

UNIVERSITY OF CALIFORNIA

Los Angeles

Physics-Informed Methods for Generative Modeling and Dynamics

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Computer Science

by

Tianyi Xie

2026

© Copyright by
Tianyi Xie
2026

ABSTRACT OF THE DISSERTATION

Physics-Informed Methods for Generative Modeling and Dynamics

by

Tianyi Xie

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 2026

Professor Demetri Terzopoulos, Chair

AI-driven content generation has transformed creative industries ranging from entertainment and gaming to education. Yet most generative models learn purely from data, lacking an intrinsic understanding of physics. The resulting outputs frequently exhibit unrealistic structures, implausible dynamics, and nonphysical artifacts, limiting their use in applications that demand physical consistency. This dissertation tackles this fundamental limitation by integrating physics-based simulation into generative modeling, enhancing the plausibility, stability, and coherence of synthesized content.

We begin with the foundations of physics-based simulation, developing a robust and efficient method for coupling deformable solids and fluids in a unified Lagrangian framework, achieving a penetration-free guarantee while significantly improving computational efficiency over prior monolithic approaches. Building on differentiable simulation, we then tackle the problem of self-supportability in generative 3D asset creation, embedding standability constraints under gravity, contact, and friction directly into text-to-3D optimization pipelines, and validating the results through real-world fabrication. Moving beyond static assets, we address the challenge of endowing multi-view reconstructions with physically grounded dynamics, integrating Newtonian continuum mechanics into the 3D Gaussian representation to synthesize novel 4D motions across diverse material types, and further extending this to real-time virtual reality through efficient deformable body simulation and two-level mesh embedding. Finally, we explore the synergy between

physics-based simulation and video foundation models for animation and interaction generation. We address the problem of generating physically plausible cartoon animations from static illustrations by coupling image-space deformable simulation with guided video diffusion. We then tackle the inverse problem of recovering physical parameters and forces from generated videos via differentiable simulation-based tracking, enabling faithful reproduction of complex 3D character dynamics. Extending this paradigm to humanoid loco-manipulation, we address the challenge of scalable 4D human-object interaction generation and physically plausible skill learning, producing a large-scale dataset of diverse interaction sequences and learning generalizable humanoid control policies through reinforcement learning in physics simulation. Together, these contributions demonstrate that embedding physical priors into generative pipelines yields content that is not only visually compelling but also physically consistent, thereby opening new avenues for computer graphics, embodied AI, and robotics.

The dissertation of Tianyi Xie is approved.

Bolei Zhou

Quanquan Gu

Chenfanfu Jiang

Demetri Terzopoulos, Committee Chair

University of California, Los Angeles

2026

To my family and friends.

TABLE OF CONTENTS

1	Introduction	1
1.1	Thesis Overview and Contributions	2
2	Efficient and Robust Solid-Fluid Coupling via Physics-Based Simulation	8
2.1	Introduction	8
2.2	Related Work	10
2.3	Formulation	13
2.3.1	Solid Domain	14
2.3.2	Fluid Domain	14
2.3.3	Coupling	17
2.3.4	Optimization Time Integrator	19
2.4	Efficient Solver	19
2.4.1	Time Splitting	20
2.4.2	Solving Linear Systems	24
2.5	Experiments	25
2.5.1	Ablation Study	27
2.5.2	Comparisons	31
2.5.3	Complex Scenarios	33
2.6	Conclusions and Future Work	37
3	Self-Supporting Text-to-3D Generation with Physics Constraints	39
3.1	Introduction	39
3.2	Related Work	42
3.3	Background	44

3.4	Method	45
3.4.1	Physics Incorporation	45
3.4.2	Additional Regularization	49
3.4.3	Method Overview	50
3.5	Experiments	51
3.5.1	Simulation Verification	51
3.5.2	Real-World Validation:	57
3.6	Conclusions	58
4	Physics-Integrated 4D Generative Dynamics for Multi-View Reconstruction	60
4.1	PhysGaussian: Physics-Integrated 3D Gaussians for Generative Dynamics	60
4.1.1	Introduction	60
4.1.2	Related Work	63
4.1.3	Method Overview	65
4.1.4	Experiments	72
4.1.5	Discussion	79
4.2	VR-GS: A Physical Dynamics-Aware Interactive Gaussian Splatting System in Virtual Reality	80
4.2.1	Introduction	81
4.2.2	Related Work	83
4.2.3	System Design	85
4.2.4	Method	87
4.2.5	Evaluation	92
4.2.6	Experiment	94
4.2.7	User Study	100

4.2.8	Conclusions and Future Work	102
5	Physics-Guided Animation Generation and Humanoid Loco-Manipulation	
		104
5.1	PhysAnimator: Physics-Guided Generative Cartoon Animation	104
5.1.1	Introduction	104
5.1.2	Related Work	106
5.1.3	Method	109
5.1.4	Experiments	115
5.1.5	Conclusions	121
5.2	AniMimic: Imitating 3D Animation From Video Priors	122
5.2.1	Introduction	122
5.2.2	Related Work	124
5.2.3	Method	127
5.2.4	Experiments	133
5.2.5	Conclusions	138
5.3	GRAIL: Generating Humanoid Loco-Manipulation From Video Foundation	
	Models	139
5.3.1	Introduction	139
5.3.2	Related Work	142
5.3.3	Method	144
5.3.4	Experiments	151
5.3.5	Discussion	157
6	Conclusions	159
A	Lagrangian Solid Fluid Coupling Technical Details	161

A.1	Derivatives of Fluid Potentials	161
A.2	Time Splitting	162
A.2.1	Baseline Time Splitting	162
A.2.2	Error Analysis	164
B	Atlas3D Additional Experiments	166
B.1	Implementation and Training Details	166
B.2	Additional Results	167
B.2.1	Qualitative Comparison With the Magic3D Baseline	167
B.2.2	Additional Comparison With the MVDream Baseline	167
B.2.3	Quantitative Comparison With the Magic3D Baseline	167
B.3	Real World Experiments	167
B.3.1	Robot Manipulation Results	167
B.3.2	User Study Results	171
C	PhysGaussian Technical Details and Additional Evaluations	173
C.1	MPM Algorithm	173
C.2	Elasticity and Plasticity Models	174
C.2.1	Fixed Corotated Elasticity	175
C.2.2	StVK Elasticity	176
C.2.3	Neo-Hookean Elasticity	176
C.2.4	Drucker-Prager Plasticity	176
C.2.5	Von-Mises Plasticity	176
C.2.6	Herschel-Bulkley Plasticity	177
C.3	Additional Evaluations	177

D PhysAnimator Technical Details and Additional Experiments	179
D.1 Motion Equation	179
D.2 Details of Interactive Animation	180
D.3 Controllable Generation	184
D.4 Ablation Study	185
D.5 More Comparison Results	185
E AniMimic Technical Details and Additional Results	187
E.1 Differentiable Simulation Details	187
E.2 Additional Results	189
F GRAIL Technical Details and Additional Results	193
F.1 Qualitative Results	193
F.2 Implementation Details	193
F.2.1 Runtime Performance	193
F.2.2 Video Generation	194
F.2.3 Initial Motion Estimation	195
F.2.4 Interaction-Aware Trajectory Refinement	195
F.2.5 Physics-based Humanoid Skill Learning	196
F.2.6 Scene-Aware Humanoid Control	198
F.2.7 Failure Case Filtering	198
F.3 Experiment Details	199
F.3.1 Experiment Settings	199
F.3.2 Additional Results	200
References	204

LIST OF FIGURES

1.1	Kicking water	3
1.2	3D-printed figurines	3
1.3	PhysGaussian pipeline	4
1.4	VR-GS pipeline	5
1.5	PhysAnimator overview	5
1.6	AniMimic pipeline overview	6
1.7	GRAIL pipeline overview	7
2.1	Timing breakdown of simulations	26
2.2	Bob simulated with different time stepping schemes	27
2.3	Viscous objects on cloth simulated with different time stepping schemes	28
2.4	Statistics of simulations with different stiffness parameter	30
2.5	Shoot armadillo	30
2.6	Dam break	32
2.7	Liquid Bunnies	33
2.8	Buoyancy	34
2.9	Varying friction	34
2.10	Twist cylinder	34
2.11	Cream	35
2.12	Angry cow	36
3.1	Simulation verification	42
3.2	2D illustration of stable equilibrium and unstable equilibrium	48
3.3	Comparison with Magic3D	51

3.4	Enhance MVDream with stable standability	52
3.5	Ablation study	53
3.6	Success rate of models standing under perturbation	53
3.7	Standability evaluation on uneven surfaces	55
3.8	TRD comparison	56
3.9	Standability test using a robotic arm	58
4.1	PhysGaussian overview	61
4.2	Gaussian kernel deformation	68
4.3	Spherical harmonics rotation	69
4.4	Internal filling illustration	71
4.5	Material Versatility	73
4.6	PhysGaussian comparison study	75
4.7	PhysGaussian ablation study	77
4.8	Internal filling	78
4.9	Volume conservation	79
4.10	Anisotropy regularizer	79
4.11	VR-GS overview	81
4.12	Two-level embedding illustration	90
4.13	Two-level embedding evaluation	93
4.14	Inpainting evaluation	93
4.15	Shadow map evaluation	94
4.16	Trade-offs between quality and performance	95
4.17	Visual quality comparison	95
4.18	Timing breakdown of demos	97

4.19	Fox, bear, and horse	98
4.20	Ring toss and table brick games	98
4.21	Toy collection	99
4.22	Just dance	99
4.23	User study	102
5.1	Geometry registration	112
5.2	Qualitative comparison	117
5.3	Sketch-guided rendering	120
5.4	Complementary dynamics enhancement	121
5.5	AniMimic qualitative comparison	135
5.6	AniMimic novel view synthesis	137
5.7	AniMimic ablation studies	137
5.8	AniMimic physics-based refinement	138
5.9	Diverse loco-manipulation skills on the Unitree G1	152
5.10	GRAIL qualitative comparison of 4D HOI generation	154
B.1	Atlas3D additional comparison with Magic3D	168
B.2	Atlas3D additional comparison with Magic3D	169
B.3	Atlas3D additional comparison with Magic3D	170
B.4	Atlas3D additional comparison with MVDream	171
B.5	Atlas3D robot manipulation experiment	172
C.1	Physgaussian additional evaluation	178
D.1	Flow particles	181
D.2	PhysAnimator additional qualitative comparison	182

D.3	PhysAnimator additional qualitative comparison	183
D.4	PhysAnimator additional qualitative comparison	184
D.5	Controllable generation with material property	185
D.6	Controllable generation with energy stroke	186
E.1	AniMimic additional qualitative comparison	190
E.2	AniMimic varying motion generation	191
E.3	AniMimic novel views	192
F.1	GRAIL diverse results overview	194
F.2	GRAIL candidate scene configurations	195
F.3	GRAIL visualization for physical plausibility evaluation	201
F.4	GRAIL additional qualitative comparison	202

LIST OF TABLES

2.1	Simulation statistics	26
2.2	Statistics of different time stepping schemes	27
2.3	Comparison of different linear system solvers	31
3.1	Comparison of success rate under perturbation (goose).	54
3.2	Quantitative evaluation	57
4.1	Lattice deformation benchmark	76
4.2	VR-GS frame time comparison	96
4.3	Parameters and timings of demos	97
5.1	Quantitative comparisons	118
5.2	User study	119
5.3	AniMimic quantitative comparisons	136
5.4	AniMimic user study	136
5.5	GRAIL quantitative comparison of 4D HOI generation	154
5.6	GRAIL physical plausibility comparison	155
5.7	GRAIL user study on 4D HOI quality	155
5.8	GRAIL ablation study on reconstruction quality	155
5.9	GRAIL ablation study on humanoid skill learning	157
B.1	Number of successes in robotic trials	167
B.2	Number of successes in user studies	171
C.1	PhysGaussian model settings	175
C.2	Material parameters	175

D.1 Ablation of dynamics enhancement	184
F.1 GRAIL runtime breakdown	194
F.2 GRAIL user study on 4D HOI quality	200

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisors, Professor Chenfanfu Jiang and Professor Demetri Terzopoulos, for their guidance and support throughout my doctoral studies. Professor Jiang’s mentorship has profoundly shaped both my research and my growth as a researcher. He taught me to look beyond the immediate and envision what is possible, a perspective that has been transformative for my work. His encouragement gave me the opportunity to pursue my PhD at UCLA, and without his support, none of this work would have been possible. I am equally grateful to Professor Terzopoulos for his unwavering support and guidance throughout this journey, and for encouraging me to pursue a PhD in the first place.

I would also like to thank my thesis committee members, Professor Bolei Zhou and Professor Quanquan Gu, for their valuable feedback and service on my committee.

I am thankful for the mentorship and collaboration I received during my industry internships, which broadened my research perspective and exposed me to real-world applications. I would like to especially thank Yiwei Zhao at Netflix, and Dr. Ye Yuan, Dr. Haotian Zhang, and Dr. Umar Iqbal at NVIDIA Research, for their tremendous support, mentorship, and collaboration on my internship projects. I also thank my other colleagues and collaborators at NVIDIA: Dr. Bowen Wen, Dr. Jiefeng Li, Dr. Zhengyi Luo, Dr. Xueting Li, Jinhyung Park, and Zi Wang. Their expertise and collaborative spirit made my internship experience both productive and inspiring.

I owe a special thanks to Professor Minchen Li for his mentorship and for teaching me the foundations of physics-based simulation research. I am also grateful to my collaborators and friends from the AIVC Lab: Yunuo Chen, Dr. Zeshun Zong, Dr. Xuan Li, Dr. Yuxing Qiu, Chang Yu, Yaowei Guo, Wenxin Du, Siyu Ma, Boqian Li, Xiyang Tan, Zeng Tao, Dr. Yadi Cao, Dr. Yu Fang, and Dr. Ziyin Qu. Together, we have gone through countless deadlines, fought through challenging problems, and explored new ideas side by side. Their camaraderie and support have been an essential part of my PhD experience. I also thank Professor Yin Yang, Professor Ying Nian Wu, Dr. Feng Gao,

Hritik Bansal, and Zongyu Lin for their collaboration and contributions to this work.

Lastly, I am deeply grateful to my parents for their unconditional love and support throughout my academic journey. I also thank my partner, Ying Jiang, for her companionship and encouragement, especially during the moments when I was struggling and feeling down. Her unwavering support helped me persevere through the most challenging times. Finally, I thank our lovely cats, Tonton and Zhuzhu, whose companionship has brought me so much happiness.

VITA

2025–2026	Research Intern, NVIDIA Research
2024–2026	Teaching Assistant, Computer Science Department, UCLA
2024	Research Intern, Netflix Research
2023–2024	Teaching Assistant, Department of Mathematics, UCLA
2023	Graduate Student Researcher, Department of Mathematics, UCLA
2023	M.S., Computer Science, UCLA
2020–2021	Research Intern, Bytedance AI Lab
2016–2020	B.S., Software Engineering, Shanghai Jiao Tong University

PUBLICATIONS

* indicates co-first authors.

Y. Chen, **T. Xie**, C. Yuksel, D. Kaufman, Y. Yang, C. Jiang, and M. Li. Multi-Layer Thick Shells. *ACM SIGGRAPH*, 2023.

T. Xie, M. Li, Y. Yang, and C. Jiang. A Contact Proxy Splitting Method for Lagrangian Solid-Fluid Coupling. *ACM Transactions on Graphics (SIGGRAPH)*, 2023.

T. Xie*, Z. Zong*, Y. Qiu*, X. Li*, Y. Feng, Y. Yang, and C. Jiang. PhysGaussian: Physics-Integrated 3D Gaussians for Generative Dynamics. *CVPR*, 2024.

Y. Jiang*, C. Yu*, **T. Xie***, X. Li*, Y. Feng, H. Wang, M. Li, H. Lau, F. Gao, Y. Yang, and C. Jiang. VR-GS: A Physical Dynamics-Aware Interactive Gaussian Splatting System in Virtual Reality. *ACM SIGGRAPH*, 2024.

Y. Chen*, **T. Xie***, Z. Zong*, X. Li, F. Gao, Y. Yang, Y. N. Wu, and C. Jiang. Atlas3D: Physically Constrained Self-Supporting Text-to-3D for Simulation and Fabrication. *NeurIPS*, 2024.

H. Bansal*, Z. Lin*, **T. Xie**[^], Z. Zong[^], M. Yarom[~], Y. Bitton[~], C. Jiang, Y. Sun, K. Chang, and A. Grover. VideoPhy: Evaluating Physical Commonsense in Video Generation. *ICLR*, 2025.

T. Xie, Y. Zhao, Y. Jiang, and C. Jiang. PhysAnimator: Physics-Guided Generative Cartoon Animation. *CVPR*, 2025.

S. Ma*, W. Du*, C. Yu*, Y. Jiang*, Z. Zong, **T. Xie**, Y. Chen, Y. Yang, X. Han, and C. Jiang. GRIP: A General Robotic Incremental Potential Contact Simulation Dataset for Unified Deformable-Rigid Coupled Grasping. *IROS*, 2025.

B. Li*, X. Li*, Y. Jiang*, **T. Xie**, F. Gao, H. Wang, Y. Yang, and C. Jiang. Garment-Dreamer: 3DGS Guided Garment Synthesis with Diverse Geometry and Texture Details. *3DV*, 2025.

X. Li*, C. Yu*, W. Du*, Y. Jiang*, **T. Xie**, Y. Chen, Y. Yang, and C. Jiang. Dress-1-to-3: Single Image to Simulation-Ready 3D Outfit with Diffusion Prior and Differentiable Physics. *ACM Transactions on Graphics (SIGGRAPH)*, 2025.

X. Tan*, Y. Jiang*, X. Li*, Z. Zong, **T. Xie**, Y. Yang, and C. Jiang. PhysMotion: Physics-Grounded Dynamics From a Single Image. *3DV*, 2026.

T. Xie*, Y. Chen*, Y. Guo*, Y. Yang, B. Zhou, D. Terzopoulos, Y. Jiang, and C. Jiang. AnimaMimic: Imitating 3D Animation from Video Priors. *CVPR*, 2026.

T. Xie*, H. Zhang*, Z. Wang, J. Park, B. Wen, J. Li, X. Li, H. Weng, D. Minor, T. Wang, J. Kautz, C. Jiang, L. Fan, Y. Zhu, Z. Luo, U. Iqbal, and Y. Yuan. GRAIL: Generating Humanoid Loco-Manipulation from Video Foundation Models. In submission, 2026.

Z. Tao*, Y. Jiang*, Y. Chen*, **T. Xie**, H. Wang, Y. N. Wu, Y. Yang, A. S. Kumar, K. Tashiro, and C. Jiang. DressWild: Feed-Forward Pose-Agnostic Garment Sewing Pattern Generation from In-the-Wild Images. *arXiv:2602.16502*, 2026.

CHAPTER 1

Introduction

Physics underpins the everyday world around us—from the way water fills a cup to the flutter of clothing in a breeze. Faithfully reproducing such phenomena in digital environments has been a central pursuit of computer graphics for decades. Physics-based simulation achieves this by encoding Newtonian dynamics, constitutive material laws, and frictional contact into numerical solvers that produce high-fidelity motion and deformation. Physics-based graphics techniques based on the Finite Element (FEM) for solids (e.g., [Sifakis and Barbic, 2012](#)) and Smoothed Particle Hydrodynamics (SPH) for fluids (e.g., [Monaghan, 1992, 2005](#)), among others, have enabled increasingly realistic virtual worlds. However, these methods remain computationally demanding, often requiring finely discretized meshes or millions of particles, and their setup typically demands domain expertise. These costs limit the scalability of physics-based simulation for large-scale content creation and interactive applications.

In parallel, AI-driven generative models have reshaped content creation by enabling the automatic synthesis of 3D assets ([Lin et al., 2023a](#); [Shi et al., 2023b](#); [Hong et al., 2023](#); [Tang et al., 2025](#)), dynamic scenes ([Ling et al., 2024](#); [Singer et al., 2023](#); [Park et al., 2021](#)), and realistic video sequences ([Chen et al., 2024a](#); [Xing et al., 2025](#); [Wang et al., 2024e](#)). These models have been widely adopted across computer graphics, gaming, robotics, and immersive applications such as augmented and virtual reality. Yet because they learn purely from data, generative models capture statistical correlations without an intrinsic understanding of physical laws. As a result, they frequently produce nonphysical artifacts—unstable structures, implausible object interactions, and temporally inconsistent dynamics—that restrict their use in applications demanding physical consistency, such as

robotics, digital twins, and scientific simulation.

A fundamental gap thus exists between these two paradigms: generative models lack physical realism, while physics-based simulations lack the scalability and accessibility of learning-based approaches. This dissertation bridges this gap by integrating physics-based constraints into AI-driven content generation. By embedding physical priors into generative pipelines, we enhance the plausibility, stability, and coherence of synthesized content while preserving the flexibility and scalability of modern generative techniques.

1.1 Thesis Overview and Contributions

The remainder of this thesis explores physics-aware generative methods across four key areas:

Chapter 2: Efficient and Robust Solid-Fluid Coupling via Physics-Based Simulation: Simulating the interaction between deformable solids and fluids is a long-standing challenge in computational physics, demanding stable two-way coupling that enforces nonpenetration while preserving realistic material behavior. We propose a novel operator-splitting strategy (Xie et al., 2023a) for Lagrangian solid-fluid coupling that introduces a variational formulation based on interior-point barrier-based frictional contact. A time-splitting scheme separates the simulation into distinct fluid and solid-coupling phases, augmented by contact proxies that accelerate convergence. The resulting method accurately simulates hyperelastic objects of varying geometries and codimensions with a penetration-free guarantee, while significantly improving computational efficiency over prior techniques (e.g., Figure 1.1).

Chapter 3: Self-Supporting Text-to-3D Generation with Physics Constraints: Diffusion-based generative models can synthesize visually realistic 3D objects from text, yet they frequently overlook physical constraints, producing structures that collapse under gravity or cannot be 3D-printed. We introduce Atlas3D (Chen et al., 2024b), a



Figure 1.1: Kicking water. Our method accurately captures the complex interactions between the water, the multi-layer skirt, and the mannequin body without any interpenetration as the skirted mannequin kicks in a swimming pool and splashes water.



Figure 1.2: 3D-printed figurines created with Atlas3D stand stably, while those without Atlas3D have fallen down.

physics-aware refinement framework that embeds differentiable simulation constraints into the optimization process. By enforcing stability under gravity, contact, and friction through a novel differentiable simulation-based loss, Atlas3D ensures structural integrity without compromising visual fidelity. The framework serves as both a physics-informed optimization module and a post-processing tool for existing Score Distillation Sampling pipelines, and we validate its effectiveness through extensive generation tasks and real-world fabrication experiments (e.g., [Figure 1.2](#)).

Chapter 4: Physics-Integrated 4D Generative Dynamics for Multi-View Reconstruction: Multi-view reconstruction methods have traditionally focused on static geometry, neglecting the physics that govern object motion and deformation. We introduce

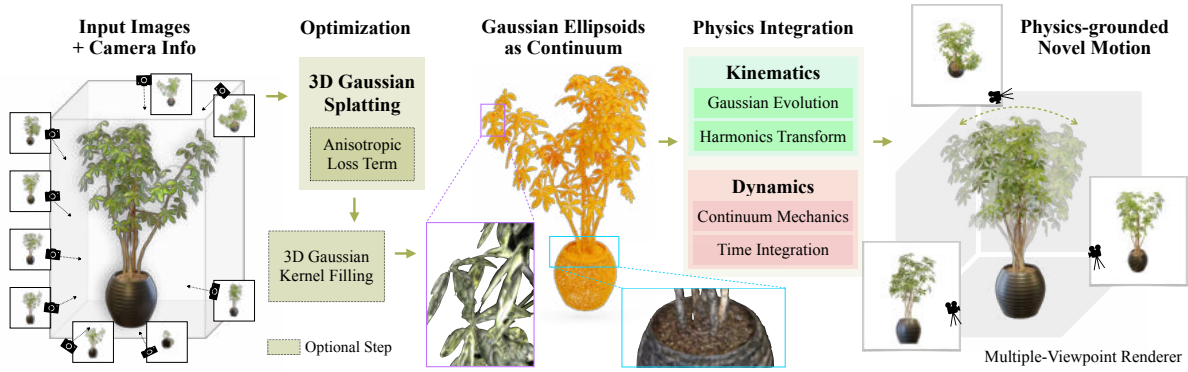


Figure 1.3: **Method Overview.** PhysGaussian is a unified simulation-rendering pipeline that incorporates 3D Gaussian splatting representation and continuum mechanics to generate physics-based dynamics and photo-realistic renderings simultaneously and seamlessly.

PhysGaussian (Xie et al., 2023b), a framework that integrates Newtonian dynamics into 3D Gaussian representations to synthesize physically grounded 4D motion (Figure 1.3). A custom Material Point Method models kinematic deformation and mechanical stress under continuum mechanics, while the same Gaussian kernels serve as both visual and physical representations—a principle we term “what you see is what you simulate” (WS²). Building on this foundation, we develop VR-GS (Jiang et al., 2024c), a real-time system that enables interactive editing and manipulation of physics-aware 3D objects in virtual reality through a two-level embedding strategy, deformable body simulation, and multi-view image inpainting (Figure 1.4).

Chapter 5: Physics-Guided Animation Generation and Humanoid Locomotion Manipulation: Video foundation models can generate visually compelling dynamic sequences, yet ensuring that the depicted motions are physically plausible and reproducible in simulation remains a major challenge. We address this through three complementary contributions spanning 2D animation, 3D character dynamics, and 4D human-object interaction. First, PhysAnimator (Xie et al., 2025b) combines image-space deformable-body simulation with sketch-guided video diffusion to synthesize physically plausible cartoon animations from static anime illustrations (Figure 1.5). Users can direct the animation through customizable energy strokes and rigging constraints that specify external forces

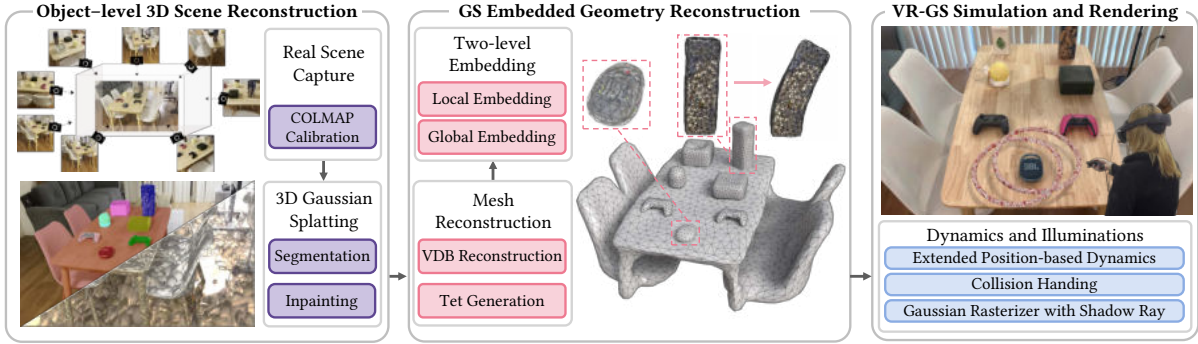


Figure 1.4: VR-GS is an interactive system designed to integrate 3D Gaussian Splatting (GS) and eXtended Position-based Dynamics (XPBD) for generating a real-time interactive experience. Beginning with multi-view images, the pipeline combines scene reconstruction, segmentation, and inpainting using Gaussian kernels. These kernels form the foundation for VR-GS’s utilization of the sparse volumetric data structure VDB, facilitating bounding mesh reconstruction and subsequent tetrahedralization. VR-GS further harnesses a novel two-level Gaussian embedding, XPBD, collision detection, and shadow casting techniques, all converging to deliver a captivating and immersive user experience.

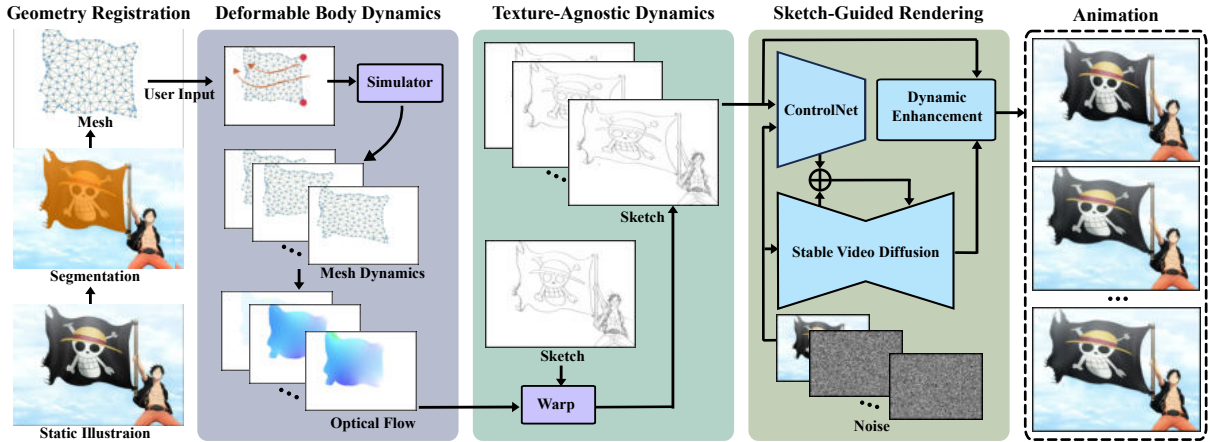


Figure 1.5: PhysAnimator overview. We begin by segmenting the object and creating a triangulated deformable mesh. Physics-based simulations are then used to generate dynamic optical flow fields, with users given the option to guide the motion through customizable energy strokes (shown as orange arrows) and rigging points (shown as red dots). The extracted sketch is warped using the computed optical flow and refined with a sketch-guided video diffusion model, producing a smooth, stylized animation sequence. Optionally, a cartoon interpolation model can further be applied to enhance the animation with expressive dynamics.

