A Real-Time and Interactive Fluid Modeling System for Mixed Reality

Yunchi Cen*, Hanchen Deng*, Yue Ma, and Xiaohui Liang

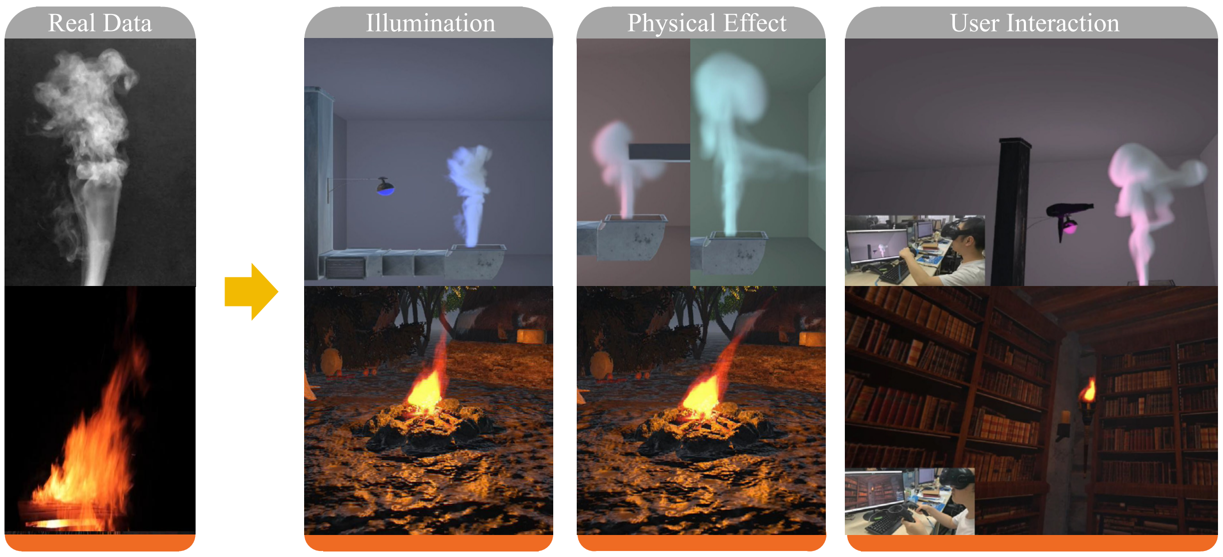


Fig. 1: We developed a real-time interactive fluid modeling system for mixed reality environments. Our system performs fluid reconstruction from input images, while concurrently considering the influence of environmental factors, such as wind, obstacles, and user interactions, during the reconstruction process. Our system is developed on the Unity platform and enables easy deployment to various head-mounted displays.

Abstract—Within the realm of mixed reality, the capability to dynamically render environmental effects with high realism plays a crucial role in amplifying user engagement and interaction. Fluid dynamics, in particular, stand out as essential elements for crafting immersive virtual settings. This includes the simulation of phenomena like smoke, fire, and clouds, which are instrumental in enriching the virtual experience. This work showcases a cutting-edge system developed to produce dynamic and interactive fluid effects that mirror real captured data in real-time for mixed reality applications. This innovative system seamlessly incorporates fluid reconstruction alongside velocity estimation processes within the Unity engine environment. Our approach leverages a novel physics-based differentiable rendering technique, grounded in the principles of light transport in participating media, to simulate the intricate behaviors of fluid while ensuring high fidelity in visual appearance. To further enhance realism, we have expanded our framework to include the estimation of velocity fields, addressing the critical need for fluid motion simulation. The practical application of these techniques demonstrates the system's capacity to offer a robust platform for fluid modeling in mixed reality environments. Through extensive evaluations, we illustrate the effectiveness of our approach in various scenes, underscoring its potential to transform mixed reality content creation by providing developers with the tools to incorporate highly realistic and interactive fluid seamlessly.

Index Terms—fluid modeling, mixed reality, differentiable rendering

1 Introduction

Mixed Reality (MR) has emerged as a transformative technology, bridging the gap between the digital and physical worlds across diverse sectors, including gaming [50], education [58], healthcare [43, 68], and industrial design [38, 69]. At the heart of immersive MR experiences lies the dynamic and realistic rendering of environmental effects, which play a crucial role in enhancing user immersion and interaction. Among these environmental effects, fluid simulation stands out as a pivotal element for creating compelling and authentic virtual environments.

However, generating realistic fluid in MR presents unique challenges. Traditional fluid reconstruction methods [10, 12, 17] often grapple with balancing computational efficiency with the need for physical accuracy and real-time performance. These challenges are compounded by the complexity of capturing the intricate behaviors of fluid, including its fluid dynamics and interaction within MR environments. Existing approaches to fluid modeling have been limited by these factors, often resulting in either overly simplified representations that lack realism or computationally intensive solutions that are not feasible for real-time applications.

To address the above challenges, we propose an integrated system designed to efficiently generate dynamic and interactive fluid from images for MR environments. Our approach is centered around a novel

- *∗ denotes that these authors contributed equally to this work.*
- *Yunchi Cen, Hanchen Deng, and Yue Ma are with the State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China.*
- *Xiaohui Liang (corresponding author) is with the State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China, and also with Zhongguancun Laboratory, Beijing, China (e-mail: liang_xiaohui@buaa.edu.cn).*

Received 14 March 2024; revised 17 June 2024; accepted 1 July 2024.
Date of publication 9 September 2024; date of current version 4 October 2024.
This article has supplementary downloadable material available at https://doi.org/10.1109/TVCG.2024.3456140, provided by the authors.
Digital Object Identifier no. 10.1109/TVCG.2024.3456140

physics-based differentiable renderer that enables efficient fluid reconstruction while maintaining high visual fidelity. To further tackle the essential demand for fluid interactivity, our system has been extended to the estimation of velocity fields. Estimating the fluid's velocity fields enables our simulated fluid to interact dynamically with the MR environment and the interaction of users. This capability introduces a new level of interactivity, where fluid responds to changes in the environment and user actions, further blurring the lines between virtual and physical realms.

The culmination of our research efforts is the development and integration of fluid reconstruction and velocity estimation into a comprehensive system based on the Unity engine. Our system provides a robust platform for generating dynamic and interactive fluid in real-time from images, showcasing the practical application of our novel approach. Through extensive evaluations conducted in various scenes, we demonstrate the effectiveness of our method in fluid motion generation for the MR environment. Our approach provides developers with the tools to integrate highly realistic and interactive fluid, thereby contributing positively to the evolution of MR content generations.

To summarize, the main contributions of our work are:

- We introduce a comprehensive system seamlessly integrated into the Unity engine, specifically engineered to create realistic fluid for mixed reality applications.
- Our approach significantly enhances the efficiency of fluid reconstruction, enabling real-time performance without compromising on visual quality. We emphasize the system's capacity for interactive user experiences, where fluid behaviors respond intuitively to user's various actions.

2 Related Work

The advent of MR technology has marked a pivotal shift in how digital information is integrated with the physical world, offering immersive experiences that blend virtual and real environments [31, 51]. This integration has found applications across various sectors, including gaming [28,56], education [58], healthcare [43], and industrial design [18], revolutionizing user interaction and engagement within these domains. It shows that MR technology has the potential to bridge the gap between human perception and the virtual world. Simondon [53] provides a philosophical insight into the connection between humans and technology. His emphasis on recognizing the human reality within technical objects resonates with our discussion on designing engaging virtual worlds with MR technology.

At the core of creating immersive MR experiences is the realistic rendering of environmental effects, which significantly enhances the sense of presence and interaction for users [54]. Among these environmental effects, the fluid plays an essential role due to its widespread application in creating compelling virtual environments. Historically, the modeling of fluid phenomena has been a topic of interest within the fields of Computational Fluid Dynamics (CFD) and computer graphics [4, 10, 12, 17, 34, 60]. The complex interplay of physical laws governing fluid motion and the intricate visual appearance of fluid presents unique challenges in accurately modeling these effects in a computationally efficient manner. As MR technologies continue to evolve, the demand for dynamic and realistic fluid that can be generated and manipulated in real-time has increased, highlighting the need for innovative approaches to fluid modeling [14,25,45,48,55,61,66]. Here we review the state of the art of fluid reconstruction, velocity field estimation, and differentiable rendering. These pivotal technologies form the foundation of our system.

2.1 Fluid Reconstruction from Visual Observations

The reconstruction of fluid density from visual observations, particularly 2D images or videos, has been an active area of research in computer graphics. Computed Tomography (CT) methods, originally developed for medical imaging applications [35], have been widely adopted for this purpose, such as reconstructing fire [32] and smoke [33] from multiple input images. These methods employed techniques like minimizing least-squares energy using conjugate gradient methods and adaptive grids to improve reconstruction efficiency. Atcheson et al. [3] captured 3D gas flows using time-resolved schlieren tomography. Subsequent efforts aimed to improve the quality of sparse tomographic reconstructions by utilizing stochastic approaches [24] or incorporating appearance transfer from additional views [47].

