FluidNexus: 3D Fluid Reconstruction and Prediction from a Single Video

Supplementary Material

A. Overview

In this supplementary material, we elaborate on the details of our approach (B), datasets (C), and experiments (D). We also include results on an in-the-wild scene from HyFluid [15] and results on using multiple views rather than a single view (E). We further discuss ablation study results (F). We compile video results in our project website https://yuegao.me/FluidNexus. We strongly encourage the readers to review the video results first.

B. Technical Details

Position-based fluid simulation. Our physics simulation is based on position-based fluid (PBF) [6, 7] which extends position-based dynamics (PBD) [7, 9]. PBD provides a simple and flexiable particle-based simulation framework based on solving position constraints. In the following, we briefly review the PBF simulation. We refer the reader to Macklin et.al. [7, 9] for more details. In short, PBD uses a set of particles to represent the scene and each particle consists of its position, mass, and velocity. PBD solves a system of non-linear equality and inequality constraints for the position correction $\mathbf{\Delta p}$ such that physical constraints are met:

$$\begin{aligned} c_i(\mathbf{p}+\mathbf{\Delta p}) &= 0, \quad i = 1, \dots, n, \\ c_j(\mathbf{p}+\mathbf{\Delta p}) &\geq 0, \quad j = 1, \dots, n, \end{aligned} \tag{S1}$$

where $\mathbf{p} = [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_N]^T$ denotes the vector of particle positions. Constraints are solved sequentially using the linearization of C around $\mathbf{p}$, and the position change $\mathbf{\Delta p}$, is restricted to lie along the constraint gradient:

$$c_i(\mathbf{p}+\mathbf{\Delta p}) \approx c_i(\mathbf{p}) + \nabla c_i(\mathbf{p})\mathbf{\Delta p} = 0, \tag{S2}$$

and the particle velocity is then given by $\mathbf{v} = \frac{\mathbf{\Delta p}}{\Delta t}$. We assume that all physical particles have a mass of 1 to simplify the formulation.

The constraint for fluid (incompressibility) is given by

$$c_{\text{fluid}}(\mathbf{p}_1, \dots, \mathbf{p}_{N_{\text{physical}}}) = \frac{\rho_i}{\rho_0} - 1 = 0, \tag{S3}$$

where the fluid density is estimated by $\rho_i = \sum_j K(\mathbf{p}_i - \mathbf{p}_j)$ and K is a kernel function. We use the cubic Poly6 kernel [8]. We also utilize the drag force as an external force to model the effect of fast moving air interacting with the surrounding environment:

$$\mathbf{f}_i = -k(\mathbf{v}_i - \mathbf{v}_{\text{env}}) \max(0, 1 - \frac{\rho_i}{\rho_o}), \tag{S4}$$

where $\mathbf{v}_{\text{env}}$ is the environmental velocity at the particle position and is set to $\mathbf{0}$ to model still air, and $k > 0$ is the drag force coefficient.

Initialization. To initialize the fluid simulation, we follow Macklin et.al. [7] to add a source region and run N_{stable} stabilization simulation steps at $t = 0$ (*i.e.*, before the reconstruction starts). We seed physical particles and visual particles within the source region at each timestep. After the stabilization steps, we obtain the initial physical particles $\{\mathbf{p}_0, \mathbf{u}_0\}$ and visual particle positions $\{\mathbf{x}_0\}$. All other visual particle attributes are initialized as constants $\{\mathbf{c}_0, \mathbf{s}_0, \mathbf{o}_0, \mathbf{r}_0\}$ for all timesteps.

Simulation and advection operators. The simulation operator for the physical particle position $\mathbf{p}_t^{\text{sim}} = \texttt{Sim}(\mathbf{u}_{t-1}, \mathbf{p}_{t-1})$ consists of three steps: (1) We generate a guess of particle velocities $\hat{\mathbf{u}}_t = \mathbf{u}_{t-1} + \Delta t \cdot \alpha \cdot \mathbf{g} + \Delta t \cdot \mathbf{f}$ where $\mathbf{g}$ denotes the gravity and $\alpha < 0$ denotes the buoyancy coefficient. $\mathbf{f}$ denotes an optional external force. (2) We generate a guess of physical particle positions $\hat{\mathbf{p}}_t = \mathbf{p}_{t-1} + \Delta t \cdot \hat{\mathbf{u}}_t$. (3) We obtain the simulation result $\mathbf{p}_t^{\text{sim}}$ by solving $c_{\text{fluid}}(\hat{\mathbf{p}}_t + \mathbf{\Delta p}_t) = 0$.