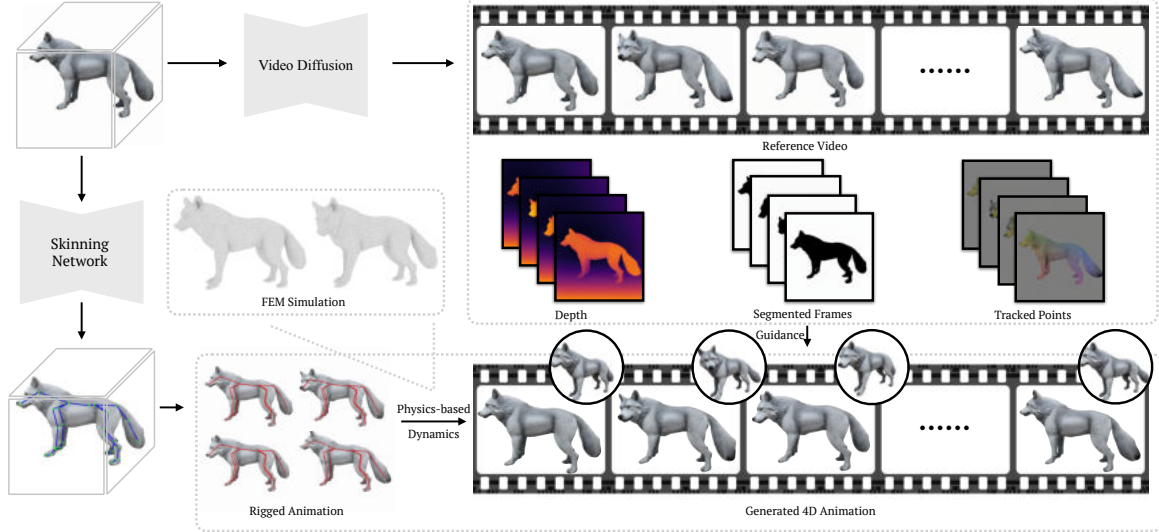


Figure 1.6: Animimic pipeline overview. From an input 3D mesh, we render a canonical view and use a video diffusion model to generate a monocular motion sequence. We construct a skeleton with skinning weights using a feed-forward rigging model and generate animation by optimizing joint motions through differentiable rendering, tracking, and depth cues. Finally, we refine mesh deformation via differentiable simulation to obtain physically grounded and temporally consistent results. Right circles indicate novel views.

such as wind, while a video diffusion model renders the simulated motion into temporally coherent, stylistically faithful frames. Second, AniMimic (Xie et al., 2025a) tackles the problem of imitating 3D animation from video priors (Figure 1.6). Given reference videos generated by video foundation models, AniMimic reconstructs per-frame 3D deformations and then recovers the underlying physical parameters—material properties and external forces—via differentiable simulation-based tracking. This closes the loop between video generation and physics, enabling faithful reproduction of complex character dynamics including large deformations, contacts, and multi-body interactions. Third, GRAIL extends this paradigm to scalable humanoid loco-manipulation (Figure 1.7). By rendering interaction videos from a fully specified 3D scene with a robot-proportioned character, GRAIL makes the subsequent 4D reconstruction well-conditioned and ensures morphological compatibility with the target robot. A holistic global optimization recovers temporally consistent full-body motion, detailed hand articulation, and coherent object dynamics. The reconstructed motions are retargeted to a humanoid robot, and a latent adaptor policy learning approach extends a whole-body control foundation model with

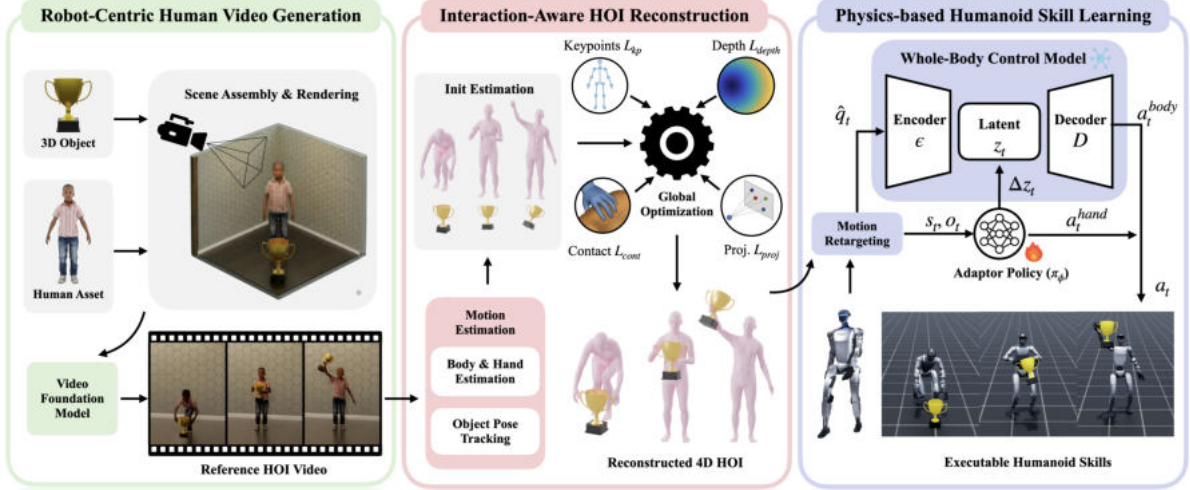


Figure 1.7: Overview of GRAIL. Given a 3D object asset, GRAIL generates physically plausible humanoid loco-manipulation trajectories and actions through three stages. First, we assemble a fully specified 3D configuration with a character sized to the target robot, render an initial frame, and feed it into a video foundation model to synthesize a reference HOI video. Second, leveraging the known 3D configuration, we reconstruct coherent 4D HOI trajectories through a holistic global optimization. Finally, we retarget the reconstructed motions to a humanoid and propose a latent adaptor policy learning approach to learn humanoid loco-manipulation skills in physics simulation.

loco-manipulation skills and scene-aware locomotion via reinforcement learning in physics simulation. GRAIL yields a large-scale dataset of over 20,000 diverse 4D HOI sequences and substantially outperforms existing approaches across geometric, perceptual, and physics-based evaluations.

Chapter 6: Conclusions: Collectively, the contributions of this thesis demonstrate that embedding physics-based constraints into generative pipelines yields visual content that is not only compelling but also physically consistent. The proposed methods bridge deep learning and physics-based simulation, offering scalable solutions for computer graphics, embodied AI, robotics, and interactive applications.

CHAPTER 2

Efficient and Robust Solid-Fluid Coupling via Physics-Based Simulation

In this chapter, we present a robust and efficient method for simulating Lagrangian solid-fluid coupling based on a new operator splitting strategy. We use variational formulations to approximate fluid properties and solid-fluid interactions, and introduce a unified two-way coupling formulation for SPH fluids and FEM solids using interior point barrier-based frictional contact. We split the resulting optimization problem into a fluid phase and a solid-coupling phase using a novel time-splitting approach with augmented *contact proxies*, and propose efficient custom linear solvers. Our technique accounts for the interaction of fluids with nonlinear hyperelastic objects of different geometries and codimensions, while maintaining an algorithmically guaranteed non-penetration criterion. Comprehensive benchmarks and experiments demonstrate the efficacy of our method.

2.1 Introduction

The coupling of solids and fluids is common in nature but challenging to simulate. While solids are typically simulated using Lagrangian meshes, fluids are often discretized using Eulerian grids to accommodate topology changes. To accurately couple these distinct discretizations, sophisticated algorithms, such as the cut-cell method (Zarifi and Batty, 2017), are often necessary. Unfortunately, they can be computationally expensive and do not handle thin shells. Purely Eulerian schemes (Teng et al., 2016; Valkov et al., 2015) or Smoothed-Particle Hydrodynamics (SPH) (Akinici et al., 2012; Gissler et al., 2019) have demonstrated successful two-way coupling by same-view discretization, but they do not

easily extend to nonlinear elastodynamics. Hybrid methods like the Material Point Method (MPM) (Jiang et al., 2016) can simulate mixed materials, but can experience artificial stickiness unless resolved with more expensive schemes (Fang et al., 2020). Furthermore, these methods do not ensure non-intersecting trajectories and often require additional correction procedures to handle accidentally penetrated fluids during advection.

We take the Lagrangian approach and present a new method for coupling Finite Element Method (FEM) solids and SPH fluids. By approximating solid, fluid, and interaction terms with potentials, we formulate two-way coupling as a monolithic optimization problem. Specifically, we draw inspiration from position-based fluids (Macklin and Müller, 2013) and model weak incompressibility using a quadratic energy and a new *updated Lagrangian update rule* to track volume changes. We further symmetrize the discrete Laplacian-based viscosity and propose a *discrete quadratic potential* for better accuracy and robustness. We follow the Incremental Potential Contact (IPC) (Li et al., 2020) model to enforce guaranteed separable boundary conditions and resolve frictional contacts at the interface.

The proposed formulation achieves strong coupling, but can be exceedingly inefficient when solved with Newton’s method due to the huge and dense Hessian of the fluid component, which is a result of the need for many particle neighbors to achieve an accurate SPH discretization. This causes a significant computational bottleneck.

To tackle this issue, we propose a robust *proxy contact energy* formulation, splitting the time integration into a fluid phase and a solid-coupling phase. The fluid phase requires only one Newton iteration per time step, resulting in increased efficiency with nonlinear optimization occurring only during the solid-coupling phase. One of the key advantages of our quadratic proxy is its ability to effectively resolve instability caused by time splitting. This is achieved through its asymptotic approximation to the solid-fluid contact force. Additionally, the consistency of the time integration is maintained through the cancellation of the proxy’s contribution in the solid-coupling phase, resulting in only a small splitting error. Finally, we design a matrix-free conjugate gradient solver and a domain-decomposed solver to further enhance the computational efficiency.

The contributions of this work include

- a unified penetration-free two-way coupling framework for weakly compressible SPH fluids and nonlinear elastic FEM solids in arbitrary codimensions;
- consistently modeled incompressibility and viscosity potentials for SPH fluids, incorporating advantages of Updated Lagrangian (UL) kinematics;
- a robust time splitting scheme with contact proxies that enables separate time integrations of solids and fluids;
- a matrix-free Conjugate Gradient (CG) solver and a Schur-complement-based domain-decomposed solver for efficiently solving the linear systems.

2.2 Related Work

Coupling Solids with Eulerian Fluids: Traditional fluid solvers typically use grid discretization to simulate fluid dynamics—aka Eulerian Fluids. Since the Lagrangian view dominates solid simulation, much existing literature focuses on coupling Eulerian fluids with Lagrangian solids by resolving interactions between the grids and irregular mesh boundaries. The ghost fluid method ([Fedkiw et al., 1999](#); [Fedkiw, 2002](#)) was proposed to additionally discretize the Eulerian/Lagrangian interface. Early efforts considered weak coupling ([Guendelman et al., 2005](#)), which advances the solids and fluids in an alternating manner. On the other hand, strong coupling ([Klingner et al., 2006](#)) solves a monolithic system and is often more robust. The cut-cell method ([Roble et al., 2005](#)) is another widely used solution to accurately model the interactions between Lagrangian solids and Eulerian fluids, often through the usage of virtual nodes. This method is first introduced to computer graphics by [Roble et al. \(2005\)](#). [Batty et al. \(2007\)](#) proposed a variational framework to strongly couple fluids and rigid bodies by casting the pressure solve as a minimization problem. Subsequent extensions support deformable objects and thin shells ([Robinson-Mosher et al., 2008, 2011](#)), where the elastic forces are explicitly applied and the solid damping was solved together with fluid dynamics in an implicit

manner. Assuming co-rotated linear elasticity, [Zarifi and Batty \(2017\)](#) incorporated the implicit solid dynamics into pressure projection, obtaining a symmetric positive-definite system. Later efforts also explored rigid-rigid ([Takahashi and Batty, 2020](#)) and rigid-fluid ([Takahashi and Batty, 2021](#)) frictional contacts.

Apart from Lagrangian solids, also explored was Eulerian discretization ([Levin et al., 2011](#)), where coupling can be conveniently achieved in a purely Eulerian fashion ([Teng et al., 2016](#); [Valkov et al., 2015](#)). However, such discretization entails increased numerical dissipation, resulting in artifacts such as volume loss, and the inability to handle structures thinner than a grid cell. More recently, [Brandt et al. \(2019\)](#) built upon the immersed boundary method ([Peskin, 2002](#)) and proposed a reduced solver to simulate real-time coupling, focusing on incompressible elastic materials and no-slip boundary conditions.

Coupling Solids with Lagrangian Fluids: Fluids can also be directly modeled with Lagrangian meshes ([Clausen et al., 2013](#); [Klingner et al., 2006](#); [Wang et al., 2020a](#); [Batty et al., 2012](#)), enabling explicit coupling with solids. However, this inevitably incurs frequent remeshing operations, thus degrading the overall performance.

Another strategy is to use particle-based methods, or meshless methods. SPH ([Koschier et al., 2022](#)) is a meshless approximation strategy based on spatial sampling for continuous functions, which has shown its ability to produce visually compelling results in physics-based simulation, especially for the fluid. Pioneering researchers ([Monaghan, 1994](#); [Becker and Teschner, 2007](#)) used the Equation of State (EOS) for weakly compressible fluids, where the pressure is proportional to the density deviation. However, this explicit formulation may strictly restrict time step sizes resulting in limited performance. Incompressibility has also been enforced by solving a Pressure Poisson Equation (PPE) ([Solenthaler and Pajarola, 2009](#); [Ihmsen et al., 2013](#); [Bender and Koschier, 2015](#)). This approach seeks to cancel out density or velocity divergence deviations caused by non-pressure forces through the use of pressure accelerations. SPH boundary handling techniques have been developed to prevent penetrations of fluid particles near solid boundaries ([Becker et al., 2009b](#); [Ihmsen et al., 2010](#); [Becker and Teschner, 2007](#)). One such method, proposed by

Akinci et al. (2012), uses a single layer of boundary samples and has been applied to the coupling of fluids with both rigid bodies and elastic solids (Akinci et al., 2013). Gissler et al. (2019) proposed a global formulation that unifies rigid body and fluid dynamics, in which the fluid pressure solver is linked to a second artificial pressure solver for rigid body particles. Koschier and Bender (2017) introduced an alternative method using density maps to represent dynamic rigid boundaries, eliminating the need for boundary particles. Bender et al. (2019) proposed using the volume contribution of boundary geometry to compute boundary forces, which reduces the cost of precomputation. However, this type of method cannot be applied to deformable bodies due to the expensive cost of computing maps at run time.

Solenthaler et al. (2007) used SPH to approximate the deformation gradient of linear elastic materials, but the resulting gradient is not rotation invariant. Becker et al. (2009a) addressed this issue by using shape matching to determine orientation and calculating forces in a rotated configuration. Peer et al. (2018) proposed an implicit scheme and applied kernel gradient correction (Bonet and Lok, 1999) to obtain a first-order consistent SPH formulation for the deformation gradient. By integrating the solid particles into the preexisting fluid pressure solver, contact handling can be inherently resolved, but SPH still faces numerical issues, such as the zero-mode (Kugelstadt et al., 2021; Ganzenmüller, 2015) when simulating elastic objects. Additionally, the pressure solver will treat solid objects as incompressible under compression, which may not be applicable in all cases.

Hybrid Methods with Coupled Phases and Contact: The MPM (Sulsky et al., 1995; Jiang et al., 2016) is a hybrid Eulerian-Lagrangian approach that treats materials as a collection of material points. Stomakhin et al. (2013) first introduced this method into computer graphics for snow simulation. The MPM has also been proven successful in capturing solid-fluid coupling (Stomakhin et al., 2014; Fei et al., 2018; Yan et al., 2018) and mixtures (Tampubolon et al., 2017; Gao et al., 2018a). While one advantage of the MPM is the automatic handling of contacts between different materials because of its hybrid nature, the separation of continuum and free-slip interaction among different objects remains

challenging. This drawback can be attributed to the fact that the traditional MPM (Jiang et al., 2016) does not inherently support discontinuous velocities, resulting in sticky boundary conditions. To allow easy material separation, Hu et al. (2018a) introduced a Compatible Particle-In-Cell (CPIC) algorithm to allow discontinuous velocities. Fang et al. (2020) proposed a free-slip treatment, but did not consider separation. Recently, a FEM-MPM coupling method based on a variational barrier formulation (Li et al., 2020) has been proposed for coupling frictional and separable elastic materials (Li et al., 2022d). Our approach for solid-fluid coupling is inspired by this method and uses a similar purely Lagrangian framework.

2.3 Formulation

The governing equations for the coupled system are

$$\rho_s \frac{D\mathbf{v}_s}{Dt} = \nabla \cdot \sigma + \rho_s \mathbf{g} + \mathbf{f}_{s \rightarrow s} + \mathbf{f}_{f \rightarrow s}, \quad (2.1)$$

$$\rho_f \frac{D\mathbf{v}_f}{Dt} = -\nabla p + \mu \nabla^2 \mathbf{v}_f + \rho_f \mathbf{g} - \mathbf{f}_{f \rightarrow s}, \quad (2.2)$$

$$\nabla \cdot \mathbf{v}_f = 0, \quad (2.3)$$

where ρ is density, $\mathbf{g}$ is gravity, $\mathbf{f}_{s \rightarrow s}$ is the self-contact force of solids, $\mathbf{f}_{f \rightarrow s}$ is the contact force exerted by fluids, σ is Cauchy stress, p is pressure, and μ is the dynamic viscosity (Bridson, 2015).

At the interface between solids and fluids, we enforce the separable boundary condition

$$0 \leq (\mathbf{v}_s - \mathbf{v}_f) \cdot \mathbf{n}_f \perp (\mathbf{f}_{f \rightarrow s} \cdot \mathbf{n}_f) \geq 0 \quad (2.4)$$

to prevent penetration while allowing separation (Batty et al., 2007). This condition helps determine the normal component of $\mathbf{f}_{f \rightarrow s}$. For the tangential component (friction), let $\mathbf{u} = (\mathbf{I} - \mathbf{n}_f \otimes \mathbf{n}_f)(\mathbf{v}_s - \mathbf{v}_f)$ be the tangential relative velocity. Following the Maximum

Dissipation Principle (Moreau, 2011), we then have

$$\begin{aligned}
(\mathbf{I} - \mathbf{n}_f \otimes \mathbf{n}_f) \mathbf{f}_{f \rightarrow s} &= \arg \min_{\boldsymbol{\beta}} \boldsymbol{\beta} \cdot \mathbf{u}, \\
\text{s.t. } \|\boldsymbol{\beta}\| &\leq \mu_t \mathbf{f}_{f \rightarrow s} \cdot \mathbf{n}_f \quad \text{and} \quad \boldsymbol{\beta} \cdot \mathbf{n}_f = 0,
\end{aligned} \tag{2.5}$$

where μ_t is the friction coefficient. We enforce exact mass conservation by adopting Lagrangian methods to discretize both domains.

2.3.1 Solid Domain

We focus on nonlinear hyperelastic solids, where the elastic force is the negative gradient of an elastic potential. After discretizing the solid domain Ω_s into Lagrangian linear finite elements (triangles in 2D and tetrahedra in 3D), the total elastic potential is a piecewise constant summation of an elastic energy density function $\psi_s(\mathbf{F})$ (e.g., neo-Hookean) over the mesh domain:

$$\Psi_s(\mathbf{x}) = \sum_e V_e \psi_s(\mathbf{F}_e), \tag{2.6}$$

where V_e is the rest volume of tetrahedron e , and $\mathbf{F} = \frac{\partial \mathbf{x}(\mathbf{X}, t)}{\partial \mathbf{X}}$ is the deformation gradient with $\mathbf{X}$ and $\mathbf{x}$ denoting the material and world space coordinates, respectively (Sifakis and Barbic, 2012). For $\mathbf{f}_{s \rightarrow s}$, we follow the smooth barrier approach of Li et al. (2020) that guarantees non-penetration. We leave the discussion of $\mathbf{f}_{f \rightarrow s}$ to Section 2.3.3.

2.3.2 Fluid Domain

Following the SPH literature (Becker and Teschner, 2007; Macklin and Müller, 2013; Ihmsen et al., 2013; Bender and Koschier, 2015), we discretize the fluid domain Ω_f using Lagrangian particles. To simulate fluids using optimization-based time integration, we approximate both the pressure and viscosity forces as conservative forces. We verify that these two proposed potential energies are both convex and quadratic in Section A.1.

Incompressibility Potential: Pressure forces help preserve the volume of incompressible fluids. We thus model the incompressibility via a quadratic energy density function

$$\psi_{f,I}(J) = \frac{k_I}{2}(J - 1)^2, \quad (2.7)$$

which penalizes the deviation from 1.0 of the volume ratio $J = \rho_0/\rho$, where ρ_0 is the initial density. The use of a large stiffness k_I in a convergent solver results in negligible visual compression, eliminating the need for higher degree polynomials in nearly incompressible fluids (Hyde et al., 2020). The incompressibility potential is obtained by integrating $\psi_{f,I}(J)$ over the fluid domain Ω_f^0 in *material space*:

$$P_I(\mathbf{x}) = \sum_i \frac{k_I}{2} V_0 (J_i(\mathbf{x}) - 1)^2, \quad (2.8)$$

where we assume that all fluid particles have equal rest volume V_0 and where J_i denotes the volume ratio of particle i as a function of $\mathbf{x}$.

The SPH literature often relates ρ to $\mathbf{x}$ through density summation in the world space. To obtain a linear relation between J and $\mathbf{x}$ so that the incompressibility potential stays quadratic in terms of $\mathbf{x}$, we track J in an *updated Lagrangian* fashion. Treating Ω^n as an intermediate reference space and differentiating the deformation map between Ω^n and Ω^{n+1} results in an update rule

$$J_i^{n+1} = J_i^n (1 + h \nabla \cdot \mathbf{v}_i^{n+1}), \quad (2.9)$$

the terms of which can be approximated via SPH as

$$J_i^n = \frac{\rho_0}{\sum_j m_j W_{ij}}, \quad (2.10)$$

$$\nabla \cdot \mathbf{v}_i^{n+1} = \sum_j \frac{m_j}{\rho_j^n} (\mathbf{v}_j^{n+1} - \mathbf{v}_i^{n+1}) \cdot \nabla_i W_{ij}, \quad (2.11)$$

and $W_{ij} = W(\mathbf{x}_i - \mathbf{x}_j)$ is a kernel function (e.g., Cubic Spline kernel (Monaghan, 1992,

2005) or Spiky kernel (Müller et al., 2003)). Here J_i^n denote the reinitialized volume ratio of fluid particle i at the start of time step n . Such reinitialization avoids accumulated density and particle distribution errors commonly seen in other updated Lagrangian solvers like MPM.

Viscosity Potential: The viscosity of fluids is a measure of their resistance to shearing at a given rate, and it plays a vital role in producing effects like buckling and coiling. It is possible to model viscosity via strain rate tensors (Peer et al., 2015; Peer and Teschner, 2016; Takahashi et al., 2015; Bender and Koschier, 2016), but this may suffer from artifacts at the surface due to particle deficiencies in the SPH approximation. Following Monaghan (2005), we use the more robust velocity Laplacian (Weiler et al., 2018) and derive its energy form.

Combining SPH first-order derivatives and finite differences, the viscosity force can be computed as

$$\begin{aligned}\mathbf{f}_i(\mathbf{x}) &= \nu m_i \nabla^2 \mathbf{v}_i^{n+1} \\ &= 2\nu(d+2) \sum_j \frac{m_i m_j}{\rho_j} \frac{\nabla_i W_{ij}(\mathbf{x}_{ij}^n)^T}{\|\mathbf{x}_{ij}^n\|^2 + 0.01\hbar^2} \mathbf{v}_{ij}^{n+1},\end{aligned}\tag{2.12}$$

where $\mathbf{x}_{ij}^n = \mathbf{x}_i^n - \mathbf{x}_j^n$ and $\mathbf{v}_{ij}^{n+1} = \mathbf{v}_i^{n+1} - \mathbf{v}_j^{n+1}$, and where $\hbar$ is the support radius of the kernel, ν is the kinematic viscosity, and $d \in \{2, 3\}$ denotes the spatial dimensionality. Directly applying this force violates momentum conservation as the mutual interaction forces are unequal. To solve this problem, we perform a further approximation:

$$\mathbf{f}_i(\mathbf{x}) \approx 4\nu(d+2) \sum_j \frac{m_i m_j}{\rho_i + \rho_j} \frac{\nabla_i W_{ij}(\mathbf{x}_{ij}^n)^T}{\|\mathbf{x}_{ij}^n\|^2 + 0.01\hbar^2} \mathbf{v}_{ij}^{n+1},\tag{2.13}$$

and also make the force integrable. Letting $\mathbf{V}_{ij} = 4(d+2) \frac{m_i m_j}{\rho_i + \rho_j} \frac{(-\nabla_i W_{ij})(\mathbf{x}_{ij}^n)^T}{\|\mathbf{x}_{ij}^n\|^2 + 0.01\hbar^2}$, we can gather and integrate all the viscosity forces and obtain a quadratic viscosity potential

$$P_V(\mathbf{x}) = \frac{1}{4} \nu \hat{h} \sum_i \sum_j \|\mathbf{v}_{ij}^{n+1}\|_{\mathbf{V}_{ij}}^2,\tag{2.14}$$

where $\hat{h}$ is a constant scalar related to the time integration scheme. For example, $\hat{h} = h$ for the implicit Euler method as $\mathbf{v}^{n+1} = (\mathbf{x}^{n+1} - \mathbf{x}^n)/h$.

2.3.3 Coupling

Barrier Potential for Non-Penetration: To couple the solid domain Ω_s with the fluid domain Ω_f , we use the separable boundary condition (2.4), which enforces non-interpenetration constraints between these two domains. To model these constraints, we first define a function

$$d(\partial\Omega_s^t, \mathbf{x}_f) = \min_{\mathbf{x}_s} \|\mathbf{x}_s - \mathbf{x}_f\|, \quad \mathbf{x}_s \in \partial\Omega_s^t, \quad \mathbf{x}_f \in \Omega_f^t, \quad (2.15)$$

which measures the distance between $\mathbf{x}_f$, a point in the fluid domain, and the surface of the solid domain. Then the primal component of the constraints can be expressed as

$$d(\partial\Omega_s^t, \mathbf{x}_f) \geq 0, \quad \forall t \geq 0, \quad \forall \mathbf{x}_f \in \Omega_f^t. \quad (2.16)$$

We then adopt the barrier formulation from Li et al. (2020) to model all the constraints in (2.4) between solids and fluids, and obtain a barrier potential

$$\int_{\partial\Omega_f^t} b(d(\partial\Omega_s^t, \mathbf{x}_f), \hat{d}) d\mathbf{x}_f, \quad (2.17)$$

where the barrier energy density $b(d, \hat{d})$ is piecewise smooth and is only activated when $d < \hat{d}$, improving efficiency and approximately satisfying the complementarity slackness condition. As d approaches 0, the value of $b(d, \hat{d})$ monotonically increases to infinity, providing arbitrarily large repulsion to avoid interpenetration.

Since our solid and fluid domains are respectively discretized as meshes and particles,

the barrier potential (2.17) in 3D can be numerically integrated as

$$\begin{aligned} B_{\text{sf}}(\mathbf{x}_s, \mathbf{x}_f) &= \sum_{q \in \mathcal{Q}_f} s_q b(\min_{e \in \mathcal{B}_s} d^{PT}(\mathbf{x}_q, e), \hat{d}) \\ &= \sum_{q \in \mathcal{Q}_f} s_q \max_{e \in \mathcal{B}_s} b(d^{PT}(\mathbf{x}_q, e), \hat{d}), \end{aligned} \quad (2.18)$$

where $\mathcal{Q}_f$ is the set of all SPH fluid particles, $\mathcal{B}_s$ is the set of all the boundary triangles of the solids, $s_q = \pi(\frac{3J_n V_0}{4\pi})^{\frac{2}{3}}$ is the integration weight (boundary area) of each fluid particle, and $d_{PT}(\mathbf{x}_q, e)$ measures the distance between particle $\mathbf{x}_q$ and triangle e . Here, the min-max transformation is based on the non-ascending property of the barrier function. However, the max operator makes the barrier potential challenging to be efficiently optimized by gradient-based methods. Fortunately, due to the local support of the barrier function $b(d, \hat{d})$ as $\hat{d}$ is small, we can simply approximate the barrier potential as

$$B_{\text{sf}}(\mathbf{x}_s, \mathbf{x}_f) = \sum_{q \in \mathcal{Q}_f} \sum_{e \in \mathcal{B}_s} s_q b(d^{PT}(\mathbf{x}_q, e), \hat{d}), \quad (2.19)$$

which may result in overestimated contact forces near the edges and nodes on the mesh boundary, but we have not observed any artifacts in our experiments.

Friction Potential: Following Li et al. (2020), we model the local friction forces $\mathbf{f}_k$ for every active solid-fluid contact pair k . Formally, the friction force is defined as

$$\mathbf{f}_k(\mathbf{x}_s, \mathbf{x}_f) = -\mu_t \lambda_k T_k(\mathbf{x}_s, \mathbf{x}_f) f_1(\|\mathbf{u}_k\|) \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|}, \quad (2.20)$$

where λ_k is the contact force magnitude, $T_k(\mathbf{x}_s, \mathbf{x}_f) \in \mathbb{R}^{3n \times 2}$ is the consistently oriented sliding basis, and $\mathbf{u}_k$ is the relative sliding displacement, which can be computed as $\mathbf{u}_k = T_k(\mathbf{x}_s, \mathbf{x}_f)^T ([\mathbf{x}_s^T, \mathbf{x}_f^T]^T - [(\mathbf{x}_s^n)^T, (\mathbf{x}_f^n)^T]^T)$. Here, f_1 is a smoothly approximated function designed for a smooth transition between the sticking and sliding modes. To make this friction formulation fit into optimization time integration, Li et al. (2020) further approximated the sliding basis $T(\mathbf{x}_s, \mathbf{x}_f)$ and contact force $\lambda_k(\mathbf{x}_s, \mathbf{x}_f)$ explicitly

as $T(\mathbf{x}_s^n, \mathbf{x}_f^n)$ and $\lambda_k(\mathbf{x}_s^n, \mathbf{x}_f^n)$. Then the semi-implicit friction force is integrable with the friction potential computed as

$$D_{\text{sf}}(\mathbf{x}_s, \mathbf{x}_f) = \sum_{k \in \mathcal{A}^n} \mu_t \lambda_k^n f_0(\|\mathbf{u}_k\|), \quad (2.21)$$

where f_0 is defined by the relation $f'_0 = f_1$ and $\mathcal{A}^n$ is the set containing all activate particle-triangle contact pairs at the previous time step n .

2.3.4 Optimization Time Integrator

With the above potential energies modeling all the solid and fluid forces, now we can build a unified two-way solid-fluid coupling framework. By stacking all the nodal positions and velocities of the SPH particles and FEM nodes as $\mathbf{x} = [\mathbf{x}_f^T, \mathbf{x}_s^T]^T$ and $\mathbf{v} = [\mathbf{v}_f^T, \mathbf{v}_s^T]^T$, we define $\Psi(\mathbf{x}) = \Psi_s(\mathbf{x}_s)$, $P(\mathbf{x}) = P_I(\mathbf{x}_f) + P_V(\mathbf{x}_f)$, and $C_{\text{sf}}(\mathbf{x}) = B_{\text{sf}}(\mathbf{x}_s, \mathbf{x}_f) + D_{\text{sf}}(\mathbf{x}_s, \mathbf{x}_f)$. Combined with the solid-solid contact potential $C_{\text{ss}}(\mathbf{x})$ from IPC, our solid-fluid coupling problem can be solved in a monolithic manner by applying implicit Euler time integration

$$\begin{cases} \mathbf{v}^{n+1} = \mathbf{v}^n + h\mathbf{M}^{-1}(\mathbf{f}_{\text{ext}} - \nabla P(\mathbf{x}^{n+1}) - \nabla \Psi(\mathbf{x}^{n+1}) - \nabla C(\mathbf{x}^{n+1})), \\ \mathbf{x}^{n+1} = \mathbf{x}^n + h\mathbf{v}^{n+1}, \end{cases} \quad (2.22)$$

which is equivalent to

$$\mathbf{x}^{n+1} = \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \hat{\mathbf{x}}^n\|_{\mathbf{M}}^2 + h^2(P(\mathbf{x}) + \Psi(\mathbf{x}) + C(\mathbf{x})) \quad (2.23)$$

with the mass matrix $\mathbf{M}$, time step size h , predictive position $\hat{\mathbf{x}}^n = \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$, and total contact potential $C(\mathbf{x}) = C_{\text{sf}}(\mathbf{x}) + C_{\text{ss}}(\mathbf{x})$.