While multi-view approaches offered some success, their practical application was hindered by high computational demands and complex calibration needs, making real-time MR applications challenging. Eckert et al.'s [12] single-view reconstruction method for plumes addressed these by integrating physics-based fluid dynamics priors. This approach improved efficiency but faced limitations in meeting the real-time performance requirements of MR environments due to its computational intensity. Thapa et al. [59] introduces a learning-based single-image approach for 3D fluid surface reconstruction, leveraging a deep neural network to estimate depth and normal maps by analyzing refractive distortion. Zhang et al. [67] combine a differentiable fluid simulator with a differentiable renderer, utilizing global physical priors across long series to reconstruct fluid dynamics from sparse view images for Virtual Reality (VR) / Augmented Reality (AR) applications.

Our approach focuses on reconstructing dynamic and interactive fluid, particularly tailored for the MR environment. It diverges from traditional multi-view reconstructions focused on exhaustive volumetric precision, opting instead for a single-view approach. This strategic choice strikes a balance between efficiency and the interactive fidelity of fluid simulations.

2.2 Velocity Field Estimation

Estimating the velocity field of fluids from captured videos is more challenging than reconstructing the volumetric density, as motion is indirectly observed by tracing temporal changes in density. There are two main approaches for estimating fluid velocity fields: tracer-based and tracer-free methods. Tracer-based methods introduce tracers (such as particles, dye, etc.) into the fluid, allowing the fluid velocity to be retrieved by tracking these tracers. Particle Image Velocimetry (PIV) and its variants, such as tomographic PIV [13], synthetic aperture PIV [5], structured-light PIV [1,62,63], and plenoptic PIV [15,57], are widely used across various fields to characterize fluid flows. However, methods like [13, 63] require carefully chosen particles, specialized hardware, and complex camera setups. Tracer-free approaches, such as Background Oriented Schlieren tomography (BOS) [2, 22], utilize the phase change due to refractive index differences in the fluid to track the flow.

An alternative approach for fluid motion reconstruction is visible light-capturing. Optical flow [29], which assumes brightness constancy and smoothness in the flow vector, has been widely used for fluid motion estimation and provides a solid foundation for velocity calculations. Liu and Shen [40] investigated the connection between optical flow and fluid flow, revealing that under certain conditions, brightness constancy is equivalent to the scalar transport equation for fluid flow, offering a physical meaning to optical flow approaches for fluid tracking.

Since the optical flow problem is physically connected to the continuity equation in fluid dynamics, it becomes feasible to introduce Navier-Stokes equations, which govern real-world fluid motions, as additional physical priors to the conventional Horn-Schunck algorithm [29]. Corpetti et al. [11] used optical flow with an additional divergence-free and curl smoothness prior to regularize 2D cloud motions. Similarly, some previous works have incorporated divergence-free constraints [27, 52, 64], although most of them suffer from the complexity in solving higher-order regularization terms. Gregson et al. [23] simplified this issue by connecting the pressure projection method with the proximal operator, allowing it to be easily handled by a convex optimization framework.

Recent works [12,17] have coupled fluid density updates and velocity calculations to improve the temporal coherence of flow reconstruction. To enhance reconstruction efficiency, our method simplifies the physical constraints in velocity field estimation compared to Eckert et al.'s [12] approach. Specifically, by leveraging differentiable rendering, we estimate the velocity field based on changes in density between consecutive frames, streamlining the process. Our method for velocity estimation aims to enhance both efficiency and interactive capabilities, which are essential for real-time MR applications, enabling dynamic fluid behavior that responds naturally to user interactions and environmental changes.

2.3 Differentiable Rendering

Differentiable rendering enables the calculation and propagation of derivatives of scene parameters through images, making it capable of handling gradient-based optimization problems, such as optimization of geometry, materials, and lighting parameters. Loper and Black [41] proposed the first differentiable renderer called *OpenDR*, which approximates the backward pass of the traditional mesh-based graphics pipeline and has inspired several follow-up works [9, 19, 26, 36, 39]. Kato et al. [36] introduced an approximate gradient for rendering a mesh, named *Neural Renderer*, which enables the integration of rendering into neural networks.

In the realm of physically-based differentiable rendering for fluid, early efforts focused on inverse transport simulation methods [20, 21] to compute derivatives of volumetric physical parameters. Zhang et al. [65] introduced differential theory for radiative transfer, combined with unbiased Monte Carlo estimation to solve derivatives for surface models and volumetric data. However, the efficiency of physically based differentiable rendering, especially in scenes with participating media, remains low. Nimier et al. [46] proposed the Radiative Backpropagation (RB) approach, treating differentiable rendering as a problem of light's backward propagation, enhancing efficiency without storing data along sampling paths, despite challenges in computational cost due to quadratic sampling time increases with scattering events. Cen et al. [7] proposed the differential diffusion theory, employing a numerical method to compute the radiance derivatives of multiple scattering with high efficiency. However, their approach still fails to meet the efficiency requirements for interactive applications.

In recent years, the rapid advancement of 3D reconstruction technology has led to widespread interest in Neural Radiance Fields [44] (NeRF) in the fields of computer vision and graphics. It achieves photorealistic image synthesis by learning continuous volumetric scene functions. Pumarola et al. [49] designed a novel NeRF network for dynamically deforming objects, not only breaking the limitation of NeRF being applicable only to surface model but also demonstrating excellent performance in the realistic representation of smoke. Feng et al. [16] proposed a fluid reconstruction method based on 3D Gaussian Splatting (GS), which aligns GS with the underlying consistency representation of Position-based Dynamics (PBD), achieving realistic fluid reconstruction while also eliminating the impact of floating noise.

In contrast to existing differentiable rendering techniques that rely on approximations of traditional graphics pipelines or learned representations, our proposed differentiable renderer is grounded in a physically-based single-scattering model, leveraging the principles of radiative transfer while avoiding the computational complexity associated with higher-order scattering events. As a physics-based differentiable rendering method, our approach does not require training on large datasets. By exploiting the single-scattering approximation, we achieve a lightweight yet accurate differentiable renderer that can efficiently compute derivatives of fluid parameters, enabling gradient-based optimization for fluid reconstruction without relying on extensive data.

3 SYSTEM IMPLEMENTATION

Our proposed system for physically-based interactive fluid reconstruction in MR is implemented within the Unity engine, ensuring compatibility with various MR applications. The system integrates three key components: Density Reconstruction, Fluid Simulation, and Display and Interaction. The framework of our system is illustrated in Fig. 2.

The **Density Reconstruction** component serves as the foundation, providing the reconstructed density fields necessary for the Fluid Simulation component. We have developed a novel differentiable renderer based on a single scattering model, which leverages the advantages of Unity's rendering pipeline. This design choice allows for efficient and accurate reconstruction of the fluid's density field from captured images while seamlessly integrating with the Unity engine. The reconstructed density fields are then passed to the **Fluid Simulatio**n component for velocity field estimation and physical simulation. To achieve real-time performance, we have developed a lightweight physical simulator that accurately simulates the fluid's motion based on the estimated velocity and reconstructed density fields. This component estimates the velocity field by analyzing changes in density between consecutive frames, employing a modified transport constraint for computational efficiency. This approach ensures realistic and physically plausible fluid dynamics while maintaining interactive frame rates. Finally, the **Display and Interaction** component renders the fluid on the Head-mounted Display (HMD), providing users with an immersive visual experience. By leveraging Unity's built-in rendering capabilities and optimizing the rendering pipeline, we achieve high-quality, real-time rendering of the fluid on the HMD. Additionally, this component processes user interactions captured by the HMD's controllers, enabling natural and realistic interactivity with the virtual fluid. The tight integration between the Display and Interaction component and the Fluid Simulation component ensures that user interactions are immediately reflected in the fluid's behavior, providing a seamless and immersive experience.

Each component will be detailed in the following, highlighting their implementation details within the Unity engine and the techniques employed to deliver realistic, dynamic, and interactive fluid effects using the HMD for display and interaction.

3.1 Density Reconstruction

The Density Reconstruction component enables the reconstruction of the fluid density field from reference images by leveraging the differential form of the rendering equation based on the single scattering model. In contrast to existing physically-based differentiable rendering methods [7, 65] that rely on Monte Carlo estimation and Automatic Differentiation (AD) to compute the scene parameter derivatives, our proposed differentiable renderer utilizes the differential form of the rendering equation to calculate the derivatives using the RB approach. Furthermore, the single scattering model simplifies the complexity of differentiable rendering, and our differentiable renderer is built on a ray marching process. Our method avoids the traditional differential volumetric path-tracing process, which requires considerable computation. Differentiable rendering involves both forward and backward rendering processes. The forward rendering process is responsible for generating images, while the backward rendering process is responsible for computing the density derivatives using the RB approach. Obtaining the density derivatives enables the optimization of the fluid density field using gradient descent methods.