For the advection operator $\texttt{Adv}$, we use a simple forward Euler integrator, *i.e.*, $\texttt{Adv}(\mathbf{V}, \mathbf{x}) = \mathbf{x} + \Delta t \cdot \mathbf{V}(\mathbf{x})$, while advanced advection schemes can also be used *e.g.*, BFECC [12].

Regularization term. To encourage temporal smoothness we add a regularization term $\mathcal{L}_{\text{reg}}$:

$$\begin{aligned} \mathcal{L}_{\text{reg}} = \lambda_c \|\mathbf{c}_t - \mathbf{c}_{t-1}\|_2^2 + \lambda_s \|\mathbf{s}_t - \mathbf{s}_{t-1}\|_2^2 \\ + \lambda_o \|\mathbf{o}_t - \mathbf{o}_{t-1}\|_2^2 + \lambda_r \|\mathbf{r}_t - \mathbf{r}_{t-1}\|_2^2 + \mathcal{L}_{\text{aniso}}, \end{aligned} \tag{S5}$$

where λ_c, λ_s, λ_o, and λ_r are weighting coefficients for the color, scale, opacity, and orientation terms respectively. The L2 terms help maintain smooth transitions in the visual attributes. We also adopt the anisotropic loss $\mathcal{L}_{\text{aniso}}$ from Xie et.al. [14] to prevent overly skinny visual particles.

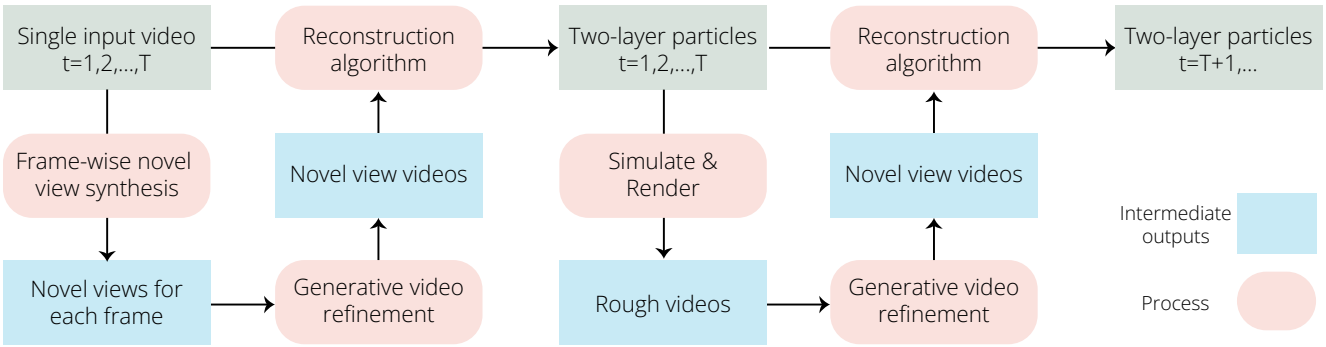

Figure S1. **Flowchart of FluidNexus.** The reconstruction algorithm is shown in Alg. S1. On the left is the reconstruction process which takes a single video as input and produces the two-layer particles within the video duration. On the right is the prediction process which takes the reconstructed two-layer particles as input and produces the particles in future timesteps.

