PhysGaia: A Physics-Aware Dataset of Multi-Body Interactions for Dynamic Novel View Synthesis

–Supplmentary document–

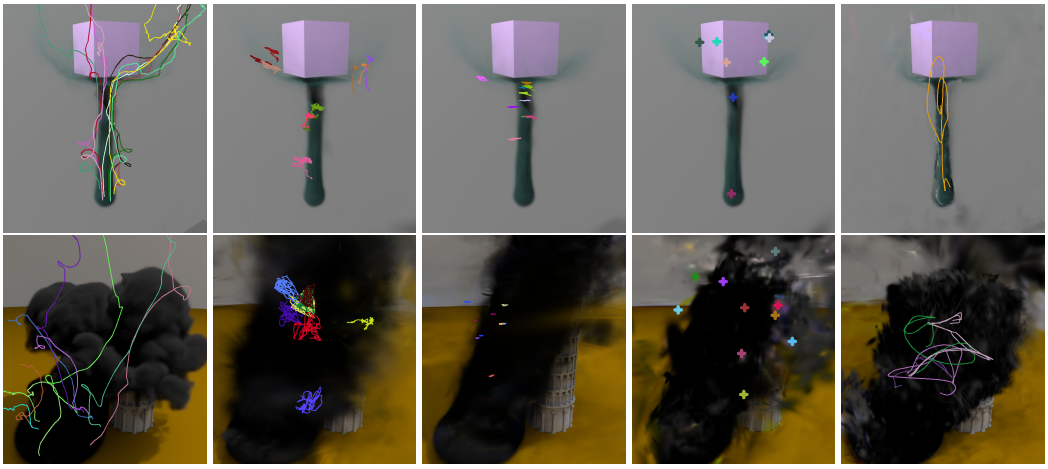
A Details of Experimental Results

A.1 Limitation on physical realism

Figure 6 in the main paper visualizes the trajectories of Gaussian primitives from 4DGS [1] alongside ground-truth particle trajectories, clearly illustrating the limitations in capturing physical realism. To further support this observation, Figure 7 demonstrates results from all the baseline algorithms.

Across all tested recent DyNVS algorithms, the reconstructed motions consistently fail to accurately follow the true physical trajectories. Specifically, trajectories produced by D-3DGS [2] and 4DGS [1] show only localized fluctuations. In the case of STG [3], the primitives remain largely static in the *Box-smoke* and *Pisa* scenes, requiring the addition of markers for better visualization. Similarly, the trajectories generated by SOM [4] exhibit erratic motion, primarily due to errors introduced by its reliance on external point trackers.

This comprehensive analysis highlights a critical limitation—**relying solely on RGB reconstruction loss ensures photorealism but does not guarantee physical realism**. We believe that our PhysGaia dataset, which provides ground-truth trajectories, will be instrumental in advancing future research toward physics-aware dynamic scene reconstruction.



(a) GT image (b) D-3DGS [2] (c) 4DGS [1] (d) STG [3] (e) SOM [4]

Figure 7: Comparison of reconstructed trajectories and its ground truth on the *Box-smoke* and *Pisa* scenes. The reconstructed trajectories do not accurately follow the true physical trajectories.

A.2 Implementation details

We implement recent DyNVS methods, including D-3DGS [2], 4DGS [1], STG [3], and SOM [4]. We follow the default training settings provided for each method. For 4DGS, we adopt the training configuration used for the HyperNeRF [5] dataset and reduce the grid learning rate to improve training stability. For SOM, we apply the recommended hyperparameters for the DyCheck [6] dataset. For

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

point cloud initialization, we run COLMAP with dense matching and fusion, followed by uniform downsampling to approximately 40,000 points. All experiments are conducted on NVIDIA RTX A5000 and A6000 GPUs, with training times ranging from 30 minutes to 2 hours depending on the method. Note that all experiments are implemented based on their public codes [1] [2] [3] [4].

A.3 Details of Quantitative Results

To supplement Table 3 of the main paper, we report more detailed quantitative results.

A.3.1 Average performance: all scenes

Table 4 presents the average performance of all methods under both monocular and multiview settings, aggregated across all 17 scenes. For SOM [4], we observed better performance when scaling the estimated depth map using the COLMAP point cloud obtained from dense matching, compared to sparse matching.