3.1.1 Forward Rendering

In the context of fluid rendering, the single scattering model assumes that light interacts with the participating medium only once before reaching the camera. This simplification is suitable for optically thin media and provides a good approximation of the general appearance of the fluid. The rendering equation for the single scattering model consists of three main components: the reduced incident radiance, the single scattering radiance, and the emission radiance. The reduced incident radiance represents the attenuated ambient light that directly reaches the viewpoint, while the single scattering radiance accounts for the light that undergoes a single scattering event within the participating medium before reaching the viewpoint. The emission radiance represents the light emitted by the participating medium itself. Mathematically, the rendering equation for the single scattering model can be expressed as:

$$L_{out}(x,\omega) = L_{ri}(x,\omega) + L_{ss}(x,\omega) + L_{em}(x,\omega). \tag{1}$$

In the equation, $L_{out}(x,\omega)$ represents the total outgoing radiance at position x in the direction ω, $L_{ri}(x,\omega)$ denotes the reduced incident radiance, $L_{ss}(x,\omega)$ signifies the single scattering radiance, and $L_{em}(x,\omega)$ is the emission radiance. For the detailed formulations of these three terms, please refer to the previous works [7, 65].

Our approach employs traditional ray marching to render the fluid volume. The ray is uniformly sampled with respect to the distance

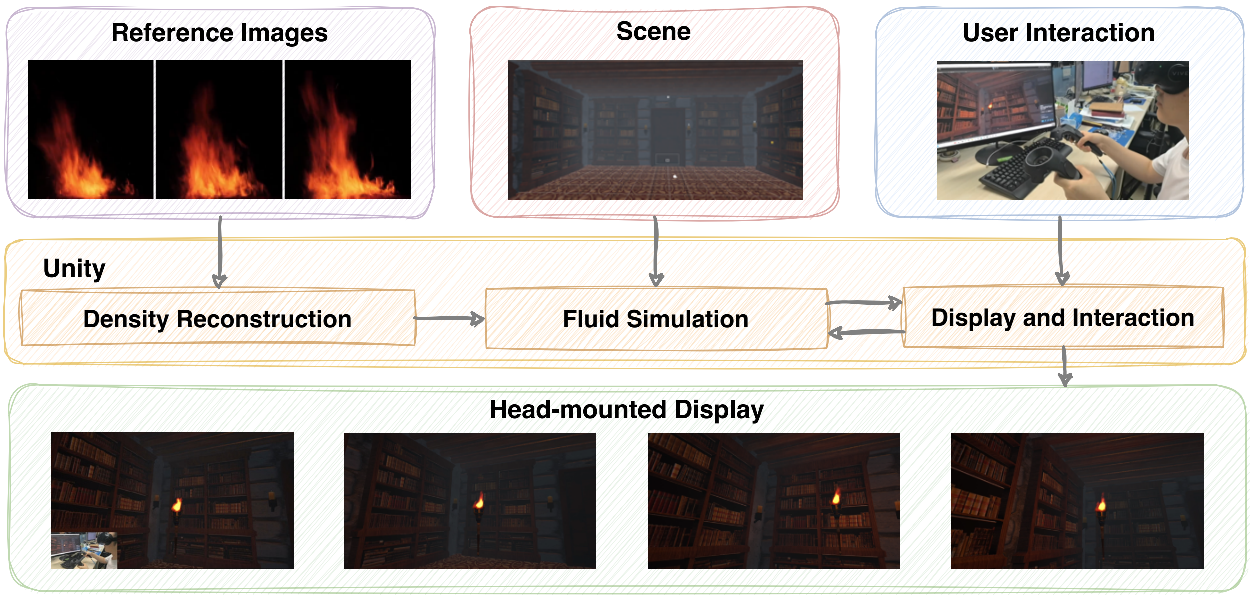


Fig. 2: Our framework comprises three main components: Density Reconstruction, Fluid Simulation, and Display and Interaction. The Density Reconstruction component efficiently reconstructs the fluid's density field using inputted target images. The Fluid Simulation component estimates the next step's velocity and density fields based on the previous two consecutive density fields, taking into account the effects of the virtual environment and user interactions. The Display and Interaction component renders the simulated fluid on the head-mounted display and updates the velocity field according to user interactions, enabling real-time interactivity and immersion within the virtual scene.

along its path. To compute the single scattering radiance at the sampling position, important sampling is utilized to estimate the radiance contributions from the ambient light.

3.1.2 Backward Rendering

The backward rendering process is responsible for computing the derivatives of the outgoing radiance with respect to the fluid density field using the RB approach [46]. By differentiating the rendering equation (Equation 1), we obtain the differential form:

$$\dot{L}_{out}(x,\omega) = \dot{L}_{ri}(x,\omega) + \dot{L}_{ss}(x,\omega) + \dot{L}_{em}(x,\omega). \tag{2}$$

Since our work focuses on fluid density field reconstruction, the derivative $\dot{L}_{out}$ can be simplified to $\dot{L}_{out} = \frac{\partial L_{out}}{\partial \rho}$, where ρ denotes the fluid density field. The derivative formulas for each term in Equation 2 can be derived by applying the chain rule and the product rule, as shown in previous works [7, 65].

Our approach employs the Radiative Backpropagation (RB) method [46] for the backward rendering process. Similar to the forward rendering described in Section 3.1.1, a traditional ray marching technique is used to compute the derivative of the outgoing radiance with respect to the density field using the differential analysis formulas derived from Equation 2. The derivatives are backpropagated from the camera to the scene, following the reverse path of the forward rendering.

We utilize a volume data structure with the same dimensions as the density field to accumulate the calculated derivatives according to the voxel position of the current sampling density field. At each sampling position along the ray, the computed derivatives are added to the corresponding voxel in the volume data structure. This process is repeated for all pixels in the image plane, resulting in a volumetric representation of the derivatives of the outgoing radiance with respect to the density field.

Having obtained the volumetric derivatives of the density field, we can now use this information to optimize the density field, as described in Section 3.1.3. The derivatives provide the necessary gradient information for gradient-based optimization algorithms to update the density field iteratively, minimizing the discrepancy between rendered and reference images.

In contrast to previous works that rely on automatic differentiation and Monte Carlo estimation [46, 65] to compute derivatives, which can be inefficient, our method offers several advantages. First, we use analytic formulas to compute the derivatives, avoiding the need for automatic differentiation. Second, we employ the RB method [46] to compute the derivatives. The RB method treats the backward rendering process as a rendering, similar to rendering derivatives. This approach eliminates the need to record the derivatives at sampling positions along the ray path, reducing memory overhead and improving performance.

3.1.3 Density Field Optimization

Having obtained the volumetric derivatives of the density field through the backward rendering process, we can now optimize the density field to minimize the discrepancy between the rendered and reference images. In our approach, instead of reconstructing a complete fluid volume (e.g., $64\times96\times64$) from a single view, we propose to use a compressed "slab-like" density field (e.g., $64\times96\times16$) to approximate the fluid body. This compressed density field can be conceptualized as a result of capturing the fluid volume from a specific viewpoint, while only retaining the fluid information within a certain depth (e.g., 16 layers) along the shooting direction and discarding the rest.

The choice of the dimensions for the compressed density field depends on the specific requirements of the application and the desired balance between reconstruction quality and performance. Using a smaller depth dimension compared to the original fluid volume can significantly reduce the data size, leading to faster reconstruction speeds and lower storage requirements.

Although this compression process inevitably loses some 3D information, the compressed density field provides a visually similar approximation to the complete fluid volume when observed from the specific viewpoint. To further mitigate the visual artifacts caused by this compression, we employ the Billboard technique for rendering. This technique ensures that the density field slab always faces the observer, thus avoiding an unnatural appearance when viewed from the side.

The optimization process of the density field is performed using gradient descent methods, such as Adam [37]. The objective function for optimization is defined as the L2 loss between the rendered image and the reference image:

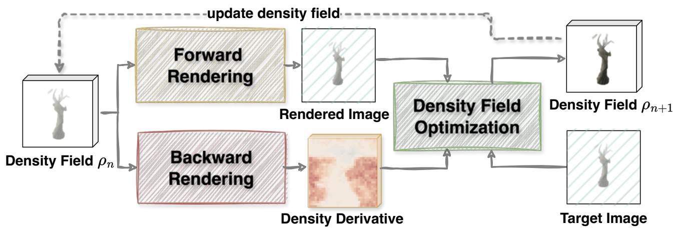


Fig. 3: Workflow of the Density Reconstruction Component, where ρ_n denotes the density field in the n^{th} iteration of the optimization process. The solid lines indicate the forward flow of data, and the dotted line indicates the optimized density field being used as the input for the next iteration.

