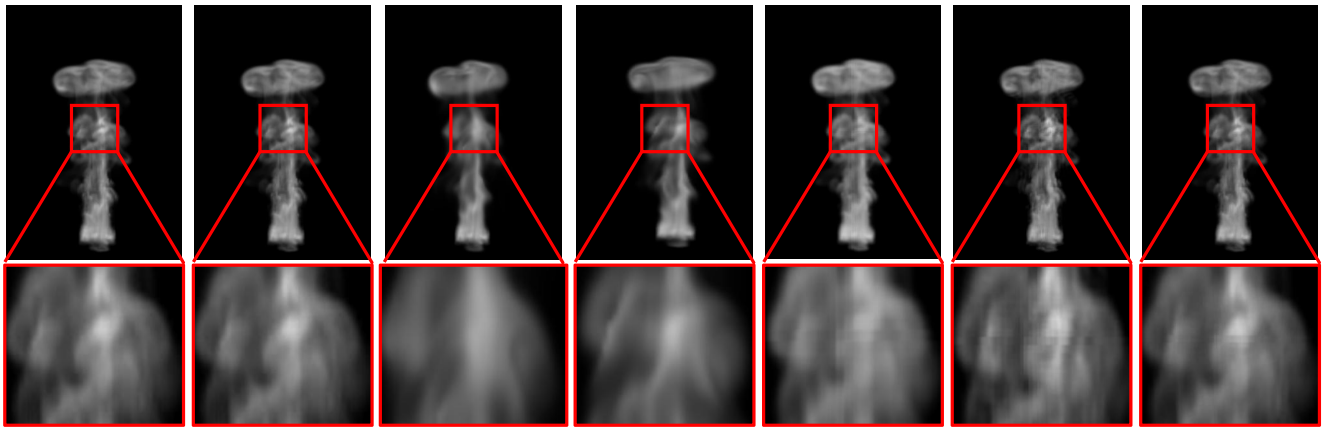


Figure 7. The top figure row presents reconstruction results from different methods based on the same input image, while the bottom figure row shows magnified views of detail-rich regions from the top figure row's reconstructions. The first text row identifies the methodology corresponding to each column. The second text row quantifies the average per-frame reconstruction loss, where 3DGS achieves optimal performance, Laplacian projection [27] single-view yields the poorest results, and our method attains reconstruction quality second only to 3DGS while substantially outperforming Laplacian's results. The third text row reports average per-frame reconstruction time, revealing that 3DGS requires the longest computation time, whereas our method with a maximum iteration count of 250 achieves the fastest reconstruction speed.

Quality-Efficiency Trade-off: Figure 7 illustrates this trade-off visually. The magnified detail regions (bottom row) demonstrate that our method preserves important structural features while significantly reducing computation time. Notably, our 250-iteration results already surpass Laplacian multi-view quality while being substantially faster, making it practical for applications requiring efficient reconstruction.

Scalability Advantages: Unlike 3DGS which optimizes numerous Gaussians, our physics-based approach directly optimizes the density field, achieving more predictable convergence behavior and better scalability across different volume resolutions.

4.3. Application

To showcase the versatility of our differentiable renderer, we integrated it into an existing smoke motion reconstruction framework [4]. In the practical implementation, we enhanced the algorithm by removing the *depth regularization* aspect, which was initially used to regulate smoke motion in depth. Our observations revealed that this regularization term undermined the sparse nature of the matrix, resulting in computationally inefficient optimization processes. We successfully reconstructed smoke motion from real-world data using the enhanced framework. The average reconstruction time per frame was dramatically reduced from 52 minutes to approximately 6 seconds, representing an efficiency enhancement of nearly two orders of magnitude. In Figure 8, we showcase an Augmented Reality (AR) application that utilizes the enhanced framework to transpose steam from a coffee video onto a teacup, highlighting the potential of our approach in generating engaging smoke-based visual effects. Further details can be found in our supplemental video.

Figure 8. Augmented Reality application demonstrating our method integrated with Eckert's framework [4]. We transfer steam from a monocular coffee video (top row) to a teacup (bottom row) using coupled density and velocity estimation.

5. Conclusion

We have presented a high-efficiency physics-based differentiable renderer specifically designed for smoke reconstruction from 2D observations. Our method successfully addresses the computational bottlenecks that have limited the practical applicability of existing differentiable rendering approaches in interactive applications.

Key Contributions and Achievements: Our approach makes two fundamental contributions that enable real-time performance. First, we derive and implement the differential form of the Radiative Transfer Equation for single scattering, allowing direct computation of density gradients without expensive Monte Carlo estimation or automatic differentiation. Second, we introduce a strategic pre-computation framework that eliminates redundant calculations by caching direction-independent radiance components, significantly accelerating both forward rendering and gradient computation.