2.4 Efficient Solver

A straightforward way to robustly solve the time-stepping optimization problem (2.23) is to apply the projected Newton's method with line search (Li et al., 2020). At every

iteration, the search direction $\mathbf{p}$ can be computed by solving the linear system

$$\begin{bmatrix} \mathbf{H}_f & \mathbf{G} \\ \mathbf{G}^T & \mathbf{H}_s \end{bmatrix} \mathbf{p} = \begin{bmatrix} \mathbf{g}_f \\ \mathbf{g}_s \end{bmatrix}, \quad (2.24)$$

where $\mathbf{H}_f$ and $\mathbf{H}_s$ are the (projected) Hessian matrices with respect to the position of fluids and solids, respectively, and $\mathbf{G} = \frac{\partial^2 E}{\partial \mathbf{x}_f \partial \mathbf{x}_s}$ denotes the coupling submatrix. Unfortunately, solving this linear system can be a severe bottleneck in practice. One reason is that SPH techniques need sufficiently many neighbors to accurately approximate physical quantities, which results in a much larger and denser fluid Hessian matrix $\mathbf{H}_f$ compared to the solid one. In addition, the optimization may require many iterations to converge due to the sharpness of barrier energy, especially in contact-rich cases.

Since our fluid energies are all quadratic, we separate them from the highly nonlinear solids and contact energies via a robust time splitting scheme (Section 2.4.1) so that the fluid part can be solved within a single Newton iteration per time step. We then propose efficient methods to solve the domain-decomposed linear systems (Section 2.4.2).

2.4.1 Time Splitting

Baseline Time Splitting: Intuitively, we can split the original time integration into a fluid phase

$$\begin{cases} \mathbf{v}_f^{n+1/2} = \mathbf{v}_f^n + h\mathbf{M}_f^{-1} \left(-\nabla_f P \left([(\mathbf{x}_f^{n+1/2})^T, (\mathbf{x}_s^n)^T]^T \right) + \mathbf{f}_f \right), \\ \mathbf{x}_f^{n+1/2} = \mathbf{x}_f^n + h\mathbf{v}_f^{n+1/2}, \end{cases} \quad (2.25)$$

and a solid-coupling phase

$$\begin{cases} \mathbf{v}^{n+1} = \begin{bmatrix} \mathbf{v}_f^{n+1/2} \\ \mathbf{v}_s^n \end{bmatrix} + h\mathbf{M}^{-1} \left(-\nabla \Psi(\mathbf{x}^{n+1}) - \nabla C(\mathbf{x}^{n+1}) + \begin{bmatrix} \mathbf{0} \\ \mathbf{f}_s \end{bmatrix} \right), \\ \mathbf{x}^{n+1} = \mathbf{x}^n + h\mathbf{v}^{n+1}, \end{cases} \quad (2.26)$$

where $\mathbf{f}_f$ and $\mathbf{f}_s$ are the external forces on the fluids and the solids, respectively. In the fluid phase, we solve for an intermediate state for the fluid particles in a single Newton iteration, ignoring contact. Then the highly nonlinear barrier force is resolved in the solid-coupling phase along with elasticity, where the fluid Hessian $\mathbf{H}_f$ reduces to a block-diagonal matrix $\frac{\partial^2 C(\mathbf{x})}{\partial \mathbf{x}_f^2}$. In this setting, nonlinear optimization only happens for fluid boundaries and solid DOFs in the solid-coupling phase. The details of this *Baseline Time Splitting Scheme* can be found in [Section A.2.1](#).

Although this baseline splitting strategy indeed brings a significant performance gain, severe instabilities can occur at the solid-fluid interface if the time step size is not sufficiently small, especially when simulating viscous fluids ([Figure 2.3](#)). For example, fluid particles may stick to the solid boundaries. This artifact is also evident in existing SPH fluid solvers, and it is typically addressed by sampling particles at solid boundaries to exert boundary pressures ([Becker et al., 2009b](#); [Ihmsen et al., 2010](#); [Akinici et al., 2012](#)). In light of this, we consistently augment the fluid phase with proxy forces for solid-fluid contact to improve stability while avoiding any particle-sampling overhead.

Time Splitting with Contact Proxy: We introduce a solid-fluid contact proxy energy $\hat{C}_{\text{sf}}(\mathbf{x})$ into the fluid phase to efficiently exert approximated interaction forces between the boundaries of solids and fluids. In the ensuing discussion, we will also write contact energy $C(\mathbf{x})$ as the sum of the solid-fluid part $C_{\text{sf}}(\mathbf{x})$ and the solid-solid part $C_{\text{ss}}(\mathbf{x})$ for clarity. To ensure consistency with the original PDE, we cancel the contribution of this contact proxy in the solid-coupling phase. The resulting time integration becomes

$$\begin{cases} \mathbf{v}^{n+1/2} = \mathbf{v}^n + h\mathbf{M}^{-1}(-\nabla P(\mathbf{x}^{n+1/2}) + \mathbf{f}_{\text{ext}} - \nabla \hat{C}_{\text{sf}}(\mathbf{x}^{n+1/2})), \\ \mathbf{x}^{n+1/2} = \mathbf{x}^n + h\mathbf{v}^{n+1/2}; \end{cases}, \quad (2.27)$$

$$\begin{cases} \mathbf{v}^{n+1} = \mathbf{v}^{n+1/2} + h\mathbf{M}^{-1}(-\nabla \Psi(\mathbf{x}^{n+1}) - \nabla C(\mathbf{x}^{n+1}) + \nabla \hat{C}_{\text{sf}}(\mathbf{x}^{n+1})), \\ \mathbf{x}^{n+1} = \mathbf{x}^n + h\mathbf{v}^{n+1}, \end{cases}$$

where the fluid phase now also implicitly updates the solid boundary nodes near the fluids to an intermediate state.

For $\hat{C}_{\text{sf}}(\mathbf{x})$, a straightforward choice is $\hat{C}_{\text{sf}}(\mathbf{x}) = \frac{1}{2}C_{\text{sf}}(\mathbf{x})$. But to ensure our fluid phase only contains linear forces, we apply the second-order Taylor expansion of $\frac{1}{2}C_{\text{sf}}(\mathbf{x})$ at $\mathbf{x}^n$ for the approximation in the fluid phase; i.e.,

$$\hat{C}_{\text{sf}}(\mathbf{x}) = \frac{1}{2} \left(C_{\text{sf}}(\mathbf{x}^n) + \nabla C_{\text{sf}}(\mathbf{x}^n)(\mathbf{x} - \mathbf{x}^n) + \frac{1}{2} \|\mathbf{x} - \mathbf{x}^n\|_{\nabla^2 C_{\text{sf}}(\mathbf{x}^n)}^2 \right), \quad (2.28)$$

while in the solid-coupling phase, we simply use $\hat{C}_{\text{sf}}(\mathbf{x}) = \frac{1}{2}C_{\text{sf}}(\mathbf{x})$. In [Section A.2.2](#), we prove that our time splitting scheme with contact proxy only has an $\mathcal{O}(h^4)$ mismatch compared to the implicit Euler solution. Reformulating both phases [\(2.27\)](#) as optimization problems, we obtain

$$\begin{aligned} \mathbf{x}^{n+1/2} &= \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \hat{\mathbf{x}}^n\|_{\mathbf{M}}^2 + h^2(P(\mathbf{x}) + \hat{C}_{\text{sf}}(\mathbf{x})), \\ \mathbf{x}^{n+1} &= \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \mathbf{x}^{n+1/2}\|_{\mathbf{M}}^2 + h^2(\Psi(\mathbf{x}) + \frac{1}{2}C_{\text{sf}}(\mathbf{x}) + C_{\text{ss}}(\mathbf{x})), \end{aligned} \quad (2.29)$$

where $\hat{\mathbf{x}}^n = \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$ with $\mathbf{f}_{\text{ext}} = [\mathbf{f}_f^T, \mathbf{f}_s^T]^T$.

In addition to avoiding fluid particle sticking issues without extra computationally expensive costs, another benefit of our method is that it helps reduce the number of Newton iterations needed to solve the problem. Typically, the barrier method takes many Newton iterations to resolve high-speed impacts. With our scheme, when high-speed fluid particles are colliding with a deformable object, their speed will be significantly reduced after the fluid phase due to the contact proxy. The reduced speed will then be taken into the solid-coupling phase, which makes the nonlinear optimization easier to solve (by having fewer contact constraint set changes). The details of our proxy-based time splitting scheme are found in [Algorithm 1](#).

Similarly, one can also separate elasticity from contact energy using the contact proxy. In this fashion, we would have a three-phase (fluid, solid, and contact) time splitting

Algorithm 1: Time Splitting with Contact Proxy

```

1:  $\mathbf{x} \leftarrow \mathbf{x}^n$ ,  $\hat{\mathbf{x}}^n \leftarrow \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$ 
2: SPH Neighbor Search & Density Update
3:  $\hat{C}_{\text{sf}}(\mathbf{x}) \leftarrow$  2nd Taylor Expansion of  $\frac{1}{2}C_{\text{sf}}(\mathbf{x})$  at  $\mathbf{x} = \mathbf{x}^n$ 
4: // Fluid Phase
5:  $\mathbf{H} \leftarrow h^2 \left( \nabla^2 P(\mathbf{x}) + \nabla^2 \hat{C}_{\text{sf}}(\mathbf{x}) \right) + \mathbf{M}$ 
6:  $\mathbf{p} \leftarrow -\mathbf{H}^{-1} \left( h^2(\nabla P(\mathbf{x}) + \nabla \hat{C}_{\text{sf}}(\mathbf{x})) + \mathbf{M}(\mathbf{x} - \hat{\mathbf{x}}^n) \right)$ 
7:  $\mathbf{x} \leftarrow \mathbf{x} + \mathbf{p}$ 
8:  $\mathbf{x}^{n+1/2} \leftarrow \mathbf{x}$ 
9: // Solid-Coupling Phase
10: do
11:    $\mathbf{H} \leftarrow h^2 \left( \nabla^2 \Psi(\mathbf{x}) + \frac{1}{2}\nabla^2 C_{\text{sf}}(\mathbf{x}) + \nabla^2 C_{\text{ss}}(\mathbf{x}) \right) + \mathbf{M}$ 
12:    $\mathbf{g} \leftarrow h^2 \left( \nabla \Psi(\mathbf{x}) + \frac{1}{2}\nabla C_{\text{sf}}(\mathbf{x}) + \nabla C_{\text{ss}}(\mathbf{x}) \right) + \mathbf{M}(\mathbf{x} - \mathbf{x}^{n+1/2})$ 
13:    $\mathbf{p} \leftarrow -\mathbf{H}^{-1}\mathbf{g}$ 
14:    $\alpha \leftarrow$  Backtracking Line Search with CCD
15:    $\mathbf{x} \leftarrow \mathbf{x} + \alpha\mathbf{p}$ 
16: while  $\frac{1}{h}\|\mathbf{p}\| > \epsilon$ 
17:  $\mathbf{x}^{n+1} \leftarrow \mathbf{x}$ ,  $\mathbf{v}^{n+1} \leftarrow (\mathbf{x} - \mathbf{x}^n)/h$ 
18: return  $\mathbf{x}^{n+1}$ ,  $\mathbf{v}^{n+1}$ 

```

scheme

$$\begin{aligned}
\mathbf{x}^{n+1/3} &= \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \hat{\mathbf{x}}^n\|_{\mathbf{M}}^2 + h^2(P(\mathbf{x}) + \hat{C}_{\text{sf}}(\mathbf{x})), \\
\mathbf{x}^{n+2/3} &= \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \mathbf{x}^{n+1/3}\|_{\mathbf{M}}^2 + h^2(\Psi(\mathbf{x}) + \hat{C}_{\text{sf}}(\mathbf{x}) + \hat{C}_{\text{ss}}(\mathbf{x})), \\
\mathbf{x}^{n+1} &= \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \mathbf{x}^{n+2/3}\|_{\mathbf{M}}^2 + h^2\left(\frac{1}{3}C_{\text{sf}}(\mathbf{x}) + \frac{1}{2}C_{\text{ss}}(\mathbf{x})\right),
\end{aligned} \tag{2.30}$$

where $\hat{C}_{\text{sf}}(\mathbf{x})$ and $\hat{C}_{\text{ss}}(\mathbf{x})$ are the second-order Taylor expansions of $\frac{1}{3}C_{\text{sf}}(\mathbf{x})$ and $\frac{1}{2}C_{\text{ss}}(\mathbf{x})$, respectively. However, this aggressive splitting scheme only applies to inversion-robust constitutive models; e.g., the fixed corotated model (Stomakhin et al., 2012). While inversion may be guaranteed to be prevented at the solid phase where the elasticity energy is considered, this may not hold in the contact phase. Despite this limitation, the three-phase splitting scheme can still work properly for inversion-robust constitutive models in practice to further accelerate the simulation.

2.4.2 Solving Linear Systems

In our time splitting scheme, solving large sparse linear systems dominates both the computational and memory costs of each phase. We thus devise matrix-free and Schur-complement-based strategies to solve them efficiently.

Fluid Phase: Since 2-ring neighbors of SPH particles need to be considered in our formulation, both constructing and directly factorizing the Hessian matrix can cost a significant amount of time and memory. Therefore, we devise a matrix-free conjugate gradient (CG) solver to efficiently solve for the intermediate state of fluids.

As all energy potentials are quadratic in this phase, the energy gradient $\mathbf{g}(\mathbf{x})$ is merely a linear function of $\mathbf{x}$ with constant coefficient matrix $\mathbf{H}(\mathbf{x})$. Thus, the product between $\mathbf{H}(\mathbf{x})$ and an arbitrary vector $\mathbf{p}$ can be expressed as

$$\mathbf{H}(\mathbf{x})\mathbf{p} = \mathbf{g}(\mathbf{p}) - \mathbf{g}(\mathbf{0}). \quad (2.31)$$

This allows us to compute gradients to evaluate the matrix-vector product, and we only need to acquire the 3×3 diagonal blocks of the Hessian for block-Jacobi preconditioning in our CG solver.

Solid-Coupling Phase: As the fluid energy potential is not included in this phase, the components of the Hessian matrix become

$$\mathbf{H}_f = \frac{\partial^2 C(\mathbf{x})}{\partial \mathbf{x}_f^2}, \quad \mathbf{G} = \frac{\partial^2 C(\mathbf{x})}{\partial \mathbf{x}_s \partial \mathbf{x}_f}, \quad \mathbf{H}_s = \frac{\partial^2 C(\mathbf{x})}{\partial \mathbf{x}_s^2} + \frac{\partial^2 \Psi(\mathbf{x})}{\partial \mathbf{x}_s^2}. \quad (2.32)$$

Although this linear system is no longer that intractable, it is not optimal to directly factorize the whole system given the considerable amount of nonzeros in $\mathbf{H}_f$ and $\mathbf{G}$ when fluid resolution is high.

Thus, we design a domain decomposed linear solver that treats $\mathbf{H}_f$ and $\mathbf{H}_s$ separately. Based on the Schur complement (Zhang, 2006), the inverse of our Hessian matrix can be

expressed as

$$\mathbf{H}^{-1} = \begin{bmatrix} \mathbf{H}_f^{-1} + \mathbf{H}_f^{-1} \mathbf{G} (\mathbf{H}/\mathbf{H}_f)^{-1} \mathbf{G}^T \mathbf{H}_f & -\mathbf{H}_f^{-1} \mathbf{G} (\mathbf{H}/\mathbf{H}_f)^{-1} \\ -(\mathbf{H}/\mathbf{H}_f)^{-1} \mathbf{G}^T \mathbf{H}_f & (\mathbf{H}/\mathbf{H}_f)^{-1} \end{bmatrix}, \quad (2.33)$$

where $\mathbf{H}/\mathbf{H}_f = \mathbf{H}_s - \mathbf{G}^T \mathbf{H}_f^{-1} \mathbf{G}$ is the Schur complement of block $\mathbf{H}_f$. Since the nonzeros of $\mathbf{H}_f$ exist only in the diagonal blocks, it is trivial to obtain the inverse matrix $\mathbf{H}_f^{-1}$. We can then apply the CHOLMOD (Chen et al., 2008) LLT solver to factorize $\mathbf{H}/\mathbf{H}_f$, the dimensionality of which is on the order of the number of solid DOFs, and then the search direction can be computed via matrix-vector products and back-solves. When there is no solid-fluid interaction, the sparsity pattern of $\mathbf{H}/\mathbf{H}_f$ remains identical to that of $\mathbf{H}_s$. Only when two solid nodes i and j are interacting with the same fluid particle will, in 3D, the 3×3 block $(\mathbf{H}/\mathbf{H}_f)_{i,j}$ become non-zero. Typically, this happens only for neighboring mesh primitives, and thus the sparsity pattern of $\mathbf{H}/\mathbf{H}_f$ is mostly nice.

Note that when the three-phase time splitting scheme (2.30) is used, our domain decomposed solver can also be applied to the solid and contact phases, since their systems share a similar structure with the solid-coupling phase.

2.5 Experiments

Our code is implemented in C++ with Eigen for basic linear algebra operations and Intel’s TBB for multi-threading. The time step size of all our simulations is adaptively chosen by the SPH CFL condition and a user-defined upper bound. We set the support radius of our SPH kernel function to $2d$, where d is the particle diameter. In our implementation, we use the cubic Spline kernel for density estimation and the Spiky kernel for gradient calculation. For the results shown in Figure 2.10, Figure 2.8, Figure 2.5, and Figure 2.2, we employ our three-phase time splitting scheme, demonstrating its efficacy when the constitutive models are compatible with mesh inversion. For the remaining simulations, we stick with our two-phase time splitting scheme. Most experiments were performed on a 24-core 3.50GHz Intel i9-10920X workstation, except for the comparative study with

Scene	N_{fluid}	N_{solid}	k_I	ν_f	d	ρ_f	E	ν_s	ρ_s	T
Figure 2.2	97K	2.3K	2×10^5	0	15	1000	1×10^5	0.3	500	0.3
Figure 2.3a	238K	0	1×10^5	100	10	1200	-	-	-	0.4
Figure 2.5	103K	16K	1×10^5	0	10	1000	1×10^5	0.3	200	1.3
Figure 2.6	280K	0	2×10^5	0.005	25	1000	-	-	-	0.4
Figure 2.7a	52K	3.7K	1×10^5	0	10	1000	4×10^3	0.49	200	0.4
Figure 2.7b	101K	4.5K	6×10^4	0	6.4	1000	1×10^3	0.49	200	1.0
Figure 2.8	787K	66K	2×10^5	1	10	1000	1×10^5	0.4	200/700/1200	5.9
Figure 2.10*	486K	12K	4×10^4	0	5	1000	-	-	500	7.9
Figure 2.11	159K	9K	3×10^4	25	3	1000	5×10^8	0.49	1000	1.8
Figure 2.12*	789K	13K	1×10^5	0.2	10	1000	1×10^5	0.45	100/700	4.9
Figure 1.1*	1M	43K	2.5×10^5	0.1	25	1000	-	-	500	37.9

Table 2.1: Simulation statistics including the number of fluid particles N_{fluid} , number of solid vertices N_{solid} , incompressibility coefficient k_I , $[Pa]$, dynamic viscosity ν_f , $[Pa \cdot s]$, fluid particle diameter d , $[mm]$, fluid density ρ_f , $[kg/m^3]$, Young’s modulus E , $[Pa]$, Poisson’s ratio ν_s , solid density ρ_s , $[kg/m^3]$, and the average simulation time T , $[min]$ for each frame. Timing statistics are measured on a 24-core 3.50GHz Intel i9-10920X machine, except for Figure 2.7, which is tested on the “e2-standard-8” (8 cores with 32GB RAM) Google Compute Engine. Note that examples marked * contain codimensional materials whose parameter settings are not covered here.

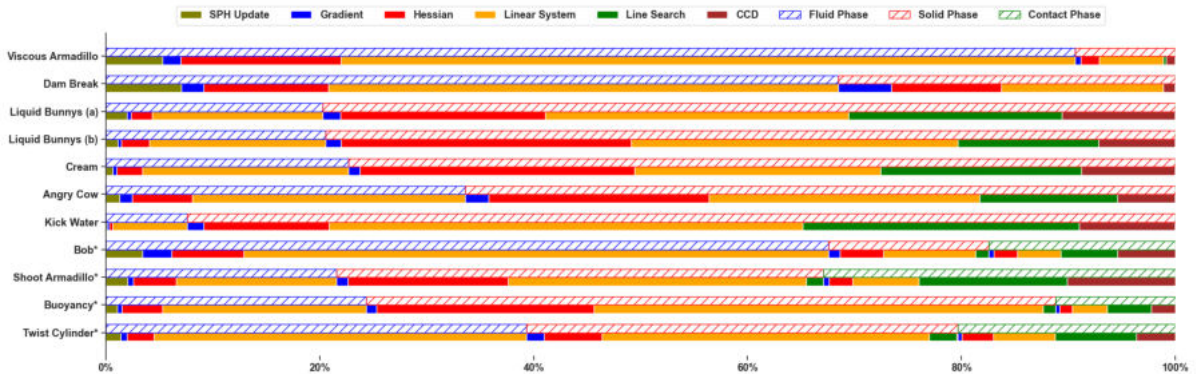


Figure 2.1: Timing breakdown. We show the timing profile of different simulation phases and plot the proportions of the major routines. Examples marked with * are simulated using our three-phase time splitting scheme. Other examples are generated with the two-phase scheme. In particular, SPH update (including neighborhood search and density update) only occurs in the fluid phase, line search happens in the solid and contact phases for non-linear optimization, and continuous collision detection (CCD) is counted when IPC contact energy is considered.

ElastoMonolith (Takahashi and Batty, 2022). We demonstrate that our method achieves efficient and robust solid-fluid coupling. The parameters and timing breakdown of our simulations are specified in Table 2.1 and Figure 2.1, respectively.



Figure 2.2: Bob simulated with (A) Joint Optimization, (B) Time Splitting with Contact Proxy, and (C) Baseline Time Splitting. For this example, baseline time splitting can also produce visually plausible results, and our proxy-assisted scheme is $3\times$ faster than joint optimization.

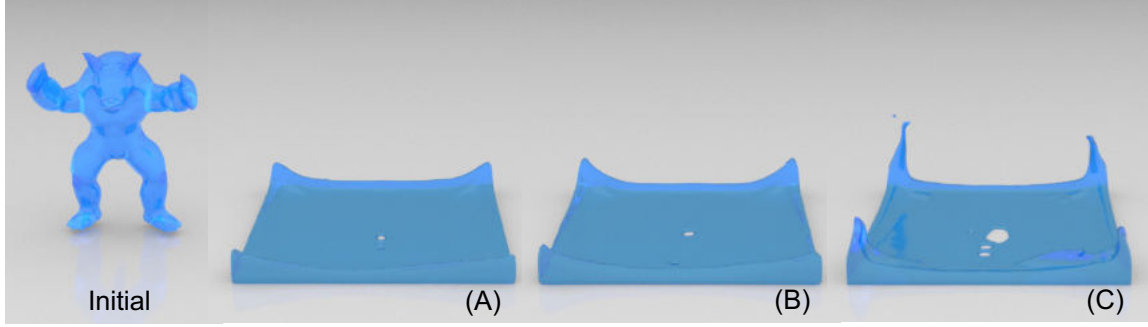
Scene	Scheme	Sec/Frame	# Newton Iter./Frame
Figure 2.2	Joint/TS/TSCP	66.1 / 38.0 / 22.5	63.5 / 117.3 / 37.1
Figure 2.3a	Joint/TS/TSCP	41.3 / 32.3 / 25.5	16.5 / 29.0 / 10.5

Table 2.2: Statistics of different time stepping schemes. Joint Optimization (Joint), Baseline Time Splitting (TS), and Time Splitting with Contact Proxy (TSCP). Our TSCP is much faster than both the Joint and TS schemes.

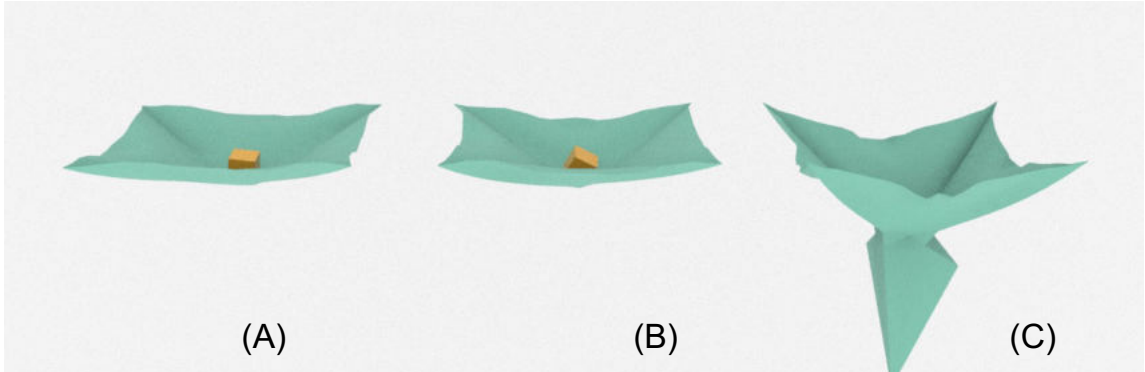
2.5.1 Ablation Study

Time Splitting Evaluation: Three simulations (Figure 2.2 and Figure 2.3) are performed to demonstrate the efficiency of time splitting and the efficacy of our contact proxy in maintaining stability.

To begin, we must choose a proper time step h . First, it must be restricted by the CFL condition, otherwise severe volume loss may be observed due to the SPH approximation error. Additionally, in contrast to the joint optimization (2.23), the time splitting scheme usually requires smaller time steps to remain stable, which imposes a second time step constraint. However, we observed that, in practice, even using the largest CFL time step, our proxy-assisted time splitting can still work properly and produce stable simulation results. Hence, for comparison, we use the largest CFL time step for both schemes to maximize their performance, as smaller h typically takes more Newton iterations in total to simulate a frame. For the joint optimization, since direct factorization is intractable, we solve (2.24) using the block-Jacobi preconditioned conjugate gradient solver with the fluid part of the matrix free.



(a) A viscous armadillo is dropped onto the ground.



(b) Cube on cloth. An elastic cube is dropped onto a square cloth with four corners fixed.

Figure 2.3: Simulation results of (A) Joint Optimization, (B) Time Splitting with Contact Proxy, and (C) Baseline Time Splitting. While directly applying time splitting results in instability at the boundaries, our results with the contact proxy are consistent with joint optimization.

As shown in [Table 2.2](#), even in these simple examples, our time splitting scheme is significantly, up to $3\times$ faster than joint optimization, especially for cases involving contacts between fluids and deformable solids (e.g., [Figure 2.2](#)). This improvement stems from no longer having to solve for the incompressibility of fluids repeatedly within a time step. Moreover, one can also find that much fewer Newton iterations are needed with our proxy-assisted time splitting scheme. As discussed in [Section 2.4.1](#), this is because the challenging high-speed impacts are already partially resolved in the fluid phase. Another benefit of time splitting is the support of different error tolerances for the two phases. Errors in the fluid phase are sourced from the solution deviation of the CG solver, while in the solid phase they are directly controlled by the tolerance of Newton’s method. Typically, setting a slightly higher tolerance for fluids yields better performance while

still producing visually convincing results.

Aside from its efficiency, our proposed contact proxy also improves the stability of the time splitting scheme. Although the simulation results of the baseline time splitting scheme look fine in the case of inviscid fluids, matters get worse when it is applied to viscous fluids. In [Figure 2.3a](#), a viscous armadillo is dropped to the ground. In this example, the baseline time splitting scheme produces severe sticky artifacts at the boundary, and the fluid surface cannot eventually calm down. By consistently applying our contact proxy to exert boundary pressure in the fluid phase, the artifacts can be well resolved, as demonstrated in [Figure 2.3a](#). Similarly, our idea of contact proxy is also applicable to further separate elasticity from IPC contact while maintaining stability, leading to our three-phase scheme ([Figure 2.3b](#)).

Linear Solver Evaluation: For the fluid phase, we designed a matrix-free conjugate gradient (CG) solver that calculates the matrix-vector product via a gradient computation to avoid the expensive computational and memory costs of direct factorization ([Section 2.4.2](#)). However, the performance improvement from this approach is less significant if the number of CG iterations required for convergence is too large, making the cost of computing gradients higher than constructing the Hessian once. In our fluid phase, the number of CG iterations is proportional to the stiffness k_I of the incompressibility energy. A larger k_I can better preserve the volume of the fluids, but also results in a worse-conditioned system, demanding more iterations to converge ([Figure 2.4](#)). In practice, by setting k_I to a proper value, we can efficiently solve the systems within 50 CG iterations without obvious fluid volume losses.

We test the performance of our matrix-free CG solver together with the domain-decomposed solver that we designed for the solid-coupling phase using an armadillo shooting example ([Figure 2.5](#)), and present our results in [Table 2.3](#). Our matrix-free CG solver significantly boosts efficiency ($20\times$ faster) and reduces memory costs by avoiding the construction of the Hessian matrix. On the other hand, our domain decomposed solver is 40% faster than directly factorizing the solid and contact systems.

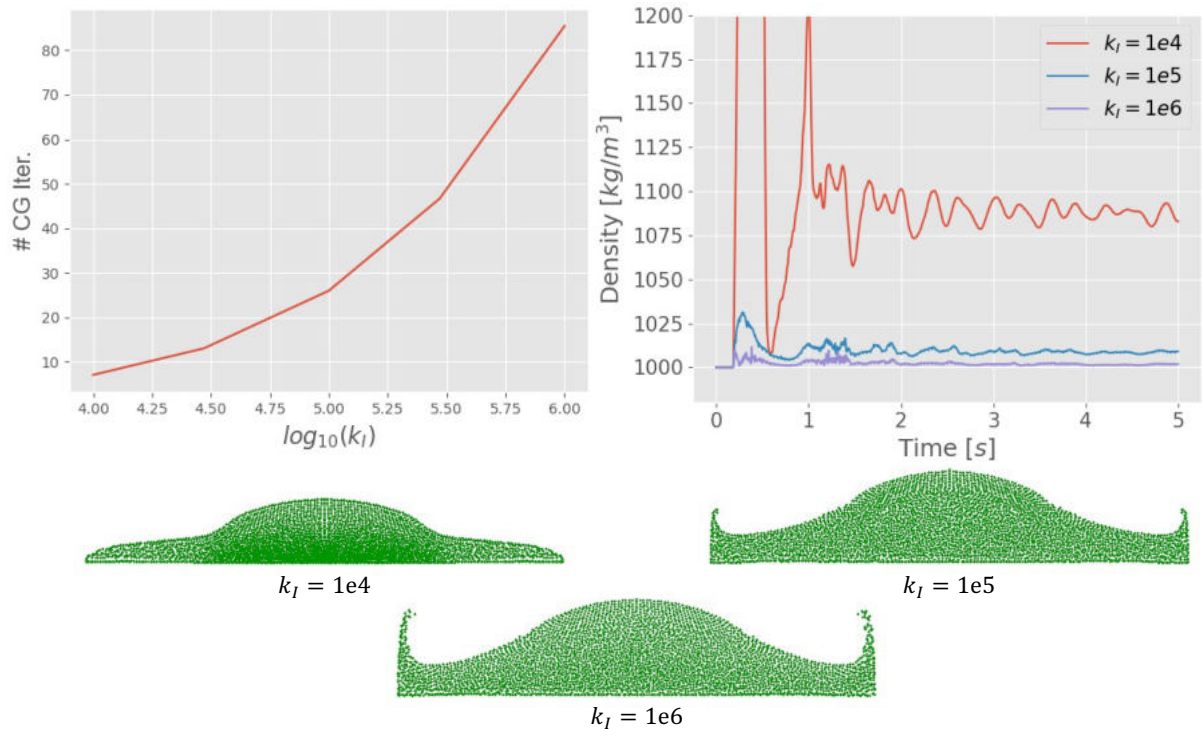


Figure 2.4: Statistics of simulations with different values of stiffness parameter k_I . A larger k_I preserves volume better, but necessitates more CG iterations. In this case, a proper $k_I = 10^5$ Pa can be set to balance the computational cost and visual artifacts.