Algorithm S1 FluidNexus: Reconstruction

Input: Multi-view videos $\{\mathcal{V}^c\}_{c=0}^{C}$, camera poses $\{\pi_c\}_{c=0}^{C}$
Initialize: Physical particles $\{\mathbf{p}_0, \mathbf{u}_0\}$ and visual particles $\{\mathbf{x}_0, \mathbf{c}_0, \mathbf{s}_0, \mathbf{o}_0, \mathbf{r}_0\}$ through stabilization

Physical particle reconstruction for $t = 1$ to T:
1. Simulate physical guess $\mathbf{p}_t^{\text{sim}} = \texttt{Sim}(\mathbf{u}_{t-1}, \mathbf{p}_{t-1})$
2. Optimize $\mathbf{p}_t$ by minimizing:
   - Simulation loss: $\mathcal{L}_{\text{sim}} = \|\mathbf{p}_t - \mathbf{p}_t^{\text{sim}}\|_2^2$
   - Incompressibility loss: $\mathcal{L}_{\text{incomp}}$ on current and next timesteps
   - Visual loss: $\mathcal{L}_{\text{visual}}$ across all views
3. Update velocity $\mathbf{u}_t = (\mathbf{p}_t - \mathbf{p}_{t-1})/\Delta t$

Visual particle reconstruction for $t = 1$ to T:
1. Compute visual positions $\mathbf{x}_t = \texttt{Adv}(\mathbf{V}_t, \mathbf{x}_{t-1})$
2. Optimize $\{\mathbf{c}_t, \mathbf{s}_t, \mathbf{o}_t, \mathbf{r}_t\}$ by minimizing:
   - Visual loss: $\mathcal{L}_{\text{visual}}$ across all views
   - Temporal regularization: $\mathcal{L}_{\text{reg}}$ with previous timestep

Output: Reconstructed fluid velocity by $\{\mathbf{p}_t, \mathbf{u}_t\}_{t=1}^{T}$ and appearance by $\{\mathbf{x}_t, \mathbf{c}_t, \mathbf{s}_t, \mathbf{o}_t, \mathbf{r}_t\}_{t=1}^{T}$

Algorithm S2 FluidNexus: Prediction

Input: Reconstructed states $\{\mathbf{p}_T, \mathbf{u}_T\}$ and $\{\mathbf{x}_T, \mathbf{c}_T, \mathbf{s}_T, \mathbf{o}_T, \mathbf{r}_T\}$, camera poses $\{\pi_c\}_{c=0}^{C}$

Initial simulation for $t = T + 1$ to T_{target}:
1. Simulate physical particles: $\mathbf{p}_t^{\text{pred}} = \texttt{Sim}(\mathbf{u}_{t-1}^{\text{pred}}, \mathbf{p}_{t-1}^{\text{pred}})$
2. Advect visual particles: $\mathbf{x}_t^{\text{pred}} = \texttt{Adv}(\mathbf{V}_t^{\text{pred}}, \mathbf{x}_{t-1}^{\text{pred}})$
3. Render rough multi-view frames: $\hat{I}_t^c = \texttt{Render}(\pi_c, \mathbf{x}_t^{\text{pred}}, \mathbf{c}_0, \mathbf{s}_0, \mathbf{o}_0, \mathbf{r}_0)$

Video refinement: Generate reference videos $\{(I_{T+1}^c, \cdots, I_{T_{\text{target}}}^c)\}_{c=0}^{C}$ using the generative video refinement model v on each rough video $(\hat{I}_{T+1}^c, \cdots, \hat{I}_{T_{\text{target}}}^c)$

Physical particle reconstruction for $t = T + 1$ to T_{target}:
1. Simulate physical guess $\mathbf{p}_t^{\text{sim}} = \texttt{Sim}(\mathbf{u}_{t-1}, \mathbf{p}_{t-1})$
2. Optimize $\mathbf{p}_t$ by minimizing:
   - Simulation loss: $\mathcal{L}_{\text{sim}} = \|\mathbf{p}_t - \mathbf{p}_t^{\text{sim}}\|_2^2$
   - Incompressibility loss: $\mathcal{L}_{\text{incomp}}$ on current and next timesteps
   - Visual loss: $\mathcal{L}_{\text{visual}}$ across all views
3. Update velocity $\mathbf{u}_t = (\mathbf{p}_t - \mathbf{p}_{t-1})/\Delta t$

