GaussFluids: Reconstructing Lagrangian Fluid Particles from Videos via Gaussian Splatting

F. Du[1], Y. Zhang[†1,2], Y. Ji[1], X. Wang[1], C. Yao[1], J. Kosinka[3], S. Frey[3], A. Telea[4], X. Ban[‡1]

[1]School of Intelligence Science & Technology, University of Science & Technology Beijing, China
[2]Shunde Innovation School, University of Science and Technology Beijing, China
[3]Bernoulli Institute, University of Groningen, Netherlands
[4]Department of Information and Computing Sciences, Utrecht University, Netherlands

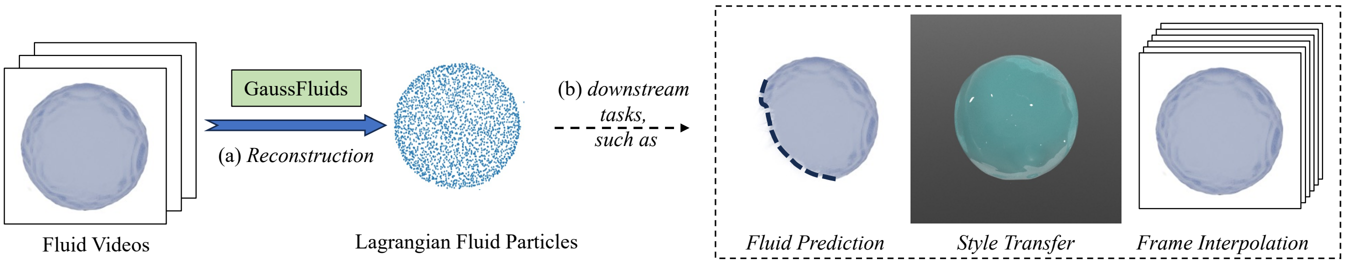

Figure 1: GaussFluids (a) reconstructs Lagrangian particle-based fluid flow from videos for (b) downstream tasks, such as novel view synthesis, style transfer, frame interpolation, fluid prediction, fluid editing, etc.

Abstract

Fluid simulation typically depends on manual modeling and visual assessment to achieve desired outcomes, which lacks objectivity and efficiency. To address this limitation, we propose GaussFluids, a novel approach for directly reconstructing temporally and spatially continuous Lagrangian fluid particles from videos. We employ a Lagrangian particle-based method instead of an Eulerian grid as it provides a direct spatial mass representation and is more suitable for capturing fine fluid details. First, to make discrete fluid particles differentiable over time and space, we extend Lagrangian particles with Gaussian probability densities, termed Gaussian Particles, constructing a differentiable fluid particle renderer that enables direct optimization of particle positions from visual data. Second, we introduce a fixed-length transform feature for each Gaussian Particle to encode pose changes over continuous time. Next, to preserve fundamental fluid physics—particularly incompressibility—we incorporate a density-based soft constraint to guide particle distribution within the fluid. Furthermore, we propose a hybrid loss function that focuses on maintaining visual, physical, and geometric consistency, along with an improved density optimization module to efficiently reconstruct spatiotemporally continuous fluids. We demonstrate the effectiveness of GaussFluids on multiple synthetic and real-world datasets, showing its capability to accurately reconstruct temporally and spatially continuous, physically plausible Lagrangian fluid particles from videos. Additionally, we introduce several downstream tasks, including novel view synthesis, style transfer, frame interpolation, fluid prediction, and fluid editing, which illustrate the practical value of GaussFluids.

CCS Concepts

• ***Computing methodologies*** → ***Computer graphics; Machine learning;*** *Modeling and simulation;*

1. Introduction

Fluid simulation is a significant research area with numerous applications in computer animation, medicine, and industrial engineering. Computer fluid simulation can be primarily categorized into Eulerian grid methods, Lagrangian particle methods, and hybrid methods, based on the underlying data structure of the fluid

[†] Corresponding author: zhangyl@ustb.edu.cn
[‡] Corresponding author: banxj@ustb.edu.cn

[KBST22]. Eulerian grid methods discretize the fluid domain into a grid, assigning physical properties such as velocity and pressure to each grid point. By solving the Navier-Stokes equations over time, these methods capture fluid behavior and are particularly suitable for large-scale simulations, such as oceans [KBST22]. In contrast, Lagrangian particle methods represent the fluid as a set of particles with a defined mass. By updating particle positions, these methods track changes in physical properties, making them well suited for simulating fluids with rich motion details, such as small-scale water splashes [KBST22].

However, achieving the desired fluid effects usually requires extensive manual modeling and visual comparison to fine-tune parameters, which demands considerable expertise and lacks objectivity and efficiency.

To address these issues, we introduce GaussFluids, a novel approach for directly extracting Lagrangian fluid particles from video data, as illustrated in Figure 1. We adopt the Lagrangian particle-based formulation because it offers a more precise spatial representation of fluid motion compared to the Eulerian grid approach. Since Lagrangian particles form a discrete point-cloud structure (i.e., discrete spheres), they do not by themselves define a spatially continuous differential field, which hinders backpropagation, and are therefore non-differentiable with respect to visual data [WQF19]. We therefore extend them with spatial Gaussian probability densities: adding Gaussians centered at the sphere centers creates the required differentiable field (see Sec. 2.3 of [KKLD23]); we refer to the resulting entities as Gaussian Particles. This extension provides each particle with a gradient receptive field in its neighborhood, thus enabling direct optimization of fluid particle positions (which often include transparency) from visual data.

Specifically, by integrating 3D Gaussian probability densities with Lagrangian fluid particles, we construct a differentiable renderer capable of mapping these Gaussian Particles to images, allowing direct optimization of particle positions from visual data. Furthermore, we introduce a fixed-length transform feature for each Gaussian Particle. Using this feature, we employ a unified temporal-spatial decoding network to determine particle pose variations at any given time. To maintain the fundamental incompressibility of fluids, we incorporate a density-based soft constraint that ensures a more uniform Gaussian Particle distribution, particularly within the fluid region. During training, we design a set of multi-dimensional loss functions, including visual-based, physics-based, and geometric-based losses, alongside an improved density optimization module. This combined strategy ensures alignment between macroscopic fluid flow and microscopic details. Extensive experiments on both synthetic and real-world datasets validate the effectiveness of GaussFluids in reconstructing Lagrangian fluid particles, facilitating subsequent tasks such as novel view synthesis, style transfer, frame interpolation, fluid prediction, fluid editing, and more.

The main contributions of this paper are as follows:

- We extend Lagrangian particles with 3D Gaussian probability densities and construct a differentiable renderer from particles to images based on splatting. This enables the direct optimization of Lagrangian particle positions from images or videos.
- We introduce a fixed-length transform feature for each Gaussian Particle and develop a unified decoder to determine particle pose variations at any given time, enabling spatiotemporally continuous fluid motion in the reconstructed simulation.
- We incorporate a density-based soft constraint to guide uniform distribution, specifically within fluid regions, in order to maintain fundamental incompressibility during reconstruction.

2. Related Work

2.1. Lagrangian-based Fluid Simulation

Fluid simulation is a significant research area in computer vision and graphics. The Lagrangian method is favored due to its ability to directly track fluid particles, making it more suitable for simulating fluids with complex motion details. A prominent method in this domain is Smoothed Particle Hydrodynamics (SPH), which represents fluid dynamics through integral particle approximation [Mon92]. To enforce fluid incompressibility, Weakly Compressible SPH [BT07] employs the state equation method, calculating pressure based on compression. Several methods have been developed to improve stability, including Predictive-Corrective Incompressible SPH [SP09] and Divergence-Free SPH [BK15]. Beyond the state equation approach, Implicit Incompressible SPH [ICS*14] implements projection methods by solving the Poisson equation for particle pressures to achieve accurate fluid simulation. The Position-Based Fluid (PBF) [MM13] combines SPH with Position-Based Dynamics to directly correct particle positions without explicitly computing pressure.

Fundamentally, fluid simulation involves solving complex partial differential equations, and machine learning approaches have been extensively applied to address these problems more efficiently [LKA*21, XR21]. For Lagrangian fluids, given their particle-based representation, specialized feature processing architectures and learning-based fluid simulators have been developed, including convolutional neural networks [UPTK20, WT24] and graph neural networks [SGGP*20, MZW*18]. However, both physics-based and machine learning-based fluid simulation methods typically lack direct connections to real-world fluids. Undoubtedly, the trend toward leveraging real fluid videos is an inevitable development.