Table 4: Average quantitative results for both monocular and multiview settings, averaged across all 17 scenes. While multiview setups generally offer better reconstruction performance than monocular ones, even multiview results achieve PSNR scores below 30. This highlights the substantial difficulty in reconstructing the complex multi-body interactions in our dataset.

Method	**Monocular**			**Multiview**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	21.7	**0.86**	**0.18**	24.2	**0.89**	**0.14**
4DGS [1]	**22.7**	0.85	0.19	**24.4**	0.87	0.17
STG [3]	19.3	0.76	0.30	21.0	0.79	0.30
SOM [4]	19.3	0.80	0.26	N/A	N/A	N/A

A.3.2 Average performance: per each materials

Table 5 shows the average quantitative results for each material category across all baseline algorithms, with updated performance of SOM. Overall, D-3DGS [2] and 4DGS [1] demonstrate the most stable performance. Performance tends to be higher on textile materials but worsens on viscoelastic materials, which typically exhibit complex motion involving many independent objects.

Table 5: Average quantitative results for each material category across all baseline algorithms

Capture Type	Method	**Liquid**			**Gas**		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Monocular	D-3DGS [2]	22.7	**0.87**	**0.22**	21.9	**0.89**	**0.16**
	4DGS [1]	**24.2**	**0.87**	0.23	21.7	0.88	0.17
	STG [3]	19.2	0.72	0.39	**21.9**	0.85	0.24
	SOM [4]	19.6	0.80	0.32	20.0	0.84	0.27
Multiview	D-3DGS [2]	22.2	0.87	0.24	23.7	**0.91**	**0.13**
	4DGS [1]	**25.1**	**0.88**	**0.22**	24.2	0.89	0.17
	STG [3]	20.8	0.75	0.40	**25.0**	**0.91**	0.19
Capture Type	Method	**Viscoelastic materials**			**Textile**		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Monocular	D-3DGS [2]	**20.1**	**0.84**	**0.15**	22.1	0.83	**0.18**
	4DGS [1]	19.5	0.82	0.18	**24.9**	**0.84**	**0.18**
	STG [3]	13.6	0.63	0.40	21.9	**0.84**	0.21
	SOM [4]	16.7	0.75	0.23	20.7	0.79	0.22
Multiview	D-3DGS [2]	**22.2**	**0.89**	**0.10**	**27.7**	**0.90**	**0.12**
	4DGS [1]	21.0	0.85	0.15	26.6	0.87	0.15
	STG [3]	17.2	0.70	0.36	21.1	0.81	0.25

[1] D-3DGS: `https://github.com/ingra14m/Deformable-3D-Gaussians`
[2] STG: `https://github.com/oppo-us-research/SpacetimeGaussians`
[3] 4DGS: `https://github.com/hustvl/4DGaussians`
[4] SOM: `https://github.com/vye16/shape-of-motion`

A.3.3 Performance breakdown: monocular setting

We provide per-scene breakdown performance of monocular video reconstruction setting in Table 6. We report the performance of D-3DGS [2], 4DGS [1], STG [3], and SOM [4], which serve as the most common and recent baselines for the DyNVS task.

Table 6: Per-scene breakdown results for all 17 scenes under the monocular setting.