Visual particle reconstruction for $t = T + 1$ to T_{target}:
1. Compute visual positions $\mathbf{x}_t = \texttt{Adv}(\mathbf{V}_t, \mathbf{x}_{t-1})$
2. Optimize $\{\mathbf{c}_t, \mathbf{s}_t, \mathbf{o}_t, \mathbf{r}_t\}$ by minimizing:
   - Visual loss: $\mathcal{L}_{\text{visual}}$ across all views
   - Temporal regularization: $\mathcal{L}_{\text{reg}}$ with previous timestep

Output: Predicted fluid dynamics $\{\mathbf{p}_t, \mathbf{u}_t\}_{t=T+1}^{T_{\text{target}}}$ and appearance $\{\mathbf{x}_t, \mathbf{c}_t, \mathbf{s}_t, \mathbf{o}_t, \mathbf{r}_t\}_{t=T+1}^{T_{\text{target}}}$

Algorithms and flowchart. We summarize the fluid reconstruction algorithm in Alg. S1, and the fluid prediction algorithm in Alg. S2. We further show a flowchart in Figure S1 to visualize the entire pipeline of FluidNexus.

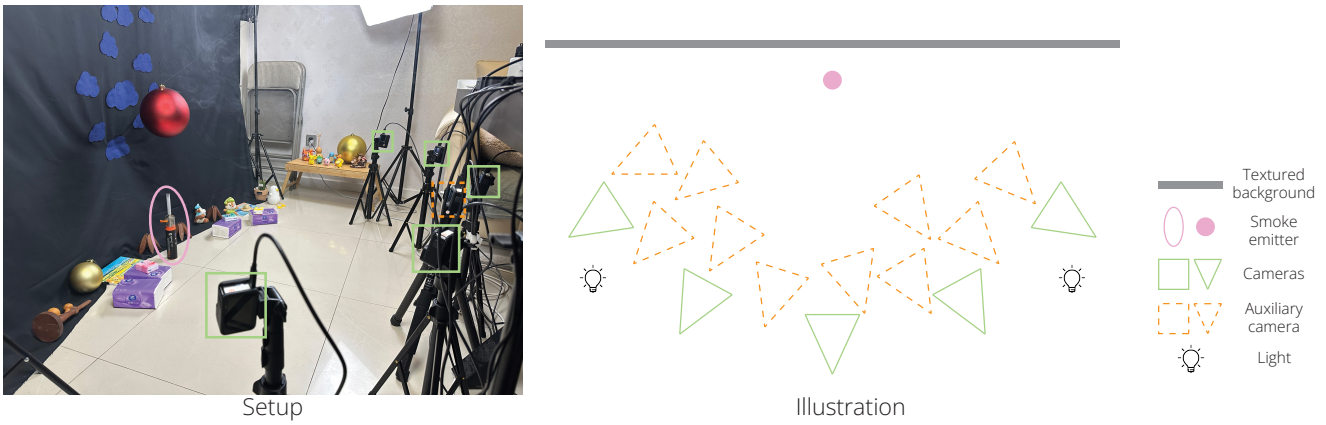

Figure S2. The configuration of our data capture system.