2.2. 3D Reconstruction from Visual Data

Reconstructing 3D scenes from 2D images remains a cornerstone of computer vision. Early techniques, such as stereo vision and structure from motion, leveraged geometric principles to deduce depth information [SSS06, HZ03]. Multi-View Stereo (MVS) techniques facilitate the creation of detailed 3D models from multiple images [GSC*07, HPP*18]. Neural networks have been progressively integrated into the task of 3D reconstruction, including novel viewpoint synthesis [FNPS16, ZTS*16] and blend weight estimation [HPP*18], predominantly following the MVS approach. The introduction of Neural Radiance Fields (NeRF) marked a significant milestone, providing a novel framework for synthesizing realistic views of complex scenes [MST*22, BMV*22]. To accelerate reconstruction, various discretization strategies have been developed to mitigate the computational inefficiency of multi-layer perceptron (MLP) queries [FKYT*22, WXZ*22, YLT*21, MESK22].

The 3D Gaussian Splatting (3DGS) method further enhances interpretability and computational efficiency by representing renderable primitives in explicit 3D Gaussian forms [KKLD23, YCH*24], enabling real-time, high-quality novel view synthesis at reduced computational cost.

In dynamic 3D reconstruction, researchers have incorporated a temporal dimension into NeRF, either by introducing additional temporal encoding to handle object motion [PCPMMN21, LCM*24, DZY*21], or by directly extending the original NeRF framework [PSB*21, TTG*21, WCS*22, LSZ*22]. Owing to their explicit representation, 3D Gaussians serve as effective primitives for dynamic modeling [YGZ*24, WYF*24, YYPZ24, LDZY24, LCLX24]. Nevertheless, these methods typically encounter challenges when applied to scenes with ill-defined boundaries, particularly for fluids. In this paper, we propose a method that leverages not only a modified Gaussian splatting approach for reconstruction but also Lagrangian fluid particles for simulation.

2.3. Radiance Field and Physics

Gaussian Splashing [FFS*25] leverages 3DGS to generate novel physics-based animations involving fluid–solid coupling. However, it can only reconstruct solid boundaries using 3DGS, while the fluid component remains synthetically defined in virtual scenes. PhysGaussian [XZQ*24] employs a custom Material Point Method (MPM) to enhance 3D Gaussian kernels with physically meaningful attributes for kinematic deformation and mechanical stress, following the principles of continuum mechanics. However, these methods cannot reconstruct fluids from videos. For smoke, a particularly challenging fluid type, FluidNexus [GYZW25] integrates 3DGS with PBF and exploits pre-trained image generation and video refinement models to reconstruct photorealistic smoke. Nevertheless, its requirement for scene-specific parameter pre-training severely limits the scalability and applicability of FluidNexus.

Current physics-based radiance field methods predominantly focus on solids or elastic bodies, with limited attention to fluid dynamics. To recover fluid density and velocity from sparse multiview videos, HyFluid [YZG*23] introduces physics-based losses to infer a physically plausible velocity field. PAC-NeRF [LQYC*23] uses a hybrid Eulerian-Lagrangian representation of the neural radiance field to estimate both unknown geometry and physical parameters for highly dynamic objects from multiview videos.

The reconstruction of fluids is often challenging, particularly due to their inherent high refractivity and reflectivity. MirrorGaussian [LTC*25] is a pioneering method for mirror scene reconstruction with real-time rendering based on 3D Gaussian Splatting. GlossyGS [LHG*25] employs microfacet geometry segmentation priors and integrates material priors to accurately reconstruct the geometry and material properties of glossy objects. TransparentGS [HYD*25] introduces transparent Gaussian primitives, enabling specular refraction via a deferred refraction strategy.

For fluid systems, NeuroFluid [GDWY22] infers state transitions and interactions within fluid particle systems from visual observations of the fluid surface, but it requires initial fluid particle positions and cannot directly reconstruct dynamic fluids from videos. 3D-IntPhys [XTT*23] adopts a conditional NeRF-style visual frontend and a 3D point-based dynamics prediction backend to learn 3D-grounded intuitive physics from videos of complex scenes, though its effectiveness has yet to be fully evaluated. Recently, DEL [JJZ*24] integrated learnable graph kernels into Discrete Element Analysis to learn material dynamics from partial 2D observations in a unified manner, but its practical performance remains unexplored. GaussFluids aims to reconstruct Lagrangian fluid particles directly from videos, enabling multiple downstream tasks.

3. Method

As illustrated in Figure 1, GaussFluids aims to reconstruct a set of moving particles over time from multiview video data. The overall process is depicted in Figure 2. GaussFluids takes particles with randomly initialized positions as input and optimizes them using multiview fluid videos. First, (a) the fluid particles are extended into Gaussian Fluid Particles, which incorporate position $\mathbf{p}$ and additional Gaussian differentiable rendering primitives for splatting. These primitives comprise spherical harmonics(SH) lighting coefficients $\mathbf{c}$ and opacity α for color control, as well as scaling $\mathbf{s}$ and rotation $\mathbf{r}$ for shape control. (b) Each particle is associated with a transform feature $\theta \in \mathbb{R}^{N_\theta}$, where N_θ is the dimension of the transform feature vector, which is used to decode the changes in the Gaussian pose over time—specifically $\Delta\mathbf{p}$, $\Delta\mathbf{s}$, and $\Delta\mathbf{r}$—contributing to the construction of the covariance matrix Σ. (c) Through splatting differentiable rendering from the camera view $\mathbf{d}$, hybrid constraints are incorporated into the loss function, optimizing both the rendering primitives and the transform features of the Gaussian Particles. (d) Finally, an improved density optimization module is applied to guide Lagrangian fluid motion reconstruction from the perspectives of visual, physical, and geometric consistency.

3.1. Gaussian Fluid Particles

GaussFluids is a novel attempt to reconstruct spatio-temporal continuity of Lagrangian fluid particles directly from videos. To achieve this, we use Gaussian particles to represent fluid particles instead of traditional point clouds. This is because point clouds are spatially discrete structures, and according to the principles of calculus, if the domain of a function is discrete, derivatives cannot be defined at those points, making them non-differentiable. Gaussian particles, on the other hand, introduce continuity and maintain differentiability, thereby enabling the optimization of particles from visual data.

Under the Lagrangian description, fluid is discretized into a set of particles, simulating interactions with relevant constraints to derive fluid motion from one frame to the next. The positions at any time t, denoted as $\mathbf{P}_t \in \mathbb{R}^{N\times 3}$ (where N denotes the total number of particles), can be obtained from the initial particle positions $\mathbf{P}_0$ using a Lagrangian fluid simulator $\mathcal{L}$, such that $\mathbf{P}_t = \mathcal{L}(\mathbf{P}_0, t)$. Note that $\mathbf{P}_0 = \mathcal{L}(\mathbf{P}_0, 0)$. Lagrangian fluid dynamics are governed by two critical constraints:

Navier-Stokes Equations describe the conservation of momentum:

$$\rho \frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{f}, \tag{1}$$

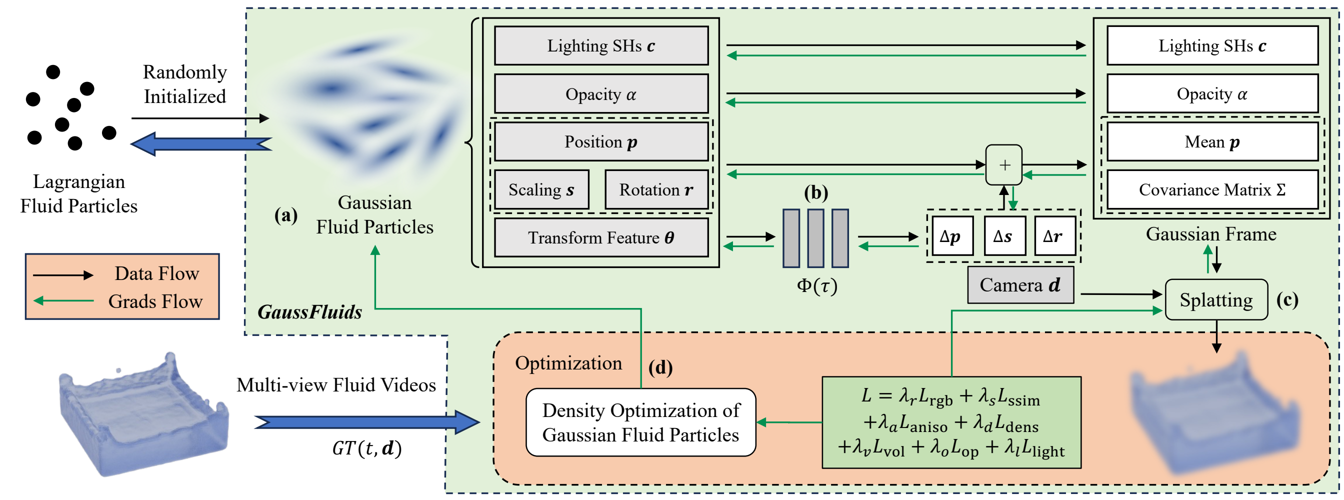
Randomly Initialized
Lagrangian Fluid Particles
Data Flow
Grads Flow
(a)
Gaussian Fluid Particles
GaussFluids
Lighting SHs c
Opacity α
Position p
Scaling s
Rotation r
Transform Feature θ
(b)
$\Phi(\tau)$
Δp
Δs
Δr
Camera d
Mean p
Covariance Matrix Σ
Gaussian Frame
Splatting
(c)
Multi-view Fluid Videos
$GT(t, d)$
Optimization
(d)
Density Optimization of Gaussian Fluid Particles
$L = \lambda_r L_{\text{rgb}} + \lambda_s L_{\text{ssim}} + \lambda_a L_{\text{aniso}} + \lambda_d L_{\text{dens}} + \lambda_v L_{\text{vol}} + \lambda_o L_{\text{op}} + \lambda_l L_{\text{light}}$