Method	Cereal			Ship			Hanok		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	23.3	0.88	0.15	25.9	0.90	0.16	16.3	0.78	0.28
4DGS [1]	26.3	0.90	0.14	25.9	0.91	0.15	15.5	0.74	0.32
STG [3]	15.8	0.54	0.54	22.8	0.87	0.23	14.8	0.60	0.41
SOM [4]	21.2	0.81	0.22	23.8	0.87	0.23	14.7	0.67	0.40
Method	**Ice**			**Pisa**			**Box-smoke**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	25.3	0.92	0.30	20.7	0.72	0.26	20.7	0.96	0.13
4DGS [1]	29.2	0.92	0.29	19.3	0.67	0.26	21.3	0.96	0.12
STG [3]	23.1	0.87	0.39	20.1	0.68	0.33	22.7	0.95	0.16
SOM [4]	18.7	0.84	0.41	17.1	0.65	0.41	23.8	0.95	0.18
Method	**Single smoke**			**Falling**			**Jelly party**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	26.5	0.97	0.08	19.6	0.90	0.18	16.3	0.81	0.21
4DGS [1]	26.1	0.97	0.08	20.1	0.90	0.19	14.9	0.70	0.28
STG [3]	24.5	0.97	0.10	20.2	0.77	0.37	11.1	0.51	0.44
SOM [4]	23.9	0.95	0.18	15.3	0.81	0.33	14.9	0.69	0.28
Method	**Pancake**			**Bouncing balls**			**Cow**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	22.8	0.88	0.13	19.8	0.77	0.15	21.6	0.92	0.10
4DGS [1]	17.0	0.78	0.22	22.7	0.86	0.10	23.6	0.92	0.11
STG [3]	10.6	0.60	0.39	11.9	0.51	0.60	20.6	0.87	0.16
SOM [4]	13.9	0.68	0.34	17.4	0.79	0.14	20.7	0.85	0.17
Method	**Lucy**			**Basin**			**Flags**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	22.8	0.92	0.10	18.2	0.67	0.36	23.6	0.94	0.12
4DGS [1]	27.5	0.94	0.08	18.0	0.68	0.38	31.9	0.96	0.08
STG [3]	19.6	0.86	0.19	16.8	0.66	0.43	24.7	0.90	0.17
SOM [4]	21.0	0.90	0.17	16.3	0.61	0.43	27.4	0.93	0.12
Method	**Single flag**			**Tube**			**Average**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	18.3	0.68	0.24	27.7	0.96	0.05	21.7	**0.86**	**0.18**
4DGS [1]	18.0	0.65	0.29	28.8	0.96	0.08	**22.7**	0.85	0.19
STG [3]	26.0	0.87	0.14	22.1	0.92	0.12	19.3	0.76	0.30
SOM [4]	16.3	0.60	0.24	22.5	0.93	0.15	19.3	0.80	0.26

A.3.4 Performance breakdown: multiview setting

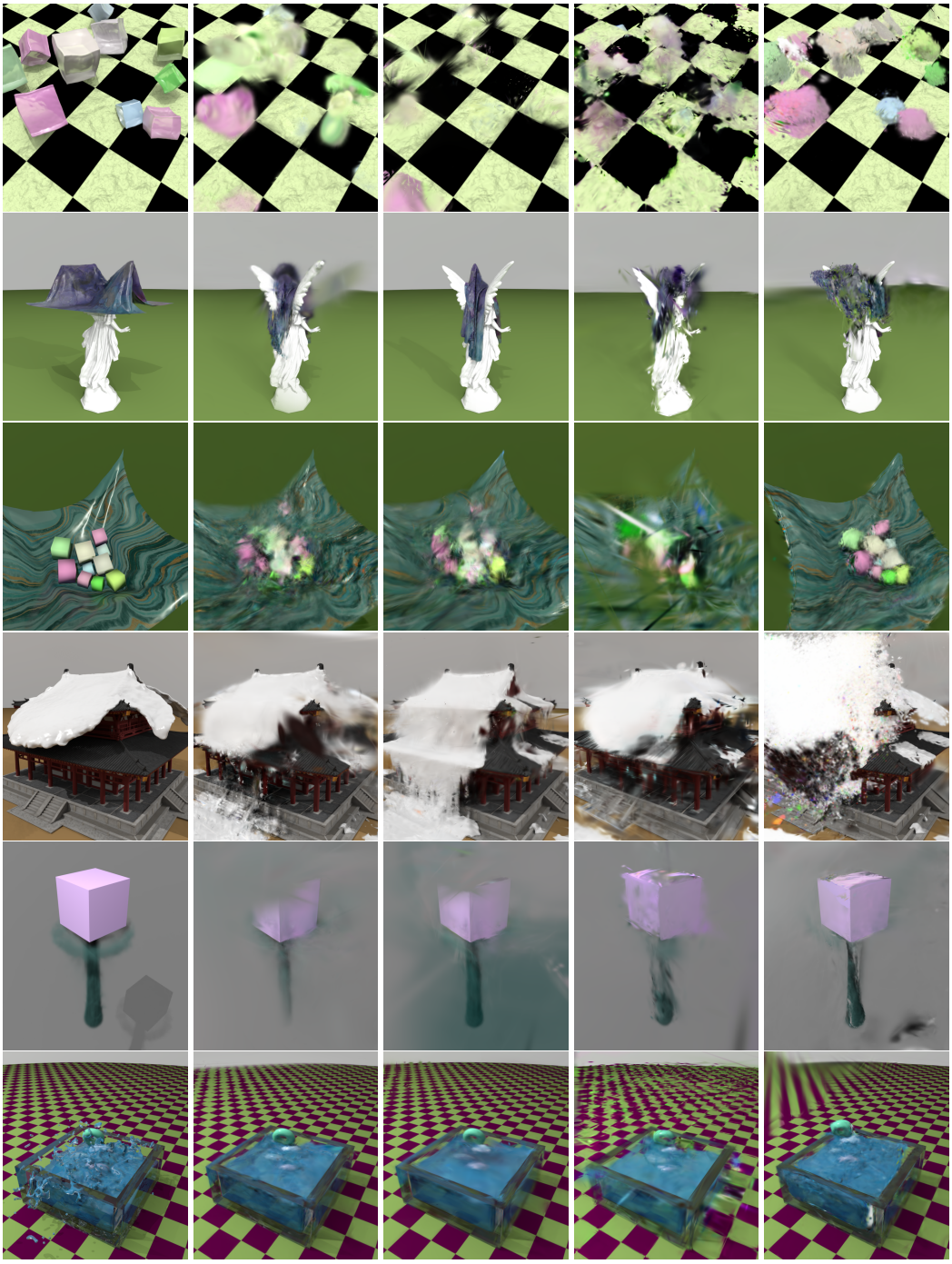
We provide per-scene breakdown performance for the multiview video reconstruction setting in Table 7. We report results for D-3DGS [2], 4DGS [1], and STG [3], which serve as the most common and recent baselines for the DyNVS task. Note that since SOM [4] is specialized for monocular video setups, we omit its performance in the multiview evaluation.