Figure 2.5: Shooting an armadillo with a high-speed water jet.

Solver	Fluid Phase		Solid Phase	Contact Phase	Mem.
	hess	solve	solve	solve	
CG + LLT	14.9	0.49	1.45	0.43	12375
Ours	0.15	0.59	1.11	0.25	1469

Table 2.3: Time and memory cost of different solvers in example [Figure 2.5](#). The costs are measured per time step in units [s] and [MB], respectively. The baseline method uses the Conjugate Gradient (CG) method for the fluid phase and CHOLMOD LLT for the solid and contact phases. Instead, our method employs a matrix-free CG solver for the fluid phase and a domain-decomposed solver for the solid and contact phases, thereby improving efficiency and saving memory.

2.5.2 Comparisons

In this section, we compare our method with several popular SPH fluid solvers and a state-of-the-art solid-fluid coupling method, ElastoMonolith ([Takahashi and Batty, 2022](#)). We leveraged the open-source library SPLisHSPlasH¹ to implement the SPH fluid simulators. To compare our method with ElastoMonolith, we set up two scenes from the aforecited publication with identical parameters and ran all the simulations using an “e2-standard-8” (8 cores with 32GB RAM) Google Compute Engine for fairness.

Fluid Dynamics: While most existing SPH fluid solvers focus on incompressible fluids, our formulation treats fluids as weakly compressible, allowing us to couple fluids with deformable solids in a unified framework. We ran a dam break simulation to compare our method with two SPH fluid solvers, IISPH ([Ihmsen et al., 2013](#)) and DFSPH ([Bender and Koschier, 2015](#)). These methods typically use particle resampling ([Akinici et al., 2012, 2013](#)) or implicit representation ([Koschier and Bender, 2017; Bender et al., 2019](#)) to exert boundary counter-forces. Our method instead employs IPC ([Li et al., 2020](#)) for more robust solid-fluid coupling, with a penetration-free guarantee. We uniformly enforce the same CFL condition for all methods along with an upperbound at 5 ms , and use the volume map ([Bender et al., 2019](#)) for their boundary handling. As shown in [Figure 2.6](#), although our formulation does not strictly enforce incompressibility, it

¹<https://github.com/InteractiveComputerGraphics/SPLisHSPlasH>

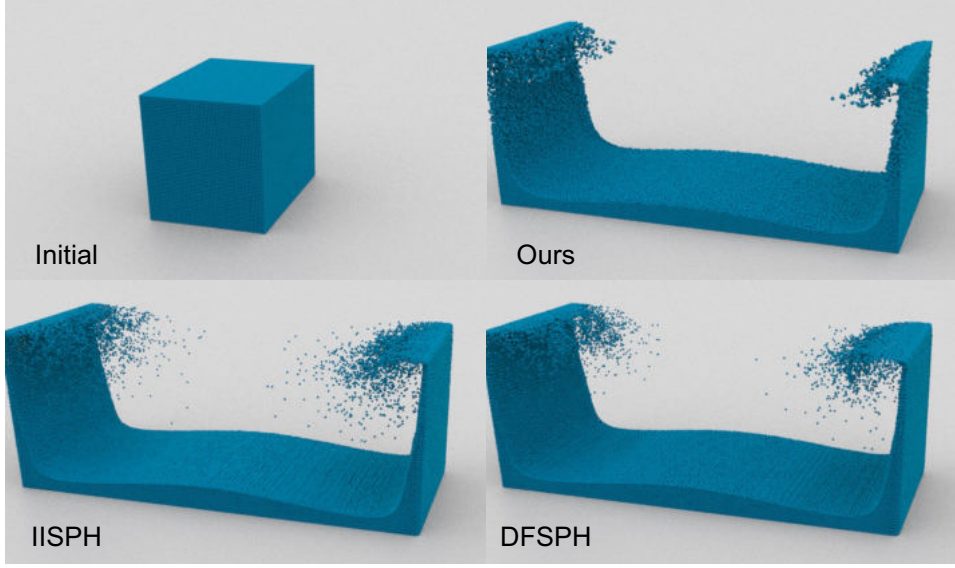


Figure 2.6: Dam break with 280K SPH particles. Our weakly compressible formulation produces stable fluid dynamics without visually evident volume loss. Compared to incompressible SPH solvers IISPH (Ihmsen et al., 2013) and DFSPH (Bender and Koschier, 2015), our simulation results demonstrate more smooth particle distribution.

produces natural fluid dynamics without visually observable volume loss. On the other hand, our method (0.45 min/frame) is slower than IISPH (0.31 min/frame) and DFSPH (0.15 min/frame) due to the more sophisticated boundary handling strategy. However, our approach can couple SPH fluids and elastic solids with arbitrary constitutive models, while most existing SPH methods (Peer et al., 2018; Kugelstadt et al., 2021) treat elastic solids as incompressible, which is not generally applicable.

Solid-Fluid Coupling: We now compare our method with ElastoMonolith (Takahashi and Batty, 2022), which couples Eulerian fluids with Lagrangian solids in a monolithic manner. Following their experiment setting, we ran two solid-fluid coupling simulations with identical parameters using our method (Figure 2.7). The timings of our method for these two scenes are 24.1 sec/frame and 62.8 sec/frame, respectively, both of which are over $5\times$ faster than ElastoMonolith according to their reported timings (253.2 sec/frame and 352.0 sec/frame). Coupling Eulerian fluids with Lagrangian solids requires dealing with geometric differences and SPD reformulation is often needed to make the linear system tractable. As stated in the publication on ElastoMonolith, this SPD reformulation can



(a) A liquid bunny dropped into a bowl.



(b) A liquid bunny and an elastic bowl dropped onto a static torus.

Figure 2.7: Liquid Bunnies. Compared to ElastoMonolith (Takahashi and Batty, 2022), our method achieves an over $5\times$ speedup for both of these examples with exactly the same scene setups.

introduce many additional non-zeros into the system, especially when contacts are rich and solids are intricately shaped. By contrast, our method treats solids and fluids from a unified Lagrangian viewpoint, where solid-solid and solid-fluid contacts are resolved in a unified manner.

2.5.3 Complex Scenarios

We next evaluate the efficiency and robustness of our method in more complicated scenarios.

Buoyancy: We drop three elastic elephants with varying densities into water (1000 kg/m^3) (Figure 2.8). The light grey elephant (200 kg/m^3) floats on the surface, the blue



Figure 2.8: Buoyancy. Three elastic elephants with different densities (left to right: 200, 700, and 1200 kg/m^3) fall into the water, demonstrating buoyancy.

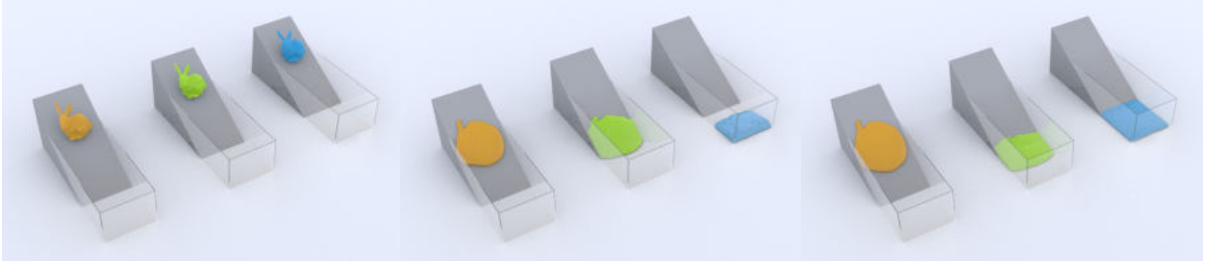


Figure 2.9: Varying friction. Three viscous bunnies are dropped onto the ramp with different coefficients of friction μ (left to right: 0.5, 0.03, 0.0). Our method supports adjustable solid-fluid boundary friction.

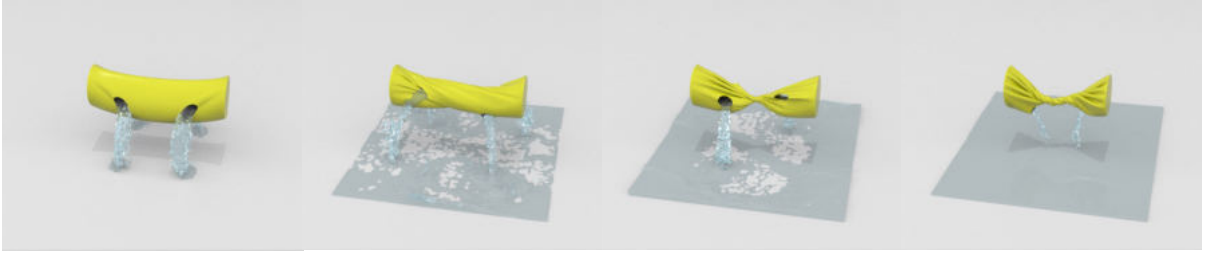


Figure 2.10: Twist cylinder. A cylindrical cloth with four holes is twisted, squeezing out water from the interior.

elephant (700 kg/m^3) is about half immersed in the water, and the red elephant (1200 kg/m^3) sinks to the bottom. This demonstrates that our method correctly captures buoyancy behavior.

Varying Friction: We drop three viscous bunnies onto the slope with different coefficients of friction (orange bunny: 0.5, green bunny: 0.03, blue bunny: 0) (Figure 2.9). All three bunnies share the same dynamic viscosity coefficients 100 kg/m^3 and the angle of slope is 30° .



Figure 2.11: Cream is stirred, causing the deformation of the spoon.

Twist Cylinder: Coupling fluids with thin shells is challenging since, without careful treatment, penetration can easily occur. As stated in (Zarifi and Batty, 2017), Eulerian fluids may flow through solids if their thickness is less than the grid cell size. Conversely, our approach adopts a unified Lagrangian view and IPC guarantees penetration-free results. In the example shown in Figure 2.10, we simulate the twisting of a cylinder full of water. The cylinder is modeled as a thin shell with a 2 mm thickness, and there are two holes in the front and back sides of this cylinder, respectively. The left side and right side are rotated at $72^\circ/s$ and are slowly moved towards each other at $2\text{ cm}/s$. As we twist the cylinder, the water gets squeezed out through the holes. This simulation demonstrates our method produces stable simulation results with a penetration-free guarantee.

Cream: The example of Figure 2.11 exhibits the coupling behaviors of viscous fluids and elastic solids. We use an elastic spoon to stir the cream in a porcelain bowl. The spoon handle rotates around y-axis at $360^\circ/s$ ($0.2\text{ m}/s$) while the bowl is fixed on the table. As shown in our simulation results, the spoon deforms due to the resistance forces subjected upon it by the viscous cream while stirring.

Angry Cow: We next show that our framework can simulate natural physical behaviors of geometries in arbitrary codimensions (0, 1, 2, and 3) as well as their interactions. In

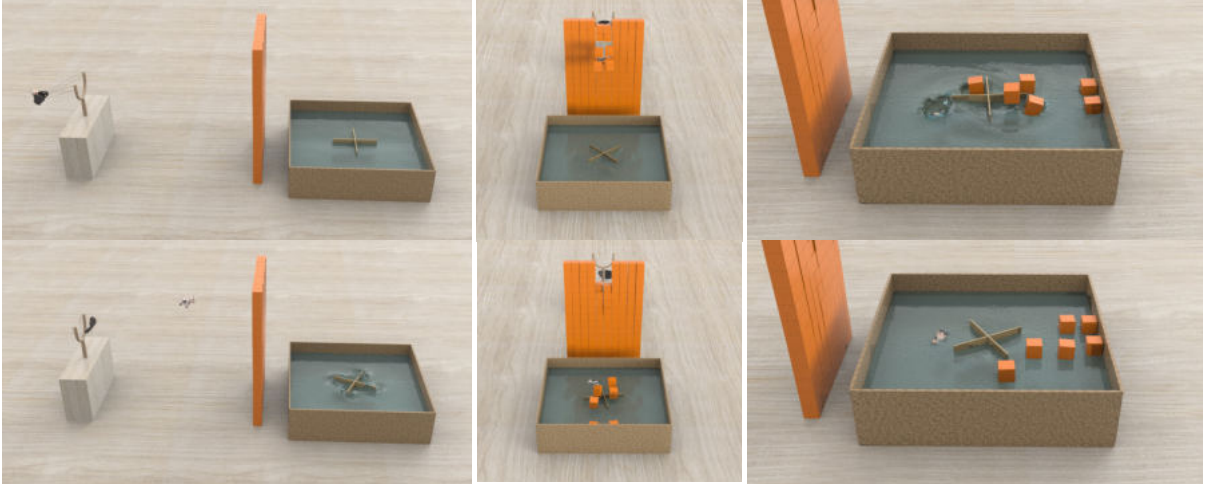


Figure 2.12: Angry cow. We show that our method can simulate the coupling of materials in arbitrary codimensions, including fluid particles, rods (rubber bands), thin shells (leather pad), deformable solids (cow), and rigid bodies (cubes). We launch an angry cow with a slingshot, and the cow penetrates through the wall and then falls into the water. All the interactions between the various materials are captured accurately.

the scene shown in Figure 2.12, the codimensional-0,1,2 objects respectively refer to fluid particles, rubber bands, and the leather pad. A deformable cow is launched by the slingshot, hitting the wall consisting of rigid cubes, and then falling into the water pool, producing interesting physical behaviors. The density of the rigid cubes and the cow are 100 kg/m^3 and 700 kg/m^3 , respectively.

Kicking Water: In the example of Figure 1.1, we show a scene in which a mannequin dressed in a multilayer skirt kicks in a large pool of water involving complex interactions between fluid particles and garments. As the mannequin kicks vigorously in and above the water, our method simulates the natural deformations of the skirt caused by the contact with the water, and the resulting water splashing is also correctly captured. Our method resolves the contacts among the fluids particles, the thin garments, and the complex, rapidly moving boundaries accurately and with penetration-free guarantees.

2.6 Conclusions and Future Work

We have presented a unified two-way strong coupling framework for weakly-compressible SPH fluids and nonlinear elastic FEM solids. To achieve this, we modeled solid-fluid interactions as contact forces between SPH particles and FEM boundary elements, applying the IPC method for guaranteed non-penetration and stability. As we track the volume change of SPH particles in an updated Lagrangian fashion, the incompressibility energy stays quadratic and nice particle distributions are maintained. Utilizing a symmetric approximation of discrete viscosity forces, we proposed a viscosity potential compatible with optimization time integration. We then proposed a time splitting scheme with a contact proxy to efficiently solve the time integration optimization while maintaining robustness. Performance is further boosted by our matrix-free conjugate gradient method and a domain-decomposed solver based on the Schur complement.

Compared to existing works ([Zarifi and Batty, 2017](#); [Takahashi and Batty, 2022](#)) coupling Eulerian fluids with Lagrangian elastic solids, our method treats both fluids and solids in a Lagrangian manner, avoiding the need to deal with different spatial discretizations. Under such a unified view, our method achieves more convenient and robust two-way coupling, even between fluids and codimensional solids. Likewise, unlike existing SPH methods ([Kugelstadt et al., 2021](#); [Peer et al., 2018](#)) that treat all materials as SPH particles, our formulation enjoys both the efficiency of SPH fluids and the accuracy of FEM solids.

There are many worthwhile future research directions. First, when fluid DOFs dominate, building and querying the spatial hash for each fluid particle can manifest a considerable cost. In fact, since there is no solid-fluid contact among interior particles, for better efficiency we can construct the spatial data structure only in the intersection between the extended bounding boxes of the fluids and each solid. Additionally, the adhesion between solids and fluids is also an interesting behavior to model. Similar to the barrier energy, adhesion forces can be exerted on close solid-fluid primitive pairs but in the opposite direction. Modeling adhesion by resolving the surface tension of fluids is

also an interesting avenue for future work.

CHAPTER 3

Self-Supporting Text-to-3D Generation with Physics Constraints

Existing diffusion-based text-to-3D generation methods primarily focus on producing visually realistic shapes and appearances, often neglecting the physical constraints necessary for downstream tasks. Generated models frequently fail to maintain balance when placed in physics-based simulations or 3D-printed. This balance is crucial for satisfying user design intentions in interactive gaming, embodied AI, and robotics, where stable models are needed for reliable interaction. Additionally, stable models ensure that 3D-printed objects, such as figurines for home decoration, can stand on their own without requiring additional support. To fill this gap, we introduce Atlas3D, an automatic and easy-to-implement method that enhances existing Score Distillation Sampling (SDS)-based text-to-3D tools. Atlas3D ensures the generation of self-supporting 3D models that adhere to physical laws of stability under gravity, contact, and friction. Our approach combines a novel differentiable simulation-based loss function with physically inspired regularization, serving as either a refinement or a post-processing module for existing frameworks. We verify Atlas3D’s efficacy through extensive generation tasks and validate the resulting 3D models in both simulated and real-world environments.

3.1 Introduction

Generating high-quality 3D content is of great importance in modern visual computing. Realistic 3D models are highly sought after in computer graphics, while robust real 3D assets are gaining attention in training embodied AI. Nevertheless, the standability of

3D models—the ability to stand steadily without additional support—is often neglected. Real-world man-made objects such as action , toys, and furniture inherently possess some degree of geometric stability, allowing them to be safely placed on the ground. Although one usually takes such standability for granted, existing generative models fail to produce steady 3D assets due to their lack of physical perception (Figure 3.1).

Incorporating this stability expectation into 3D generation will significantly reduce the human effort required for tasks such as sorting out unqualified meshes, post-processing geometries, or adding external supports before actually using the 3D asset in any simulator or the real world. Furthermore, creating physically plausible 3D content will enhance the fidelity of simulations and policy training with these objects, potentially narrowing the sim-to-real gap and empowering embodied AI in robotic tasks. Toward this goal, we develop a 3D generation framework that can produce high-quality models adhering to basic physical laws, such as gravity, stability, and frictional contact.

Several attempts have been made to incorporate physical constraints into 3D generation. Yang et al. utilized the spatial and physical sense of LLM to design floor plans and furniture arrangements (Yang et al., 2024d). PhyScene introduced physical guidance, such as collision and reachability constraints, to diffusion models to generate furniture layouts (Yang et al., 2024c). However, both works primarily consider straightforward spatial constraints, such as non-collision, and fail to incorporate more complex physics. Mezghanni et al. proposed a GAN-based network to generate physically-aware geometries by training a neural stability predictor using datasets labeled by Bullet (Mezghanni et al., 2021). Another GAN-based work by Wang et al. employed CFD software to compute vehicle drag coefficients, guiding the generation of streamlined vehicle meshes (Wang and He, 2019). Such indirect incorporation of physical simulations, however, results in suboptimal efficiency and accuracy. Furthermore, due to the low expressibility of the backbone latent representation, the versatility of the generated results is very limited compared to current state-of-the-art diffusion-based models, as the results are typically confined to specific categories (e.g., furniture and vehicles). Most recently, Ni et al. bridged differentiable physical simulation with differentiable rendering to obtain virtual

3D reconstructions from real-world images that are physically plausible in simulators. Their work primarily focused on simple four-leg-supported objects such as tables and chairs (Ni et al., 2024). Moreover, the evaluations of all aforementioned works are conducted in virtual simulators, leaving their performance in the real world untested. This limitation hinders potential downstream applications such as industrial manufacturing and robotic manipulation.

Since the pioneering work DreamFusion (Poole et al., 2022), Score Distillation Sampling (SDS) has demonstrated efficacy in elevating 2D content to 3D, inspiring numerous follow-up studies (Chen et al., 2023c; Lin et al., 2023a; Metzger et al., 2023; Shi et al., 2023b; Wang et al., 2024d; Zhu and Zhuang, 2023). These advancements have enhanced both the versatility of generated content and the quality of textures. However, none have addressed the crucial issue of physical stability. On the other hand, traditional computational fabrication has concentrated on employing topology and shape optimization to ensure that 3D printed objects can stand in a balanced state (Prévost et al., 2013). Directly integrating these methods with 3D generative AI as a postprocessing module is suboptimal. Shape optimization disregards the original input conditions of diffusion models, while topology optimization produces internal structures that defy intuitive physics, rendering them unsuitable for training embodied AI systems designed to emulate human-like reasoning about physical objects.

Observing this gap, we introduce Atlas3D, a generation pipeline that produces physically plausible, self-supporting 3D models from text. Incorporating differentiable physics-based simulation into our process, we generate models that are both simulation- and fabrication-ready. That is, they can be directly utilized in physical simulators, or 3D-printed for real-world applications; see Figure 3.1 and Figure 1.2. As our method is orthogonal to previous SDS-based techniques, which focus on non-physical qualities, it can be seamlessly integrated into many existing generation frameworks, functioning either as part of the refinement stage in a multi-stage method or as a post-processing step in a single-stage method. We demonstrate the efficacy of Atlas3D by comparing the stability of our models with those produced by existing methods. Validation examples reveal



Figure 3.1: Simulation in ABD (Lan et al., 2022): (a) 3D models generated from our Atlas3D framework can stand steadily on the ground; (b) those generated from existing methods tend to fall over.

that our generated models can be deployed as virtual simulation assets. Their stability transfers directly to the real world, as evidenced by our 3D printed results, suggesting further applications in robot training.

3.2 Related Work

Diffusion-based 3D Generation: Due to the abundance of information encoded in large image latent diffusion models (LDMs) (Rombach et al., 2021), extensive studies have used pre-trained LDMs to distill 3D content. One approach is to fine-tune LDMs to support novel view synthesis, with a separate multiview fusion step to produce 3D content (Liu et al., 2023b; Shi et al., 2023a; Ye et al., 2023a; Liu et al., 2024b, 2023a; Weng et al., 2023; Yang et al., 2023a; Long et al., 2023; Tang et al., 2023d; Kong et al., 2024; Zheng and Vedaldi, 2023; Liu et al., 2023e; Chen et al., 2024c; Voleti et al., 2024). Another approach, which is more related to our paper, is using LDMs as likelihood discriminators. A differentiable renderer is connected to a 3D representation, and the

LDMs guide the optimization of the representation parameters. (Poole et al., 2022) proposed Score Distillation Sampling (SDS). Efficiency has been improved by coarse-to-fine strategies (Lin et al., 2023a; Qian et al., 2024; Tang et al., 2023c; Chen et al., 2023c) and timestep scheduling (Huang et al., 2023b; Yi et al., 2024). 3D priors are involved to improve multiview consistency (Zhou and Tulsiani, 2023; Seo et al., 2023; Li et al., 2023h; Anciukevičius et al., 2023; Xu et al., 2023d; Lin et al., 2024; Tang et al., 2023b). Multiview diffusions can also be used to evaluate SDS (Shi et al., 2023b; Wang and Shi, 2023; Zhao et al., 2023a; Yang et al., 2024a). Other researchers have explored SDS variants or improvements (Wang et al., 2023b; Tang et al., 2023a; Wang et al., 2024d; Katzir et al., 2024; Yu et al., 2024; Hong et al., 2024; Zhu and Zhuang, 2023; Yang et al., 2023b). 3D LDMs that directly generate 3D representations are also explored, such as compositional scenes (Huang et al., 2023a; Po and Wetzstein, 2023), point clouds (Luo and Hu, 2021; Vahdat et al., 2022; Mo et al., 2024), SDFs (Shim et al., 2023; Zheng et al., 2023b; Li et al., 2023f), occupancy fields (Gupta et al., 2023; Shue et al., 2023; Mercier et al., 2024; Erkoç et al., 2023) and NeRFs (Bautista et al., 2022; Dupont et al., 2022; Müller et al., 2023; Chen et al., 2023a; Ntavelis et al., 2023; Cao et al., 2023; Szymanowicz et al., 2023; Karnewar et al., 2023).

Physics-Aware 3D Generation: Most existing 3D generative models focus only on geometry or appearance modeling, with physics priors being underexplored. Time-independent physical constraints, such as penetrations, can be directly defined by penalties (Yang et al., 2024c; Huang et al., 2023a; Vilesov et al., 2023; Liu et al., 2023d; Yuan et al., 2023). For time-dependent physical qualities, such as stability and comfort, data-driven quality checkers trained with offline simulators can be applied (Dong et al., 2024; Mezghanni et al., 2021; Blinn et al., 2021). Offline simulators can also be used as validators to augment the training dataset (Shu et al., 2020) and update the design with reinforcement learning (Wang and He, 2019), and as dynamics generators for generated 3D assets (Xie et al., 2023b; Jiang et al., 2024c; Qiu et al., 2024). Another direction is to utilize differentiable simulations, which have been widely used in tasks like robotic control

(Sundaresan et al., 2022; Huang et al., 2021; Qiao et al., 2021; Li et al., 2022f; Takahashi et al., 2021) or inverse problems (Li et al., 2023k; Huang et al., 2022b; Zhang et al., 2023e; Murthy et al., 2020; Li et al., 2022e; Strecke and Stueckler, 2021; Chen et al., 2022; Zhang et al., 2024b). They can also be applied in 3D content generation to define physics-based losses to aid per-instance generation (Ni et al., 2024; Xu et al., 2024b) or model training (Mezghanni et al., 2022).

3.3 Background

Score Distillation Sampling: SDS-based methods are shown to be effective in distilling 3D models from 2D images. They utilize a 3D representation such as implicit density field, implicit Signed Distance Field (SDF), or tetrahedral SDF (Shen et al., 2021), a differentiable renderer like NeRF (Mildenhall et al., 2021), NeuS (Wang et al., 2021), or Nvdiffrast (Laine et al., 2020), and a pre-trained text-to-image model such as Stable Diffusion (Rombach et al., 2022) serving as diffusion guidance. The generation process involves optimizing the parameters θ of the underlying 3D representation, where the 3D shape is differentially rendered to 2D images $\mathbf{z} = g(\theta)$ and compared against the real distribution from the diffusion model with text guidance y :

$$\mathcal{L}_{\text{SDS}} = \mathbb{E}_{t,\epsilon} \left[w(t) \|\hat{\epsilon}_{\phi}(\mathbf{z}_t, y, t) - \epsilon\|^2 \right], \quad (3.1)$$

where $\mathbf{z}_t$ is the noisy image at noise level t , $w(t)$ is a weighting function, and $\hat{\epsilon}_{\phi}$ is the predicted noise. [Appendix B](#) provides more technical details on SDS optimization.

Rigid Body Dynamics: To incorporate physics into our framework, we propose predicting the dynamics of the generated 3D models using a differentiable simulator, where all objects are treated as rigid bodies. We follow the conventions in (Baraff, 2001) to define the dynamical states of the simulation.

The kinematics of a rigid body are described by its mass M and body-space inertia

tensor $\mathbf{I}_{\text{body}}$, which remains constant. Assuming the center of mass of the body initially lies at the origin, the physical state Ψ of the body at time t (not to be confused with the noise level in diffusion) includes position $\mathbf{T}(t)$ and orientation $\mathbf{R}(t)$ (spatial information), and its linear and angular momentum $\mathbf{P}(t)$ and $\mathbf{L}(t)$ (velocity information). The rigid body equations of motion are given by

$$\frac{d}{dt}\Psi(t) = \frac{d}{dt} \begin{pmatrix} \mathbf{T}(t) \\ \mathbf{R}(t) \\ \mathbf{P}(t) \\ \mathbf{L}(t) \end{pmatrix} = \begin{pmatrix} \mathbf{v}(t) \\ \boldsymbol{\omega}(t) * \mathbf{R}(t) \\ \mathbf{F}(t) \\ \boldsymbol{\tau}(t) \end{pmatrix}, \quad (3.2)$$

where $\mathbf{F}(t)$ and $\boldsymbol{\tau}(t)$ are the total force and torque exerted on the body, $\mathbf{v}(t) = \frac{\mathbf{P}(t)}{M}$ is the linear velocity, $\boldsymbol{\omega}(t) = \mathbf{I}(t)^{-1}\mathbf{L}(t)$ is the angular velocity, $\mathbf{I}(t) = \mathbf{R}(t)\mathbf{I}_{\text{body}}\mathbf{R}(t)^T$ is the world-space inertia tensor, and $*$ denotes cross product of $\boldsymbol{\omega}$ with the columns of $\mathbf{R}$. The physical state at a later time can be derived via time integration: $\Psi(t) = \Psi(0) + \int_0^t \frac{d\Psi}{ds}(s)ds$, which can be solved by numerical methods. By optimizing the physical states together with the SDS loss, we can jointly refine both the 3D geometry and the physical attributes of the generated results.

3.4 Method

We introduce Atlas3D, a plug-and-play algorithm for generating 3D models from text. Focusing on man-made objects such as action figures and toys, which generally do not deform, Atlas3D treats generated models as rigid bodies and incorporates physics-based guidance into the generation process.

3.4.1 Physics Incorporation

As mentioned in [Section 3.3](#), we predict the dynamic behavior of generated models by rigid body simulations. While various explicit or implicit representations of 3D shapes

can be chosen in a generation network, we opt for triangular meshes in our framework as they facilitate frictional contact modeling and simplify kinematics computation. Given a triangle surface mesh representation $\mathbf{X}(\theta)$, where θ is the implicit parameter, we integrate $\mathbf{X}$ into a rigid body represented by the dynamic state $\Psi(t) = [\mathbf{T}(t), \mathbf{R}(t), \mathbf{P}(t), \mathbf{L}(t)]^T$, where the world-space location $\mathbf{x}$ of any point $\mathbf{X}$ on the body is $\mathbf{x}(t) = \mathbf{R}(t)\mathbf{X} + \mathbf{T}(t)$. Assuming the 3D model is initially placed upright² on the ground with the bottom point touching the surface, we define standability as:

$$\lim_{t \rightarrow \infty} \Psi(t) = \Psi(0). \quad (3.3)$$

Standability intuitively indicates an equilibrium state where all external forces acting on the object are balanced, and the physical state remains unchanged over time. However, perfectly placing an object straight on the ground without initial velocity is impractical in the real world. For example, when manually placing a cube on a flat table, the bottom face is unlikely to be perfectly parallel to the table surface. A stable 3D model should recover its initial state under mild perturbations, such as minor shaking. This state is known as stable equilibrium. Motivated by this, we augment standability with stable equilibrium $\tilde{\Psi}(t)$ defined as

$$\tilde{\Psi}(t) = \Psi(0) + \epsilon_0 + \int_0^t \frac{d\Psi}{ds}(s)ds \quad \text{and} \quad \lim_{t \rightarrow \infty} \tilde{\Psi}(t) = \Psi(0), \quad (3.4)$$

where ϵ_0 represents mild perturbations to the initial physical state. We first describe how to incorporate the standability criterion (3.3) into the optimization process of 3D generation, and then explain how to further augment it with stable equilibrium (3.4). Additionally, we introduce geometry regularization to enhance the smoothness of generated meshes.

²We define the upright pose by setting the upward axis to coincide with the upward axis from the pre-trained text-to-3D model, as the upright direction is semantics-driven and generative models inherently learn these semantics from the training data.