Figure 2: The pipeline of GaussFluids. The optimization process begins with randomly initialized Lagrangian fluid particles. GaussFluids optimizes these particles using multiview fluid videos. (a) The Lagrangian particles are extended to Gaussian Particles with 3D Gaussian probability density [KKLD23], carrying extra rendering information that makes fluid particles differentiable for visual data. (b) We introduce a transform feature θ for each Gaussian Particle. Through the unified neural network Φ, it estimates changes in the Gaussian pose over time τ (represented in the frequency domain from t). (c) By employing splatting differentiable rendering from camera **d**, the carefully designed constraint loss L is backpropagated to each Gaussian Particle, enabling efficient optimization. (d) Improved density optimization uses gradient information to strategically densify and prune the particles.

where ρ is the fluid density, $\mathbf{u} \in \mathbb{R}^3$ is the velocity field, p is the pressure field, μ is the viscosity, and $\mathbf{f} \in \mathbb{R}^3$ represents external forces acting on the fluid.

Incompressibility Condition arises from the continuity equation in fluid dynamics, stating that the fluid density remains constant if the divergence of the velocity field at any position is zero:

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = -\rho \nabla \cdot \mathbf{u} = 0. \tag{2}$$

This condition is fundamental for fluid simulation, as it ensures that the internal fluid pressure maintains consistency throughout the fluid body. Lagrangian fluid particles only carry kinematic information, including positions $\mathbf{P}_t$ and velocities $\mathbf{U}_t$, both $\in \mathbb{R}^{N\times 3}$. To render these particles, additional parameters such as particle radius R, opacity α, and color $\mathbf{c} \in \mathbb{R}^3$ must be introduced.

Because fluid particles, like other point clouds, are not differentiable with respect to visual data [WQF19], we incorporate Gaussian probability density into each fluid particle, creating Gaussian Particles. Specifically, each Gaussian Particle is characterized by its position $\mathbf{p} \in \mathbb{R}^3$, as well as scaling $\mathbf{s} \in \mathbb{R}^3$ and rotation $\mathbf{r} \in \mathbb{R}^4$ to construct the covariance matrix Σ following [KKLD23]. This formulation defines the particle pose $\mathbf{A} = \{\mathbf{p}, \mathbf{s}, \mathbf{r}\}$. During rendering, opacity α controls transparency, and a first-order SH coefficients $\mathbf{c}$ determines the particle's color. Higher-frequency lighting effects are omitted for simplicity.

We implement splatting as our differentiable rendering mechanism, projecting each Gaussian Particle onto the image plane. At pixel x, the color $\mathcal{C}_x$ is computed by

$$\mathcal{C}_x = \sum_k G_k(x)\alpha_k \mathcal{S}(c_k) \prod_{j=1}^{k-1} \left(1 - G_j(x)\alpha_j\right), \tag{3}$$

where k indexes the Gaussian Particles in ascending order of their depth from the camera. $G_k(x)$ is the 2D Gaussian weight at pixel x, derived from $\mathbf{p}$ and Σ as in [ZPvBG01], and $\mathcal{S}$ denotes SH.

Rather than applying uniform values across the entire scene, we assign each particle independent parameters for pose, opacity, and color. This approach acknowledges that particles in different regions may require distinct appearance characteristics. However, we incorporate a consistency soft constraint and employ density optimization (Section 3.3) for efficient training.

By computing the difference between the splatted and ground-truth images, gradients are backpropagated to each Gaussian Particle's attributes, including pose and rendering parameters. This mechanism enables GaussFluids to directly optimize Gaussian Particles from multiview video data.

3.2. Spatio-Temporal Encoding

To model the temporal and spatial evolution of fluid particles, we propose a novel spatio-temporal encoding strategy. Traditional fluid simulation methods typically employ time-stepping techniques for temporal discretization to track the motion trajectory of each fluid particle [Mon92, BT07, BK15]. These methods generally assume that forces and accelerations remain constant within timesteps. However, finding an efficient yet effective timestep size is challenging.

To represent the complex dynamics of fluid particle motion, instead of discrete representations, GaussFluids encodes each particle with a transform feature θ—a fixed-length vector—and stores trajectory features at the particle level. A globally shared multi-layer perceptron (MLP) network Φ decodes θ together with time τ, represented in the frequency domain from t using the method described in [RBA*19], to obtain the transformation $\Delta\mathbf{A}^t$ at time t. This formulation enables continuous representation of particles in both temporal and spatial domains, thereby avoiding temporal discretization. Based on the notation introduced in Section 3.1, this process is formulated as follows:

$$\Delta\mathbf{A}^t = \Phi(\theta,\tau),\, \mathbf{A}^t = \mathbf{A}^0 + \Delta\mathbf{A}^t. \tag{4}$$

With this encoding method, the spatial position of each particle at any given time can be accurately determined. This approach not only enhances the temporal and spatial continuity of the fluid representation but also enables downstream tasks such as time interpolation, as demonstrated in Section 4.3.2.

3.3. Density Optimization and Constraint Conditions

Using MLPs to model the motion of particles ensures visual consistency but neglects the fundamental physical properties of the fluid, such as incompressibility. According to Equation (2), density constraints ensure that the mass per unit volume remains constant regardless of how the fluid moves or deforms, which is key to achieving mass conservation.

We implement a density soft constraint while preserving visual fidelity. Specifically, the mass of fluid particles is defined as unit mass, and a smoothing kernel function W is used to compute the density estimate $\mathcal{D}_i$ at the location of fluid particle i. This smoothing kernel approach is widely adopted in particle-based fluid simulations to distribute attributes across neighboring particles [Mon92, MCG03]. We define the density loss as:

$$L_{\text{dens}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\mathcal{D}_i}{\bar{\mathcal{D}}}-1\right)^2,\, \mathcal{D}_i = \sum_{j=1}^{N_i} W(\|\mathbf{p}_i-\mathbf{p}_j\|_2, h), \tag{5}$$

where N_i represents the number of neighboring particles of particle i, and $\bar{\mathcal{D}} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{D}_i$ is the average density across all particles. Here, $\mathbf{p}_i$ and $\mathbf{p}_j$ denote the positions of particles i and j, respectively. The smoothing kernel function chosen here is Poly6:

$$W(r,h) = \frac{315}{64\pi h^9}\begin{cases}(h^2-r^2)^3, & \text{if } 0 \le r \le h,\\ 0, & \text{otherwise,}\end{cases} \tag{6}$$

where r denotes the distance between two particles and h denotes the smoothing length.

To enable Gaussian Particles to approximate Lagrangian particles, all particles should maintain volume consistency via:

$$L_{\text{vol}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{V_i}{\bar{V}}-1\right)^2, \quad V_i = \mathbf{s}_{ix}\mathbf{s}_{iy}\mathbf{s}_{iz}, \tag{7}$$

where $\bar{V} = \frac{1}{N}\sum_{i=1}^{N} V_i$ is the average volume across all particles, and geometric isotropy via:

$$L_{\text{aniso}} = \frac{1}{N}\sum_{i=1}^{N}\left\|\frac{\mathbf{s}_i}{V_i^{\frac{1}{3}}}-(1,1,1)\right\|^2. \tag{8}$$

Here, $\mathbf{s}_i$ represents the scaling of the i-th particle, and $\mathbf{s}_{ix}$, $\mathbf{s}_{iy}$, $\mathbf{s}_{iz}$ represent its components in the x, y, and z directions, respectively. Additionally, for uniform fluid particles, different particles should maintain consistent opacity and color. Thus, we define:

$$L_{\text{op}} = L_{\text{MSE}}(\alpha), \quad L_{\text{light}} = L_{\text{MSE}}(\mathbf{c}), \tag{9}$$

$$L_{\text{MSE}}(y) = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \frac{1}{N}\sum_{i=1}^{N} y_i\right)^2. \tag{10}$$

Here, α_i and $\mathbf{c}_i$ represent the opacity and SH lighting coefficients of the i-th particle, respectively.