Table 7: Per-scene breakdown results for all 17 scenes under the multiview setting.

Method	Cereal			Ship			Hanok		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	28.1	0.93	0.11	28.8	0.93	0.11	15.6	0.76	0.31
4DGS [1]	27.6	0.92	0.13	25.6	0.91	0.15	15.8	0.75	0.33
STG [3]	15.8	0.58	0.59	25.9	0.90	0.20	15.7	0.65	0.43
Method	**Ice**			**Pisa**			**Box-smoke**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	16.4	0.86	0.43	22.3	0.75	0.23	20.5	0.96	0.11
4DGS [1]	31.4	0.93	0.28	22.8	0.74	0.24	22.6	0.97	0.07
STG [3]	25.9	0.89	0.38	28.7	0.90	0.18	19.0	0.95	0.17
Method	**Single-smoke**			**Falling**			**Jelly party**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	26.4	0.98	0.07	25.7	0.95	0.11	18.2	0.87	0.16
4DGS [1]	28.2	0.98	0.07	23.4	0.87	0.30	16.1	0.80	0.24
STG [3]	27.0	0.98	0.07	25.1	0.83	0.34	12.5	0.60	0.39
Method	**Pancake**			**Bouncing balls**			**Cow**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	25.3	0.90	0.08	25.7	0.91	0.07	19.8	0.87	0.10
4DGS [1]	18.7	0.84	0.18	25.0	0.84	0.09	24.1	0.92	0.10
STG [3]	11.7	0.65	0.36	12.9	0.59	0.54	31.8	0.95	0.15
Method	**Lucy**			**Basin**			**Flags**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	30.2	0.96	0.06	23.6	0.82	0.24	30.5	0.96	0.09
4DGS [1]	28.6	0.95	0.07	20.5	0.75	0.33	32.8	0.96	0.08
STG [3]	21.2	0.88	0.19	17.5	0.70	0.45	26.0	0.91	0.17
Method	**Single flag**			**Tube**			**Average**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
D-3DGS [2]	21.6	0.81	0.14	32.6	0.98	0.05	24.2	**0.89**	**0.14**
4DGS [1]	19.7	0.72	0.21	31.3	0.97	0.07	**24.4**	0.87	0.17
STG [3]	16.9	0.62	0.36	23.8	0.93	0.11	21.0	0.79	0.30

A.4 Additional Qualitative Results

Figure 8 shows additional qualitative results, where all methods fail to accurately capture multi-body interactions, producing blurred or under-reconstructed outputs. For the 4DGS [1], using the default grid learning rate frequently causes NaN values in the loss. Reducing the learning rate stabilizes training but results in poor dynamic capture, with outputs resembling static scenes.