Limitations and Future Directions: Our single scattering assumption limits applicability to scenarios where multiple scattering effects are negligible, such as smoke with relatively low optical density. For denser media like cumulus clouds, multiple scattering contributions may become significant. Future work could explore efficient approximation techniques for incorporating higher-order scattering effects while preserving computational advantages. Additionally, extending our pre-computation strategy to handle surface-media interactions would broaden the method's applicability to more complex scenes.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (No. 62272019).

References

[1] B. Atcheson, I. Ihrke, W. Heidrich, A. Tevs, D. Bradley, M. Magnor, and H.-P. Seidel. Time-resolved 3d capture of non-stationary gas flows. *ACM transactions on graphics (TOG)*, 27(5):1–9, 2008. 2

[2] S. Chandrasekhar. *Radiative transfer*. Courier Corporation, 2013. 2

[3] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, pages 333–350. Springer, 2022. 2

[4] M.-L. Eckert, W. Heidrich, and N. Thuerey. Coupled fluid density and motion from single views. In *Computer Graphics Forum*, volume 37, pages 47–58. Wiley Online Library, 2018. 1, 2, 8, 13

[5] M.-L. Eckert, K. Um, and N. Thuerey. Scalarflow: a large-scale volumetric data set of real-world scalar transport flows for computer animation and machine learning. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019. 1, 2

[6] M.-L. Eckert, K. Um, and N. Thuerey. Scalarflow: a large-scale volumetric data set of real-world scalar transport flows for computer animation and machine learning. *ACM Trans. Graph.*, 38(6), Nov. 2019. 12

[7] E. Franz, B. Solenthaler, and N. Thuerey. Global transport for fluid reconstruction with learned self-supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1632–1642, 2021. 1, 2

[8] I. Gkioulekas, A. Levin, and T. Zickler. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision*, pages 685–701. Springer, 2016. 2

[9] J. Gregson, M. Krimerman, M. B. Hullin, and W. Heidrich. Stochastic tomography and its applications in 3d imaging of mixing fluids. *ACM Transactions on Graphics (TOG)*, 31(4):1–10, 2012. 2

[10] Y. Hu, L. Anderson, T.-M. Li, Q. Sun, N. Carr, J. Ragan-Kelley, and F. Durand. Difftaichi: Differentiable programming for physical simulation. *ICLR*, 2020. 9

[11] Y. Hu, T.-M. Li, L. Anderson, J. Ragan-Kelley, and F. Durand. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)*, 38(6):201, 2019. 9

[12] H. C. Hulst and H. C. van de Hulst. *Light scattering by small particles*. Courier Corporation, 1981. 2

[13] I. Ihrke and M. Magnor. Image-based tomographic reconstruction of flames. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 365–373, 2004. 2

[14] I. Ihrke and M. Magnor. Adaptive grid optical tomography. *GRAPHICAL MODELS*, 68(5-6):484–495, SEP-NOV 2006. 2

[15] W. Jakob, S. Speierer, N. Roussel, and D. Vicini. Dr. jit: a just-in-time compiler for differentiable rendering. *ACM Transactions on Graphics (TOG)*, 41(4):1–19, 2022. 2

[16] A. C. Kak and M. Slaney. *Principles of computerized tomographic imaging*. SIAM, 2001. 2

[17] H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3907–3916, 2018. 2

[18] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4), July 2023. 12

[19] M. M. Loper and M. J. Black. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pages 154–169. Springer, 2014. 2

[20] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[21] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 2

[22] M. Nimier-David, T. Müller, A. Keller, and W. Jakob. Unbiased inverse volume rendering with differential trackers. *ACM Transactions on Graphics (TOG)*, 41(4):1–20, 2022. 1, 9, 12

[23] M. Nimier-David, S. Speierer, B. Ruiz, and W. Jakob. Radiative backpropagation: an adjoint method for lightning-

fast differentiable rendering. *ACM Transactions on Graphics (TOG)*, 39(4):146–1, 2020. 1, 2

[24] M. Nimier-David, D. Vicini, T. Zeltner, and W. Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)*, 38(6):1–17, 2019. 2

[25] M. Okabe, Y. Dobashi, K. Anjyo, and R. Onai. Fluid volume modeling from sparse multi-view images by appearance transfer. *ACM Transactions on Graphics (TOG)*, 34(4):1–10, 2015. 2

[26] D. Vicini, S. Speierer, and W. Jakob. Path replay backpropagation: differentiating light paths using constant memory and linear time. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 1, 2

[27] S. Xiao, C. Tong, Q. Zhang, Y. Cen, F. W. B. Li, and X. Liang. Laplacian projection based global physical prior smoke reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 30(12):7657–7671, 2024. 12, 13

[28] T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4389–4398, 2024. 2

[29] C. Zhang, L. Wu, C. Zheng, I. Gkioulekas, R. Ramamoorthi, and S. Zhao. A differential theory of radiative transfer. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019. 2

[30] C. Zhang, Z. Yu, and S. Zhao. Path-space differentiable rendering of participating media. *ACM Transactions on Graphics (TOG)*, 40(4):1–15, 2021. 2