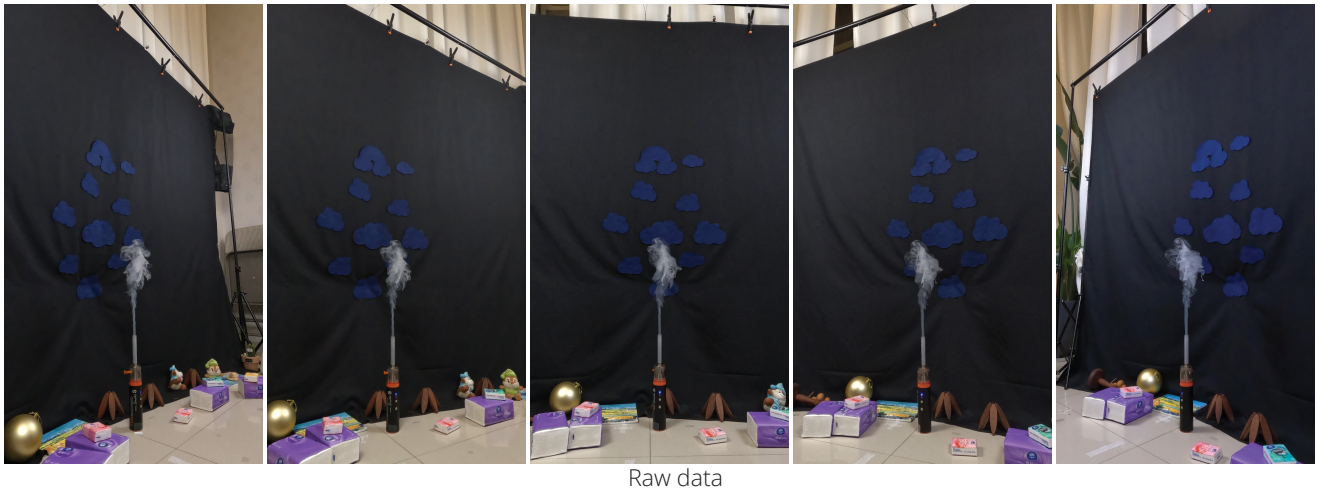

Figure S3. The raw frames captured by our five cameras.

C. Dataset Details

We show a photo and an illustration of our dataset capture setup in Figure S2. A black cloth was used as the background, onto which patches of various shapes were attached. To eliminate the influence of ambient light, we employed two adjustable color-temperature light sources. A handheld portable fog generator (Fog Machine Model S) with remote start and stop control created the desired smoke effects. The other objects on the ground were only used to hold down the black cloth to prevent it from moving and are not directly related to our setup. Other objects were used to add textures to help camera calibration.

The setup included six GoPro HERO 12 cameras, five of which were fixed to their locations and used as primary data recording cameras, while the sixth served as a secondary camera to capture multiple images for camera calibration using COLMAP [10, 11]. The GoPro cameras were mounted on tripods and configured to a 5K (2988×5312) resolution with $1.4\times$ magnification. The raw frames captured by the cameras are shown in Figure S3. We center-cropped the frames to 1440×2560 and resized them to 1080×1920. All experiments and camera calibrations were conducted at a resolution of 1080×1920. The frame rate was set to 50 fps to address power-line flicker. During preprocessing, all videos were converted into individual frames, and we sampled the frames with a step size of 2 for all experiments. The frame rate was set to 30 fps for presenting videos and conducting experiments.

Although we used a program to control the start and stop of the cameras, slight mis-synchronization could still occur. To address this, we manually labeled the starting frame of each video to ensure frame synchronization across all viewpoints.

For both of our datasets, we recorded 120 scenes each. We used 100 scenes as training data and evaluated the model on the remaining 20 scenes. Additionally, for the ScalarFlow [2] dataset, we used 94 scenes for training and evaluated on the remaining 10 scenes.

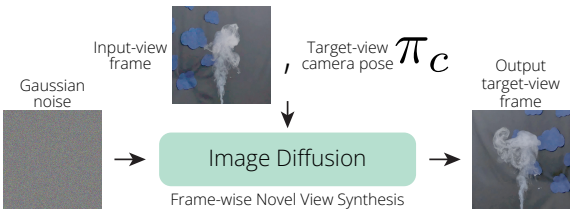

Figure S4. Illustration: Frame-wise Novel View Synthesis module.

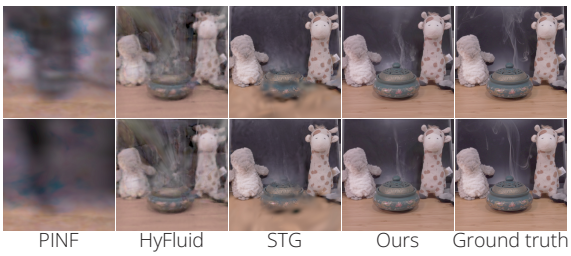

Figure S5. Qualitative results of re-simulation on in-the-wild data.