(a) GT image (b) D-3DGS [2] (c) 4DGS [1] (d) STG [3] (e) SOM [4]

Figure 8: Qualitative results of recent DyNVS methods on the *Jelly party*, *Lucy*, *Basin*, *Hanok*, *Box-smoke*, and *Cereal*s scenes with monocular training setup. These results show that all methods frequently exhibit needle-like artifacts and fail to reconstruct dynamic elements accurately.

B Details of PhysGaia

B.1 Scene Lists

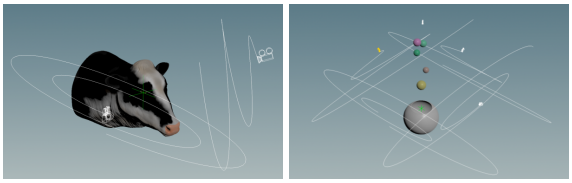
Our dataset consists of 17 scenes divided into four categories: liquid, gas, viscoelastic substances, and textile. Each category contains 4 to 5 scenes, and the detailed scene names are listed in Table 8.

Table 8: List of scenes included in our PhysGaia dataset

	Liquid	**Gas**	**Viscoelastic substance**	**Textile**
Scene name	Cereal	Pisa	Jelly party	Lucy
	Ship	Box-smoke	Pancake	Basin
	Hanok	Single smoke	Bouncing balls	Flags
	Ice	Falling	Cow	Single flag
	–	–	–	Tube

B.2 Camera Trajectories

Figure 9 visualizes our training camera trajectories, where we adopt either 2 or 4 cameras depending on the properties of each scene. For the *Cow*, *Single flag*, and *Tube* scenes–which resemble 180° settings with background walls placed behind the objects–we use only 2 cameras, as shown in Figure 9a. In contrast, for 360° scenes, we place 4 cameras at evenly spaced viewpoints around the scene, enabling full coverage from four directions. All cameras follow spiral trajectories to capture diverse views over time. For the monocular setup, we select one of the training cameras to generate the training data.



(a) 180° scene: 2 cams (b) 360° scene: 4 cams

Figure 9: Visualization of training camera's traj.

B.3 Implementation Details

All simulations were conducted on 4 NVIDIA Titan Xp GPUs and Intel Xeon E5-2630 v4 CPUs with 40 cores. While most scenes rendered in 2–3 hours, the *Lucy* and *Hanok* scenes required 12–24 hours due to their complex geometry and increased simulation costs. To supplement Section 4, we provide additional detailed simulation implementation along with the value of physics parameters. These parameters can be served as ground-truth labels for downstream physical reasoning tasks.

Liquid We assume incompressible, non-Newtonian fluids in our simulations. Table 9 summarizes the density and viscosity values, which are the most important factors governing fluid motion. External forces are limited to gravity. For the *Ship* scene, we reduced the particle separation from 0.1 to 0.05 to decrease initial particle displacement and improve simulation accuracy. Also, the particle separation during dynamic simulation is set to 0.1 for the *Cereal* scene and 0.05 for the *Ship* scene.

Table 9: Physics parameters used for liquid materials.

Scene name	**Object**	**Density** (kg/m^3)	**Viscosity**
Ship	Water	800	0
	Ship	300	-
Cereal	Water	1000	0
	Cereal	1000	-
Hanok	Snow	1000	10
Ice	Water	1000	0

Gas In gas simulations, temperature is the most critical physical parameter, governing buoyancy, expansion, and flow dynamics. We initialized the temperature to 3000K for gas materials. As temperature evolves locally due to gas movement and interactions, its spatiotemporal variation is represented as a temperature field defined on a voxel grid. These temperature fields are accessible via the provided simulation source file with Python API, similar to how flow fields are accessed for generating ground-truth trajectory data.

Viscoelastic substances Gravity is applied to all scenes; however, for the *Cow* scene, we introduce additional internal forces to induce cow motion and increase the complexity of the physical reasoning scenarios. Boundaries are set to be open in all directions, allowing objects to move freely without collisions against invisible walls. Table 10 lists detailed physics parameters, including Young's modulus (E) and Poisson's ratio (P), used in the simulations.

Table 10: Physics parameters used for viscoelastic materials.