Standability Through Differentiable Simulation: We utilize a differentiable rigid-body simulator to obtain the physical state $\Psi(t)$. Assuming that the 3D model will eventually reach its steady state, $\exists T \in \mathbb{R}$ large enough such that $\forall t > T, \Psi(t) = \Psi(T)$. Let $\mathcal{S}$ denote a differentiable simulation function. We approximate $\Psi(T)$ via simulation:

$$\Psi(T) = \mathcal{S}(\mathbf{X}(\theta), \Psi(0), \mu, T). \quad (3.5)$$

Here T is the simulation end time, μ captures material parameters such as density and friction coefficient, as well as simulator parameters such as time step and damping. We adopt the semi-implicit Euler time-integrator in Warp (Macklin, 2022) for simulation.

Assuming the initial translation, rotation, and velocity are all zero, the difference between $\Psi(T)$ and $\Psi(0)$ arises only from discrepancies in spatial location, as the velocity at the final steady state is also zero. Therefore, we propose a standability loss to penalize rotational deviation due to instability:

$$\mathcal{L}_{\text{stand}} = \|\mathbf{R}(T) - \mathbf{R}(0)\|_2^2 = \|\mathbf{R}(T) - \mathbf{I}\|_2^2, \quad (3.6)$$

where $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ represents the rotation matrix. We disregard the translation $\mathbf{T}$, as real-world instability mostly leads to rotational deviation from the initial state, such as falling to one side, while most translations, like falling due to gravity, are irrelevant to standability.

With a differentiable simulator, the standability loss can be backpropagated to mesh vertex coordinates and then to the implicit parameter θ as $\frac{d\mathcal{L}_{\text{stand}}}{d\theta} = \frac{d\mathcal{L}_{\text{stand}}}{d\mathbf{X}} \frac{d\mathbf{X}}{d\theta}$. In theory, any differentiable simulator is compatible with our framework.

Stable Equilibrium: Although the standability loss directly penalizes the non-standability of a 3D object, it can be slow to compute, especially when T is large and many time steps are required, creating a huge computational graph. Consequently, both the simulation itself and the backpropagation of gradients through the simulation trajectory are time-consuming. Additionally, standability does not necessarily imply stable equilibrium, which

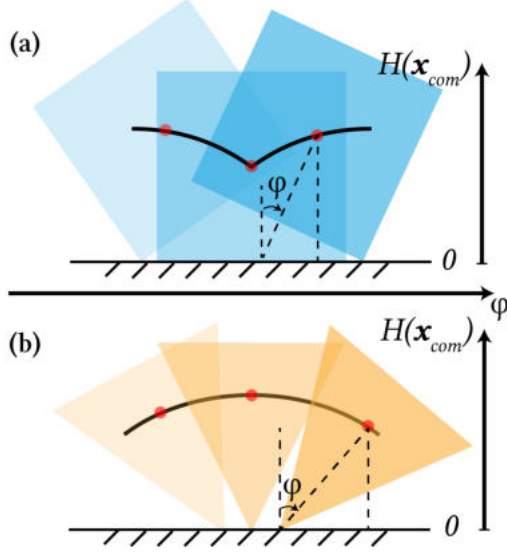


Figure 3.2: 2D illustration of stable equilibrium and unstable equilibrium. (a) A square is stable as a small perturbation of ϕ increases in $H(\mathbf{x}_{com})$; (b) An upside-down triangle is unstable as tilting decreases $H(\mathbf{x}_{com})$.

is crucial for real-world 3D objects such as action figures and toys. Without this property, an object remains unstable even if standability is achieved, known as unstable equilibrium. Unstable equilibrium means that when a disturbance force is applied, the object moves away from its original position instead of recovering. [Figure 3.2](#) visualizes the difference between stable and unstable equilibrium. In the absence of perturbation, geometries like the upside-down triangle may remain standable in a simulator but are clearly unstable in the real world. Thus, we augment standability with stable equilibrium [\(3.4\)](#). One straightforward way to incorporate this property is to introduce initial perturbation ϵ_0 into the simulator. However, this would require many more simulations with various perturbations and subsequent loss backpropagation, which is extremely time-consuming.

Inspired by the concept of a potential well ([Griffiths and Schroeter, 2018](#)), we augment our optimization objective with a robust and efficient stable equilibrium loss $\mathcal{L}_{stable}$. Specifically, for an object to be robustly standable, it needs to reside at a local minimum of potential energy—specifically, gravitational potential energy—so that if perturbed, gravity will act as a restoring force that returns the object to its original state. For a rigid body, gravitational potential energy is determined by the height of its center of mass.

Thus, for any object in a stable equilibrium state, the center of mass would rise if it is slightly perturbed. This leads to our formulation of the stable equilibrium loss $\mathcal{L}_{\text{stable}}$. Let $\mathbf{x}_{\text{com}}$ denote the position of the center of mass of the underlying geometry and $H(\mathbf{x})$ denote the distance of the point $\mathbf{x}$ to the ground, assuming the object’s pivot point is at $z = 0$. The stable equilibrium loss is defined as:

$$\mathcal{L}_{\text{stable}} := \mathbb{E}_{\mathbf{v} \in \mathbb{R}^2, \|\mathbf{v}\|=1} [\max\{H(\mathbf{x}_{\text{com}}(\mathbf{X})) - H(\mathbf{x}_{\text{com}}(\mathbf{P}_{\mathbf{v}}^{\phi}\mathbf{X})), 0\}], \quad (3.7)$$

where $\mathbf{P}_{\mathbf{v}}^{\phi}$ represents the rotation of ϕ radian about axis $[\mathbf{v}^T, 0]^T$. Mathematically, a local minimum of gravitational potential energy is reached if $\exists \phi_0$ such that $\forall \phi \in (0, \phi_0)$, $\mathcal{L}_{\text{stable}} = 0$. In practice, we fix the perturbation scale ϕ and uniformly sample 20 perturbation directions $\mathbf{v}$ in xy -plane.

3.4.2 Additional Regularization

While standability loss $\mathcal{L}_{\text{stand}}$ and stable equilibrium loss $\mathcal{L}_{\text{stable}}$ provide a well-defined objective for robust standing, they may lead to distorted optimized meshes due to the high-dimensional searching space of implicit parameter θ without constraint. To constrain the optimization space and obtain smooth meshes, we add a normal consistency term that favors smooth solutions:

$$\mathcal{L}_{\text{normal}} = \frac{1}{|\mathcal{T}|} \sum_{(i,j) \in \mathcal{T}} (1 - \mathbf{n}_i \cdot \mathbf{n}_j), \quad (3.8)$$

where $\mathcal{T}$ is the set of the triangle pairs sharing a common pair with $\mathbf{n}_i, \mathbf{n}_j$ being their normals respectively. This term maximizes the cosine similarity between neighboring surface triangle normals, leading to smoother meshes. Considering the bottom surfaces of most robust standing objects are flat, we apply the Laplacian loss to a subset $\mathcal{B}$ of vertices with a height lower than a threshold h_b :

$$\mathcal{L}_{\text{b-lap}} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \|\delta_i\|_2, \quad (3.9)$$

where $\delta_i = (\mathbf{L}\mathbf{V})_i \in \mathbb{R}^3$ calculates the differential coordinates of vertex i with $\mathbf{L}$ being the Laplacian matrix of the mesh graph and $\mathbf{V}$ representing mesh vertices. Intuitively, this loss term attempts to minimize the distance between vertex i and the average position of adjacent vertices.

3.4.3 Method Overview

With the physically-inspired loss terms derived above, we now describe how to incorporate them into the text-to-3D framework. SDS-based methods and their variants start optimization with a random initialization of the implicit parameters, which initially have no knowledge of the model’s geometry. Adding physical constraints at this early stage would be ineffective. Therefore, we propose a two-stage training strategy: the coarse stage and the refine stage. In the coarse stage, we generate a rough shape of the model using a text prompt. We can adopt any SDS-based generation framework as our baseline model, offering various choices of implicit representation and differentiable renderers. In the refine stage, we optimize the geometry with our physical constraints included. For this, we use a tetrahedral SDF representation (Shen et al., 2021) and employ Deep Marching Tetrahedra (DmTet) to differentially convert the coarse geometry from implicit density or SDF as necessary. We utilize Nvdiffrast (Laine et al., 2020) as the differentiable renderer and Stable Diffusion v2.1 (Rombach et al., 2022) for guidance. We propose the following loss function for the joint optimization of texture, geometry, and stability:

$$\mathcal{L} = \lambda_{\text{SDS}}\mathcal{L}_{\text{SDS}} + \lambda_{\text{stand}}\mathcal{L}_{\text{stand}} + \lambda_{\text{stable}}\mathcal{L}_{\text{stable}} + \lambda_{\text{normal}}\mathcal{L}_{\text{normal}} + \lambda_{\text{b-lap}}\mathcal{L}_{\text{b-lap}} \quad (3.10)$$

In practice, we observe that adding standability loss once every 10 iterations is sufficient to ensure a significant reduction in loss without notably increasing computational overhead.

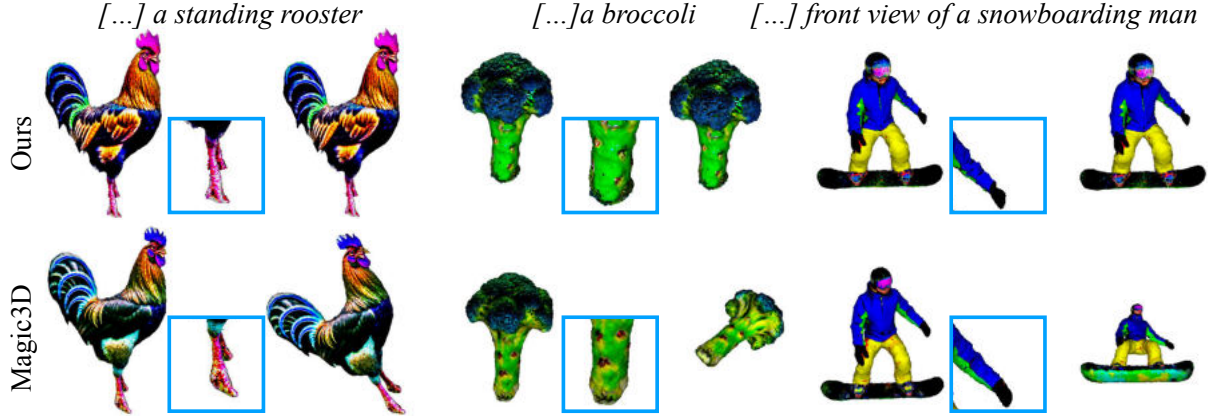


Figure 3.3: Comparison with Magic3D (Lin et al., 2023a) includes zoom-in views that highlight the detailed changes in geometry. Our method enhances Magic3D with physics priors to generate self-supporting meshes.

3.5 Experiments

In this section, we devise comprehensive experiments (both virtual and real-world) to demonstrate the efficacy of our method. We use a series of text prompts to generate 3D models that we expect to be self-supporting, and compare the generated results with baseline models. Our models are verified by simulation for stability and then fabricated using a 3D printer for real-world testing. [Appendix B](#) provides more details about implementation, training, and experiment setup.

3.5.1 Simulation Verification

Qualitative Comparison: Using the same text prompts, we compare our generated models with previous methods. The quality of the results is assessed by their stability, which is verified by a forward simulation: we simulate the generated models in an upright initial position close to the ground for a sufficiently long time and record whether they fall. Using Magic3D (Lin et al., 2023a) as the baseline model, we visualize the initial state and a later state in [Figure 3.3](#). Our generated meshes remain stable throughout the simulation, while the baseline models fail under the same conditions due to a lack of consideration for physics.



Figure 3.4: Atlas3D is also compatible with MVDream (Shi et al., 2023b), enhancing it with stable standability.

We highlight the main changes in mesh topology that enable the models to stand (see Figure 3.3). These changes include modifications to both the overall shape and specific local geometries. More specifically, our physical adjustments alter the contact surface to gain more support from the ground and shift the center of mass to be slightly lower and more centered above the contact surface. These macroscopic and microscopic optimizations jointly increase the support to the models, thus ensuring their stability. Note that our method slightly modifies the texture of the generated results as we are jointly optimizing our physical adjustments with the SDS loss.

Since we do not assume a specific baseline model in the first stage, we can vary the model used in the coarse stage to generate versatile, physically-aware 3D meshes based on different existing SDS generation models. We use MVDream (Shi et al., 2023b) as another baseline model and compare the results side-by-side with ours in Figure 3.4. Our method improves the geometry of the mesh and ensures stability in simulation. The quality of the texture and the main part of the shape is determined by the underlying model used in the coarse stage, while we focus on improving the physical stability in the refinement stage. More qualitative comparisons with previous methods are provided in Appendix B.

Ablation Study: We perform ablation studies to demonstrate the necessity of our proposed losses. It can be observed in Figure 3.5(b), that without standability loss L_{stand} , the model fails to stand. While the model can still stand without stable equilibrium

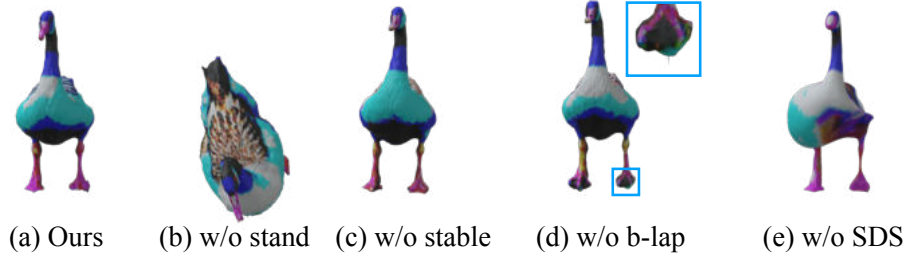


Figure 3.5: Ablation study of each loss term.

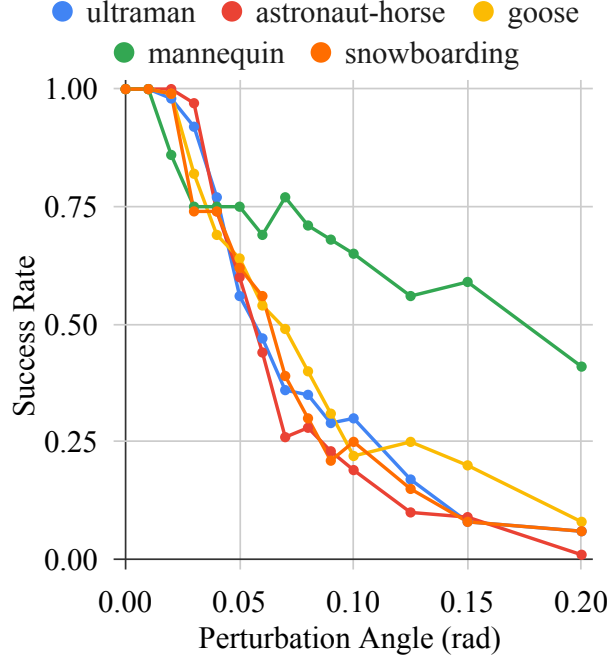


Figure 3.6: Success rate of models standing under perturbation.

loss L_{stable} , as demonstrated in Figure 3.5(c), it is less stable under perturbation (see the next section for the details). The geometry regularization loss term $L_{\text{b-lap}}$ helps smooth the geometry and avoid spiky artifacts on the surface as shown in Figure 3.5(d). Additionally, we show that applying mesh regularization as a post-processing step, rather than integrating it with SDS loss in a joint optimization, can degrade text alignment as this neglects the semantics during the deformation process (see Figure 3.5(e)).

Stability Under Perturbation: In the real world, placing an object on the ground always involves some noise, as both human and robot manipulation have imprecision in angles and directions. Hence, the standability of an object under small perturbations is

Perturbation Angle $\theta_{\max}$	0	0.01	0.02	0.04	0.08
w/ stability loss	1	1	0.99	0.69	0.4
w/o stability loss	1	0.97	0.71	0.62	0.23

Table 3.1: Comparison of success rate under perturbation (goose).

crucial for improving the success rate of such tasks.

To mimic this uncertainty in our framework, we evaluate the stability of our generated models under a small initial rotation. With a given precision $\phi_{\max}$, we rotate the generated mesh at random angles $\phi_y, \phi_x \in (-\phi_{\max}, \phi_{\max})$ in both the y and x axes, respectively (with the z axis being the up direction). The mesh is then placed close to the ground and tested in a simulation to see if it can still stand. We choose 13 different values of the maximum perturbation angle $\phi_{\max}$ and perform 100 random tests with each angle on 6 of our generated models. We report the success rate in [Figure 3.6](#). We define a successful test as: after a sufficiently long time period, the maximum height of the model stays within 3% of the initial maximum height.

Due to the presence of physical constraints, our generated models can withstand small initial perturbations, while the baseline models fail to stand when placed straight up (without rotation), let alone with perturbations. Furthermore, introducing the stable equilibrium loss consistently increases the success rate of standing under different scales of perturbations, as shown in [Table 3.1](#).

Standability on Different Platforms: Our pipeline can be generalized to learn standability on various platforms, not just flat ground, by incorporating them as boundary conditions in the simulator. To demonstrate this, we use a 10-degree inclined plane and a sphere, then train our 3D model separately to stand still on each. Modeling frictional contact is crucial for achieving stability in such scenarios. As shown in [Figure 3.7](#), our optimized mesh stands stably on both the incline and the sphere with the help of static friction, whereas the baseline model fails as expected.



Figure 3.7: Standability evaluation on uneven surfaces.

Simulator Cross-Validation: While we base our method on a differentiable rigid-body simulator with a semi-implicit Euler time-integrator, our pipeline is compatible with any other physics-based simulator as the backbone, with differentiability required for the training stage. To verify the reliability of models generated with our simulator, we include an external simulator in the testing stage to verify the correctness of the simulated dynamics. We choose the Incremental Potential Contact (IPC) method (Lan et al., 2022; Li et al., 2020), which has been proven accurate for frictional contact. We validate the correctness of every single generated model and visualize the simulation results in Figure 3.1.

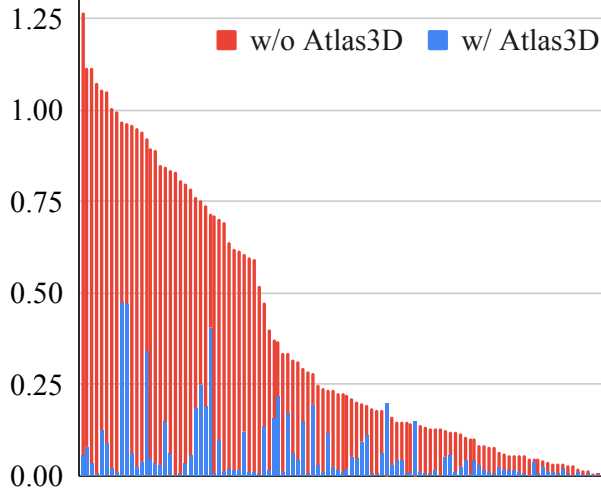


Figure 3.8: TRD results from 107 prompts using the Magic3D baseline and our method.

Quantitative Evaluation: We examine the versatility of our method. We randomly select 150 prompts from (Poole et al., 2022) and manually exclude 43 prompts deemed unfeasible (for instance, it does not make sense to require “a swan and its cygnets swimming in a pond” to be standable), leaving a total of 107 prompts. We use the two-stage Magic3D (Lin et al., 2023a) as the baseline model and compare our optimized mesh with the results from the refine stage of the baseline under the same settings (e.g., iterations, loss weights).

To evaluate the standability of the baseline method and our method, we run the rigid body simulation in Warp with simulation end time $T = 2.0$ at which almost all objects have reached the steady state. We propose Time-Averaged Rotation Deviation Loss (TRD) defined as

$$\text{TRD} = \frac{1}{T} \int_0^T \|\mathbf{R}(t)\hat{\mathbf{z}} - \mathbf{R}(0)\hat{\mathbf{z}}\|_2 dt \quad (3.11)$$

to assess the standability, as a representation of the average tilting of the upward direction $\hat{\mathbf{z}}$ (of the object) over time. We approximate the integral ((3.11)) with discrete quadrature $\Delta t = 0.02$. Results are plotted in Figure 3.8.³ The mean TRD score is reduced by more than six times compared with the baseline method. As shown in Table 3.2, we calculate the average CLIP score (Radford et al., 2021) of 107 generated shapes for both our

³Results are ranked by TRD scores of the baseline approach.

Metrics	Ours	Magic3D
TRD ↓	0.060	0.389
CLIP ↑	25.356	25.781
Elo (GPT-4o) ↑	970.774	1029.226

Table 3.2: Quantitative Evaluation

proposed method and baseline. Furthermore, Elo (GPT-4o) (Wu et al., 2024b) scores are presented. Both metrics illustrate that our method not only implements physics-based stability but also maintains the fidelity of the generated 3D shapes in terms of content alignment and overall shape quality. More details are provided in [Appendix B](#).

3.5.2 Real-World Validation:

One major advantage of incorporating physics-based simulation into the optimization pipeline is that it bridges the gap between the generated model and the real world. Our method ensures the direct usability of the model for fabrication, with success primarily dependent on the accuracy of both the simulator and the manufacturing machine.

3D Printing and User Studies: We test the readiness of our generated meshes for real-world application by producing eight figures using a 3D-printing device (Zortrax M200 with Z-ABS filament material). For reference, we also print the corresponding baseline meshes generated without physical constraints. Our physically constrained figures can steadily support themselves when gently placed on an even surface, while the baseline figures either fail to stand at all, or require extensive adjustments and fall easily with little perturbation ([Figure 1.2](#)).

We conduct user studies with these printed figures to assess their stability under different types of human manipulation. Ten users were asked to place the figures upright on a table, with five trials for each figure, resulting in a total of 800 trials. Fabricated figures generated from the baseline has a success rate of 7%, while figures generated from our method has an overall success rate of 92.25%. Itemized results are provided in [Appendix B](#). Our method significantly increases the physical stability of the models

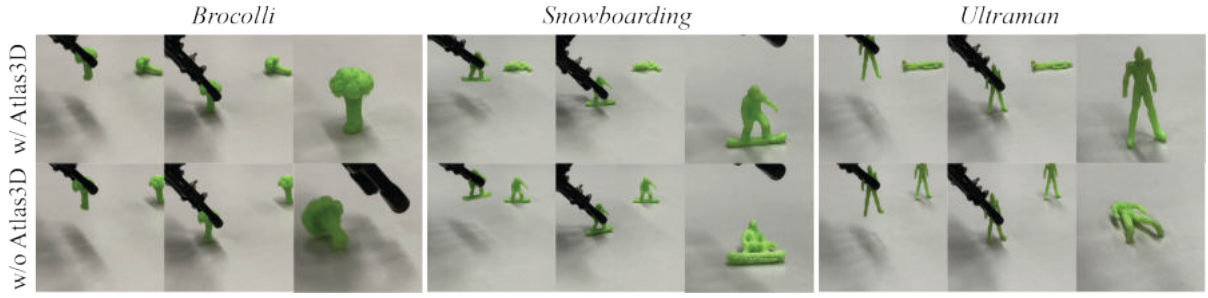


Figure 3.9: Standability test using a robotic arm. More results are shown in [Appendix B](#) under varying human efforts.

Validation with a Robot Arm: To showcase the compatibility of our framework with robotic applications, we test our fabricated figures with a teleoperated LewanSoul LeArm robot arm outfitted with a two-finger parallel gripper (see [Figure 3.9](#)). The gripper is set to initially grasp a figure above the ground. It slowly moves downward, and is then gently released to place the figure on the ground. Four trials were performed for each figure, yielding 64 trials in total. For the baseline method, 6.25% trials resulted in successful standing figures, while ours has a success rate of 90.6%. Experimental data are provided in [Appendix B](#).

3.6 Conclusions

We have presented Atlas3D, a physically constrained SDS-based framework that generates self-supporting 3D models from text prompts. Our framework can learn standability through a differentiable physics-based simulator and other physics-inspired loss functions. The generated 3D models can be directly imported into a physics simulator and are ready to be manufactured and deployed in the real world. Our method has wide potential for generation tasks, as it can be easily integrated into many existing pipelines and improve the physical plausibility of their generated results.

Limitations and Future Work: Our physical adjustments are optimized over all mesh vertices, resulting in a large degree of freedom in optimization. This may lead to undesired distorted meshes (Michel et al., 2022). Future work may consider adding a latent embedding or skeleton rigging to limit the variety of mesh deformation. Our framework focuses on SDS-based methods as a backbone. It would be interesting to further generalize our physical constraints to other non-SDS or non-diffusion based methods (Jun and Nichol, 2023; Michel et al., 2022; Li et al., 2023a). Finally, we have only considered text-to-3D tasks in this work. An exciting extension would be to generalize our work to image-to-3D tasks (Qian et al., 2024; Liu et al., 2023b).

CHAPTER 4

Physics-Integrated 4D Generative Dynamics for Multi-View Reconstruction

4.1 PhysGaussian: Physics-Integrated 3D Gaussians for Generative Dynamics

We introduce PhysGaussian, a new method that seamlessly integrates physically grounded Newtonian dynamics within 3D Gaussians to achieve high-quality novel motion synthesis (Figure 4.1). Employing a custom Material Point Method (MPM), our approach enriches 3D Gaussian kernels with physically meaningful kinematic deformation and mechanical stress attributes, all evolved in line with continuum mechanics principles. A defining characteristic of our method is the seamless integration between physical simulation and visual rendering: both components utilize the same 3D Gaussian kernels as their discrete representations. This negates the necessity for triangle/tetrahedron meshing, marching cubes, “cage meshes,” or any other geometry embedding, highlighting the principle of “what you see is what you simulate (WS²).” Our method demonstrates exceptional versatility across a wide variety of materials—including elastic entities, plastic metals, non-Newtonian fluids, and granular materials—showcasing its strong capabilities in creating diverse visual content with novel viewpoints and movements.

4.1.1 Introduction

Recent strides in Neural Radiance Fields (NeRFs) have showcased significant advancements in 3D graphics and vision (Mildenhall et al., 2021). Such gains have been further augmented



Figure 4.1: PhysGaussian is a unified simulation-rendering pipeline based on 3D Gaussians and continuum mechanics.

by the cutting-edge 3D Gaussian Splatting (GS) framework (Kerbl et al., 2023). Despite many achievements, a noticeable gap remains in the application towards generating novel dynamics. While there exist endeavors that generate new poses for NeRFs, they typically cater to quasi-static geometry shape editing tasks and often require meshing or embedding visual geometry in coarse proxy meshes such as tetrahedra (Yuan et al., 2022; Xu and Harada, 2022; Peng et al., 2022b; Jambon et al., 2023).

Meanwhile, the traditional physics-based visual content generation pipeline has been a tedious multi-stage process: constructing the geometry, making it simulation-ready (often through techniques like tetrahedralization), simulating it with physics, and finally rendering the scene. This sequence, while effective, introduces intermediary stages that can lead to discrepancies between simulation and final visualization. Even within the NeRF paradigm, a similar trend is observed, as the rendering geometry is embedded into a simulation geometry. This division, in essence, contrasts with the natural world, where the physical behavior and visual appearance of materials are intrinsically intertwined. Our overarching philosophy seeks to align these two facets by advocating for a unified representation of a material substance, employed for both simulation and rendering. In essence, our approach champions the principle of *“what you see is what you simulate”* (WS^2) (Müller et al., 2016), aiming for a more coherent integration of simulation, capturing, and rendering.

Building towards this goal, we introduce PhysGaussian: physics-integrated 3D Gaussians for generative dynamics. This novel approach empowers 3D Gaussians to encapsulate

physically sound Newtonian dynamics, including realistic behaviors and inertia effects inherent in solid materials. More specifically, we impart physics to 3D Gaussian kernels, endowing them with kinematic attributes such as velocity and strain, along with mechanical properties like elastic energy, stress, and plasticity. Notably, through continuum mechanics principles and a custom Material Point Method (MPM), PhysGaussian ensures that both physical simulation and visual rendering are driven by 3D Gaussians. This eradicates the necessity for any embedding mechanisms, thus eliminating any disparity or resolution mismatch between the simulated and the rendered.

We present PhysGaussian’s versatile adeptness in synthesizing generative dynamics across various materials, such as elastic objects, metals, non-Newtonian viscoplastic substances (e.g. foam or gel), and granular mediums (e.g. sand or soil). To summarize, our contributions include

- **Continuum Mechanics for 3D Gaussian Kinematics:** We introduce a continuum mechanics-based strategy tailored for evolving 3D Gaussian kernels and their associated spherical harmonics in physical Partial Differential Equation (PDE)-driven displacement fields.
- **Unified Simulation-Rendering Pipeline:** We present an efficient simulation and rendering pipeline with a unified 3D Gaussian representation. Eliminating the extra effort for explicit object meshing, the motion generation process is significantly simplified.
- **Versatile Benchmarking and Experiments:** We conduct a comprehensive suite of benchmarks and experiments across various materials. Enhanced by real-time GS rendering and efficient MPM simulations, we achieve *real-time* performance for scenes with simple dynamics.

4.1.2 Related Work

Radiance Fields Rendering for View Synthesis: Radiance field methods have gained considerable interest in recent years due to their extraordinary ability to generate novel-view scenes and their great potential in 3D reconstruction. The adoption of deep learning techniques has led to the prominence of neural rendering and point-based rendering methods, both of which have inspired a multitude of subsequent works. On the one hand, the NeRF framework employs a fully-connected network to model one scene (Mildenhall et al., 2021). The network takes spatial position and viewing direction as inputs and produces the volume density and radiance color. These outputs are subsequently utilized in image generation through volume rendering techniques. Building upon the achievements of NeRF, further studies have focused on enhancing reconstruction quality and improving training speeds (Fridovich-Keil et al., 2022; Müller et al., 2022; Sun et al., 2022; Barron et al., 2022; Xu et al., 2022a; Lin et al., 2022). On the other hand, researchers have also investigated differentiable point-based methods for real-time rendering of unbounded scenes. Among the current investigations, the state-of-the-art results are achieved by the recently published 3D Gaussian Splatting framework (Kerbl et al., 2023). Contrary to prior implicit neural representations, GS employs an explicit and unstructured representation of one scene, offering the advantage of straightforward extension to post-manipulation. Moreover, its fast visibility-aware rendering algorithm also enables real-world dynamics generations.

Dynamic Neural Radiance Field: An inherent evolution of the NeRF framework entails the integration of a temporal dimension to facilitate the representation of dynamic scenes. For example, both Pumarola et al. (2021) and Park et al. (2021) decompose time-dependent neural fields into an inverse displacement field and canonical time-invariant neural fields. In this context, the trajectory of query rays is altered by the inverse displacement field and then positioned within the canonical space. Subsequent studies have adhered to the aforementioned design when exploring applications related to NeRF deformations, such as static scene editing and dynamic scene reconstruction (Li et al.,

2023l; Peng et al., 2022b; Yuan et al., 2022; Chen et al., 2022; Qiao et al., 2022, 2023; Liu et al., 2023c). Additionally, Yuan et al. (2022); Qiao et al. (2022); Liu et al. (2023c) have contributed to the incorporation of physics-based deformations into the NeRF framework. However, the effectiveness of these methodologies relies on the usage of exported meshes derived from NeRFs. To circumvent this restriction, explicit geometric representations have been explored for forward displacement modeling (Xu et al., 2022a; Kerbl et al., 2023). In particular, Chen et al. (2023b); Luiten et al. (2023); Yang et al. (2023c); Wu et al. (2023a); Yang et al. (2023d) directly manipulate NeRF fields. Li et al. (2023i) extends this approach by including physical simulators to achieve more dynamic behaviors. In this study, we leverage the explicit 3D Gaussian Splatting ellipsoids as a unified representation for both physics and graphics. In contrast to previous dynamic GS frameworks, which either maintain the shapes of Gaussian kernels or learn to modify them, our approach uniquely leverages the first-order information from the displacement map (deformation gradient) to assist dynamic simulations. In this way, we are able to deform the Gaussian kernels and seamlessly integrate the simulation within the GS framework.