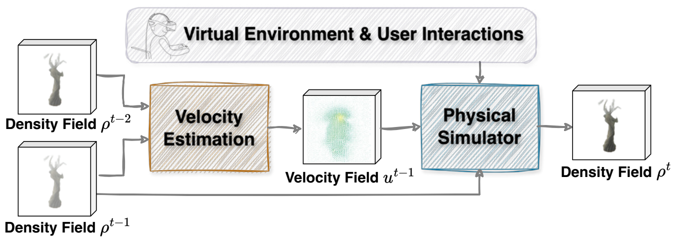

Virtual Environment & User Interactions
Density Field ρ^{t-2}
Velocity Estimation
Velocity Field u^{t-1}
Physical Simulator
Density Field ρ^{t}
Density Field ρ^{t-1}

Fig. 4: Workflow of the Fluid Simulation component. The Velocity Estimator takes the reconstructed density fields from the previous two-time steps as input and estimates the velocity field by solving a modified optimization problem. The estimated velocity field is then passed to the Physical Simulator, which simulates the fluid's motion for the current time step while considering any external forces applied by the user interactions. The simulated fluid motion is then used to update the density field for the current time step, which serves as input for the next iteration of the Velocity Estimator. This iterative process continues, allowing for the creation of dynamic and realistic fluid motion in real time that is responsive to user interactions.

$$\mathscr{L} = \frac{1}{2}\sum_{i=1}^{N}(I_i - \hat{I}_i)^2, \tag{3}$$

where I_i and $\hat{I}_i$ represent the pixel values of the rendered and the reference images, respectively, and N is the total number of pixels.

By minimizing the loss function in Equation 3, the density field is iteratively updated using the computed volumetric derivatives. The optimization process is performed for a fixed number of iterations or until convergence is reached, resulting in a reconstructed density field that closely matches the reference image.

In summary, the "slab-like" density field reduces computational overhead and memory requirements, enabling real-time reconstruction and rendering within the Unity engine. The Billboard rendering technique ensures that the fluid appears visually plausible from the user's perspective. By integrating the forward rendering, backward rendering, and the density field optimization process into the Density Reconstruction component, we achieve a balance between reconstruction quality and performance, allowing the Fluid Simulation component to support interactive engagement with the virtual environment and respond to the user's actions. The workflow of the Density Reconstruction is shown in Figure 3.

3.2 Fluid Simulation

The Fluid Simulation component estimates the velocity field and simulates the fluid's motion based on the reconstructed density fields provided by the Density Reconstruction component. This component comprises a velocity estimator and a physical simulator, which is essential for creating dynamic and physically plausible fluid dynamics while maintaining interactive frame rates. The workflow of the Fluid Simulation component is shown in Figure 4.

3.2.1 Velocity Estimation

To estimate the velocity field, we applied a coupled density and velocity estimation method that leverages the advantages of the Unity engine. Our approach is inspired by the work of Eckert et al. [12] but introduces a simplified strategy to enable interactive fluid motion reconstruction. While Eckert et al.'s approach provides a solid foundation for coupled density and velocity estimation, their method is computationally expensive, which is unsuitable for interactive applications. Our method addresses this limitation by introducing the modified constraints below:

$$\min_{\mathbf{u}^{t-1}} f\left(\mathbf{u}^{t-1}\right) = \left|\frac{\rho^{t-1}-\rho^{t-2}}{\Delta t} + \mathbf{u}^{t-1}\cdot\nabla\rho^{t-1}\right|^2 + E_{smooth}(\mathbf{u}^{t-1}) \tag{4}$$

where ρ^{t-1} and ρ^{t-2} indicate the previous two-time step density field. $\mathbf{u}^{t-1}$ denotes the velocity field of $t-1$ step. The first term in the objective function represents the optical flow constraint, which ensures that the estimated velocity field is consistent with the observed changes in the density field over time. The second term is the smoothness regularization term, $E_{smooth}(\mathbf{u}^{t-1})$, which is designed to penalize large variations in the velocity field. This is achieved using a carefully chosen regularization function that can be efficiently solved using proximal operators. The smoothness regularization term is given by:

$$E_{smooth}(\mathbf{u}^{t-1}) = \alpha\left\|\nabla\mathbf{u}^{t-1}\right\|^2 \tag{5}$$

where α is the weight of the smoothness regularization term. It penalizes large changes in the velocity field by minimizing the square of the L2-norm of $\nabla\mathbf{u}^{t-1}$.

To solve this optimization problem efficiently, we applied the Fast Primal-dual method (Fast-PD) for convex optimization [8]. Our solution strategy leverages the sparse structure and introduces a tailored set of proximal operators that can be efficiently computed on the GPU. By carefully designing these proximal operators and the overall optimization workflow, we can achieve significant performance improvements compared to Eckert et al.'s approach, enabling real-time and interactive fluid motion reconstruction.

Once the velocity field $\mathbf{u}^{t-1}$ is estimated, we predict the velocity field $\mathbf{u}^{t}$ and density field ρ^{t} for the current time step t by advecting the velocity and density fields from the previous time step using our lightweight physical simulator.

3.2.2 Physical Simulator

To achieve real-time performance, we have developed a lightweight physical simulator that accurately simulates the fluid's motion based on the estimated velocity and reconstructed density fields. Due to the Density Reconstruction component and the Velocity Estimation module being developed by *Python*, in order to reduce the latency caused by the communication between Unity and *Python* processes, we integrated all the modules and implemented them by Unity's native *Compute Shader*.

We initially wrote the *Vulkan* backend code in the *Taichi* programming language [30], then exported it as *spv* files using the *Ahead of Time* compilation mode, and finally converted it to *HLSL* (High-Level Shading Language) format using *SpirV-Cross* to run in Unity. Parts of the source code are available on GitHub at `https://github.com/cenyc/Interactive-Fluid-Modeling`, which includes the *Taichi* program and *HLSL* shaders.

In terms of the physical simulator, we implemented a classic Eulerian approach [6] to simulate fluid dynamics. The simulator operates with a time step of 0.04 and performs 60 iterations per frame using the Gauss-Seidel iteration method. The Fluid Simulation component seamlessly integrates with the Density Reconstruction component, receiving the reconstructed density fields as input and providing the simulated fluid motion as output. This tight integration between the Density Reconstruction and Fluid Simulation components ensures that the user's interactions with the virtual fluid are immediately reflected in the fluid's behavior, resulting in a highly responsive and immersive experience.

Table 1: System performance on different scenes evaluated with average time per 100 iterations.

Sence	Reconstruction (ms)	Simulation (ms)	HDM Rendering (ms)	Total Time (ms / FPS)	Memory Usage (MB)
Chimney	7.20	15.41	0.38	22.99 / 43.5	1723
Bonfire	12.88	16.79	0.42	30.09 / 33.2	1893
Fireplace	16.04	16.71	0.40	33.15 / 30.2	1868

3.3 Display and Interaction

The Display and Interaction component is responsible for rendering the simulated fluid and handling user interactions. This component ensures that the fluid is seamlessly integrated into the mixed reality environment and responds to user interactions in real-time. The main challenges addressed by this component are: (1) updating the fluid's velocity field based on user interactions and environmental factors, and (2) efficiently rendering the fluid on the head-mounted display while maintaining a high frame rate.

3.3.1 User Interaction and Velocity Field Update

In our mixed reality fluid modeling system, user interaction is handled through a combination of the HTC VIVE and Unity engine. Users can interact with the virtual fluid using virtual objects or tools, such as a ball or a hair dryer, which are tracked using the HTC VIVE's controllers. These interactions allow users to manipulate the fluid's motion, apply external forces, and create dynamic effects in real time.

To incorporate user interactions into the fluid simulation, we employed an interaction handling mechanism that seamlessly updates the fluid's velocity field based on user input. When a user interacts with the virtual fluid, our system captures the interaction data, such as the position, velocity, and force of the input, and translates it into a corresponding external force field. During each step of the simulation, this externally applied force $\mathbf{F}$, which is derived from user interactions, is incorporated into the velocity field update. The equation used to update the velocity field $\mathbf{u}^*$ at each step is given by:

$$\mathbf{u}^* = \mathbf{u}^{t-1} + \Delta t(\mathbf{u}^{t-1} \cdot \nabla \mathbf{u}^{t-1} + \mathbf{F}) \tag{6}$$

where $\mathbf{u}^{t-1}$ represents the fluid's velocity field from the previous time step, and Δt, the simulation time step, is set to 0.04. The term $\mathbf{u}^{t-1} \cdot \nabla \mathbf{u}^{t-1}$ captures the fluid's convective effects, where the velocity field interacts with its own spatial gradients. The external force $\mathbf{F}$ directly adds to this dynamic, simulating the influence of real-time interactions on the fluid's movement. This formulation ensures that the simulation not only reflects the intrinsic properties of the fluid but also responsively adapts to external influences, thereby enhancing the realism and interactivity of the simulation environment.

Finally, the updated velocity field $\mathbf{u}^*$ is then passed to the Physical Simulator, which performs a pressure projection step to enforce incompressibility and account for user-induced forces.

3.3.2 Fluid Rendering on Head-Mounted Display

To achieve efficient and visually convinced fluid rendering on head-mounted displays within the Unity engine, our method integrates a ray marching algorithm grounded in a single scattering model with the Billboard algorithm. Below, we present the technical specifics of both the ray marching and Billboard algorithms.