Scene name	Object	Type	E	P	Viscosity
Pancake	Honey	Viscous	2.5×10^5	0.10	0.125
	Pancakes	Chunky	8.0×10^4	0.23	-
Bouncing balls	Balls	Elastic	1.0×10^5	0.50	-
	Fishbowl	Static	-	-	-
Jelly party	Jelly	Elastic	8.0×10^4	0.45	-
Cow	Cow	Elastic	5.0×10^4	0.45	-

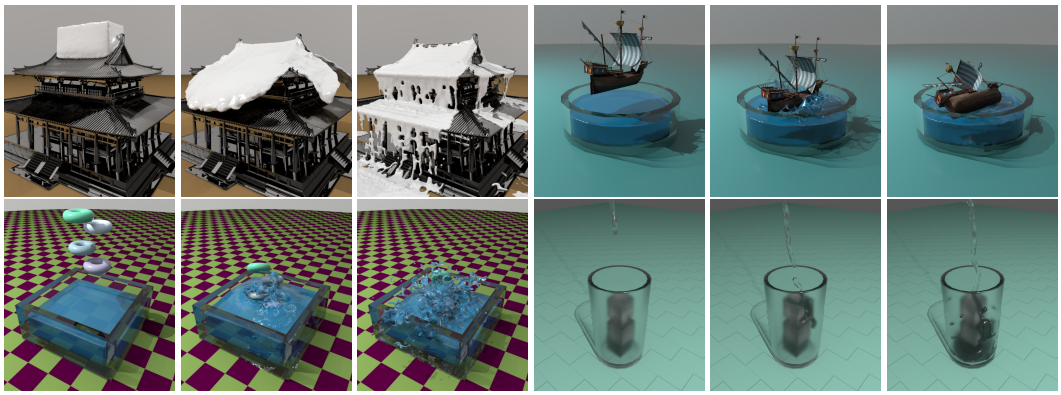
Textile For textiles, we primarily adopt the default configuration from the Vellum solver, with detailed modifications summarized in Table 11.

Table 11: Physics parameters used for textile materials.

Scene name	Object	Stretch		Bend	
		Stiffness	Damping Ratio	Stiffness	Damping Ratio
Flags	Flag1 (small)	1.0×10^{10}	1.0×10^{-3}	1.0×10^{-4}	1.0×10^{-2}
	Flag2 (big)	1.0×10^{11}	1.0×10^{-3}	1.0×10^{-4}	1.0×10^{-2}
Basin	Boxes	1.0×10^{10}	1.0×10^{-3}	N/A	N/A
	Cloth	1.0×10^{13}	1.0×10^{-3}	1.0×10^{-4}	1.0×10^{-2}
Single flag	Flag	1.0×10^{10}	1.0×10^{-3}	1.0×10^{-4}	1.0×10^{-2}
Lucy	Cloth	1.0×10^{10}	1.0×10^{-3}	1.0×10^{-4}	1.0×10^{-2}
Tube	Tube flag	1.0×10^{10}	1.0×10^{-3}	1.0×10^{-4}	1.0×10^{-2}