D. Further Implementation Details

Simulation details. In our experiments, we set the timestep $\Delta t = 1/30$ seconds. We set the drag force coefficient $k = 3$. We set the regularization loss weights $\lambda_c = 10$, $\lambda_s = 0$, $\lambda_o = 8$, $\lambda_r = 0.1$. For the ScalarFlow dataset, we set the buoyancy coefficient $\alpha = -3$ and environmental density $\rho_0 = 2$. For the FluidNexus-Smoke and FluidNexus-Ball datasets, we set $\alpha = -6$ and $\rho_0 = 1.5$. We set the stabilization simulation steps to $N_{\text{stable}} = 20$. We set the PBF constraint solver iteration count to 10. This maintains mean simulation speed of 0.04493 seconds per simulation timestep (amounting to 22.26 FPS).

Frame-wise novel view synthesis model training. We illustrate our frame-wise novel view synthesis model g in Fig. S4. It is based on an image diffusion model. In particular, it takes an input-view frame and a target-view camera pose as control signals, and it gradually denoises a Gaussian noise to generate the target-view frame. The generated target-view frames are the input to the generative video refinement module. We utilized Zero123 [5] as our image diffusion model. We performed full-parameter fine-tuning based on the official implementation and the pre-trained Zero123-XL [5] model, which is trained on the Objaverse-XL [1] dataset. We used a smaller batch size of 92 while keeping other hyperparameters (such as the base learning rate) unchanged.

We fine-tune it on three datasets individually. For the ScalarFlow dataset, we fine-tune the model for $15,000$ iterations. For FluidNexus-Smoke and FluidNexus-Ball, we fine-tune it for $50,000$ iterations. We keep all other official settings and use the official implementation of Zero123 [5]. For our experiments, we apply square padding and resize it to 256×256 to match the input and output frame size of the original implementation. We maintain the aspect ratio by cropping the output results and resizing them back to a resolution of 1080×1920.

Generative video refinement model training. For the generative video refinement model v, we use CogVideoX [3] as our base model and fine-tune it. We use the official implementation of CogVideoX [3]. We utilized CogVLM2 [13] to generate captions for all videos, and fine-tuned the model for $10,000$ iterations across all datasets using the official LoRA [4] implementation included in CogVideoX's official implementation. Similar to training the view synthesis model, we pad and resize our dataset to the resolution required by CogVideoX [3] (720×480), and inversely transform its output back to our experimental resolution (1080×1920).

Interaction simulation. For the counterfactual interaction simulation in Sec. 4.2, we consider two types of interaction: external body force (*e.g.*, the wind) and one-way coupling with rigid body (*e.g.*, the ball). External body force is simply implemented by setting $\mathbf{f}$. Specifically, we set the force $\mathbf{f}$ such that it is exponentially increasing with the physical particle position along the y axis (opposite gravity direction). The one-way coupling is implemented by fixing the rigid object still and use the simplified contact constraints from Macklin et.al. [7]. In particular, we use 3D Gaussian splatting to reconstruct the ball, which gives particles along the surface of the ball. The constraint for the fluid's physical particles is that if a particle enters the interior of the rigid object, we find the nearest object surface particle and update the physical fluid particle position to the exterior of the rigid body.

E. More Results

In-the-wild scene. We reconstruct an in-the-wild example from the HyFluid [15] paper. We use 3 available input views, as our video synthesizer requires training data. We showcase our re-simulation results in Fig. S5, and please check our website for video results. Ours significantly outperforms other methods.

Multi-view reconstruction. We use 4 views as input and 1 holdout view for testing. We show results in Tab. S1 and Fig. S6, as well as video results in our website. Our approach significantly outperforms prior methods. It indeed improves performances compared to ours using a single-view input.

F. Further Ablation Comparison

We provide all variants of the ablation experiments in the video results, showcasing the two tasks of novel view synthesis and re-simulation.