The final loss function combines visual consistency terms (L_{rgb} and $L_{\text{d-ssim}}$ from [KKLD23]), physical constraints, and geometric regularization, with each term weighted by a corresponding λ parameter:

$$\begin{aligned} L = {} & \lambda_r L_{\text{rgb}} + \lambda_s L_{\text{d-ssim}} + \lambda_d L_{\text{dens}} + \lambda_a L_{\text{aniso}} \\ & + \lambda_v L_{\text{vol}} + \lambda_o L_{\text{op}} + \lambda_l L_{\text{light}}. \end{aligned} \tag{11}$$

4. Experiments

To validate the efficacy of GaussFluids, we conducted experiments on both synthetic and real-world datasets. We focus on two main aspects: (1) the effectiveness of reconstructing fluid particles' motion from videos, and (2) the applications of these reconstructed particles in subsequent tasks.

4.1. Experiments Setup

4.1.1. Environment

All experiments were performed on a machine equipped with an NVIDIA RTX 3090, an Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz, and 64 GB DDR4 main memory. The software environment includes Python 3.7, PyTorch 1.12.1, and CUDA 11.8.

4.1.2. Implementation Details

The iteration process consists of three phases: the first phase focuses on training canonical frame particles, the second phase concentrates on learning the dynamic scene, and the third phase emphasizes particle refinement. Each phase commences after the previous phase has converged. We found this to be an effective strategy for reconstructing temporally and spatially continuous fluid particles from videos.

Table 1: The weights assigned to each loss term

Weight	λ_r	λ_s	λ_d	λ_a	λ_v	λ_o	λ_l
Setting	0.2	0.1	0.1	0.1	0.1	0.01	1

The canonical frame phase follows the official 3DGS [KKLD23] procedure and setup. In the Spatio-Temporal Encoding module, the transform feature dimension N_θ is set to 64. The decoder network Φ has 4 hidden layers, each containing 256 neurons, with ReLU activation in every layer. Both are initialized to zero. The learning rates for the feature space and the MLP network are set to 10^{-2} and 10^{-4}, respectively. The ablation studies related to this setup are discussed in Section 4.2.3. The weights assigned to each loss term, as shown in Table 1, are chosen primarily to ensure that the magnitudes of the different loss terms remain within a similar range. Additionally, we employ a dynamic weighting schedule during the three phases described above. In Phase 1, λ_d is set to 0 (density ignored), while λ_a and λ_v are set to 10% of their final values to prioritize visual consistency. In Phase 2, λ_d increases linearly to 10% of its final value to encourage a more uniform particle distribution. In Phase 3, λ_a, λ_v, and λ_d are gradually raised to their final values. GaussFluids is not highly sensitive to the final absolute weights, but gradual scheduling is crucial to avoid early over-regularization and structural distortion. Ablation results on WaterSphere, shown in Table 2, confirm the effectiveness of the dynamic weighting schedule.

We employ the K-Nearest Neighbors algorithm ($K = 64$) for operations involving neighboring particles. Additionally, the smoothing kernel radius h in Equation (5) is set to 0.3 to balance accuracy and computational efficiency. Empirically, 0.3 performed best among tested values and is consistent with prior recommendations in the literature [JYJY22, KBST19].

4.1.3. Dataset information

The synthetic dataset NeuroFluid [GDWY22] was generated using the DFSPH algorithm [BK15] and rendered in Blender. This dataset features three scenarios: WaterCube, WaterSphere, and HoneyCone, each captured from five frontal-view cameras without posterior angles. The scenarios simulate fluid dynamics over a one-second interval, with WaterCube and HoneyCone at 99 fps and WaterSphere at 60 fps. All images were rendered at a resolution of 400×400 pixels. Material properties for ‘Water’ include a Glass BSDF shader with a viscosity of 0.08 and a density of 1000 kg/m^3. ‘Honey’ is modeled with a Principled BSDF shader, a viscosity of 0.8, and a density of 1420 kg/m^3.

The ScalarFlow dataset [EUT19] consists of real, five-view smoke recordings containing 104 different smoke flows, each with 150 frames of 2D grayscale images at a resolution of 1080×1920 pixels. We performed XOR operations between each frame and the first frame to remove the background, and we use one channel for the SH lighting since the images are grayscale.

4.2. Reconstructed Lagrangian Fluid Particles

4.2.1. Particle Reconstruction

The Lagrangian method uses particles to represent objects. Figure 3 illustrates the reconstruction of the HoneyCone and WaterSphere scenes from the NeuroFluid dataset, representing two distinct material properties. GaussFluids effectively reproduces the fluid morphologies, with detailed reconstructions evident at both macro and micro scales. HoneyCone exhibits denser particle formations due to its higher viscosity, as shown in Table 3, with 62,927 particles and smaller radii (0.007 m). WaterSphere demonstrates more extensive movement patterns, resulting in fewer and larger particles.

The ScalarFlow reconstruction results are presented in Figure 6. The particle radius after reconstruction with GaussFluids is approximately 0.006 m. The second row depicts the rendering with 80% Gaussian scaling, demonstrating GaussFluids’ effective interpretation of the scene. The third row shows the reconstruction results in the particle state; the particles can be observed gradually diffusing upwards from the bottom while maintaining the same smoke morphology as in the original video.

4.2.2. Fluid Dynamics

In Lagrangian fluid simulation, maintaining incompressibility—where each fluid particle’s density remains constant—is a fundamental soft constraint, as described by Equation (2). Using the WaterSphere scene as an example, we compute the standard deviation of $\mathcal{D}_i$ for each frame, as shown in Figure 4. Compared with D3G [YGZ*24] and SGFS [LCLX24], which are purely vision-based 4D reconstruction methods lacking physical guidance, GaussFluids achieves the lowest and most consistent standard deviation values, indicating superior preservation of incompressibility. These results demonstrate that GaussFluids not only reconstructs surface particles but effectively captures interior fluid physics.

Mechanical energy (the sum of gravitational potential energy and kinetic energy) is another crucial physical metric. As shown in Figure 5, we also use the WaterSphere scene to visualize changes in mechanical energy. The kinetic energy initially increases due to gravitational acceleration, while gravitational potential energy gradually decreases in the absence of collisions. Upon the fluid’s contact with the container’s bottom, gravitational potential energy stabilizes, causing kinetic energy to diminish until the total mechanical energy reaches a steady state. Compared to D3G and SGFS, GaussFluids demonstrates more physically consistent energy dissipation, suggesting that the reconstructed Lagrangian fluids better conform to established physical principles.

4.2.3. Ablation Study

To investigate how the parameter selection of the network Φ affects reconstruction efficiency and accuracy, we performed an ablation study using the 20th frame of the WaterCube scene. This frame was chosen because it contains an intense collision state, placing significant demands on temporal feature encoding and decoding. As shown in Table 4, a feature-vector dimension of 64 with 256 neurons provides an optimal trade-off between performance and efficiency. When the feature dimension is 32, the visual results show particles exiting the boundaries directly, leading to reconstruction failure. Parameter settings beyond these values offer only marginal performance improvements while incurring substantial computational costs.

For the synthetic dataset, the current setup employs five camera views. We examine the effect of reducing the synthetic dataset’s camera views to four, three, and two, respectively. As shown in Table 5, with only two views, reconstruction fails which is consistent with established principles in computer vision where two views are insufficient for reliable 3D reconstruction. Increasing from three to

Table 2: Ablation of dynamic loss weight scheduling on WaterSphere. “Dynamic” denotes the proposed dynamic weighting schedule.

Condition	PSNR↑	SSIM↑	Comment
Current coefficients (dynamic)	32.45	0.912	Three-phase dynamic weighting schedule
Current coefficients (fixed)	23.85	0.764	Early strong constraints cause distortion
50% coefficients (dynamic)	32.37	0.913	Minimal difference from original
200% coefficients (dynamic)	32.03	0.909	Slight reduction, not significant

Table 3: Reconstruction results on the NeuroFluid dataset.

Dataset	Particles (Count)	Radius (m)	Opacity	Averaged Density	Time (min)
HoneyCone	62927	0.007	0.6%	3407.51	16
WaterCube	33650	0.0150	1.4%	1962.27	14
WaterSphere	41280	0.0120	1.1%	2738.42	16

Table 4: Effects of different feature dimensions and neuron quantities on reconstruction efficiency and quality, using the 20th frame of the WaterCube scene as an example. The "Total" column represents the sum of the time for both "Splatting" and "Φ" operations. Note that a feature dimension of 64 and 256 neurons achieves a favorable balance between performance and efficiency.