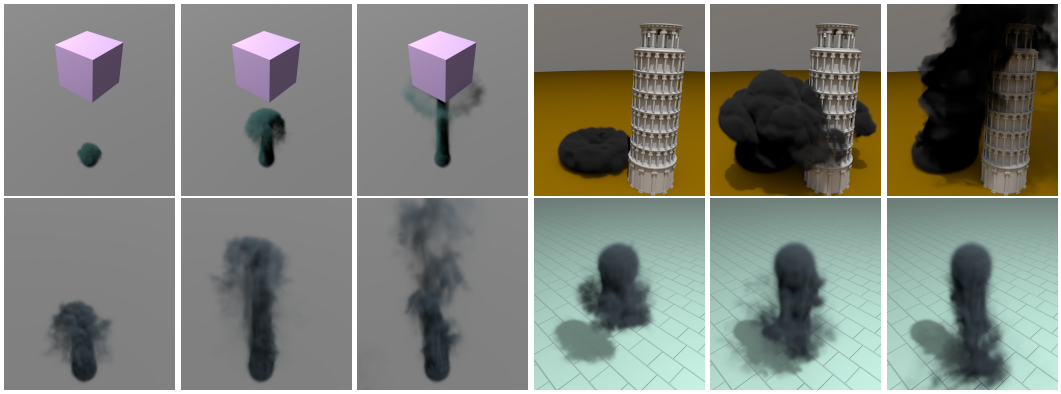
B.4 Scene Visualization

Figure 10 and Figure 11 visualize more scenes in PhysGaia to supplement the Figure 1 in our main paper. As shown in Table 2 of the main paper, textile materials remain underexplored in physics-based datasets. To facilitate progressive research development in this domain, we include few simpler scenes such as *single flag* and *tube* that employ with only basic wind interactions. We believe this approach effectively supports advancements by providing more accessible starting points. Similarly, our dataset also includes few simpler gas scenes, *simple smoke* and *falling*, for the same reason as with textile scenarios: to facilitate progressive development. For the *cow* scene, internal forces intrinsic to the cow itself introduce unique and more complex dynamics, rather than using only gravity.

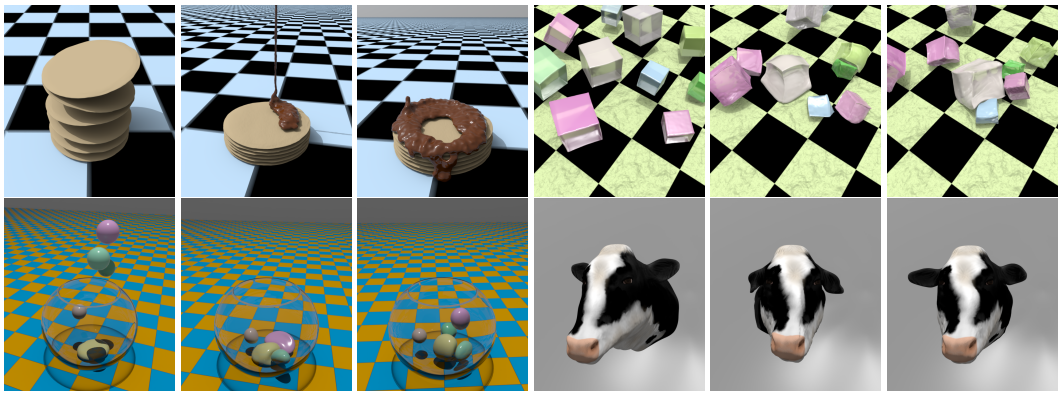


(a) Liquid: *Hanok* (top-left), *Ship* (top-right), *Cereal* (bottom-left), *Ice* (bottom-right)

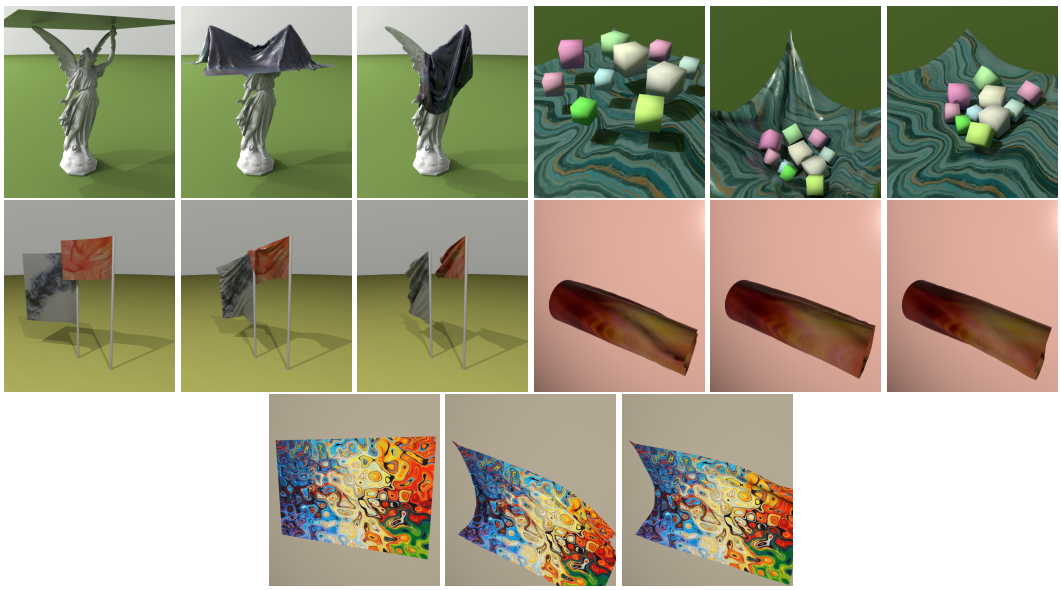
Figure 10: Examples from the proposed physics-aware dataset, PhysGaia.