Material Point Method: The Material Point Method (MPM) is a widely used simulation framework for a broad range of multi-physics phenomena (Hu et al., 2018a). The inherent capability of the MPM system allows for topology changes and frictional interactions, making it suitable for simulating various materials, including but not limited to elastic objects, fluids, sand, and snow (Stomakhin et al., 2013; Jiang et al., 2015; Klár et al., 2016). MPM can also be expanded to simulate objects that possess codimensional characteristics (Jiang et al., 2017). In addition, the efficacy of utilizing GPU(s) to accelerate MPM implementations has also been demonstrated in (Gao et al., 2018b; Hu et al., 2019; Wang et al., 2020b; Qiu et al., 2023). Owing to its well-documented advantages, we employ the MPM to support the latent physical dynamics. This choice allows us to efficiently import dynamics into various scenarios with a shared particle representation alongside the Gaussian Splatting framework.

4.1.3 Method Overview

We propose PhysGaussian (Figure 1.3), a unified simulation-rendering framework for generative dynamics based on continuum mechanics and 3D GS. Adopted from Kerbl et al. (2023), we first reconstruct a GS representation of a static scene, with an optional anisotropic loss term to regularize over-skinny kernels. These Gaussians are viewed as the discretization of the scene to be simulated. Under our novel kinematics, we directly splat the deformed Gaussians for photo-realistic renderings. For better physics compliance, we also optionally fill the internal regions of objects. We detail these in this section.

4.1.3.1 3D Gaussian Splatting

3D Gaussian Splatting method (Kerbl et al., 2023) reparameterizes NeRF (Mildenhall et al., 2021) using a set of unstructured 3D Gaussian kernels $\{\mathbf{x}_p, \sigma_p, \mathbf{A}_p, \mathcal{C}_p\}_{p \in \mathcal{P}}$, where $\mathbf{x}_p$, σ_p , $\mathbf{A}_p$, and $\mathcal{C}_p$ represent the centers, opacities, covariance matrices, and spherical harmonic coefficients of the Gaussians, respectively. To render a view, GS projects these 3D Gaussians onto the image plane as 2D Gaussians, differing from traditional NeRF techniques that emit rays from the camera. The final color of each pixel is computed as

$$\mathbf{C} = \sum_{k \in \mathcal{P}} \alpha_k \text{SH}(\mathbf{d}_k; \mathcal{C}_k) \prod_{j=1}^{k-1} (1 - \alpha_j). \quad (4.1)$$

Here α_k represents the z -depth ordered effective opacities, e.g., products of the 2D Gaussian weights and their overall opacities σ_k ; $\mathbf{d}_k$ stands for the view direction from the camera to $\mathbf{x}_k$. Per-view optimizations are performed using L_1 loss and SSIM loss. This explicit representation of the scene not only significantly accelerates training and rendering speeds, but also enables direct manipulation of the NeRF scene. The data-driven dynamics are supported by making $\mathbf{x}_p, A_p$ time-dependent (Wu et al., 2023a) and minimizing rendering losses over videos. In Section 4.1.3.1, we show that this time-dependent evolution can be given by the continuum deformation map.

4.1.3.2 Continuum Mechanics

Continuum mechanics describes motions by a time-dependent continuous deformation map $\mathbf{x} = \boldsymbol{\phi}(\mathbf{X}, t)$ between the undeformed material space Ω^0 and the deformed world space Ω^t at time t . The deformation gradient $F(\mathbf{X}, t) = \nabla_{\mathbf{X}} \boldsymbol{\phi}(\mathbf{X}, t)$ encodes local transformations including stretch, rotation, and shear (Bonet and Wood, 1997). The evolution of the deformation $\boldsymbol{\phi}$ is governed by the conservation of mass and momentum. Conservation of mass ensures that the mass within any infinitesimal region $B_\epsilon^0 \in \Omega^0$ remains constant over time:

$$\int_{B_\epsilon^t} \rho(\mathbf{x}, t) \equiv \int_{B_\epsilon^0} \rho(\boldsymbol{\phi}^{-1}(\mathbf{x}, t), 0), \quad (4.2)$$

where $B_\epsilon^t = \boldsymbol{\phi}(B_\epsilon^0, t)$ and $\rho(\mathbf{x}, t)$ is the density field characterizing material distribution. Denoting the velocity field with $\mathbf{v}(\mathbf{x}, t)$, the conservation of momentum is given by

$$\rho(\mathbf{x}, t) \dot{\mathbf{v}}(\mathbf{x}, t) = \nabla \cdot \boldsymbol{\sigma}(\mathbf{x}, t) + \mathbf{f}^{\text{ext}}, \quad (4.3)$$

where $\boldsymbol{\sigma} = \frac{1}{\det(\mathbf{F})} \frac{\partial \Psi}{\partial \mathbf{F}}(\mathbf{F}^E) \mathbf{F}^{E^T}$ is the Cauchy stress tensor associated with a hyperelastic energy density $\Psi(\mathbf{F})$, and $\mathbf{f}^{\text{ext}}$ is the external force per unit volume (Bonet and Wood, 1997; Jiang et al., 2016). Here the total deformation gradient can be decomposed into an elastic part and a plastic part $\mathbf{F} = \mathbf{F}^E \mathbf{F}^P$ to support permanent rest shape changes caused by plasticity. The evolution of $\mathbf{F}^E$ follows some specific plastic flow such that it is always constrained within a predefined elastic region (Bonet and Wood, 1997).

4.1.3.3 Material Point Method

Material Point Method (MPM) solves the above governing equations by combining the strengths of both Lagrangian particles and Eulerian grids (Stomakhin et al., 2013; Jiang et al., 2016). The continuum is discretized by a collection of particles, each representing a small material region. These particles track several time-varying Lagrangian quantities such as position $\mathbf{x}_p$, velocity $\mathbf{v}_p$, and deformation gradient $\mathbf{F}_p$. The mass conservation in Lagrangian particles ensures the constancy of total mass during movement. Conversely,

momentum conservation is more natural in Eulerian representation, which avoids mesh construction. We follow [Stomakhin et al. \(2013\)](#) to integrate these representations using C^1 continuous B-spline kernels for two-way transfer. From time step t^n to t^{n+1} , the momentum conservation, discretized by the forward Euler scheme, is represented as

$$\frac{m_i}{\Delta t}(\mathbf{v}_i^{n+1} - \mathbf{v}_i^n) = - \sum_p V_p^0 \frac{\partial \Psi}{\partial \mathbf{F}}(\mathbf{F}_p^{E,n}) \mathbf{F}_p^{E,n T} \nabla w_{ip}^n + \mathbf{f}_i^{ext}. \quad (4.4)$$

Here i and p represent the fields on the Eulerian grid and the Lagrangian particles respectively; w_{ip}^n is the B-spline kernel defined on i -th grid evaluated at $\mathbf{x}_p^n$; V_p^0 is the initial representing volume, and Δt is the time step size. The updated grid velocity field $\mathbf{v}_i^{n+1}$ is transferred back onto particle to $\mathbf{v}_p^{n+1}$, updating the particles' positions to $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}$. We track $\mathbf{F}^E$ rather than both $\mathbf{F}$ and $\mathbf{F}^P$ ([Simo and Hughes, 2006](#)), which is updated by $\mathbf{F}_p^{E,n+1} = (\mathbf{I} + \Delta t \nabla \mathbf{v}_p) \mathbf{F}_p^{E,n} = (\mathbf{I} + \Delta t \sum_i \mathbf{v}_i^{n+1} \nabla w_{ip}^{n T}) \mathbf{F}_p^{E,n}$ and regularized by an additional return mapping to support plasticity evolution: $\mathbf{F}_p^{E,n+1} \leftarrow \mathcal{Z}(\mathbf{F}_p^{E,n+1})$. Different plasticity models define different return mappings. We refer to the supplemental document for details of the simulation algorithm and different return mappings.

4.1.3.4 Physics-Integrated 3D Gaussians

We treat Gaussian kernels as discrete particle clouds for spatially discretizing the simulated continuum. As the continuum deforms, we let the Gaussian kernels deform as well. However, for a Gaussian kernel defined at $\mathbf{X}_p$ in the material space, $G_p(\mathbf{X}) = e^{-\frac{1}{2}(\mathbf{X} - \mathbf{X}_p)^T \mathbf{A}_p^{-1}(\mathbf{X} - \mathbf{X}_p)}$, the deformed kernel under the deformation map $\phi(\mathbf{X}, t)$,

$$G_p(\mathbf{x}, t) = e^{-\frac{1}{2}(\phi^{-1}(\mathbf{x}, t) - \mathbf{X}_p)^T \mathbf{A}_p^{-1}(\phi^{-1}(\mathbf{x}, t) - \mathbf{X}_p)} \quad (4.5)$$

is not necessarily Gaussian in the world space, which violates the requirements of the splatting process. Fortunately, if we assume particles undergo local affine transformations characterized by the first-order approximation

$$\tilde{\phi}_p(\mathbf{X}, t) = \mathbf{x}_p + \mathbf{F}_p(\mathbf{X} - \mathbf{X}_p), \quad (4.6)$$

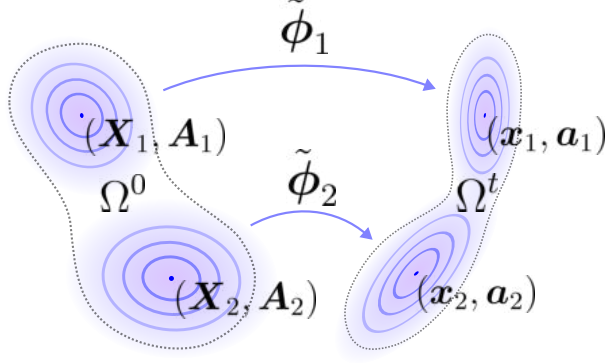


Figure 4.2: Local affine approximation of the deformation map transforms a Gaussian kernel in material space into a Gaussian in world space.

the deformed kernel (Figure 4.2) becomes Gaussian as desired:

$$G_p(\mathbf{x}, t) = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}_p)^T(\mathbf{F}_p\mathbf{A}_p\mathbf{F}_p^T)^{-1}(\mathbf{x}-\mathbf{x}_p)}. \quad (4.7)$$

This transformation naturally provides a time-dependent version of $\mathbf{x}_p$ and $\mathbf{A}_p$ for the 3D GS framework:

$$\begin{aligned} \mathbf{x}_p(t) &= \phi(\mathbf{X}_p, t), \\ \mathbf{a}_p(t) &= \mathbf{F}_p(t)\mathbf{A}_p\mathbf{F}_p(t)^T. \end{aligned} \quad (4.8)$$

In summary, given the 3D GS of a static scene $\{\mathbf{X}_p, \mathbf{A}_p, \sigma_p, \mathcal{C}_p\}$, we use simulation to dynamize the scene by evolving these Gaussians to produce dynamic Gaussians $\{\mathbf{x}_p(t), \mathbf{a}_p(t), \sigma_p, \mathcal{C}_p\}$. Here we assume that the opacity and the coefficients of spherical harmonics are invariant over time, but the harmonics will be rotated as discussed in the next section. We also initialize other physical quantities in (4.4): the representing volume of each particle V_p^0 is initialized as background cell volume divided by the number of contained particles; the mass m_p is then inferred from user-specified density ρ_p as $m_p = \rho_p V_p^0$. To render these deformed Gaussian kernels, we use the splatting from the original GS framework (Kerbl et al., 2023). It should be highlighted that the integration of physics into 3D Gaussians is *seamless*: on the one hand, the Gaussians themselves are viewed as the discretization of the continuum, which can be simulated directly; on the

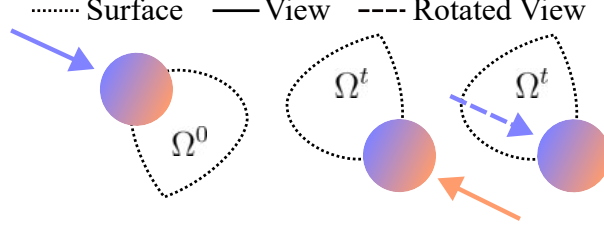


Figure 4.3: Rotating the spherical harmonic bases along with the ellipsoid ensures view-consistent appearance under object rotation.

other hand, the deformed Gaussians can be directly rendered by the splatting procedure, avoiding the need for commercial rendering software in traditional animation pipelines. Most importantly, we can directly simulate scenes reconstructed from real data, achieving WS^2 .

4.1.3.5 Evolving Orientations of Spherical Harmonics

Rendering the world-space 3D Gaussians can already obtain high-quality results. However, when the object undergoes rotations, the spherical harmonic bases are still represented in the material space, resulting in varying appearances even if the view direction is fixed relatively to the object. The solution is simple: when an ellipsoid is rotated over time, we rotate the orientations of its spherical harmonics as well. However, the bases are hard-coded inside the GS framework. We equivalently achieve this evolution by applying inverse rotation on view directions (Figure 4.3). We remark that the rotation of view directions is not considered in Wu et al. (2023a). Chen et al. (2023b) tackles this issue in the Point-NeRF framework, but requires tracking of surface orientation. In our framework, the local rotation is readily obtained in the deformation gradient $\mathbf{F}_p$. Denote $f^0(\mathbf{d})$ as a spherical harmonic basis in material space, with $\mathbf{d}$ being a point on the unit sphere (indicating view direction). The polar decomposition, $\mathbf{F}_p = \mathbf{R}_p \mathbf{S}_p$, leads us to the rotated harmonic basis:

$$f^t(\mathbf{d}) = f^0(\mathbf{R}^T \mathbf{d}). \quad (4.9)$$

4.1.3.6 Incremental Evolution of Gaussians

We also propose an alternative way for Gaussian kinematics that better fits the updated Lagrangian framework, which avoids the dependency on the total deformation gradient $\mathbf{F}$. This approach also paves the way for physical material models that do not rely on employing $\mathbf{F}$ as the strain measure. Following conventions from computational fluid dynamics (McKiver and Dritschel, 2003; Chandrasekhar, 1967), the update rule for the world-space covariance matrix $\mathbf{a}$ can also be derived by discretizing the rate form of kinematics $\dot{\mathbf{a}} = (\nabla \mathbf{v})\mathbf{a} + \mathbf{a}(\nabla \mathbf{v})^T$:

$$\mathbf{a}_p^{n+1} = \mathbf{a}_p^n + \Delta t(\nabla \mathbf{v}_p \mathbf{a}_p^n + \mathbf{a}_p^n \nabla \mathbf{v}_p^T). \quad (4.10)$$

This formulation facilitates the incremental update of the Gaussian kernel shapes from time step t^n to t^{n+1} without the need to obtain $\mathbf{F}_p$. The rotation matrix $\mathbf{R}_p$ of each spherical harmonics basis can be incrementally updated in a similar manner. Starting from $\mathbf{R}_p^0 = \mathbf{I}$, we extract the rotation matrix $\mathbf{R}_p^{n+1}$ from $(\mathbf{I} + \Delta t \mathbf{v}_p)\mathbf{R}_p^n$ using the polar decomposition.

4.1.3.7 Internal Filling

The internal structure is occluded by the object’s surface, as the reconstructed Gaussians tend to distribute near the surface, resulting in inaccurate behaviors of volumetric objects. To fill particles into the void internal region, inspired by Tang et al. (2023b), we borrow the 3D opacity field from 3D Gaussians

$$d(\mathbf{x}) = \sum_p \sigma_p \exp \left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}_p)^T \mathbf{A}_p^{-1}(\mathbf{x} - \mathbf{x}_p) \right). \quad (4.11)$$

This continuous field is discretized onto a 3D grid. To achieve robust internal filling, we first define the concept of “intersection” within the opacity field, guided by a user-defined threshold σ_{th} . Specifically, we consider it an intersection when a ray passes

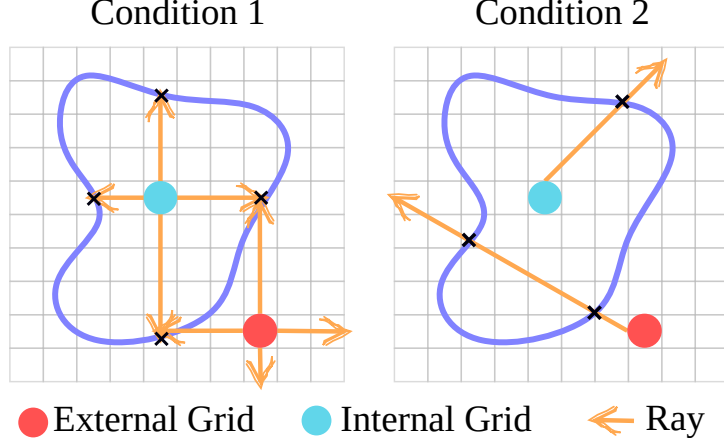


Figure 4.4: Internal filling procedure: candidate interior grid cells are identified by casting rays along 6 axes and checking opacity intersections.

from a lower opacity grid ($\sigma_i < \sigma_{th}$) to a higher opacity one ($\sigma_j > \sigma_{th}$). Based on this definition, we identify candidate grids by casting rays along 6 axes and checking intersections (Condition 1). Rays originating from internal cells will always intersect with the surface (Figure 4.4). To further refine our selection of candidate grids, we employ an additional ray to assess the intersection number (Condition 2), thus ensuring greater accuracy.

Visualization of these internal particles is also crucial as they may get exposed due to large deformation. Those filled particles inherit $\sigma_p, \mathcal{C}_p$ from their closet Gaussian kernels. Each particle’s covariance matrix is initialized as $\text{diag}(r_p^2, r_p^2, r_p^2)$, where r is the particle radius calculated from its volume: $r_p = (3V_p^0/4\pi)^{\frac{1}{3}}$. Alternatively, one may also consider employing generative models for internal filling, potentially leading to more realistic results.

4.1.3.8 Anisotropy Regularizer

The anisotropy of Gaussian kernels increases the efficiency of 3D representation while over-skinny kernels may point outward from the object surface under large deformations, leading to unexpected plush artifacts. We propose the following training loss during 3D

Gaussian reconstruction:

$$\mathcal{L}_{aniso} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \max\{\max(\mathbf{S}_p) / \min(\mathbf{S}_p), r\} - r, \quad (4.12)$$

where $\mathbf{S}_p$ are the scalings of 3D Gaussians (Kerbl et al., 2023). This loss essentially constrains that the ratio between the major axis length and minor axis length does not exceed r . If desired, this term can be added to the training loss.

4.1.4 Experiments

In this section, we show the versatility of our approach across a wide range of materials. We also evaluate the effectiveness of our method across a comprehensive suite of benchmarks.

4.1.4.1 Evaluation of Generative Dynamics

Datasets: We evaluate our method for generating diverse dynamics using several sources of input. In addition to the synthetic data (*sofa suite*) generated by BlenderNeRF (Raafat, 2023), we utilize *fox*, *plane*, and *ruins* from the datasets of Instant-NGP (Müller et al., 2022), Nerfstudio (Tancik et al., 2023) and the DroneDeploy NeRF (Pilkington, 2022), respectively. Furthermore, we collect two real-world datasets (referred to as *toast* and *jam*) with an iPhone. Each scene contains 150 photos. The initial point clouds and camera parameters are obtained using COLMAP (Schönberger et al., 2016; Schönberger and Frahm, 2016).

Simulation Setups: We build upon the MPM from Zong et al. (2023). To generate novel physics-based dynamics of a 3D Gaussian scene, we manually select a simulation region and normalize it to a cube with edge length 2. The internal particle filling can be performed before simulation. The cuboid simulation domain is discretized by a 3D dense grid. We selectively modify the velocities of specific particles to induce controlled movement. The remaining particles follow natural motion patterns governed by the

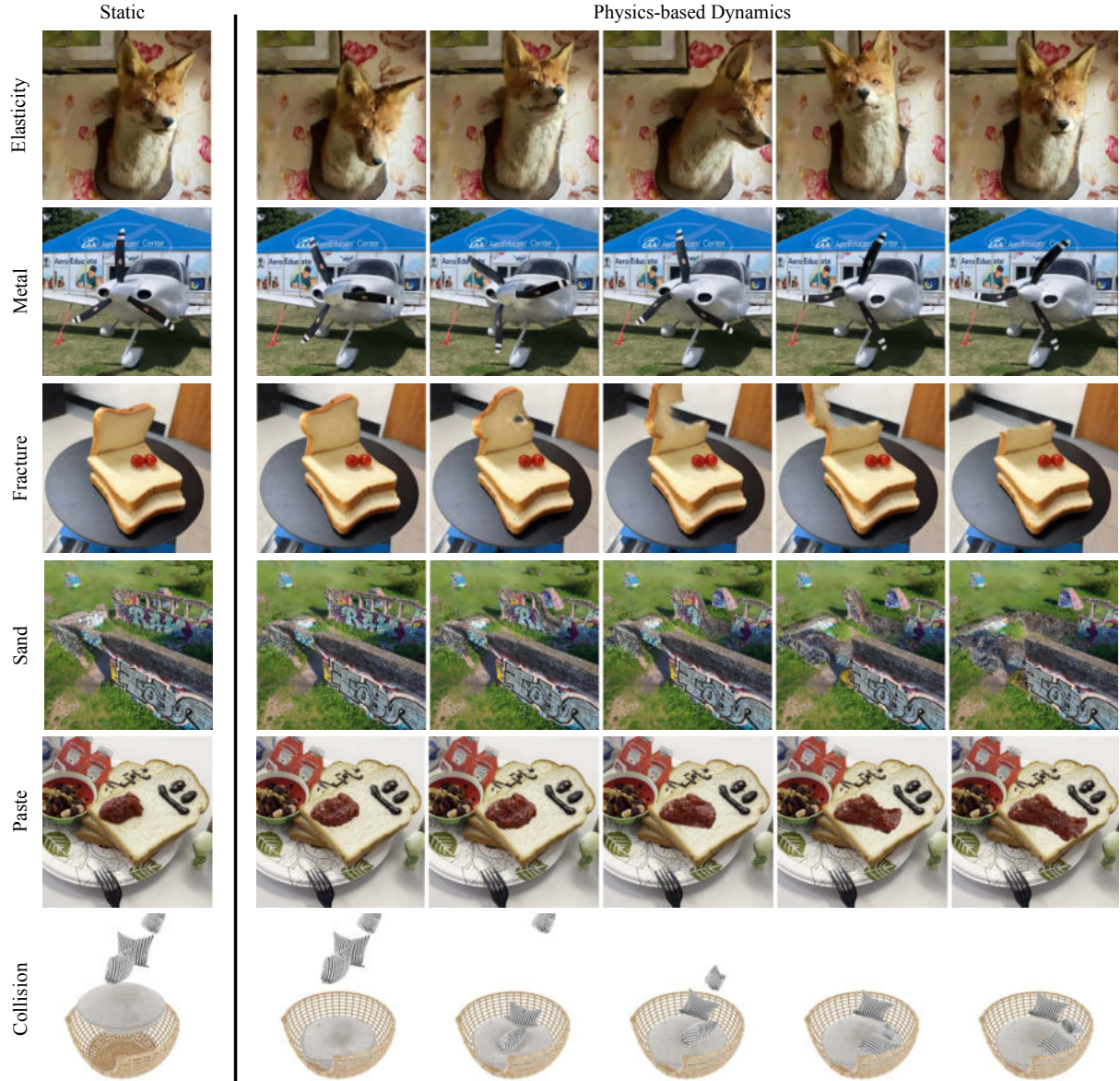


Figure 4.5: Material Versatility. We demonstrate exceptional versatility of our approach across a wide variety of examples: *fox* (elastic entity), *plane* (plastic metal), *toast* (fracture), *ruins* (granular material), *jam* (viscoplastic material), and *sofa suite* (collision).

established physical laws. All our experiments are performed on a 24-core 3.50GHz Intel i9-10920X machine with a Nvidia RTX 3090 GPU.

Results: We simulate a wide range of physics-based dynamics. For each type of dynamics, we visualize one example with its initial scene and deformation sequence, as shown in [Figure 4.5](#). Additional experiments are included in the supplemental document. The dynamics include: *Elasticity* refers to the property where the rest shape of the object remains invariant during deformation, representing the simplest form of daily-life dynamics. *Metal* can undergo permanent rest shape changes, which follows von-Mises plasticity model. *Fracture* is naturally supported by MPM simulation, where large deformations can cause particles to separate into multiple groups. *Sand* follows Drucker-Prager plasticity model ([Klár et al., 2016](#)), which can capture granular-level frictional effects among particles. *Paste* is modeled as viscoplastic non-Newtonian fluid, adhering to Herschel-Bulkley plasticity model ([Yue et al., 2015](#)). *Collision* is another key feature of MPM simulation, which is automatically handled by grid time integration. Explicit MPM can be highly optimized to run on GPUs. We highlight that some of the cases can achieve real-time based on the 1/24-s frame duration: *plane* (30 FPS), *toast* (25 FPS) and *jam* (36 FPS). While utilizing FEM may further accelerate the elasticity simulation, it will involve an additional step of mesh extraction and lose the generalizability of MPM in inelasticity simulation.

4.1.4.2 Lattice Deformation Benchmarks

Dataset: Due to the absence of ground truth for post-deformation, we utilize BlenderNeRF ([Raafat, 2023](#)) to synthesize several scenes, applying bending and twisting with the lattice deformation tool. For each scene, we create 100 multi-view renderings of the undeformed state for training, and 100 multi-view renderings of each deformed state to serve as ground truth for the deformed NeRFs. The lattice deformations are set as input to all methods for fair comparisons.

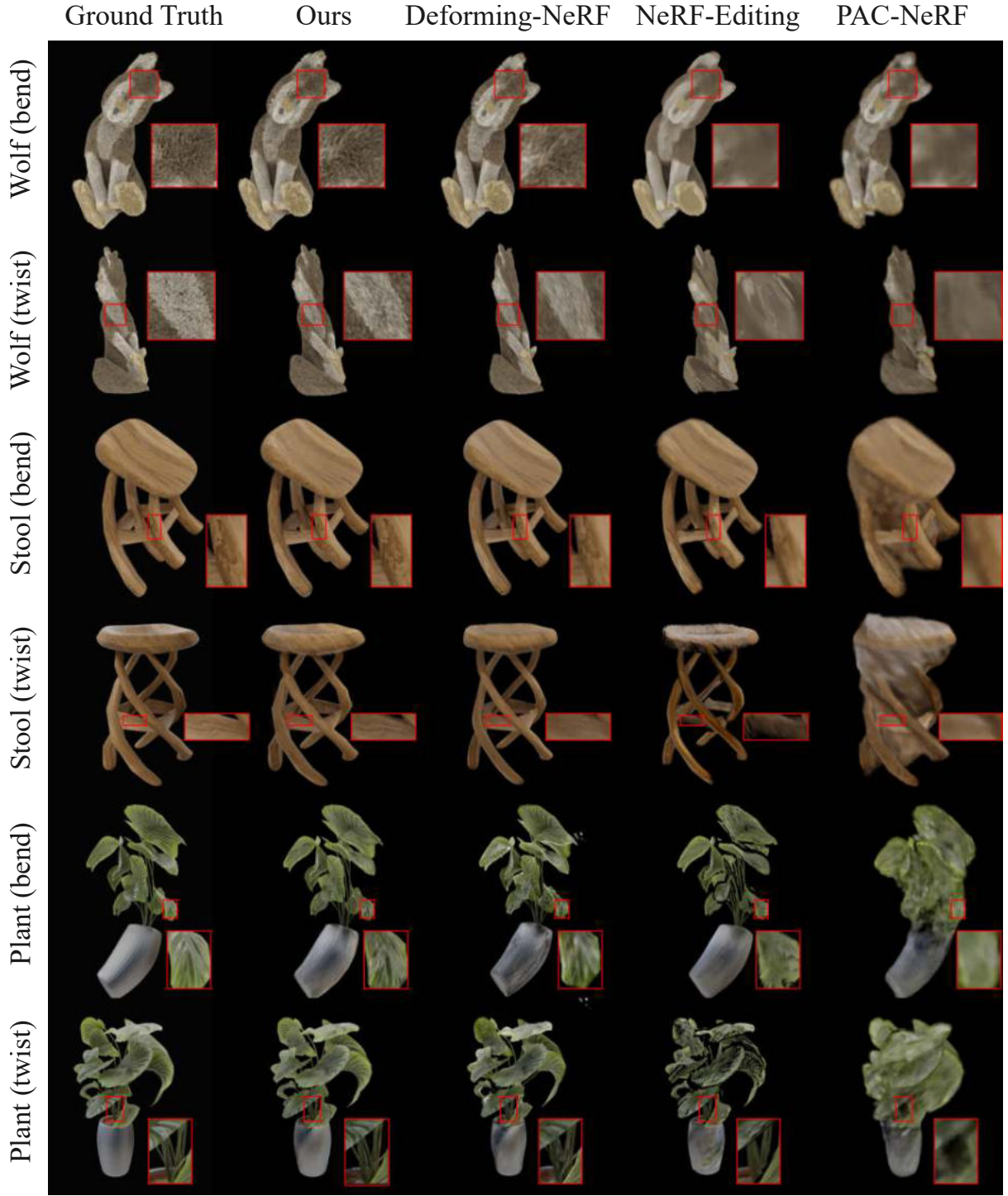


Figure 4.6: Comparisons. For each benchmark case, we select one test viewpoint and visualize all comparisons. We zoom in on some regions to highlight the ability of our method to maintain high-fidelity rendering quality after deformations. We use a black background to avoid the interference of the background.

Test Case	Wolf		Stool		Plant	
Deformation Operator	Bend	Twist	Bend	Twist	Bend	Twist
NeRF-Editing	26.74	24.37	25.00	21.10	19.85	19.08
Deforming-NeRF	21.65	21.72	22.32	21.16	17.90	18.63
PAC-NeRF	26.91	25.27	21.83	21.26	18.50	17.78
Fixed Covariance	26.77	26.02	29.94	25.31	23.95	23.09
Rigid Covariance	26.84	26.16	30.28	25.70	24.09	23.53
Fixed Harmonics	26.83	26.02	30.87	25.75	25.09	23.69
Ours	26.96	26.46	31.15	26.15	25.81	23.87

Table 4.1: We synthesize a lattice deformation benchmark dataset to compare with baselines and conduct ablation studies to validate our design choices. PSNR scores are reported (higher is better). Our method outperforms the others across all cases.

Comparisons: We compare our method with several state-of-the-art NeRF frameworks that support manual deformations: 1) *NeRF-Editing* (Yuan et al., 2022) deforms NeRF using an extracted surface mesh, 2) *Deforming-NeRF* (Xu and Harada, 2022) utilizes a cage mesh for deformation, and 3) *PAC-NeRF* (Li et al., 2023i) manipulates individual initial particles.