Feature dim.	Neurons per layer	Reconstruction quality		Time (s)		
		Max. density	Max. speed (m/s)	Splatting	Φ	Total
32	128	1462.07	341.96	0.021	0.007	0.028
32	256	1626.70	298.05	0.027	0.019	0.046
64	256	2583.63	89.43	0.045	0.032	0.077
64	512	2635.25	87.59	0.048	1.035	1.083
128	512	2597.80	85.24	0.047	1.835	1.882

four views significantly improves performance, while the transition from four to five views offers diminishing returns, due to the limited angular diversity in the NeuroFluid dataset’s camera positions.

Table 5: Ablation study results on the HoneyCone scene. S1: Covariance matrix is hardcoded to a scaled identity matrix. S2: All particles share identical opacity and color. S3: Allows SH to vary over time. "FAILED" indicates that the method did not converge or produce meaningful results.

Setup	Particles Number	Particles Radius (m)	Opacity	Averaged Density
5 views	62927	0.007	0.6%	3407.51
4 views	61215	0.007	1.2%	3150.77
3 views	11475	0.037	13.6%	1075.84
2 views	FAILED	FAILED	FAILED	FAILED

4.3. Downstream Applications

4.3.1. Novel Viewpoint Synthesis

Novel viewpoint synthesis is a fundamental task in 3D reconstruction. Conventional methods predominantly reconstruct surface color fields without capturing the underlying physical structure. In contrast, GaussFluids reconstructs fluid particles with continuous temporal coherence while maintaining high-fidelity visual reconstruction. To evaluate novel viewpoint synthesis performance, we selected the HoneyCone scene as an experimental case, reconstructing it from four viewpoints and reserving an additional one for testing. We compare our approach with the NeRF-based D-NeRF [PCPMMN21] method, as well as the 4DGS-based D3G [YGZ*24] and SGFS [LCLX24] approaches.

As illustrated in Figure 7, D-NeRF [PCPMMN21], which builds upon NeRF, exhibits limitations with restricted views, resulting in poor generalization and suboptimal visual quality from novel viewpoints. Similarly, the D3G method [YGZ*24] exhibits visual inconsistencies due to insufficient input views and its surface-based reconstruction approach, as highlighted by the red boxes. In contrast, the SGFS method [LCLX24] achieves more effective reconstruction from sparse viewpoints owing to its explicit spatial feature storage mechanism. GaussFluids, by utilizing inter-frame information (Equation (4)), consistently produces robust reconstruction results with sparse viewpoints and accurately captures fluid details. Although GaussFluids primarily employs first-order SH lighting, we have enhanced its performance by incorporating third-order high-frequency components to improve lighting representation fidelity.

We provide a quantitative evaluation on the WaterCube scene at frame 60, as presented in Table 6. PSNR (Peak Signal-to-Noise Ratio) measures overall reconstruction fidelity, with higher values indicating closer correspondence to ground truth. SSIM (Structural Similarity Index) quantifies structural preservation, where higher scores reflect enhanced visual consistency. GaussFluids achieves superior performance across these metrics compared to alternative methods, partially attributable to its capacity to reconstruct not only the fluid surface but also the interior fluid physics.

Table 6: Quantitative evaluation of novel viewpoint synthesis on the WaterCube scene (frame 60). Higher PSNR and SSIM values indicate better visual quality.

Method	**GaussFluids**	D3G	SGFS	D-NeRF
PSNR↑	29.65	25.65	27.34	8.14
SSIM↑	0.874	0.801	0.834	0.6402

4.3.2. Temporal Resampling

Table 7: Temporal Resampling Results (HoneyCone scene from 60 fps to 120 fps). Higher PSNR/SSIM values indicate better visual quality, and lower Jitter values represent smoother temporal consistency. Jitter is measured by calculating the standard deviation of per-pixel temporal differences between adjacent frames.

Method	PSNR↑	SSIM↑	Jitter↓
GaussFluids	30.08	0.905	0.046
D3G	28.92	0.871	0.096
SGFS	29.31	0.883	0.078

Our approach, which represents fluid particle motion via continuous temporal encoding rather than discrete time steps, enables precise temporal resampling of fluid animations and facilitates high-fidelity frame interpolation. We evaluate this capability using the HoneyCone scene by upsampling the frame rate from 60 fps to 120 fps. As demonstrated in Table 7, GaussFluids attains superior performance with the highest PSNR/SSIM values and the lowest Jitter measure, owing to its physics-aware reconstruction methodology. Visual demonstrations of these results are provided in the supplementary video materials.

4.3.3. Style Transfer

GaussFluids can reconstruct fluid motion in a Lagrangian particle state, enabling style transfer. As shown in Figure 8, we assign new materials to the reconstructed particles and subsequently render them, achieving different visual appearances. In Figure 6 from the ScalarFlow dataset, the fourth row is generated by re-rendering particles with a pink color to demonstrate additional style transfer results.

4.3.4. Fluid Prediction

Table 8: Fluid prediction on WaterSphere. Higher PSNR/SSIM values indicate better quality.

Method	PSNR↑	SSIM↑
GaussFluids	32.45	0.912
3DGS+NeuroFluid	15.32	0.412
GaussFluids (w/o density constraint)	13.55	0.328

Because GaussFluids reconstructs the fluid in a Lagrangian particle state, it can be employed for fluid motion prediction. We use the WaterSphere dataset and reconstruct the initial 20 frames as a particle representation. As shown in Figure 9, GaussFluids learns both fluid particle rendering properties (e.g., color and opacity) for future rendering, and fluid particle motion properties (e.g., velocity and rest density) for subsequent simulation. We then use these properties with PBF [MM13] to simulate the next 40 frames. For comparison, we first rely on 3DGS [KKLD23] to reconstruct the initial frame's particles, followed by NeuroFluid [GDWY22] to predict future frames. While GaussFluids' prediction remains consistent with the ground truth, 3DGS+NeuroFluid fails to produce meaningful results because 3DGS only reconstructs surface particles, leaving NeuroFluid unable to refine them. Furthermore, to underscore the importance of the density constraint (Section 3.3) for capturing volumetric fluid properties essential for prediction, we compare against an ablated version of our method, GaussFluids (w/o density constraint). Without this constraint, the reconstruction tends to focus on surface particles, similar to 3DGS. Consequently, the subsequent PBF simulation, which relies on volumetric particle interactions, fails to produce physically plausible predictions. The numerical indicators in Table 8 confirm that GaussFluids excels across multiple metrics, and the ablated version performs poorly, underscoring the effectiveness of its fluid prediction capabilities when volumetric consistency is maintained.

4.3.5. Fluid Editing

The Lagrangian particle representation of GaussFluids allows direct fluid editing. In Figure 10, 50% of WaterSphere particles are removed, and the simulation continues, maintaining physical coherence.

4.4. Discussion

To verify the effectiveness of GaussFluids in extracting continuous spatiotemporal Lagrangian particles of flowing fluids from multi-view videos, this section presents a comprehensive analysis based on both synthetic and real-world datasets. First, we conduct visual and physical evaluations to demonstrate the physical plausibility of the reconstructed particles. Then, we introduce several downstream tasks to illustrate the practical value of GaussFluids, including novel viewpoint synthesis, temporal interpolation, style transfer, and fluid prediction.

Nonetheless, due to imaging limitations, scenarios involving occlusion or low-contrast regions are inevitable, occasionally leading to suboptimal reconstruction outcomes. Mitigating this issue may require further consideration of fluid-specific properties to enhance reconstruction accuracy.

5. Conclusion

To address the issues of reliance on manual modeling and parameter tuning in fluid simulation, which often lack objectivity and efficiency, this paper proposes GaussFluids, a method that directly reconstructs flowing Lagrangian fluid particles from video data. The Lagrangian-based representation captures more detailed fluid motion but is inherently non-differentiable with respect to visual data. GaussFluids extends Lagrangian fluid particles with 3D Gaussian probability densities, enabling the direct optimization of fluid particle positions from visual data. GaussFluids achieves both temporal and spatial continuity of fluid particles during reconstruction while maintaining fundamental fluid incompressibility, especially within the fluid region. Experiments on both synthetic and real-world datasets demonstrate its effectiveness and broad applicability in novel view synthesis, temporal resampling, style transfer, fluid prediction, and fluid editing. Unlike traditional forward modeling approaches in fluid simulation, GaussFluids provides a novel computational framework for fluid research, especially when leveraging real-world fluid data.