Ray Marching with Single Scattering Model We employ a ray marching algorithm that traverses the volumetric fluid data, represented as a 3D grid of density values stored in Unity's 3D texture format. The algorithm accumulates density along each ray using Unity's compute shaders, which leverage GPU parallel processing for real-time performance. The rendering equation considers reduced incident radiance L_{ri}, single scattering radiance L_{ss}, and emission radiance L_{em}. At each sampling point, the shader calculates the contributions of these components based on local density and lighting conditions, accumulating them along the ray to determine the final color and opacity of the fluid. The importance sampling technique is employed in the computation of single scattering radiance, enhancing the efficiency of illumination estimation across the entire solid angle.

Billboard Algorithm To maintain a consistent fluid appearance from different viewpoints, we utilize the Billboard algorithm. Implemented using a custom Unity shader, the algorithm transforms the vertices of the fluid's bounding geometry to align with the camera's view direction, ensuring that the fluid always faces the viewer. The shader considers the camera's position and orientation to calculate the correct fluid orientation. Unity's depth testing and alpha blending capabilities are used to properly integrate the fluid with the virtual environment, ensuring accurate occlusion and seamless blending with surrounding objects.

The combination of the ray marching algorithm based on the single scattering model and the Billboard algorithm enables visually convinced rendering of fluids on HMD while maintaining real-time performance.

4 System Evaluation

We conducted comprehensive evaluations to assess our system from three key aspects: (1) the performance validation of our system, (2) the visual evaluation of the reconstructed fluid and the realism of the interactions, and (3) the user study to subjectively assess the visual and interactive authenticity of our results. We used three different scenes in our experiments: Chimney, Bonfire, and Fireplace, each with a volumetric data resolution of $64 \times 96 \times 16$. The target images for smoke reconstruction were derived from the ScalarFlow [42] dataset. The size of the smoke and flame target images are 600×1062 and 512×512, respectively. Our experiments were conducted on a high-performance workstation consisting of two Intel Xeon E5-2620 V4 processors with 128 GB of memory and an NVIDIA GeForce RTX 2080 Ti graphics card.

4.1 Performance Validation

We measured the computation time and resource utilization of each component in our pipeline across three scenes: Chimney, Bonfire, and Fireplace. As shown in Table 1, our system achieves real-time performance with interactive frame rates ranging from 30.2 to 43.5 FPS, while efficiently managing memory usage.

The Fluid Reconstruction component demonstrates efficient performance, with computation times ranging from 7.20 ms to 16.04 ms. The Fluid Simulation component maintains consistent computation times across scenes, ensuring stable real-time performance. The HDM Rendering component achieves low computation times, contributing minimally to the overall performance overhead. The total computation time, including all components, varies from 22.99 ms to 33.15 ms, depending on the scene complexity. Our system efficiently manages memory resources, consuming between 1723 MB and 1893 MB across the tested scenes.

Table 2: Performance comparison of differentiable rendering methods.

Method	Computation Time (ms)	Memory Usage (MB)
Ours	**8.51**	**423**
Cen et al. [7]	78.56	1823
RB [46]	5249.31	1,274

Differentiable rendering is one of the most critical techniques for our system. To demonstrate the high performance of our differentiable rendering method, we conducted a comparison experiment with Cen

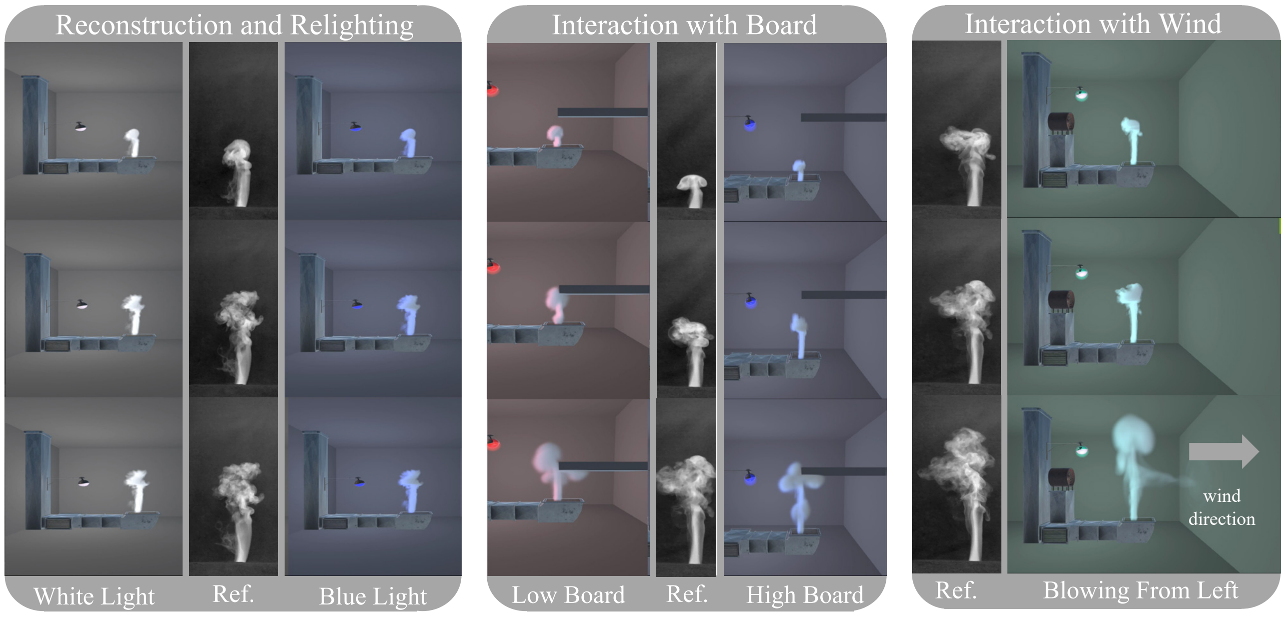


Fig. 5: This figure showcases the dynamic smoke motions reconstructed from real captured images and rendering in the Chimney scene. The left column demonstrates the relighting of the reconstructed fluid sequences under distinct colors of a light source, highlighting the adaptability of our reconstructed fluid applied to different lighting conditions. Ref. means the reference images. The middle and right columns present experiments investigating the fluid's behavior when interacting with virtual objects and user-induced wind forces, respectively, validating the realism and responsiveness of our reconstructed fluid in various interaction scenes.

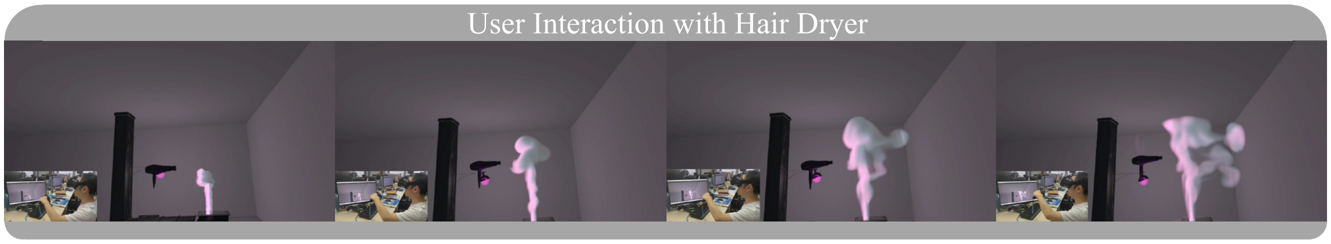


Fig. 6: A user wearing HTV VIVE interacts with the fluid using a virtual hair dryer. As the user points the hair dryer at the fluid and activates it, the fluid responds to the simulated wind force, exhibiting dynamic behavior and turbulence.

et al.'s method [7] and the RB method [46] using the Chimney scene. We measured the average computation time and memory usage for 100 iterations. Table 2 presents the performance data for this comparison experiment. Our differentiable rendering method achieves the best balance between computation time and memory usage among the compared methods. Our method takes only 8.51 ms for each iteration, which is 9.23 times faster than Cen et al.'s method [7] (78.56 ms) and 616.96 times faster than the RB method [46] (5249.31 ms). In terms of memory usage, our method consumes 423 MB, which is 4.31 times less than Cen et al.'s method [7] (1,823 MB) and 3.01 times less than the RB method [46] (1,274 MB).

The significant performance improvement of our method can be attributed to several factors when compared to both the RB method [46] and Cen et al.'s method [7]. First, our approach incorporates RB processing and uses analytic derivative formulas to compute the derivatives directly, eliminating the need for Monte Carlo estimation based on differential path tracing, which is used in the RB method. This contributes to the substantial speedup of our method compared to the RB method. Second, our method does not account for multiple scattering in differentiable rendering, which further reduces computational complexity compared to Cen et al.'s method, which requires solving the differential diffusion equation to determine the derivatives of multiple scattering radiance. This simplification allows our method to achieve faster computation times and lower memory usage compared to Cen et al.'s method while still maintaining high performance.