We show qualitative results in Figure 4.6 and quantitative results in Table 4.1. NeRF-Editing uses NeuS (Wang et al., 2021) as the scene representation, which is more suited for surface reconstructions rather than high-fidelity renderings. Consequently, its rendering quality is inherently lower than that of 3DGS. Furthermore, the deformation highly depends on the precision of the extracted surface mesh and the dilated cage mesh – an overly tight mesh might not encompass the entire radiance field, while an excessively large one could result in a void border, as observed in the twisting stool and plant examples. Deforming-NeRF, on the other hand, provides clear renderings and potentially leads to enhanced results if higher-resolution deformation cages are provided. However, it employs a smooth interpolation from all cage vertices, thus filtering out fine local details and failing to match lattice deformations. PAC-NeRF is designed for simpler objects and textures in system identification tasks. While offering flexibility through its particle representation, it does not achieve high rendering fidelity. Our method utilizes both zero-order information (the deformation map) and first-order information (the deformation gradient) from each

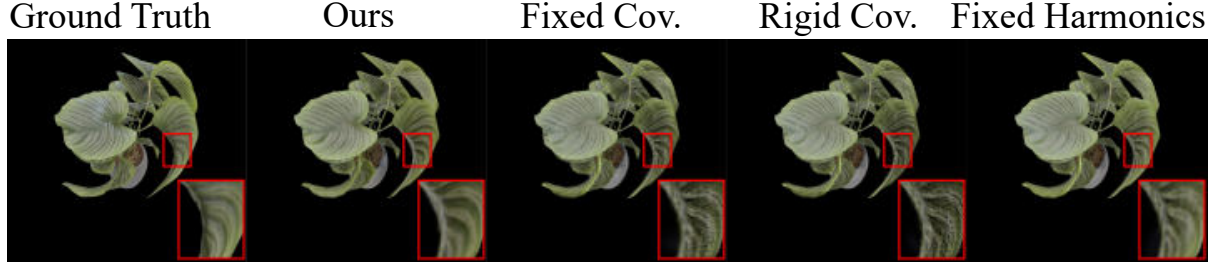


Figure 4.7: Ablation Studies. Non-extensible Gaussians can lead to severe visual artifacts during deformations. Although direct rendering the deformed Gaussian kernels can already obtain good results, additional rotations on spherical harmonics can improve the rendering quality.

lattice cell. It outperforms the other methods across all cases, as high rendering qualities are well preserved after deformations. Although not primarily designed for editing tasks, this comparison showcases our method’s significant potential for realistic editing of static NeRF scenes.

Ablation Studies: We further conduct several ablation studies on these benchmark scenes to validate the necessity of the kinematics of Gaussian kernels and spherical harmonics: 1) *Fixed Covariance* only translates the Gaussian kernels. 2) *Rigid Covariance* only applies rigid transformations on the Gaussians, as assumed by [Luiten et al. \(2023\)](#). 3) *Fixed Harmonics* does not rotate the orientations of spherical harmonics, as assumed by [Wu et al. \(2023a\)](#).

Here we visualize one example in [Figure 4.7](#). We can observe that Gaussians will not properly cover the surface after deformation if they are non-extensible, leading to severe visual artifacts. Enabling the rotation of spherical harmonics can slightly improve the consistency with the ground truth. We include quantitative results on all test cases in [Table 4.1](#), which shows that all these enhancements are needed to achieve the best performance of our method.

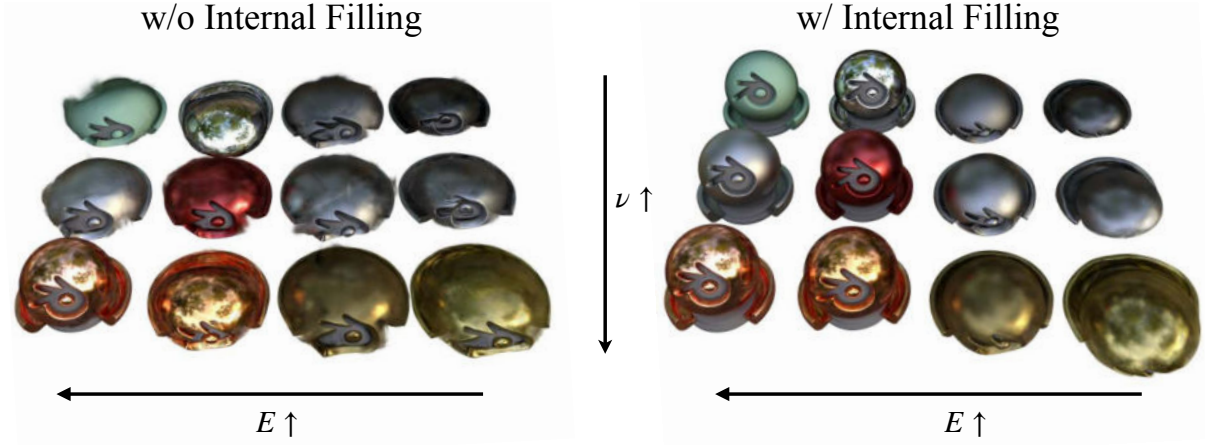


Figure 4.8: Internal filling enables more realistic simulation results. Our method also supports flexible control of dynamics via material parameters. A larger Young’s modulus E indicates higher stiffness while a larger poission ratio ν leads to better volume preservation.

4.1.4.3 Additional Qualitative Studies

Internal Filling: Typically, the 3D Gaussian splatting framework focuses on the surface appearance of objects and often fails to capture their internal structure. Consequently, the interior of the modeled object remains hollow, resembling a mere shell. This is usually not true in the real world, leading to unrealistic simulation results. To address this challenge, we introduce an internal filling method leveraging a reconstructed density field, which is derived from the opacity of Gaussian kernels. [Figure 4.8](#) showcases our simulation results with varying physical parameters. Objects devoid of internal particles tend to collapse when subjected to gravity forces, irrespective of the material parameters used. In contrast, our approach assisted by internal filling allows for nuanced control over object dynamics, effectively adjusting to different material characteristics.

Volume Conservation: Existing approaches to NeRF manipulation focus primarily on geometric adjustments without incorporating physical properties. A key attribute of real-world objects is their inherent ability to conserve volume during deformation. In [Figure 4.9](#), we conduct a comparison study between our method and NeRF-Editing ([Yuan et al., 2022](#)), which utilizes surface As-Rigid-As-Possible (ARAP) deformation ([Sorkine and Alexa, 2007](#)). Unlike NeRF-Editing, our approach accurately captures and maintains

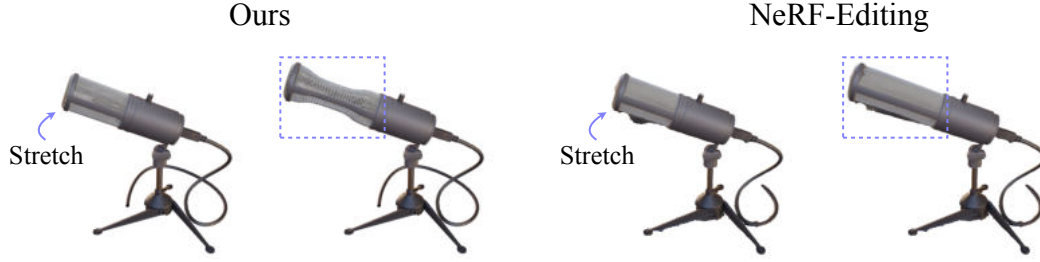


Figure 4.9: Volume Conservation. Compared to the geometry-based editing method (Yuan et al., 2022), our physics-based method is able to capture volumetric behaviors, leading to more realistic dynamics.



Figure 4.10: Anisotropy Regularizer. We introduce an anisotropy constraint for Gaussian kernels, effectively enhancing the fidelity of the Gaussian-based representation under dynamic conditions.

the volume of the deformed objects.

Anisotropy Regularizer: 3D Gaussian models inherently represent anisotropic ellipsoids. However, excessively slender Gaussian kernels can lead to burr-like visual artifacts, especially pronounced during large deformations. To tackle this issue, we introduce an additional regularization loss (4.12) to constrain anisotropy. As demonstrated in Figure 4.10, this additional loss function effectively mitigates the artifacts induced by extreme anisotropy.

4.1.5 Discussion

Conclusions: This paper introduces PhysGaussian, a unified simulation-rendering pipeline that generates physics-based dynamics and photo-realistic renderings simultane-

ously and seamlessly.

Limitations: In our framework, the evolution of shadows is not considered, and material parameters are manually set. Automatic parameter assignment could be derived from videos by combining GS segmentation (Cen et al., 2023; Ye et al., 2023b) with a differentiable MPM simulator. Additionally, incorporating geometry-aware 3DGS reconstruction methods (Guédon and Lepetit, 2023) could enhance generative dynamics. Future work will also explore handling more versatile materials like liquids and integrating more intuitive user controls, possibly leveraging advancements in Large Language Models (LLMs).

4.2 VR-GS: A Physical Dynamics-Aware Interactive Gaussian Splatting System in Virtual Reality

As 3D content becomes increasingly prevalent, there is a growing focus on the development of engagements with 3D virtual content. Unfortunately, traditional techniques for creating, editing, and interacting with this content are fraught with difficulties. They tend to be not only engineering-intensive but also require extensive expertise, which adds to the frustration and inefficiency in virtual object manipulation. Our proposed VR-GS system represents a leap forward in human-centered 3D content interaction, offering a seamless and intuitive user experience. By developing a physical dynamics-aware interactive Gaussian Splatting (GS) in a Virtual Reality (VR) setting, and constructing a highly efficient two-level embedding strategy alongside deformable body simulations, VR-GS ensures real-time execution with highly realistic dynamic responses. The components of our system are designed for high efficiency and effectiveness, starting from detailed scene reconstruction and object segmentation, advancing through multi-view image in-painting, and extending to interactive physics-based editing. The system also incorporates real-time deformation embedding and dynamic shadow casting, ensuring a comprehensive and engaging virtual experience.



Figure 4.11: Utilizing the VR-GS system, individuals can engage in intuitive and interactive physics-based game-play with deformable virtual animals and realistic environments represented with 3D Gaussian Splatting.

4.2.1 Introduction

As digital technology advances, the importance of 3D content is becoming increasingly prominent across various industries, from entertainment to education. This growth is driving the demand for high-fidelity 3D content within the Computer Graphics (CG) and Computer Vision (CV) communities. Despite their capabilities of being rendered efficiently, traditional 3D/4D content creation, reliant on 3D modeling tools and game engines, is time-intensive and complex, often beyond the reach of non-expert users. This accessibility barrier limits broader user engagement in high-quality content creation.

Recognizing the advantages and limitations of the traditional graphics pipeline, our goal is to find a modern variant. To enhance the visual quality and ease of creation for non-expert users, we move away from traditional 3D models in graphics pipelines and instead, adopt state-of-the-art radiance field techniques for rendering. In this context, Neural Radiance Fields (NeRF) emerge as a natural choice. Despite its versatility, NeRF’s volume rendering falls short in efficiency for interactive applications, which demand high frame rates. Additionally, NeRF’s approach to handling deformations—requiring bending of query rays via an inverse deformation map—is slow (Pumarola et al., 2021). Fortunately, 3D Gaussian Splatting (GS) (Kerbl et al., 2023) has been recently introduced as an efficient and explicit alternative to NeRF. This method not only excels in rendering efficiency but also provides an explicit geometric representation that can be directly deformed or edited. The explicit nature of 3D GS can simplify the direct manipulation of

geometry by solving a Partial Differentiable Equation (PDE) that governs the motions of Gaussian kernels. Furthermore, GS eliminates the need for high-fidelity meshes, UV maps, and textures, offering the potential for naturally photorealistic appearances. It is worth noting that many studies have already demonstrated the use of 3D GS for 4D dynamics (Park et al., 2021; Pumarola et al., 2021; Yang et al., 2023d) and the integration of physics-based simulations for more realistic 4D content generation (Xie et al., 2023b; Feng et al., 2024c), as well as for creating animatable avatars (Zielonka et al., 2023; Xu et al., 2023c).

In this paper, we introduce a physics-aware interactive system for immersive manipulation of 3D content represented with GS. To ensure an interactive experience, we utilize eXtended Position-based Dynamics (XPBD) (Macklin et al., 2016), a highly adaptable and unified physical simulator, for real-time deformation simulation. Direct incorporation of a simulator onto GS kernels presents challenges, as the simulation and rendering processes have distinct geometrical representations. To address this, we construct a tetrahedral cage for each segmented GS kernel group and embed these kernel groups into corresponding meshes. The deformed mesh, driven by XPBD, subsequently guides the deformation of the GS kernels. Noticing that simplistic embedding techniques can lead to undesirable, spiky deformations in GS kernels, we propose a novel two-level embedding approach. This method allows each Gaussian kernel to adapt to a smoothed average deformation of the surrounding tetrahedra. The intricate combination of GS and XPBD through our two-level embedding not only achieves real-time physics-based dynamics but also upholds high-quality, realistic rendering. In summary, our contributions include:

- *A Physics-Aware Interactive System:* Development of a system that enables interactive, physics-aware manipulation of 3D content represented with GS.
- *Two-Level Deformation Embedding:* Introduction of a novel two-level embedding approach that allows Gaussians to adapt smoothly to the mesh, enhancing the deformation realism and preventing undesirable spiky artifacts.

We deploy our system on VR devices, enriched by segmentation, inpainting, and shadow

map, offering users a rich platform for 3D content manipulation.

4.2.2 Related Work

4.2.2.1 Radiance Fields Rendering

A variety of 3D representations, such as mesh (Sorkine and Cohen-Or, 2004), point clouds (Qi et al., 2017), signed distance fields (Wang et al., 2021), and grids (Zhu and Bridson, 2005) are exploited in early work for visual computing tasks, including synthesis, estimation, manipulation, animation, reconstruction, and transmission of data about objects and scenes (Kerbl et al., 2023; Xie et al., 2022b; Gao et al., 2023). Since neural radiance fields (NeRF) (Mildenhall et al., 2021) was proposed for novel view synthesis with differentiable volume rendering, it has been applied to versatile computer graphics and computer vision tasks, such as scene reconstruction (Oechsle et al., 2021; Meng et al., 2023; Yariv et al., 2021), synthesis (Sun et al., 2023; Huang et al., 2023c), rendering (Lombardi et al., 2021; Wu et al., 2023b), interactive games (Xia et al., 2024) and animation (Chen et al., 2021; Peng et al., 2021a), simulation (Li et al., 2023j), etc. While Neural Radiance Fields (NeRF) have achieved remarkable image quality, they suffer from significant time and memory inefficiencies (Lindell et al., 2021). To mitigate these issues, various methods have been introduced. Sparse representations (Liu et al., 2020), decomposed strategies (Niemeyer and Geiger, 2021; Rebain et al., 2021), and multi-resolution encoding (Müller et al., 2022) have all been proposed to enhance volume rendering efficiency while maintaining high quality. However, NeRFs are implicit representations, making it challenging to detect and resolve collisions (Qiao et al., 2023). Recently, 3D Gaussian splatting (GS) proposed by (Kerbl et al., 2023) utilizes an array of 3D Gaussian kernels to represent scenes explicitly, facilitating faster optimization and achieving state-of-the-art results.

4.2.2.2 Editing in Radiance-based Scene

A natural extension of NeRF is to support user-guided editing. [Li and Pan \(2023\)](#) made use of two proxy cages to offer interactive control of the shape deformation of NeRF. Besides geometry editing, text prompts ([Wang et al., 2023a](#)) or a user-edited image ([Bao et al., 2023](#)) can also be adopted to conduct style transfer of NeRF. Besides, [Lin et al. \(2023b\)](#) put forward exploiting 2D sketches to edit high-quality facial NeRF. Concerning NeRF texture editing, [Huang et al. \(2023c\)](#) utilized a coarse-fine disentanglement representation and a patch-matching algorithm to synthesize textures of different shapes from multi-view images. De-Nerf ([Wu et al., 2023b](#)) exploited a hybrid light representation that enables users to relight NeRF. In contrast, Gaussian Splatting is more appropriate for post-editing tasks thanks to its explicit nature and has inspired a series of follow-up works ([Chen et al., 2023f](#); [Fang et al., 2023](#); [Ye et al., 2023b](#); [Duisterhof et al., 2023](#); [Huang et al., 2023d](#)).

Another significant direction in NeRF research is the incorporation of dynamic components into static scenes. To achieve this, a popular strategy ([Pumarola et al., 2021](#); [Park et al., 2021](#)) is to consider time as an additional input to the system. This method typically splits a dynamic NeRF into a canonical static field and a deformation field, with the latter mapping this canonical representation to a deformed one. More recently, [Yang et al. \(2023d\)](#) drew inspiration from 3D GS to reconstruct a dynamic scene using 4D Gaussian primitives that change over time. However, the dynamics in these methods are generally confined to motions captured from input data, limiting their ability to synthesize unseen dynamics. To generate novel dynamics, [Xie et al. \(2023b\)](#) and [Feng et al. \(2024c\)](#) have integrated physics-based simulations into 3D static representations, paving the way for generating physically plausible and novel dynamic scenes.

4.2.2.3 Real-Time Neural Radiance Fields

Rendering NeRF is computationally expensive for real-time applications, such as VR, which causes high latency and low quality ([Song et al., 2023a](#); [Li et al., 2022b](#)). To address these challenges and enable high-fidelity rendering with minimal latency on a

single GPU, various techniques have been proposed. These include gaze-contingent 3D neural representations (Deng et al., 2022), variable rate shading (Rolff et al., 2023), and hybrid surface-volume representations (Turki et al., 2023), all aimed at accelerating NeRF. In contrast to single-GPU solutions, VR-NeRF (Xu et al., 2023a) leveraged multiple GPUs to achieve high-quality volumetric rendering of NeRF. Beyond software acceleration techniques, RT-NeRF (Li et al., 2022a) utilized an algorithm-hardware co-design framework to provide real-time NeRF solutions for immersive VR rendering. While these methods primarily focus on enhancing NeRF rendering, our work is dedicated to facilitating interactive editing.

Li et al. (2023d) integrated affine transformation with NeRF to enable interactive features, such as exocentric manipulation, editing, and VR tunneling effects. However, their approach is limited to filter-based edits such as edge blurring and fails to support the deformation of virtual objects’ geometry. RealityGit (Li et al., 2023c) and Magic NeRF Lens (Li et al., 2023e) modified a NeRF model by erasing or revealing a portion of NeRF. Nevertheless, neither system permits deforming the geometrical structure of the model. In contrast to these works which concentrate on interactive editing of NeRF, our system enables users to interact with 3D GS in a physically realistic way in real time.

4.2.3 System Design

We propose a unified physics dynamics-aware interactive VR system for real-time interactions with Gaussian Splatting (GS).

Immersive and Realistic Generative Dynamics: Our targeted system aims to offer users an immersive and realistic experience, characterized by dynamic motions, 3D virtual shapes, and illuminations that closely mirror the real world (Kalawsky, 1999). This goal sets our system apart, especially in how it handles physics-based 3D deformations and dynamics, as opposed to those that rely solely on geometric deformation. The proposed system leverages physics-based simulation techniques to produce realistic dynamics and facilitate 3D shape editing. Unlike methods that reconstruct motion from time-dependent

datasets or utilize generative AI to drive Gaussian models, our system adheres to physical principles. To create an immersive experience, we choose VR as the interaction platform.

Real-Time Interaction: For an interactive system, low latency is crucial to prevent disorientation or motion sickness (Sutcliffe et al., 2019; Deng et al., 2022). To offer instantaneous output and feedback, most interactive systems support basic transformations including rotation, scaling, and translation. Our system additionally offers users real-time physics-based interaction. To our knowledge, VR-GS represents the first interactive physics-based GS editor. To achieve both real-time performance and physically realistic editing, we implement a reduced representation model and parallel-friendly algorithms within our frame budget. While the per-Gaussian-based discretization as done in Phys-Gaussian (Xie et al., 2023b) offers detailed dynamics, its time cost is impractical for our system’s needs. Instead, we utilize a tetrahedral mesh embedding reconstructed from Gaussian kernels to drive GS motion. We adopt XPBD with finite-element constraints to achieve real-time simulations.

Unified Framework: VR-GS is a framework that integrates rendering and simulation within a unified pipeline. Our design principles adhere to the goal of “what you see is what you simulate” (Müller et al., 2016). Unlike Xie et al. (2023b), our system employs a hybrid approach, combining cage mesh and Gaussian kernels, to achieve real-time performance. The deformation gradient field of Gaussian kernels is embedded in the cage mesh through piecewise constant interpolation. With each Gaussian kernel resembling a rotated ellipsoid, this simple embedding strategy leads to spiky artifacts during large deformation, a challenge highlighted in (Xie et al., 2023b). To mitigate this, we employ a two-level interpolation scheme: initially embedding each Gaussian within an individual bounding tetrahedron, followed by embedding these tetrahedra within the simulated cage mesh to elevate the smoothness of the deformation field.

Additional Components: Our framework is designed to construct VR-compatible interactive settings derived from real-world captures. Consequently, it introduces complexities associated with facilitating intuitive user interactions and delivering real-time feedback that meets user expectations and common sense perceptions (Stanney et al., 2003; Hale and Stanney, 2014; Gabbard et al., 1999). In particular, when users engage with objects within a scene, they anticipate natural movements and realistic appearances from both the selected objects and the surrounding environment. Moreover, it is necessary to address voids when an object is removed from its supporting base. To manage these challenges, our system additionally integrates functionalities for object segmentation and scene inpainting.

4.2.4 Method

4.2.4.1 Gaussian Splatting

Gaussian splatting, proposed by Kerbl et al. (2023), is an explicit 3D representation to encapsulate 3D scene information using a set of 3D anisotropic Gaussian kernels, each with learnable mean μ , opacity σ , covariance matrix Σ , and spherical harmonic coefficients $\mathcal{C}$. Spherical harmonics (SH) are expansions of the view-dependent color function defined on the unit sphere. To render a view, the 3D splats are projected to 2D screen space ordered by their z -depth. The color C of a pixel is computed by α -blending of these 2D Gaussians from near to far:

$$C = \sum_{i \in N} c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (4.13)$$

where c_i is the evaluated color by SHs viewed from the camera to the kernel’s mean. α_i is the product of the kernel’s opacity and 2D Gaussian weight evaluated at the pixel coordinate. Leveraging a differentiable implementation, the rendering loss towards the ground truth image can be backpropagated to Gaussians’ parameters for optimization.

In contrast to traditional NeRFs based on implicit scene representations, GS provides

an explicit representation that can be seamlessly integrated with post-processing manipulations, such as animating and editing. Moreover, the efficient rasterization and superior rendering quality of 3D Gaussians facilitate their integration with VR.

4.2.4.2 VR-GS Assets Preparation

Each 3D asset in our VR system consists of a high-fidelity 3D GS reconstruction and an enveloping simulatable tetrahedral mesh in a moderate resolution to enable real-time physics-aware dynamics. These preparations are conducted offline before immersive and interactive editings within our VR environment, which includes:

- *Segmented GS Generation*: we support interactions with individual objects in a large scene, achieved by segmented GS reconstructions.
- *Inpainting*: The occluded parts between the objects and their supporting planes usually have no texture. We inpaint 3D GS representations leveraging a 2D inpainting technique.
- *Mesh Generation*: A simulation-ready tetrahedral mesh is generated for each object.

Segmentation: The segmented GS is constructed during GS training. We first generate 2D masks on the multi-view RGB images by utilizing a 2D segmentation model (Cheng et al., 2023). Each segmented part is assigned a different color that is consistent across different views. Subsequently, we enhance the scene representation by integrating three additional learnable RGB attributes into the 3D Gaussian kernels. During the reconstruction process, each 3D Gaussian kernel will automatically learn what object it belongs to utilizing a segmentation loss function L_{seg} :

$$L_{\text{seg}} = L_1(M_{2d}, I), \quad (4.14)$$

where M_{2d} represents colored 2D segmentation results and I denotes the rendering of 3D Gaussian kernels with colors replaced by the extended RGB attributes instead of

evaluated from SHs. Thus, the total loss becomes

$$L_{\text{total}} = (1 - \lambda)L_1 + \lambda L_{\text{SSIM}} + \lambda_{\text{seg}}L_{\text{seg}}, \quad (4.15)$$

where L_1 and L_{SSIM} are computed between the normally rendered images and the multi-view ground truths. We use $\lambda = 0.2$ and $\lambda_{\text{seg}} = 0.1$ in all our experiments.

Inpainting: After 3D GS segmentation, we extract all objects separately from the scene. This process of object removal, however, results in the emergence of holes within the regions that are previously occluded. To alleviate the issue, we utilize a 2D inpainting tool LaMa (Suvorov et al., 2022) to guide the 3D inpainting of Gaussian kernels. We freeze the Gaussian kernels located outside of the holes and then use an inpainting loss $L_{\text{inpaint}} = L_1(I_{\text{inpainted}}, I)$ to optimize a Gaussian kernel patch under the guidance of the 2D inpainted images $I_{\text{inpainted}}$ for the current 3D GS rendering I . The result of our inpainting is validated in Section 4.2.5.

Mesh Generation: Due to our design choice to use mesh-based simulation, we generate a tetrahedral mesh for each group of the segmented 3D GS kernels. These meshes will not be rendered during the interaction but only serve as the media of dynamics. Hence we can use a moderate mesh resolution which does not hinder performance. To construct a simulation mesh, we first use internal filling proposed by (Xie et al., 2023b) to fill particles into the void internal region that is not reconstructed by GS. Then we treat centers of Gaussians as a point cloud and convert it to a voxelized VDB representation (Museth, 2013). A water-tight surface mesh is then extracted using marching cubes (Lorensen and Cline, 1987) and tetrahedralized into a finite element mesh using TetGen (Si, 2015).

4.2.4.3 Unified Framework for Simulation and Rendering

As shown in Xie et al. (2023b), Gaussian kernels can be deformed by the simulated deformation field. We follow the Gaussian kinematics of this work but replace the

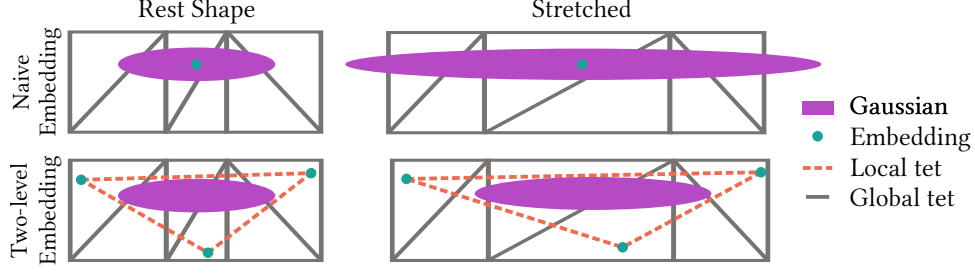


Figure 4.12: Our two-level embedding effectively resolves spiky artifacts. Each Gaussian kernel is embedded into a local tetrahedron. The vertices of the local tetrahedron are independently embedded into the global mesh.

simulator with XPBD (Macklin et al., 2016) to achieve real-time interactions. We employ the strain energy constraint as the elastic model and adopt the velocity-based damping model in the XPBD framework.

Physical Parameters: The physical parameters, such as densities and material stiffness, are tunable hyperparameters. Following PhysGaussian (Xie et al., 2023b), we manually set physical parameters for all interactable objects, including Young’s modules (E), Poisson ratio (ν), and density (ρ). We start from an initial configuration $E = 1000$ Pa, $\nu = 0.3$, $\rho = 1000$ kg for each object, then tune them to produce visually plausible dynamics. The Young’s modulus governs the overall stiffness of an object and is manually adjusted by analyzing the severity of deformation of its free fall collision with the ground. Additionally, we adjust the relative density between two objects based on their deformations under collision. In practice we usually arrive at a finalized parameter setting that gives visually plausible results within 10 iterations of this manual process.

Embedding: In our mesh-based simulation, the deformation map is piece-wise linear, with the resulting deformation gradient piece-wise constant within each tetrahedron. Given a tetrahedron with the rest-shape configuration $\{\mathbf{x}_0^0, \mathbf{x}_1^0, \mathbf{x}_2^0, \mathbf{x}_3^0\}$ and the current configuration $\{\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$, the deformation gradient is defined as

$$\mathbf{F} = \begin{bmatrix} \mathbf{x}_1 - \mathbf{x}_0 & \mathbf{x}_2 - \mathbf{x}_0 & \mathbf{x}_3 - \mathbf{x}_0 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1^0 - \mathbf{x}_0^0 & \mathbf{x}_2^0 - \mathbf{x}_0^0 & \mathbf{x}_3^0 - \mathbf{x}_0^0 \end{bmatrix}^{-1}, \quad (4.16)$$

where the inverse of the rest-shape basis can be computed pre-simulation. The mean and the covariance matrix of the deformed Gaussian kernel inside this tetrahedron is given by

$$\boldsymbol{\mu} = \sum_i w_i \mathbf{x}_i, \quad \boldsymbol{\Sigma} = \mathbf{F} \boldsymbol{\Sigma}_0 \mathbf{F}^T, \quad (4.17)$$

where w_i is the barycentric coordinates of the initial center $\boldsymbol{\mu}_0$ in the rest-shape configuration, and $\boldsymbol{\Sigma}_0$ is the initial covariance matrix (Xie et al., 2023b). The deformed Gaussian kernels can be directly rendered by the point splatting procedure. However, the direct embedding of Gaussian centers cannot guarantee that every ellipse shape is completely enveloped by some tetrahedron inside the simulation mesh. As shown in Figure 4.12, this can lead to spiky artifacts. Observe that kernels that are completely inside the tetrahedron will always be inside. This motivates us to propose a two-level embedding procedure:

1. Local embedding: we independently envelope every Gaussian kernel by an as-tight-as-possible tetrahedron. There is no connectivity between these local tetrahedra.
2. Global embedding: we embed the vertices of local tetrahedra into the global simulation mesh.

As the global mesh is deformed by the simulation, the vertices of local tetrahedra are kept inside the boundary, driving the kinetic evolution of Gaussian kernels. A local tetrahedron could overlap with multiple global tetrahedra. The deformation map on it can be understood as the average of the surrounding global tetrahedra, hence eliminating sharp, spiky artifacts, as validated in Section 4.2.5.

Shadow Map: While the original global illumination of the scene can be accurately learned and baked by the spherical harmonics on each Gaussian, the shadow will no longer be aligned with the object when it is moving or deforming. Bringing shadow map (Wanger et al., 1992) into the GS framework can enhance the immersive experience in VR. More importantly, it can guide human perception of the spatial relationships between

objects: during manipulation, users will rely on the shadow to determine the distance between objects. The shadow map is a fast real-time algorithm and is well-aligned with the GS rasterization pipeline. We follow (4.13) to estimate the depth map from the light source and test the visibility for each Gaussian using this depth map. The influence of shadow maps is studied in Section 4.2.5.

4.2.5 Evaluation

In this section, we assess the essential components of our proposed system, including two-level GS embedding and shadow map.

Two-level Embedding: The two-level embedding is a crucial component in our physical dynamics-aware system, integrating the tetrahedra cage mesh with the embedded Gaussian. In Figure 4.13, we conduct an ablation study to validate the effectiveness of our two-level embedding approach. Under conditions of extreme stretching or twisting, the naive embedding method, which simply embeds the Gaussian kernel within the closest tetrahedron, leads to severe spiky artifacts. Our two-level embedding strategy addresses this by initially embedding each Gaussian into a localized, independent tetrahedron, followed by embedding the vertices of this tetrahedron into the cage mesh. The deformation of each Gaussian kernel is determined by averaging the deformations at these vertices, resulting in a smoother deformation gradient field than the naive approach and significantly reducing spiky artifacts.