References

[BK15] BENDER J., KOSCHIER D.: Divergence-free smoothed particle hydrodynamics. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (Los Angeles, California, 2015), Association for Computing Machinery, pp. 147–155. 2, 4, 6

[BMV*22] BARRON J. T., MILDENHALL B., VERBIN D., SRINIVASAN P. P., HEDMAN P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 5460–5469. 2

[BT07] BECKER M., TESCHNER M.: Weakly compressible sph for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar, DEU, 2007), Eurographics Association, pp. 209–217. 2, 4

[DZY*21] DU Y., ZHANG Y., YU H.-X., TENENBAUM J. B., WU J.: Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision* (2021), IEEE Computer Society, pp. 14304–14314. 3

[EUT19] ECKERT M.-L., UM K., THUEREY N.: Scalarflow: a large-scale volumetric data set of real-world scalar transport flows for computer animation and machine learning. *ACM Transactions on Graphics 38*, 6 (2019), 1–16. 6

[FFS*25] FENG Y., FENG X., SHANG Y., JIANG Y., YU C., ZONG Z., SHAO T., WU H., ZHOU K., JIANG C.: Gaussian splashing: Unified particles for versatile motion synthesis and rendering. In *Proceedings of the Computer Vision and Pattern Recognition Conference* (2025), pp. 518–529. 3

[FKYT*22] FRIDOVICH-KEIL S., YU A., TANCIK M., CHEN Q. H., RECHT B., KANAZAWA A.: Plenoxels: Radiance fields without neural networks. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 5491–5500. 2

[FNPS16] FLYNN J., NEULANDER I., PHILBIN J., SNAVELY N.: Deepstereo: Learning to predict new views from the world's imagery. In *2016 Ieee Conference on Computer Vision and Pattern Recognition* (2016), pp. 5515–5524. 2

[GDWY22] GUAN S. Y., DENG H. Y., WANG Y. B., YANG X. K.: Neurofluid: Fluid dynamics grounding with particle-driven neural radiance fields. In *International Conference on Machine Learning* (2022), vol. 162. 3, 6, 8

[GSC*07] GOESELE M., SNAVELY N., CURLESS B., HOPPE H., SEITZ S. M.: Multi-view stereo for community photo collections. In *2007 Ieee 11th International Conference on Computer Vision* (2007), vol. 1-6, pp. 825–+. 2

[GYZW25] GAO Y., YU H.-X., ZHU B., WU J.: Fluidnexus: 3d fluid reconstruction and prediction from a single video. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2025), pp. 26091–26101. 3

[HPP*18] HEDMAN P., PHILIP J., PRICE T., FRAHM J.-M., DRETTAKIS G., BROSTOW G.: Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph. 37*, 6 (2018), Article 257. 2

[HYD*25] HUANG L., YE D., DAN J., TAO C., LIU H., ZHOU K., REN B., LI Y., GUO Y., GUO J.: Transparentgs: Fast inverse rendering of transparent objects with gaussians. *ACM Transactions on Graphics* (2025). 3

[HZ03] HARTLEY R., ZISSERMAN A.: *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2

[ICS*14] IHMSEN M., CORNELIS J., SOLENTHALER B., HORVATH C., TESCHNER M.: Implicit incompressible sph. *IEEE Transactions on Visualization and Computer Graphics 20*, 3 (2014), 426–435. 2

[JJZ*24] JIAXU W., JINGKAI S. U. N., ZIYI Z., JUNHAO H., QIANG Z., MINGYUAN S., RENJING X.: Del : Discrete element learner for learning 3d particle dynamics with neural rendering. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems* (2024). 3

[JYJY22] JIAO T., YE M., JIN M., YANG J.: Decoupled finite particle method with normalized kernel (dfpm-nk): A computationally efficient method for simulating solute transport in heterogeneous porous media. *Water Resources Research 58*, 8 (2022), e2022WR032308. 6

[KBST19] KOSCHIER D., BENDER J., SOLENTHALER B., TESCHNER M.: Smoothed particle hydrodynamics techniques for the physics based simulation of fluids and solids. *Eurographics 2019-Tutorials* (2019). 6

[KBST22] KOSCHIER D., BENDER J., SOLENTHALER B., TESCHNER M.: A survey on sph methods in computer graphics. *Computer Graphics Forum 41*, 2 (2022), 737–760. 2

[KKLD23] KERBL B., KOPANAS G., LEIMKUEHLER T., DRETTAKIS G.: 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics 42*, 4 (2023), 1–14. 2, 3, 4, 5, 8

[LCLX24] LI Z., CHEN Z., LI Z., XU Y.: Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 8508–8520. 3, 6, 7

[LCM*24] LIU J.-W., CAO Y.-P., MAO W., ZHANG W., ZHANG D. J., KEPPO J., SHAN Y., QIE X., SHOU M. Z.: Devrf: fast deformable voxel radiance fields for dynamic scenes. In *Proceedings of the 36th International Conference on Neural Information Processing Systems* (2024), Curran Associates Inc., p. Article 2664. 3

[LDZY24] LIN Y., DAI Z., ZHU S., YAO Y.: Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 21136–21145. 3

[LHG*25] LAI S., HUANG L., GUO J., CHENG K., PAN B., LONG X., LYU J., LV C., GUO Y.: Glossygs: Inverse rendering of glossy objects with 3d gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics* (2025), 1–14. 3

[LKA*21] LI Z., KOVACHKI N. B., AZIZZADENESHELI K., BHATTACHARYA K., STUART A., ANANDKUMAR A.: Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations* (2021). 2

[LQYC*23] LI X., QIAO Y.-L., YICHEN CHEN P., MURTHY JATAVALLABHULA K., LIN M., JIANG C., GAN C.: Pac-nerf: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. In *The Eleventh International Conference on Learning Representations* (2023). 3

[LSZ*22] LI T., SLAVCHEVA M., ZOLLHOEFER M., GREEN S., LASSNER C., KIM C., SCHMIDT T., LOVEGROVE S., GOESELE M., NEWCOMBE R., LV Z.: Neural 3d video synthesis from multi-view video. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 5511–5521. 3

[LTC*25] LIU J., TANG X., CHENG F., YANG R., LI Z., LIU J., HUANG Y., LIN J., LIU S., WU X., XU S., YUAN C.: Mirrorgaussian: Reflecting 3d gaussians for reconstructing mirror reflections. In *Computer Vision – ECCV 2024* (2025), Leonardis A., Ricci E., Roth S., Russakovsky O., Sattler T., Varol G., (Eds.), Springer Nature Switzerland, pp. 377–393. 3

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003), Eurographics Association, p. 154–159. 5

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics 41*, 4 (2022), 1–15. 2

[MM13] MACKLIN M., MÜLLER M.: Position based fluids. *ACM Transactions on Graphics 32*, 4 (2013), 1–12. 2, 8

[Mon92] MONAGHAN J. J.: Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics 30* (1992), 543–574. 2, 4, 5

[MST*22] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the Acm 65*, 1 (2022), 99–106. 2

[MZW*18] Mrowca D., Zhuang C., Wang E., Haber N., Fei-Fei L., Tenenbaum J. B., Yamins D. L. K.: Flexible neural representation for physics prediction. In *the 32nd International Conference on Neural Information Processing Systems* (2018), Curran Associates Inc., p. 8813–8824. 2

[PCPMMN21] Pumarola A., Corona E., Pons-Moll G., Moreno-Noguer F.: D-nerf: Neural radiance fields for dynamic scenes. In *2021 Ieee/Cvf Conference on Computer Vision and Pattern Recognition* (2021), pp. 10313–10322. 3, 7

[PSB*21] Park K., Sinha U., Barron J. T., Bouaziz S., Goldman D. B., Seitz S. M., Martin-Brualla R.: Nerfies: Deformable neural radiance fields. In *2021 Ieee/Cvf International Conference on Computer Vision* (2021), pp. 5845–5854. 3

[RBA*19] Rahaman N., Baratin A., Arpit D., Draxler F., Lin M., Hamprecht F., Bengio Y., Courville A.: On the spectral bias of neural networks. In *the 36th International Conference on Machine Learning* (2019), Kamalika C., Ruslan S., (Eds.), vol. 97, PMLR, pp. 5301–5310. 5

[SGGP*20] Sanchez-Gonzalez A., Godwin J., Pfaff T., Ying R., Leskovec J., Battaglia P.: Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning* (Virtual, 2020), Hal Daumé I., Aarti S., (Eds.), vol. 119, PMLR, pp. 8459–8468. 2

[SP09] Solenthaler B., Pajarola R.: Predictive-corrective incompressible sph. *ACM Transactions on Graphics 28*, 3 (2009), 1–6. 2

[SSS06] Snavely N., Seitz S. M., Szeliski R.: Photo tourism: Exploring photo collections in 3d. *Acm Transactions on Graphics 25*, 3 (2006), 835–846. 2