4.2 Qualitative Evaluation

The qualitative evaluation of our interactive fluid modeling system in mixed reality focuses on three key aspects: visual realism, interactivity, and user engagement. Section 4.2.1 evaluates the visual and interaction realism for smoke reconstruction, while Section 4.2.2 is dedicated to luminous fluid reconstruction, validating the verisimilitude of visual representation and the interaction with the environment. Ultimately, we conduct a user study in Section 4.2.3 to assess the system's capability to replicate realistic fluid behaviors and respond to user interactions within virtual environments.

4.2.1 Visual Realism and Interactivity Evaluation

In the Chimney scene, we conducted experiments to validate the reconstruction adaptability and the dynamic realism of the interaction. Figure 5 showcases a relighting and two interaction experiments. Fluid motions are reconstructed from the captured real images. The left column demonstrates the versatility and reusability of our reconstructed fluid in novel scenes with distinct light source colors. The middle column, Interaction with Board experiment, evaluates the dynamic realism of interaction with virtual objects by placing a virtual board at different heights and observing the fluid's realistic motion and adaptation to the new constraint. The right column, Interaction with Wind experiment, assesses the realism of the reconstructed fluid's interaction with environmental factors, showing the reconstructed fluid accurately captures the dynamic behavior and turbulence caused by the wind.

Figure 6 presents the User Interaction with Hair Dryer experiment, where a user wearing HTC VIVE interacts with the fluid using a virtual hair dryer. The fluid's motion is visibly influenced by the virtual hair dryer's wind, exhibiting realistic dynamics and turbulence as the user moves the hair dryer around. The fluid's response to the user's actions is both visually convincing and interactive, closely resembling the behavior of real smoke in the presence of wind.

The experiments conducted in the Chimney scene collectively validate our reconstructed fluid's visual quality, adaptability, and interactive capabilities, showcasing the fluid's ability to adapt to different lighting conditions, interact with virtual objects, respond to user-induced forces, and provide an immersive and engaging user experience.

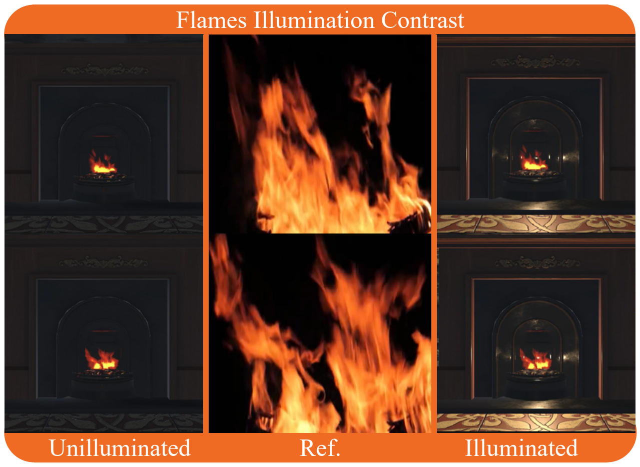


Fig. 7: This figure presents the Flames Illumination Contrast experiment, which compares the reconstructed fire with an unilluminated scene (left column) and an illuminated scene (right column) to validate the realism of the reconstructed fire's interaction with its surrounding environment. The middle column showcases the reference images(Ref.) for the reconstruction.

4.2.2 Validations of Reconstructed Luminous Fluid

We designed the Flames Illumination Contrast experiment to validate the effects of the reconstructed luminous fluid's interaction with its surrounding environment. Figure 7 presents the results, with the left column depicting the unilluminated rendering results and the right column showing the illuminated rendering results produced by the reconstructed flames. This comparison allows us to validate the authenticity and convincing nature of the reconstructed fire's interaction with its surroundings.

The Changing Viewpoints experiment, shown in Figure 8, validates the effectiveness of our Billboard-based rendering method by showcasing the reconstructed fire from three different viewpoints. As the viewpoint changes, the reconstructed fire adapts to the different perspectives while maintaining its volumetric appearance and luminous properties. We recommend that readers view our live demo in the supplemental video for visual validation.

To further evaluate the effects of luminous fluid reconstruction under diverse virtual environments, including both calm and windy conditions. The resultant outcomes are depicted in Figure 9. In the Bonfire scene, the left column displays the flame reconstructed in a breezeless environment, illuminating the surrounding ground. The right column demonstrates the reconstructed flame's response to wind, illuminating its surroundings while being pushed and deformed in the wind's direction.

In conclusion, these experiments demonstrate the effectiveness of our method in reconstructing luminous fluids that realistically interact with their environment, representing a significant step forward in creating immersive and believable virtual experiences.

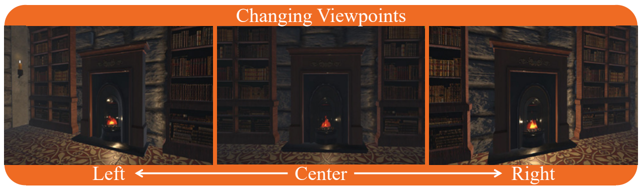


Fig. 8: The reconstructed fire is shown from three different viewpoints: left, center, and right. As the camera perspective changes, the Billboard-based rendering ensures that the fire faces the observer.

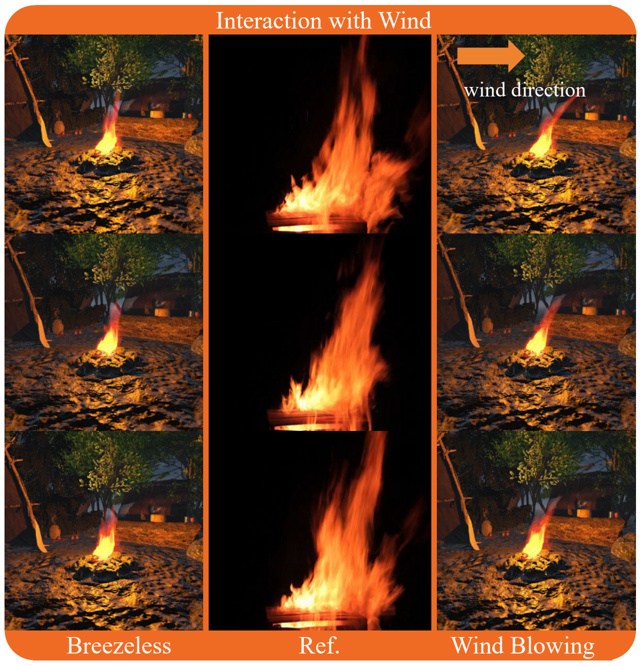


Fig. 9: Flame reconstruction in the Bonfire scene with and without wind. The left column shows that the reconstructed flame illuminated its surroundings in a breezeless environment. In contrast, the right column illustrates the reconstructed flame being influenced by wind, exhibiting a drifting motion. Ref. means the reference images.

4.2.3 User Study

To further assess the visual realism of the generated fluids, we have recruited 20 participants to assess their subjective impressions of our system's visual effects. Among these participants, 14 are male and 6 are female, ranging in age from 20 to 40. In terms of prior experience, 8 participants reported having no prior experience with MR.

Evaluation Metrics Qualitatively, participants are required to assess the naturalness of generations, the smoothness of generations, the visual similarity between the reference and the generation, and the authenticity of interactions using an 10-point Likert scale (0: Extremely disagree, 10: Extremely agree).

- The measurement of naturalness corresponds to the question: "I thought the visual effect of the generated fluid in this scene to be natural."
- The smoothness rating measures the efficiency of the system, encompassing the entire process from fluid reconstruction to interactions, and corresponds to the question: "I thought the generated fluid in this scene to be smooth."
- The visual similarity rating gauges to which the generated fluid mirrors the reference fluid, and corresponds to the question: "I thought the generated fluid and the reference fluid look similar."
- The authenticity rating measures the reliability of interaction

response, and corresponds to the question: "I thought the variation of the fluid in this scene conformed to my operation."

Scenes and Tasks During the subjective experiment, participants are required to experience 6 VR scenes. The first four scenes, named Chimeny1, Chimeny2, Chimeny3, and HairDryer, are all based on the Chimney scene depicted above, each varying in the type of interaction. In Chimeny1, no predefined interactions exist. Chimeny2 and Chimeny3 feature predefined interactions, with a board present in the former and fixed wind in the latter. In HairDryer, participants have the freedom to blow wind arbitrarily using a virtual hair dryer. The fifth scene is the same as the Fireplace (also named as Fireplace), and the sixth scene is the Bonfire scene with fixed wind (also named as Bonfire)

In scenes except for HairDryer, participants are asked to observe the generated fluid multiple times from different views and assess both the naturalness and the smoothness of generations. In HairDryer, participants are requested to interact with the fluid multiple times and assess both the naturalness and the smoothness of the fluids affected by the user interaction. Additionally, participants in HairDryer must evaluate the authenticity of their interactions. In Chimeny1 and Fireplace, the other evaluation participants must perform is to compare the visual similarity between the generated fluid and the corresponding reference.