(a) Gas: *Box-smoke* (top-left), *Pisa* (top-right)), *Simple smoke* (bottom-left), and *Falling* (bottom-right)



(b) Viscoelastic substances: *Pancake* (top-left), *Jelly party* (top-right), *Bouncing balls* (bottom-left), and *Cow* (bottom-right)



(c) Textile: *Lucy* (top-left), *Basin* (top-right), *Flags* (mid-left), *Tube* (mid-right), and *Singe flag* (bottom)

Figure 11: Examples from the proposed physics-aware dataset, PhysGaia. They exhibit complex physical interactions between multiple bodies composed of diverse materials such as liquid, gas, viscoelastic substance, and textile. This dataset will foster physics reasoning in dynamic scenes.

B.5 License

Our dataset is released under the Creative Commons Attribution-NonCommercial (CC BY-NC) license. The mesh used for the *Lucy* scene is sourced from the Stanford 3D Repository, which permits usage for research purposes. Other mesh objects are licensed under CC-BY 4.0, as detailed in Table 12. For the texture maps in the textile categories, we utilized images from Pixabay, which are freely available with contributor consent and comparable to a CC-BY 4.0 license, also summarized in Table 12. Therefore, all data included in our dataset comply with usage rights and do not pose intellectual property issues.

Table 12: License of the sources used for generating data.

Scene	Resource name	License	Access
Ice	Glass cup	CC BY 4.0	Sketchfab
Hanok	Korean building	CC BY 4.0	Sketchfab
Ship	Ship	CC BY 4.0	Sketchfab
Pisa	Torre Pisa	CC BY 4.0	Sketchfab
Bouncing balls	Fish bowl	CC BY 4.0	Sketchfab
Cow	Cow	CC BY 4.0	Sketchfab
Lucy	Lucy	research only	Stanford 3D Scan Repo
Scene	**Contributor**	**License**	**Access**
Flags	PaftDrunk	Content License	Pixabay
	Zoeysmom	Content License	Pixabay
Single flag	–	Content License	Pixabay
Lucy	WalterClark	Content License	Pixabay
Tube	–	Content License	Pixabay
Basin	Yourialka	Content License	Pixabay

C Limitations & Broad Impact

Limitations Although our *PhysGaia* dataset holds a crucial position in the DyNVS field by enabling accurate evaluation with ground-truth physical properties, it is inherently limited to synthetic scenarios. To bridge this gap towards more realistic applications, we believe domain adaptation approaches, such as those from Syn2Real [8], could be adopted. However, such methods must be carefully designed to maintain multi-view consistency across frames. We leave this important domain transfer for future work, confident that our dataset's precise physical information will be invaluable for developing these advanced techniques.

Broad impact From a positive perspective, our work advances research on physically plausible 4D reconstruction. This, in turn, significantly improves monocular video reconstruction, an essential technology for future AR/VR applications. However, such progress in 4D reconstruction, especially given its potential for scene editing of existing videos, may raise intellectual property concerns regarding the original video content.

References

[1] Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X.: 4d gaussian splatting for real-time dynamic scene rendering. In CVPR. (2024)

[2] Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In CVPR. (2024)

[3] Li, Z., Chen, Z., Li, Z., Xu, Y.: Spacetime gaussian feature splatting for real-time dynamic view synthesis. In CVPR. (2024)

[4] Wang, Q., Ye, V., Gao, H., Austin, J., Li, Z., Kanazawa, A.: Shape of motion: 4d reconstruction from a single video (2024) arXiv preprint arXiv:2407.13764.

[5] Park, K., Sinha, U., Hedman, P., Barron, J.T., Bouaziz, S., Goldman, D.B., Martin-Brualla, R., Seitz, S.M.: Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. ACM Trans. Graph. (2021)

[6] Gao, H., Li, R., Tulsiani, S., Russell, B., Kanazawa, A.: Monocular dynamic view synthesis: A reality check. In NeurIPS. (2022)

[7] Laboratory, S.C.G.: The stanford 3d scanning repository. http://graphics.stanford.edu/data/3Dscanrep/ (1994)

[8] Peng, X., Usman, B., Saito, K., Kaushik, N., Hoffman, J., Saenko, K.: Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. CoRR (2018)