[TTG*21] Tretschk E., Tewari A., Golyanik V., Zollhöfer M., Lassner C., Theobalt C.: Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *2021 Ieee/Cvf International Conference on Computer Vision* (2021), pp. 12939–12950. 3

[UPTK20] Ummenhofer B., Prantl L., Thuerey N., Koltun V.: Lagrangian fluid simulation with continuous convolutions. In *International Conference on Learning Representations* (Virtual, 2020). 2

[WCS*22] Weng C. Y., Curless B., Srinivasan P. P., Barron J. T., Kemelmacher-Shlizerman I.: Humannerf: Free-viewpoint rendering of moving people from monocular video. In *2022 Ieee/Cvf Conference on Computer Vision and Pattern Recognition* (2022), pp. 16189–16199. 3

[WQF19] Wu W., Qi Z., Fuxin L.: Pointconv: Deep convolutional networks on 3d point clouds. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 9613–9622. 2, 4

[WT24] Winchenbach R., Thuerey N.: Symmetric basis convolutions for learning lagrangian fluid mechanics. In *The Twelfth International Conference on Learning Representations* (2024). 2

[WXZ*22] Wu X., Xu J., Zhu Z., Bao H., Huang Q., Tompkin J., Xu W.: Scalable neural indoor scene rendering. *ACM Transactions on Graphics 41*, 4 (2022), 1–16. 2

[WYF*24] Wu G., Yi T., Fang J., Xie L., Zhang X., Wei W., Liu W., Tian Q., Wang X.: 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 20310–20320. 3

[XR21] Xu R.-J., Ren B.: Solving the fluid pressure with an iterative multi-resolution guided network. *The Visual Computer 38*, 2 (2021), 433–442. 2

[XTT*23] Xue H., Torralba A., Tenenbaum J., Yamins D., Li Y., Tung H.-Y.: 3d-intphys: Towards more generalized 3d-grounded visual intuitive physics under challenging scenes. In *Advances in Neural Information Processing Systems* (2023), Oh A., Naumann T., Globerson A., Saenko K., Hardt M., Levine S., (Eds.), vol. 36, Curran Associates Inc., pp. 7116–7136. 3

[XZQ*24] Xie T., Zong Z., Qiu Y., Li X., Feng Y., Yang Y., Jiang C.: Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 4389–4398. 3

[YCH*24] Yu Z., Chen A., Huang B., Sattler T., Geiger A.: Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 19447–19456. 3

[YGZ*24] Yang Z., Gao X., Zhou W., Jiao S., Zhang Y., Jin X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 20331–20341. 3, 6, 7

[YLT*21] Yu A., Li R. L., Tancik M., Li H., Ng R., Kanazawa A.: Plenoctrees for real-time rendering of neural radiance fields. In *2021 Ieee/Cvf International Conference on Computer Vision* (2021), pp. 5732–5741. 2

[YYPZ24] Yang Z., Yang H., Pan Z., Zhang L.: Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In *The Twelfth International Conference on Learning Representations* (2024). 3

[YZG*23] Yu H.-X., Zheng Y., Gao Y., Deng Y., Zhu B., Wu J.: Inferring hybrid neural fluid fields from videos. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (2023), Curran Associates Inc., p. Article 2777. 3

[ZPvBG01] Zwicker M., Pfister H., van Baar J., Gross M.: Ewa volume splatting. *Visualization 2001, Proceedings* (2001), 29–36. 4

[ZTS*16] Zhou T., Tulsiani S., Sun W., Malik J., Efros A. A.: View synthesis by appearance flow. In *Computer Vision – ECCV 2016* (2016), vol. 9908 of *Computer Vision – ECCV 2016*, Springer International Publishing, pp. 286–301. 2

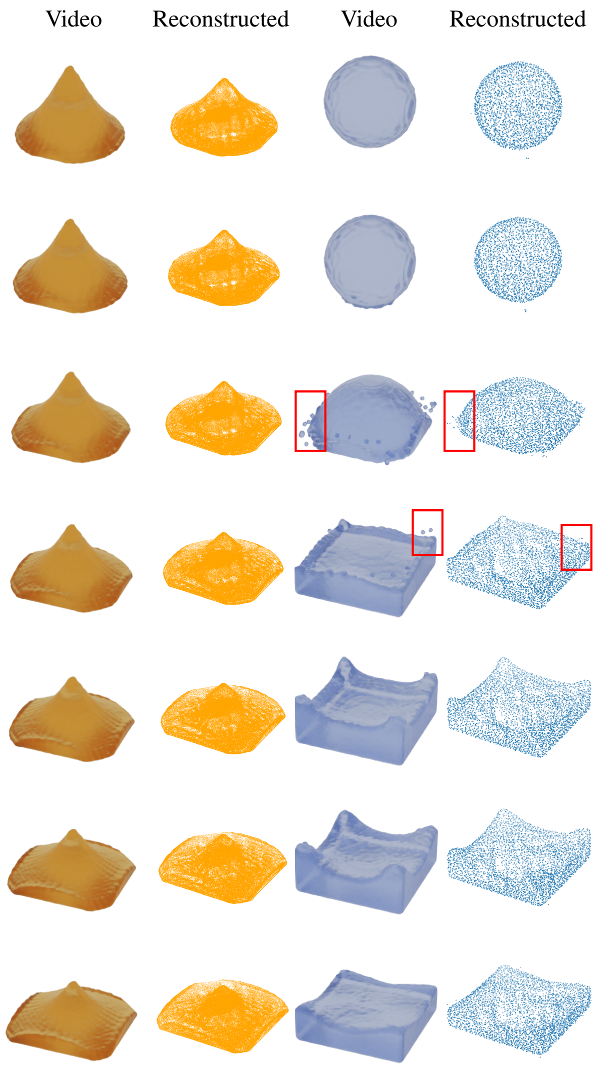

Figure 3: Reconstructed particles for the HoneyCone and WaterSphere scenes of the NeuroFluid dataset. GaussFluids uses only video input to reconstruct complete, temporally and spatially continuous Lagrangian fluid particles. As highlighted in the red boxes, fluid details are also preserved.

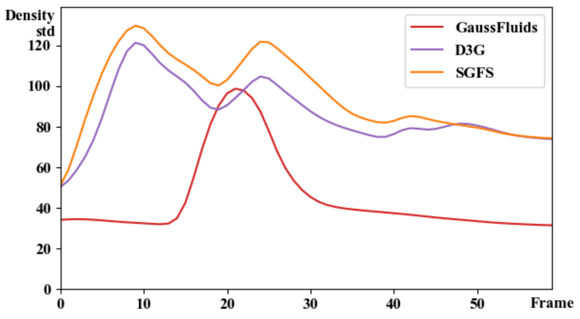

Figure 4: Standard deviation of $\mathcal{D}_i$ in the WaterSphere scene, where the horizontal axis represents time frames. GaussFluids maintains the lowest standard deviation, underscoring its strong adherence to incompressibility.

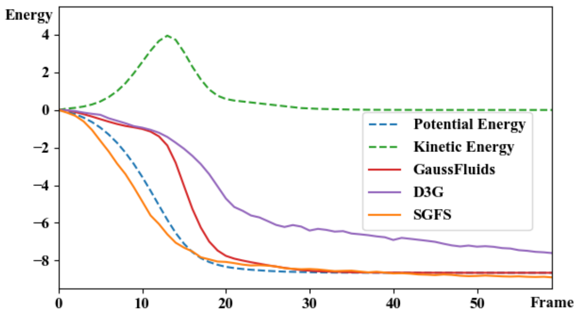

Figure 5: Temporal variations of average mechanical energy for the reconstructed particles, using WaterSphere. GaussFluids exhibits gradually dissipating mechanical energy, with the most significant drop occurring as the fluid collides with the container bottom. By contrast, D3G shows fluctuations in mechanical energy, occasionally even increasing, whereas SGFS experiences an excessively rapid initial energy drop. These results collectively indicate that GaussFluids more closely aligns with physical laws. Subplots of kinetic and gravitational potential energy are also provided.