Results and Analysis The statistical results are presented in Figure 10. The boxplots showcase the the distribution of ratings for naturalness smoothness, visual similarity, and authenticity in scenes.

The median naturalness ratings in Chimney1, Chimney2, Chimney3, Fireplace, Bonfire and HairDryer are 8, 9, 7, 8, 9, 8 and 8, respectively, with corresponding mean values of 8.25, 8.25, 6.85, 8.4, 8.5 and 7.45. Notably, even the lowest rating in most scenes, higher than 5, reflects a positive perception. These high median and mean values, along with the compact interquartile ranges of the boxplots, indicate that participants consistently found the generated fluid to be highly natural, even when environmental and user interactions exist in the scene.

The median smoothness ratings in six scenes are 8, 8.5, 8, 9, 9 and 8, respectively, with corresponding mean values of 8.35, 8.3, 7.25, 8.45, 8.65 and 8. High median and mean values, coupled with the compact interquartile ranges depicted in the boxplots, suggest a consistent recognition among participants of the efficiency of our system.

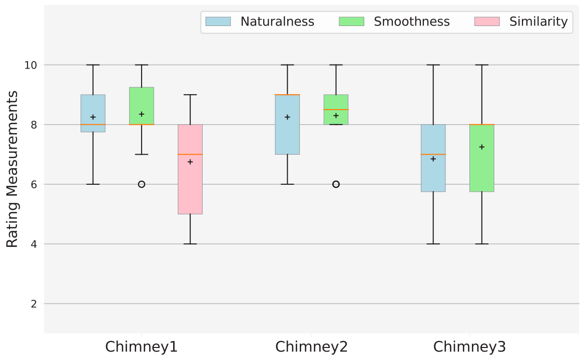
The median visual similarity ratings in the Chimney1 and Fireplace scenes are 7 and 7.5, respectively, with corresponding mean values of 6.75 and 7.6. In Chimney1, the lower quartile is at 5, suggesting that most participants perceive a positive similarity between the ground truth and the generated fluid. In Fireplace, the minimum value of the boxplot serves as the indicator of neutrality, suggesting a consistent recognition of visual similarity among participants. These metrics demonstrate our system's ability to faithfully replicate real fluid phenomena within MR environments.

The median authenticity rating in HairDryer, which involved user interaction, is 8.5, with a mean of 7.75. Notably, the lower quartile of the authenticity rating stands at 6.5, significantly surpassing the neutrality indicator. These metrics validate the system's ability to provide a reliable and convincing interactive experience, in line with expectations for user interactions.

5 Conclusions and Future Work

In this work, we have presented a comprehensive system for real-time, interactive fluid modeling in mixed reality environments. Our approach integrates fluid reconstruction, velocity estimation, and a lightweight physical simulator within the Unity engine to generate dynamic and visually compelling fluid effects that respond to user interactions and environmental factors. Our system not only aims to replicate realistic fluid behaviors but also to enhance the mixed reality experience by introducing novel interactive capabilities and visual effects. By pushing the boundaries of fluid modeling in virtual environments, we strive to unlock new possibilities for user engagement and creativity in mixed reality applications.

However, our work has several limitations that should be acknowledged. Firstly, our method focuses on reconstructing optically thin fluids using a single scattering model. While this simplification enables real-time performance, it may not accurately represent the appearance of optically thick or highly scattering media. Future work might enhance the visual fidelity of the reconstructed fluids in these scenarios by integrating more complex multiple scattering models. This extension would require a careful balance between computational efficiency and visual accuracy to maintain real-time performance.



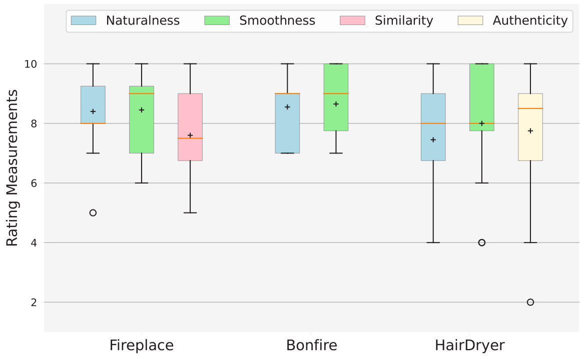


Fig. 10: Boxplot of subjective impressions of system's visual effects in 6 scenes. The metrics includes the the naturalness and the smoothness of generations, the visual similarity between references and generations, and the authenticity of interactions.

Secondly, the simplified velocity estimation approach, while enabling interactive fluid motion reconstruction, may not capture all the intricate details and complex behaviors of real-world fluids. Our method prioritizes computational efficiency to achieve real-time interaction, but this comes at the cost of some accuracy compared to more computationally expensive methods like Eckert et al.'s [12] approach. Future work could explore ways to improve the correctness of the velocity estimation while maintaining interactive frame rates, possibly by leveraging advanced machine learning techniques or adaptive simulation grids.

Another limitation of our current system is the lack of two-way coupling between the fluid and solid objects. While the fluid can interact with virtual objects and respond to user actions, the objects themselves do not experience forces from the fluid. Future work could integrate two-way coupling, allowing for more realistic interactions, such as objects being pushed or deformed by the fluid. This extension would require the development of efficient collision detection and response algorithms that can handle the complex interactions between fluid and solid geometry.

In conclusion, our work makes valuable contributions to the field of fluid modeling for mixed reality applications. Our main contribution is the development of an end-to-end system within the Unity engine. Despite the limitations discussed, our work provides a valuable foundation for future research and development in the field of fluid modeling for mixed reality applications.

Acknowledgments

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

REFERENCES

[1] A. Aguirre-Pablo, A. B. Aljedaani, J. Xiong, R. Idoughi, W. Heidrich, and S. T. Thoroddsen. Single-camera 3d ptv using particle intensities and structured light. *Experiments in Fluids*, 60(2):25, 2019. 2

[2] B. Atcheson, W. Heidrich, and I. Ihrke. An evaluation of optical flow algorithms for background oriented schlieren imaging. *Experiments in fluids*, 46(3):467–476, 2009. 2

[3] 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

[4] K. Bai, C. Wang, M. Desbrun, and X. Liu. Predicting high-resolution turbulence details in space and time. *ACM Transactions on Graphics (TOG)*, 40(6):1–16, 2021. 2

[5] J. Belden, T. T. Truscott, M. C. Axiak, and A. H. Techet. Three-dimensional synthetic aperture particle image velocimetry. *Measurement Science and Technology*, 21(12):125403, 2010. 2

[6] R. Bridson and M. Müller-Fischer. Fluid simulation: Siggraph 2007 course notesvideo files associated with this course are available from the citation page. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07, 81 pages, p. 1–81. Association for Computing Machinery, New York, NY, USA, 2007. doi: 10.1145/1281500.1281681 5

[7] Y. Cen, C. Li, F. W. Li, B. Yang, and X. Liang. A differential diffusion theory for participating media. In *Computer Graphics Forum*, vol. 42, p. e14956. Wiley Online Library, 2023. 3, 4, 6, 7

[8] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of mathematical imaging and vision*, 40(1):120–145, 2011. 5

[9] W. Chen, H. Ling, J. Gao, E. Smith, J. Lehtinen, A. Jacobson, and S. Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. *Advances in Neural Information Processing Systems*, 32:9609–9619, 2019. 3

[10] M. Chu, L. Liu, Q. Zheng, E. Franz, H.-P. Seidel, C. Theobalt, and R. Zayer. Physics informed neural fields for smoke reconstruction with sparse data. *ACM Transactions on Graphics (TOG)*, 41(4):1–14, 2022. 1, 2

[11] T. Corpetti, É. Mémin, and P. Pérez. Dense estimation of fluid flows. *IEEE Transactions on pattern analysis and machine intelligence*, 24(3):365–380, 2002. 2

[12] M.-L. Eckert, W. Heidrich, and N. Thuerey. Coupled fluid density and motion from single views. In *Computer Graphics Forum*, vol. 37, pp. 47–58. Wiley Online Library, 2018. 1, 2, 5, 9

[13] G. E. Elsinga, F. Scarano, B. Wieneke, and B. W. van Oudheusden. Tomographic particle image velocimetry. *Experiments in fluids*, 41(6):933–947, 2006. 2

[14] B. K. ER and A. S. Behr. Teaching fluid mechanics in a virtual-reality based environment. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pp. 1563–1567. IEEE, 2020. 2

[15] T. W. Fahringer, K. P. Lynch, and B. S. Thurow. Volumetric particle image velocimetry with a single plenoptic camera. *Measurement Science and Technology*, 26(11):115201, 2015. 2

[16] Y. Feng, X. Feng, Y. Shang, Y. Jiang, C. Yu, Z. Zong, T. Shao, H. Wu, K. Zhou, C. Jiang, et al. Gaussian splashing: Dynamic fluid synthesis with gaussian splatting. *arXiv preprint arXiv:2401.15318*, 2024. 3