Firstly, in the novel view synthesis task, we observe that when our method excludes the novel-view video synthesizer

Model	Novel View Synthesis			Future Prediction			Re-simulation		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
PINF	24.45	0.8568	0.4973	20.42	0.7816	0.5883	19.44	0.6176	0.5672
HyFluid	26.42	0.8676	0.4146	22.91	0.8181	0.6049	22.46	0.8381	0.5623
STG	19.58	0.7101	0.4224	18.48	0.6105	0.4929	19.40	0.6098	0.5219
Ours-1	30.43	0.9212	0.1812	25.74	0.8609	0.2675	30.42	0.9211	0.1812
Ours	**31.15**	**0.9216**	**0.1233**	**26.84**	**0.8654**	**0.2382**	**31.14**	**0.9215**	**0.1233**
PINF	21.17	0.7507	0.5317	20.08	0.7195	0.5344	18.38	0.5618	0.5533
HyFluid	23.92	0.8327	0.4346	21.65	0.8088	0.5907	20.29	0.8165	0.5689
STG	18.79	0.7209	0.3884	18.45	0.5982	0.5054	18.68	0.6209	0.4883
Ours-1	28.24	0.9080	0.1719	24.82	0.8431	0.2607	28.23	0.9079	0.1720
Ours	**30.42**	**0.9211**	**0.1121**	**26.26**	**0.8479**	**0.2105**	**30.41**	**0.9211**	**0.1122**

Table S1. Quantitative results on FluidNexus-Smoke (upper) and FluidNexus-Ball (lower) with 4 input views. "Ours-1" denotes the performances using a single view as input.

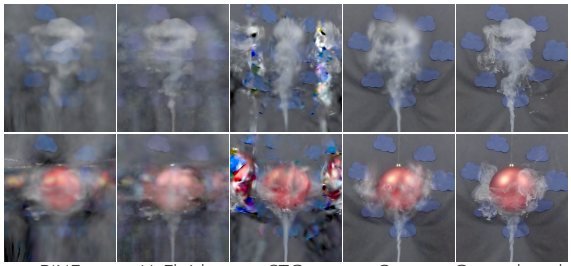

Figure S6. Novel view synthesis results with 4 input views.

("w/o NVS"), the results degrade significantly due to the lack of multi-view constraints. Additionally, when we remove the generative video refinement component ("w/o GVR"), the results include noticeably more jittering artifacts. Furthermore, we can observe abrupt transitions in the video results when the long video generation is removed ("w/o LVG").

To validate the importance of our physics constraints, we conducted comparisons on the re-simulation task. First, when the physical loss is removed ("w/o $\mathcal{L}_{\text{physics}}$"), the video results degrade significantly. This is because the optimized velocity field lacks physical accuracy, leading to implausible particle dynamics during advection in the re-simulation, which results in numerous artifacts. Additionally, when we remove the incompressibility loss ("w/o $\mathcal{L}_{\text{incomp}}$"), the visual dynamics look unnatural with mild jittering. When the simulation loss is removed ("w/o $\mathcal{L}_{\text{sim}}$"), the visual results also look unnatural with abrupt transitions.

References

[1] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-XL: A universe of 10M+ 3D objects. *arXiv preprint arXiv:2307.05663*, 2023. S4

[2] Marie-Lena Eckert, Kiwon Um, and Nils Thuerey. Scalarflow: a large-scale volumetric data set of real-world scalar transport flows for computer animation and machine learning. *ACM TOG*, 2019. S3

[3] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022. S4

[4] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. S4

[5] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3D object. In *CVPR*, 2023. S4

[6] Miles Macklin and Matthias Müller. Position based fluids. *ACM Transactions on Graphics (TOG)*, 2013. S1

[7] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014. S1, S4

[8] Matthias Müller, David Charypar, and Markus Gross. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 154–159. Citeseer, 2003. S1

[9] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109–118, 2007. S1

[10] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. S3

[11] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. S3

[12] Andrew Selle, Ronald Fedkiw, Byungmoon Kim, Yingjie Liu, and Jarek Rossignac. An unconditionally stable maccormack method. *Journal of Scientific Computing*, 2008. S1

[13] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, Jiazheng Xu, Bin Xu, Juanzi Li, Yuxiao Dong, Ming Ding, and Jie Tang. Cogvlm: Visual expert for pretrained language models, 2023. S4

[14] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4389–4398, 2024. S1

[15] Hong-Xing Yu, Yang Zheng, Yuan Gao, Yitong Deng, Bo Zhu, and Jiajun Wu. Inferring hybrid neural fluid fields from videos. *Advances in Neural Information Processing Systems*, 36, 2024. S1, S4