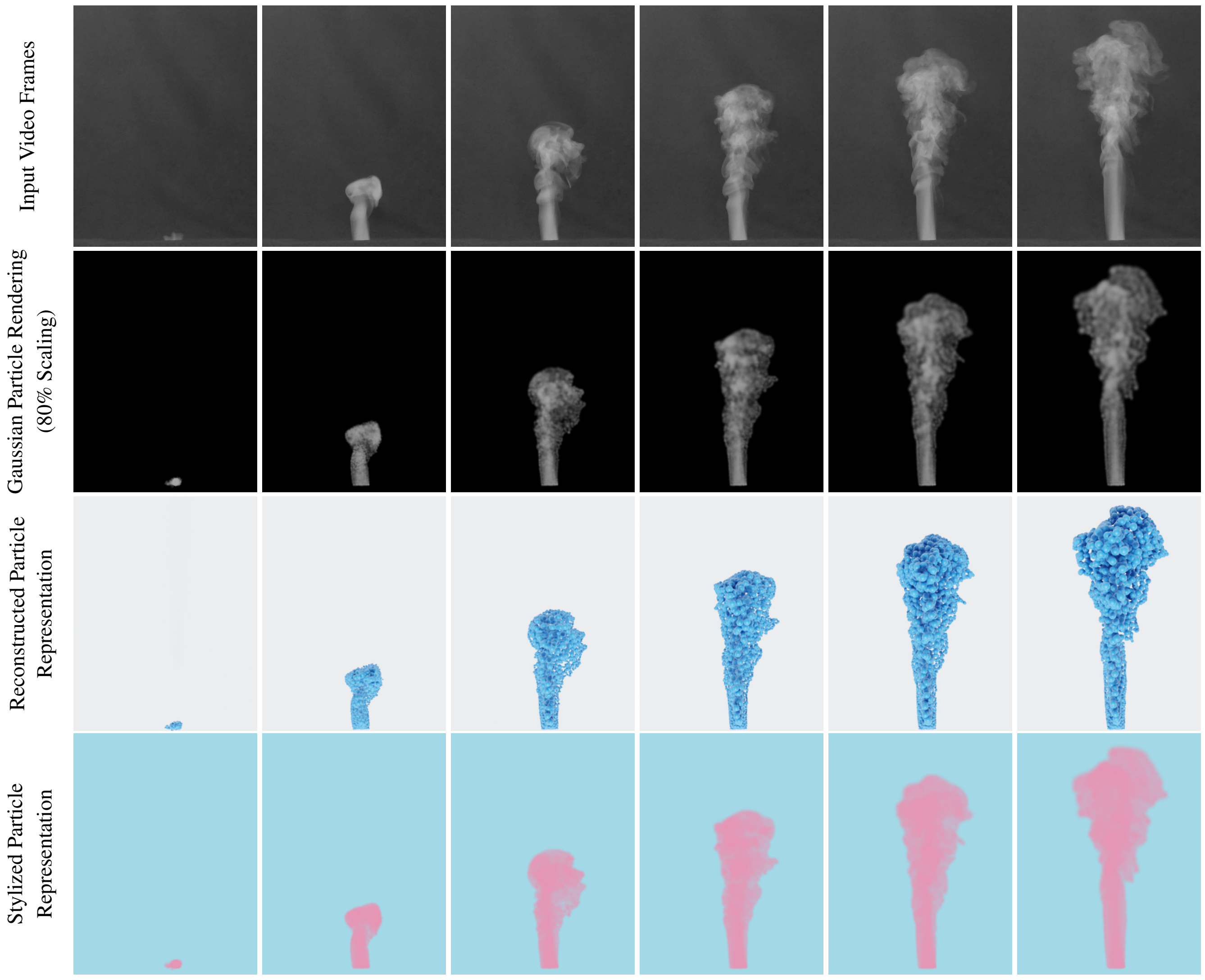
Input Video Frames
Gaussian Particle Rendering (80% Scaling)
Reconstructed Particle Representation
Stylized Particle Representation

Figure 6: Visual results of GaussFluids on the ScalarFlow dataset. The first row shows the input video frames. The second row presents Gaussian Particles extracted by GaussFluids, rendered with an 80% scaling factor. The third row visualizes the reconstructed particle representations, while the fourth row demonstrates the application of a novel style to the extracted Gaussian Particles. These results highlight GaussFluids' capability to effectively handle smoke-like fluid phenomena and enable subsequent fluid style transfer tasks.

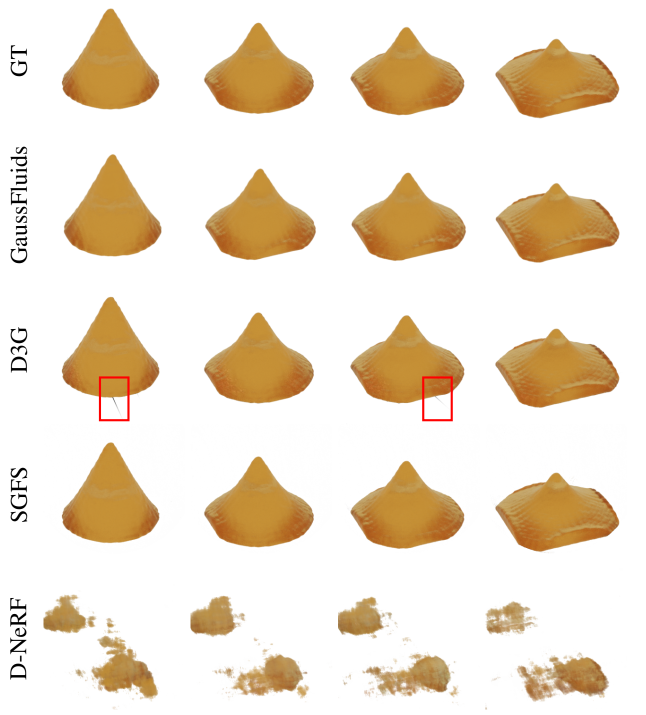

Figure 7: Visual reconstruction effects. Renderings from novel viewpoints of NeRF-type methods are often unsatisfactory. The D3G and SGFS methods, although able to generate alternatives for new viewpoints, face occlusion issues caused by surface reconstruction, highlighted by the red boxes. In contrast, GaussFluids effectively handles such challenges and delivers high-quality results.

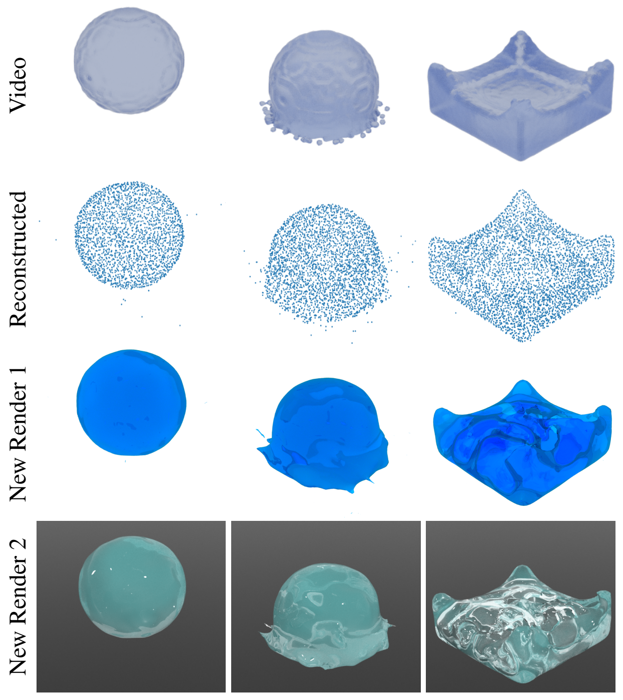

Figure 8: Style Transfer Demonstration. Using the WaterSphere scene as an example, we assign new materials to the reconstructed particles to achieve various stylized rendering effects.

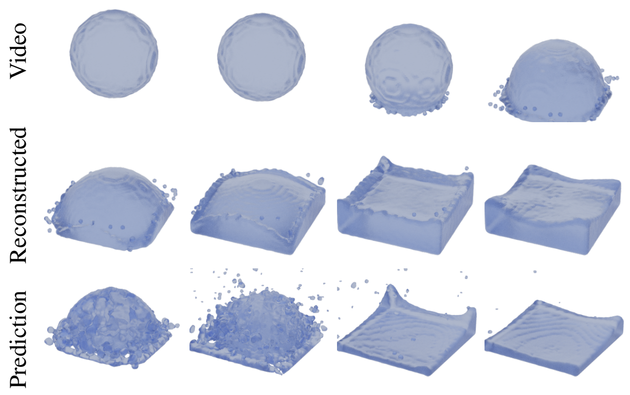

Figure 9: Fluid Prediction. The first row shows the original video frames, while the second row presents the reconstructed particles. The third row displays the simulated results using PBF, demonstrating its ability to predict fluid motion.

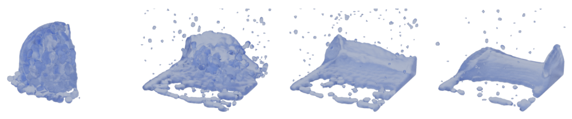

Figure 10: Fluid editing demonstration. Half of the WaterSphere particles are removed, and their motion is predicted.