[17] 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*, pp. 1632–1642, 2021. 1, 2

[18] M. Gattullo, A. E. Uva, M. Fiorentino, and J. L. Gabbard. Legibility in industrial ar: text style, color coding, and illuminance. *IEEE computer graphics and applications*, 35(2):52–61, 2015. 2

[19] K. Genova, F. Cole, A. Maschinot, A. Sarna, D. Vlasic, and W. T. Freeman. Unsupervised training for 3d morphable model regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8377–8386, 2018. 3

[20] I. Gkioulekas, A. Levin, and T. Zickler. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision*, pp. 685–701. Springer, 2016. 3

[21] I. Gkioulekas, S. Zhao, K. Bala, T. Zickler, and A. Levin. Inverse volume rendering with material dictionaries. *ACM Transactions on Graphics (TOG)*, 32(6):1–13, 2013. 3

[22] E. Goldhahn and J. Seume. The background oriented schlieren technique: sensitivity, accuracy, resolution and application to a three-dimensional density field. *Experiments in fluids*, 43(2):241–249, 2007. 2

[23] J. Gregson, I. Ihrke, N. Thuerey, and W. Heidrich. From capture to simulation: connecting forward and inverse problems in fluids. *ACM Transactions on Graphics (TOG)*, 33(4):1–11, 2014. 2

[24] 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

[25] P.-H. Han, T.-H. Wang, and C.-H. Chou. Groundflow: Liquid-based haptics for simulating fluid on the ground in virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2670–2679, 2023. 2

[26] P. Henderson and V. Ferrari. Learning to generate and reconstruct 3d meshes with only 2d supervision. *arXiv preprint arXiv:1807.09259*, 2018. 3

[27] I. Herlin, D. Béréziat, N. Mercier, and S. Zhuk. Divergence-free motion estimation. In *European Conference on Computer Vision*, pp. 15–27. Springer, 2012. 2

[28] W. Höhl. Ambiguity in utopian xr-games: Basic principles for the design of virtual worlds. *European Journal of Futures Research*, 11(1):6, 2023. 2

[29] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 2

[30] Y. Hu, J. Liu, X. Yang, M. Xu, Y. Kuang, W. Xu, Q. Dai, W. T. Freeman, and F. Durand. Quantaichi: a compiler for quantized simulations. *ACM Transactions on Graphics (TOG)*, 40(4):1–16, 2021. 5

[31] C. Huang and Y. Huang. Application of computer in mixed reality technology. In *Frontier Computing: Theory, Technologies and Applications (FC 2019) 8*, pp. 760–766. Springer, 2020. 2

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

[33] I. Ihrke and M. Magnor. Adaptive grid optical tomography. *Graphical Models*, 68(5-6):484–495, 2006. 2

[34] C. Jiang, C. Schroeder, A. Selle, J. Teran, and A. Stomakhin. The affine particle-in-cell method. *ACM Transactions on Graphics (TOG)*, 34(4):1–10, 2015. 2

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

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

[37] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[38] C. Lee, G. A. Rincon, G. Meyer, T. Höllerer, and D. A. Bowman. The effects of visual realism on search tasks in mixed reality simulation. *IEEE transactions on visualization and computer graphics*, 19(4):547–556, 2013. 1

[39] S. Liu, W. Chen, T. Li, and H. Li. Soft rasterizer: Differentiable rendering for unsupervised single-view mesh reconstruction. *arXiv preprint arXiv:1901.05567*, 2019. 3

[40] T. Liu and L. Shen. Fluid flow and optical flow. *Journal of Fluid Mechanics*, 614:253–291, 2008. 2

[41] M. M. Loper and M. J. Black. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pp. 154–169. Springer, 2014. 3

[42] N. T. Marie-Lena Eckert, Kiwon Um. 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):239, 2019. 6

[43] G. Martin, L. Koizia, A. Kooner, J. Cafferkey, C. Ross, S. Purkayastha, A. Sivananthan, A. Tanna, P. Pratt, J. Kinross, et al. Use of the hololens2 mixed reality headset for protecting health care workers during the covid-19 pandemic: prospective, observational evaluation. *Journal of medical Internet research*, 22(8):e21486, 2020. 1, 2

[44] 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. 3

[45] D. Mourtzis, J. Angelopoulos, and N. Panopoulos. Integration of mixed reality to cfd in industry 4.0: A manufacturing design paradigm. *Procedia CIRP*, 107:1144–1149, 2022. 2

[46] 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. 3, 4, 6, 7

[47] 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

[48] G. Psomathianos, N. Sourdakos, and K. Moustakas. Smoke diffusion

simulation and physically-based rendering for vr. In *2021 International Conference on Cyberworlds (CW)*, pp. 117–120. IEEE, 2021. 2

[49] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10318–10327, 2021. 3

[50] G. E. Raptis, C. Fidas, and N. Avouris. Effects of mixed-reality on players' behaviour and immersion in a cultural tourism game: A cognitive processing perspective. *International Journal of Human-Computer Studies*, 114:69–79, 2018. 1

[51] S. Rokhsaritalemi, A. Sadeghi-Niaraki, and S.-M. Choi. A review on mixed reality: Current trends, challenges and prospects. *Applied Sciences*, 10(2):636, 2020. 2

[52] P. Ruhnau, A. Stahl, and C. Schnörr. Variational estimation of experimental fluid flows with physics-based spatio-temporal regularization. *Measurement Science and Technology*, 18(3):755, 2007. 2

[53] G. Simondon, N. Mellamphy, and J. Hart. *On the mode of existence of technical objects*. University of Western Ontario London, 1980. 2

[54] M. Slater. Immersion and the illusion of presence in virtual reality. *British journal of psychology*, 109(3):431–433, 2018. 2

[55] S. Solmaz and T. Van Gerven. Interactive cfd simulations with virtual reality to support learning in mixing. *Computers & Chemical Engineering*, 156:107570, 2022. 2

[56] P. Song, H. Yu, and S. Winkler. Vision-based 3d finger interactions for mixed reality games with physics simulation. In *Proceedings of the 7th ACM SIGGRAPH international conference on virtual-reality continuum and its applications in industry*, pp. 1–6, 2008. 2

[57] Z. P. Tan and B. S. Thurow. Time-resolved 3d flow-measurement with a single plenoptic-camera. In *AIAA Scitech 2019 Forum*, p. 0267, 2019. 2

[58] Y.-M. Tang, K. M. Au, H. C. Lau, G. T. Ho, and C.-H. Wu. Evaluating the effectiveness of learning design with mixed reality (mr) in higher education. *Virtual Reality*, 24(4):797–807, 2020. 1, 2

[59] S. Thapa, N. Li, and J. Ye. Dynamic fluid surface reconstruction using deep neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21–30, 2020. 2

[60] K. Um, X. Hu, and N. Thuerey. Liquid splash modeling with neural networks. In *Computer Graphics Forum*, vol. 37, pp. 171–182. Wiley Online Library, 2018. 2

[61] Y. Wang. [dc] foveated fluid animation in virtual reality. In *2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pp. 987–988. IEEE, 2023. 2

[62] J. Xiong, Q. Fu, R. Idoughi, and W. Heidrich. Reconfigurable rainbow piv for 3d flow measurement. In *2018 IEEE International Conference on Computational Photography (ICCP)*, pp. 1–9. IEEE, 2018. 2

[63] J. Xiong, R. Idoughi, A. A. Aguirre-Pablo, A. B. Aljedaani, X. Dun, Q. Fu, S. T. Thoroddsen, and W. Heidrich. Rainbow particle imaging velocimetry for dense 3d fluid velocity imaging. *ACM Transactions on Graphics (TOG)*, 36(4):1–14, 2017. 2

[64] J. Yuan, C. Schörr, and G. Steidl. Simultaneous higher-order optical flow estimation and decomposition. *SIAM Journal on Scientific Computing*, 29(6):2283–2304, 2007. 2

[65] 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. 3, 4

[66] F. Zhang, Q. Wei, and L. Xu. An fast simulation tool for fluid animation in vr application based on gpus. *Multimedia Tools and Applications*, 79:16683–16706, 2020. 2

[67] Q. Zhang, S. Xiao, Y. Cen, J. Han, and X. Liang. Global physical prior based fluid reconstruction for vr/ar. In *2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pp. 823–824. IEEE, 2023. 2

[68] K. Zhou, R. Cai, Y. Ma, Q. Tan, X. Wang, J. Li, H. P. Shum, F. W. Li, S. Jin, and X. Liang. A video-based augmented reality system for human-in-the-loop muscle strength assessment of juvenile dermatomyositis. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2456–2466, 2023. 1

[69] K. Zhou, C. Chen, Y. Ma, Z. Leng, H. P. Shum, F. W. Li, and X. Liang. A mixed reality training system for hand-object interaction in simulated microgravity environments. In *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 167–176. IEEE, 2023. 1