Inpainting: With the guidance of 2D image segmentation results, our system achieves object-level 3D scene reconstruction, facilitating convenient post-physics-based manipulation. However, GS is limited to reconstructing surfaces visible in the provided multi-view training images. Consequently, some object and background areas, unseen in these images, remain unreconstructed by GS, leading to “black hole” like artifacts when foreground objects are moved. To address this, our system integrates the object segmentation mask with LAMA (Suvorov et al., 2022), a 2D inpainting model, to generate inpainted multi-

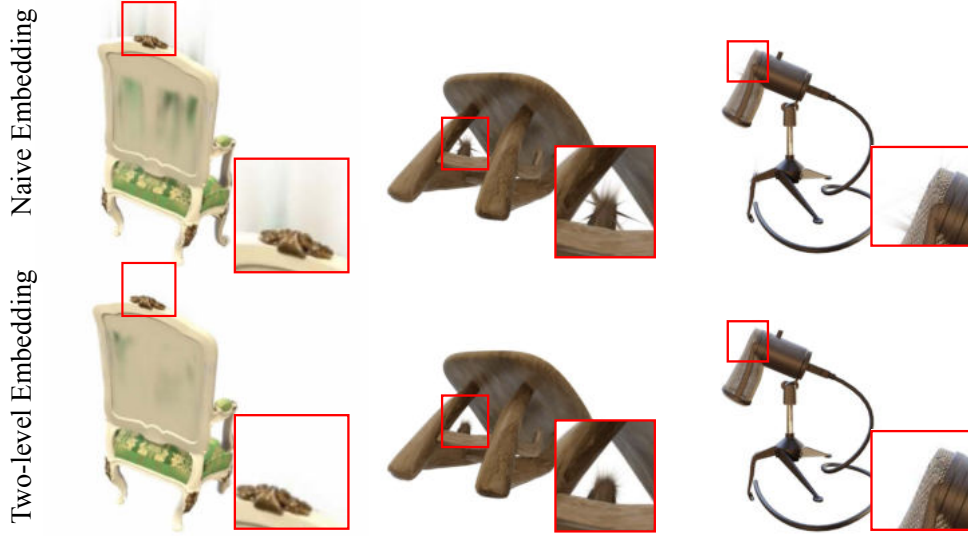


Figure 4.13: Two-level Embedding Evaluation. Our two-level embedding alleviates the commonly seen spiky artifacts for deformed GS kernels.



Figure 4.14: Inpainting Evaluation. GS struggles to reconstruct occluded surfaces. By leveraging LAMA (Suvorov et al., 2022), we produce 2D inpainted results that guide the 3D scene inpainting, enhancing the realism of the 3D representation.

view images. These images then guide the fine-tuning and inpainting of our 3D GS scene, yielding a more complete and realistic user experience, as evidenced in Figure 4.14.

Shadow Map: Furthermore, we take advantage of the shadow map to add extra time-dependent shadows into the GS scene, as shown in Figure 4.15. Original GS represents shadows as textures and thus fails to provide dynamic shadows when objects move. VR-GS allows users to choose the parameters, e.g. position and direction, of the light source to reproduce the lighting setting of the original scene, leading to a more realistic VR environment.

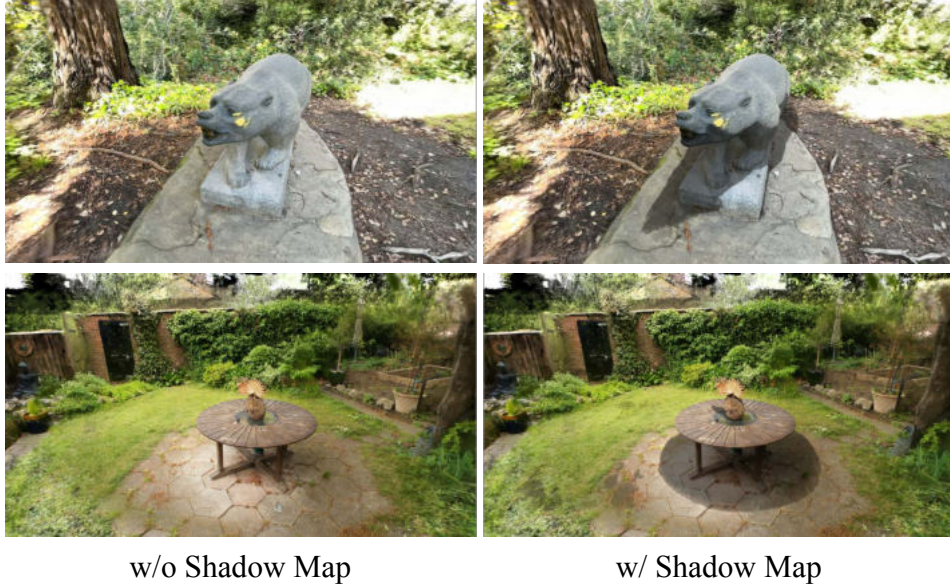


Figure 4.15: Shadow Map Evaluation. GS traditionally models shadows as static surface textures. Our approach, employing a shadow map, generates dynamic shadows for a more immersive experience.

4.2.6 Experiment

In this section, we benchmark our VR-GS system’s simulation performance against other methods in GS and NeRF manipulation. Additionally, we showcase interactive demonstrations on VR devices. Our prototype, developed in Unity with a CUDA-implemented plugin, is tested using a Quest Pro Head-Mounted Display (HMD) and corresponding controllers, on an Intel Core i9-14900KF CPU with 32GB memory and an NVIDIA GeForce RTX 4090 GPU.

4.2.6.1 Performance

VR-GS is designed for real-time, physics-aware interactions by integrating Gaussian kernels within a cage mesh for simulation. This cage mesh is derived from the VDB representation of Gaussians, with adjustable mesh resolution to influence quality. We first conduct experiments to explore the trade-offs between mesh quality and system performance. As shown in [Figure 4.16](#), using a coarse cage mesh facilitates higher frame rates but may lead to the loss of fine details. Conversely, a higher-resolution mesh inevitably increases

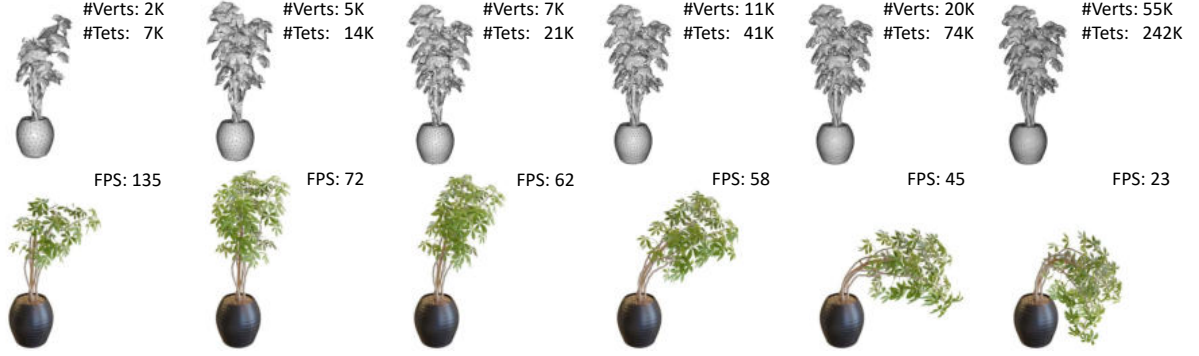


Figure 4.16: Trade-offs between Quality and Performance. The top row displays cage meshes at varying resolutions, and the bottom row illustrates the corresponding simulation dynamics. Low-resolution meshes fail to capture fine dynamic details, whereas high-resolution meshes compromise real-time performance and can result in overly soft artifacts in the simulated object due to non-converging simulations. We employ mid-resolution meshes in practice to achieve an optimal balance between high frame rates and realistic physical dynamics.

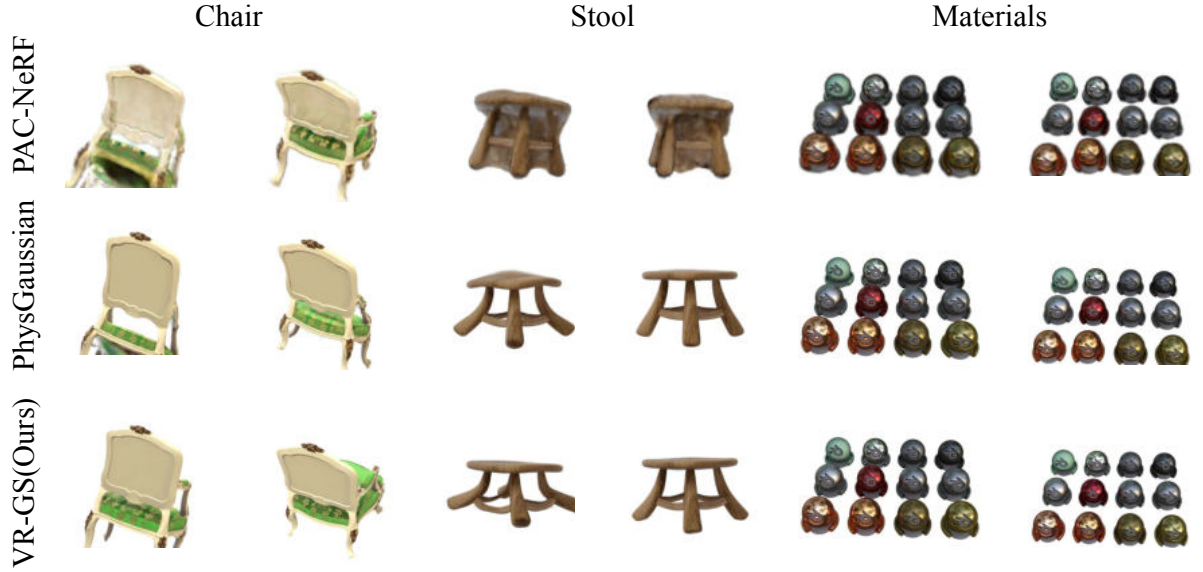


Figure 4.17: Visual Quality Comparison. Our method synthesizes competitive visual results compared to PhysGaussian (Xie et al., 2023b) and significantly outperforms PAC-NeRF (Li et al., 2023j).

Example	PAC-NeRF	PhysGaussian	VR-GS
Stool	0.750	0.112	0.017
Chair	0.813	0.219	0.022
Materials	0.625	0.39	0.021

Table 4.2: Frame time (seconds) comparison between PAC-NeRF, PhysGaussian, and VR-GS.

the computational cost. Moreover, XPBD also requires much more iterations to achieve convergence for finer meshes. Without sufficient iterations, the simulated object may appear to be overly soft, yielding unrealistic dynamics. In practice, we constrain the number of mesh vertices to between 10K and 30K to maintain a balance between high frame rates and accurate physical dynamics.

We then compare VR-GS against two state-of-the-art NeRF/GS physics-based manipulation methods: PAC-NeRF (Li et al., 2023j) and PhysGaussian (Xie et al., 2023b). PAC-NeRF primarily concentrates on estimating material parameters from multi-view videos in reconstructed NeRF scenes. Although it offers novel dynamic generation capabilities, the resulting visuals often fall short in rendering quality. In contrast, PhysGaussian, leveraging GS, produces superior photorealistic dynamics. However, the Material Point Method (MPM) used by both systems can limit real-time performance in complex scenes. For a fair comparison, we standardize the frame time ($\Delta t_{\text{frame}} = 1/25$ sec) and the simulation step time ($\Delta t_{\text{step}} = 0.0001$ sec) across all methods. For VR-GS, we set the XPBD iteration per substep to 1, as the small Δt_{step} sufficiently resolves the dynamics. As depicted in Figure 4.17 and Table 4.2, VR-GS not only matches the visual quality of PhysGaussian but also outperforms PAC-NeRF in clarity and realism. Crucially, our XPBD-based simulation framework allows for significantly faster simulations, making VR-GS ideal for real-time physics-aware interactions.

4.2.6.2 VR Demos

We then showcase VR-GS’s capability to replicate real-world scenarios through several representative demos in VR. Detailed simulation setups and a timing breakdown for these

Example	#Gaussians	#Verts.	#Iter [*]	CD [*]	FPS
Fox (Figure 4.19)	304,875	28,205	1	/	161.2
Bear (Figure 4.19)	2,525,891	19,472	5	/	76.9
Horse (Figure 4.19)	1,295,223	10,611	5	/	115.4
Ring Toss (Figure 4.20)	1,456,209	9,002	10	10	37.7
Table Brick Game (Figure 4.20)	1,866,835	33,279	10	10	41.4
Toy Collection (Figure 4.21)	1,665,128	46,428	20	20	24.3
Box Moving (Figure 4.14)	1,769,412	1,434	10	/	73.8
Animal Crossing (Figure 4.11)	1,312,670	36,386	10	5	34.7
Just Dance (Figure 4.22)	1,059,054	13,936	50	/	33.9

^{*}#Iter: XPBD iterations per substep; CD: collision detections per step.

Table 4.3: Parameters and Timings of Demos in Section 4.2.6.2.

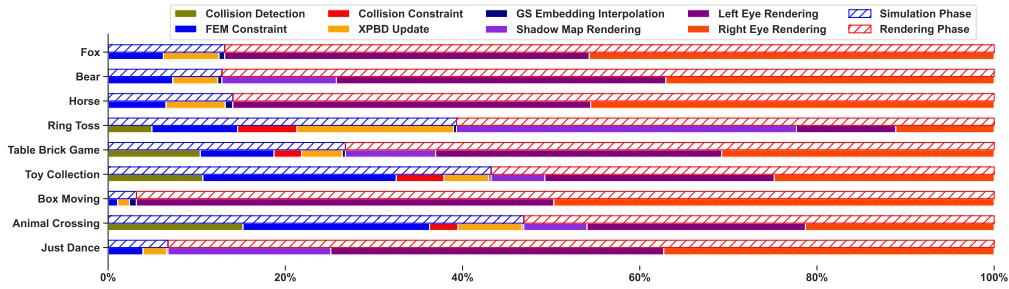


Figure 4.18: Timing Breakdown of Demos in Section 4.2.6.2.

demos are provided in Table 4.3 and Figure 4.18.

Fox, Bear, and Horse Manipulation: VR-GS empowers users to edit 3D Gaussian kernels intuitively and efficiently, utilizing our XPBD-based physics engine for dynamic manipulation. We demonstrate this through three examples as depicted in Figure 4.19: a fox, a bear, and a horse, reconstructed from the Instant-NGP dataset (Müller et al., 2022), the Instruct-NeRF2NeRF (Haque et al., 2023), and the Tanks and Temples dataset (Knapitsch et al., 2017), respectively.

Ring Toss and Table Brick Game: In this example, we demonstrate the system’s ability to seamlessly integrate new *virtual objects* into existing scenes. We use a room scene reconstructed from real-world footage and virtual objects (rings and bricks modeled in Blender) to create ring tossing and table brick game (Figure 4.20). The dining table



Figure 4.19: Fox, Bear, and Horse. VR-GS allows users to manipulate 3D GS at a real-time rate with physically plausible responses.



Figure 4.20: Ring Toss and Table Brick Games. Participants can engage in interactive ring toss and table brick breakout games simulated within authentic environments.

and objects on it serve as collision boundaries for the elastic ring and rigid bricks (Deul et al., 2016).

Toy Collection and Animal Crossing: We reconstructed a living room scene and a set of plush toys (Figure 4.21) for users to interact with. The yellow spherical toy is generated using a text-to-3D generator LucidDreamer (Liang et al., 2023), while the others are reconstructed. Additionally, four animal plush toys were placed on chairs, allowing users to manipulate and deform them freely (Figure 4.11).



Figure 4.21: Toy Collection. We place all the toys on the ground first, and then move them into the basket.

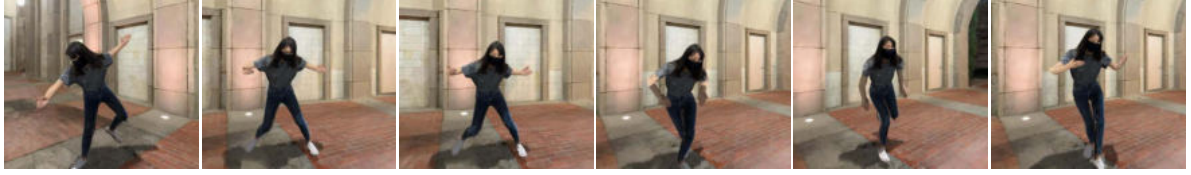


Figure 4.22: Just Dance. VR-GS generates high-fidelity dance given a reconstructed human body.

Just Dance: VR-GS also supports animation of reconstructed GS humans (Figure 4.22). Utilizing a reconstructed human character, we leverage Mixamo’s⁴ auto-rig to generate motion sequences of its surface mesh. These sequences then serve as the boundary condition for the simulation, finally producing the GS human animation that can be viewed in our immersive system.

4.2.6.3 Modeling Details

We employ the *Toy Collection* (Figure 4.21) as a case study to demonstrate the time and labor required to create a fully interactive VR environment from scratch. The setup process is as follows:

1. The initial stage involves setting up the real-world scene. For the *Toy Collection*, the indoor scene was captured directly as one scene, with all toys separately hung with strings as another scene. We completed it within 20 minutes.
2. Next, we capture multi-view images for Gaussian Splatting by recording videos

⁴<https://www.mixamo.com/>

of the two scenes with a consumer-level camera and converting them into image sequences. Camera intrinsic and extrinsic parameters are collected using COLMAP, with the entire process taking around 60 minutes, of which COLMAP occupies approximately 40 minutes.

3. We then proceed to image processing for segmentation and inpainting. Object classes are manually designated in the first video frame. Subsequent frames are automatically segmented using the method from [Cheng et al. \(2023\)](#), taking about 2 minutes. The inpainting of what’s below the basket is fully automated ([Suvorov et al., 2022](#)) and requires 5 minutes.
4. GS reconstruction is performed on the processed images, undergoing 30,000 training steps as specified by [Kerbl et al. \(2023\)](#) without additional treatment for the number of Gaussians, which takes about 30 minutes on a single RTX 4090.
5. After generating 3D GS, we execute internal filling, VDB reconstruction, and TetGen, all within 10 minutes.
6. Lastly, object world positions and physical parameters are specified and estimated in Unity, as detailed in [Section 4.2.4.3](#), taking 20 minutes.

In summary, the total preparation time for the *Toy Collection* is approximately 2 hours and 30 minutes. Manual tasks include scene staging, video recording, object class specification, and physical parameter selection. All other processes are automated. Once modeled, the scene is ready for a physics-aware interactive VR experience.

4.2.7 User Study

We conducted a user study including two tasks with 10 participants: 2 professionals (P1–P2) with 5 and 7 years of experience in 3D VFX software (including Houdini and Blender), and 8 novices (P3–P10), with P2 also having 2 years of VR experience. Our study used the hardware specified in the experimental section and included the Fox, Bear, House, Ring Toss, and Toy Collection demos. Participants received a 10-minute

video tutorial on our system before completing two tasks. After task 2, they answered a questionnaire covering usability (System Usability Scale (Bangor et al., 2009)) and subjective feedback on individual and overall system features, as shown in Section 4.2.7.2. Additionally, we conducted a 20-minute semi-structured interview for more in-depth feedback.

4.2.7.1 Tasks

Task 1: Goal-directed (30 minutes): The participants were required to play two VR games developed by our system: ring toss and toy collection (Figure 4.21) in two different settings (S1: physics-based and S2: transform-based interaction). Each game had a 5-minute duration under each configuration. A 5-minute break was allowed between sessions. The arrangement of sessions and tasks used a Latin square design to counter potential learning effects. We requested users to rate the immersive and realistic experience in VR using a 7-point Likert scale.

Task 2: Open-ended (30 minutes): The participants are asked to edit the scene and generate dynamics freely in aforementioned scenes (shown in Figure 4.19, Figure 4.20 and Figure 4.21) with our prototype system, which includes geometry-based editing, physics-based editing, transform, rotation, duplicating, undoing, rescaling, etc. Task 2 is designed to evaluate the detailed potential factors impacting VR immersive experience.

4.2.7.2 Results and Discussion

According to a paired t-test conducted on the score collected from task 1, it could be noticed that physics-based interaction significantly enhanced user immersion and realism compared to transform-based interaction (physics-based: 6.1 vs transform-based: 4.8 on average, $p = .0227$). As shown in Figure 4.23, our system has received overall positive opinions. Users spoke highly of the placement of virtual content. Transforming objects without physics rules resulted in floating objects, while in our system, all virtual content



Figure 4.23: Study Results. The left chart summarizes individual (M1-M5: object manipulation, scene inpainting, illumination, dynamics, physics placement) and system feedback (M6-M9: ease of use, latency, functionality, satisfaction). The right two figures show that physics-based interaction enhances immersion and realism in editing, whereas transform-based interaction yields less authentic outcomes, e.g. an undeformed and penetrated toys.

placements, such as putting toys onto a sofa, are followed by physics rules and are consistent with the user’s understanding. P7 commented, *“I love this a lot. I have a dog. Petting the fox is really like what I did to my dog at home. I felt so real. Moreover, the placement of the toy is awesome. After they all fell to the basket, the basket even shook for a while.”* The VR-GS system’s realistic lighting and physics-based dynamics create an immersive experience. As a professional user, P2 spoke highly of high-fidelity generative dynamics and illumination. *“Those motions of virtual content are just like what I see in the physical world. I can’t wait to take photos of my own house and then put them in VR with my video game character.”* Users rated the system highly on ease of use (4.6/5) and overall satisfaction (4.8/5). With a System Usability Scale (SUS) score of 83.5, VR-GS is classified as “excellent” according to (Bangor et al., 2009).

4.2.8 Conclusions and Future Work

We presented a physical dynamics-aware interactive Gaussian Splatting system for addressing challenges in editing real-time high-fidelity virtual content. By leveraging the advancements in Gaussian Splatting, VR-GS bridges the quality gap traditionally observed between machine-generated and manually created 3D content. Our system not only enhances the realism and immersion via physically-based dynamics, but also provides fine-grained interaction and manipulation controllability.

Although all the study participants appreciated the efficiency and effectiveness of our system, it still remains to be improved. Firstly, rendering high-fidelity Gaussian kernels in VR is computationally demanding. As a result, rendering generative dynamics in a large scene with 2K resolution might lead to potential latency issues in our system. Secondly, the physical parameters in our system are manually defined. Estimating parameters from videos like PAC-NeRF (Li et al., 2023j), or leveraging large-vision models, would be interesting directions to automate this process. In future work, we aim to incorporate a broader range of materials, such as fluid (Feng et al., 2024a) and cloth, to enhance the system’s capabilities. Furthermore, it is also interesting to explore how to utilize large multimodal models to assess the fidelity of the generated dynamics.

CHAPTER 5

Physics-Guided Animation Generation and Humanoid Loco-Manipulation

5.1 PhysAnimator: Physics-Guided Generative Cartoon Animation

Creating hand-drawn animation sequences is labor-intensive and demands professional expertise. We introduce PhysAnimator, a novel approach for generating physically plausible meanwhile anime-stylized animation from static anime illustrations. Our method seamlessly integrates physics-based simulations with data-driven generative models to produce dynamic and visually compelling animations. To capture the fluidity and exaggeration characteristic of anime, we perform image-space deformable body simulations on extracted mesh geometries. We enhance artistic control by introducing customizable energy strokes and incorporating rigging point support, enabling the creation of tailored animation effects such as wind interactions. Finally, we extract and warp sketches from the simulation sequence, generating a texture-agnostic representation, and employ a sketch-guided video diffusion model to synthesize high-quality animation frames. The resulting animations exhibit temporal consistency and visual plausibility, demonstrating the effectiveness of our method in creating dynamic anime-style animations.

5.1.1 Introduction

Dynamic visual effects are essential to the immersive quality of 2D animation. From the subtle sway of a character’s hair to the fluid motion of garments in response to wind,

realistic and pleasing dynamics create a captivating visual experience. Traditionally, these effects are achieved through meticulous, hand-drawn techniques, where animators painstakingly draw each frame to bring these dynamic elements to life. This process is labor-intensive and requires not only artistic skill but also a deep understanding of natural forces and environmental interactions.

To alleviate the challenges of manual animation, researchers have explored both traditional and data-driven approaches. Traditional animation tools (Kazi et al., 2016; Su et al., 2018) provide interactive systems that assist artists in creating animated illustrations based on established principles of 2D animation. These methods often rely on user-provided stroke inputs to specify motion trajectories and utilize geometry constraints to produce deformation-based animation. However, such approaches typically assume simple inputs like lineart or drawings with separated layers, limiting their applicability to more complex, in-the-wild anime illustrations that feature intricate textures and details. In contrast, data-driven video generative models (Xing et al., 2025; Blattmann et al., 2023a; Ma et al., 2024) offer a promising alternative by leveraging neural synthesis to generate dynamic effects directly from images, bypassing the need for manual sketching or layered inputs. Recent methods even enable interactive motion design, allowing users to specify object trajectories via drag-based inputs (Wu et al., 2025a; Niu et al., 2024; Shi et al., 2024). These approaches typically rely on predicting a sequence of optical flow fields to drive the desired motion and warp the frames accordingly. However, the quality of the generated results is often limited by inaccuracies in the predicted optical flow, which frequently exhibits artifacts due to a lack of geometric understanding and physical constraints. As a result, imprecise motion estimation can lead to noticeable distortions and unsatisfactory visual quality.

Recognizing the limitations of both traditional and purely data-driven methods, we introduce PhysAnimator, a novel framework that integrates physics-based animation with data-driven generative models to synthesize visually compelling dynamic animations from static anime illustrations. Our approach combines the physical consistency of simulation-based methods with the flexibility and expressiveness of generative models, overcoming

the drawbacks of previous approaches. To achieve this, we model objects of interest in the anime illustration as deformable bodies, capturing the fluidity and exaggerated motion characteristic of anime. We solve the motion equations to compute a sequence of optical flow fields that represent consistent dynamics. Users can enhance these animations with customized energy strokes, defining the effects of external forces such as wind. To render the motion dynamics as high-quality frames, we first extract and warp the sketch using the optical flow fields to generate a texture-agnostic video sequence. We then apply a sketch-guided video diffusion model to colorize the sketch sequence based on the reference illustration, ensuring stylistic coherence. Finally, to incorporate dynamic effects that cannot be easily captured by physical simulation, we employ a data-driven cartoon interpolation model, enriching the results with complementary, expressive dynamics. In summary, our contributions are as follows:

- We introduce a novel framework that combines physics-based simulations with data-driven generative models, achieving both physically consistent and stylistically expressive animations.
- We develop an image-space deformable body simulation technique that models anime objects as deformable meshes, capturing fluid and exaggerated dynamics
- We propose to leverage a sketch-guided video diffusion model to render simulation dynamics as high-quality frames, combined with a cartoon interpolation model to introduce additional dynamic effects beyond physical laws.

5.1.2 Related Work

5.1.2.1 Video Diffusion Models

Building on the success of image diffusion models (Dhariwal and Nichol, 2021; Rombach et al., 2022), recent work has introduced video diffusion models to streamline video synthesis from text prompts or images, significantly reducing labor and time costs compared to traditional commercial video editing and creation tools (Ho et al., 2022;

Xing et al., 2023). These approaches either extend pre-trained image diffusion models by incorporating temporal mixing layers in various forms (Singer et al., 2022; Blattmann et al., 2023b; Ge et al., 2023; Esser et al., 2023) or train video diffusion models with temporal layers from scratch on large-scale text-video paired datasets (He et al., 2022; Zhang et al., 2024e). However, using text prompts or a single image as input provides limited control over fine-grained aspects of video generation, making it challenging to define complex structural attributes such as spatial layouts, poses, and shapes (Guo et al., 2025; Yin et al., 2023). To allow for more detailed control over object motion and camera movements, additional guidance in the form of motion trajectories (Jeong et al., 2024; Shi et al., 2024; Yin et al., 2023; Chen et al., 2023d), pose (Zhang et al., 2024c), depth (He et al., 2023), and optical flow (Guo et al., 2025) has been integrated into video diffusion models to produce more controllable videos. These powerful video diffusion models have also been applied to various downstream tasks, such as video editing (Ku et al., 2024; Gu et al., 2024), image animation (Gal et al., 2024; Xing et al., 2025; Wu et al., 2024a), video understanding (Wang et al., 2024a; Lee et al., 2024; Shao et al., 2024a), video interpolation (Xing et al., 2024; Jain et al., 2024) and 3D reconstruction and generation (Voleti et al., 2025; Han et al., 2025; Kwak et al., 2024; Chen et al., 2024c; Liu et al., 2024a). Nevertheless, these data-driven approaches usually produce artifacts due to a lack of geometric understanding and physical constraints.

5.1.2.2 Physics-based Animation

Researchers have explored various methods for animation authoring. Xing et al. (2015) extend local similarity techniques to global similarity, enabling automatic sketch completion based on previous frames. Building on this, Peng et al. (2020) propose a keyframe-based sculpting system that autocompletes user edits using an intuitive brushing interface. Willett et al. (2018) tackle the challenge of textured anime images by segmenting the illustration into layers for different objects and allowing users to provide scribbles as trajectory guides. Leveraging animation principles, some works (Ma et al., 2022; Kazi et al., 2016) use geometry-based deformations to create stylized animation effects such

as “squash” and “stretch.” Similarly, [Xing et al. \(2016\)](#) employ physics-based simulations and introduce energy brushes to generate elemental dynamics ([Gilland, 2012](#)), such as smoke and fire. Other approaches ([Coros et al., 2012](#); [Jones et al., 2015](#); [Barbič et al., 2009](#)) incorporate physical laws like gravity, collision, and elasticity to animate deformable characters, enabling the efficient creation of complex scenes. For instance, [Jones et al. \(2016\)](#) introduces an example-based plasticity model based on linear blend skinning for animating the failure of near-rigid, man-made materials. [Coros et al. \(2012\)](#) exploit two rest-pose adaptation strategies using only internal energy to animate curve, shell, and solid-based characters. Physics-based modeling has also proven effective in adding secondary dynamics ([Willett et al., 2017](#); [Zhang et al., 2020a](#)) to enhance details in rigged animations. Despite their effectiveness, these 2D animation methods typically assume simple inputs, such as sketches, or require layered separation, which limits their applicability for animating complex, in-the-wild anime illustrations.

5.1.2.3 Generative Image Dynamics

With the advancement of generative models, there is growing interest in leveraging these powerful techniques to synthesize dynamic animations from static images, guided by motion features extracted from various user inputs. These inputs can be sparse, such as text prompts ([Xing et al., 2025](#); [Chen et al., 2023e](#); [Li et al., 2024c](#)), trajectories ([Wang et al., 2024e](#); [Wu et al., 2025a](#); [Shi et al., 2024](#); [Zhang et al., 2024d](#); [Li et al., 2024f](#)), or camera movements ([Wang et al., 2024e](#); [Yang et al., 2024b](#)), or dense, like reference videos ([Wei et al., 2024](#); [Zhao et al., 2023b](#); [Jeong et al., 2024](#)). To achieve controllable dynamics, prior works often incorporate ControlNet ([Zhang et al., 2023d](#)) into image or video generative models during the decoding stage, utilizing motion features such as Canny edges, depth maps ([Guo et al., 2025](#)), 2D Gaussian maps ([Wu et al., 2025a](#)), and optical flow maps ([Shi et al., 2024](#)). In addition, several methods employ specialized motion fusion modules ([Zhang et al., 2024d](#); [Wang et al., 2024e](#); [Li et al., 2024c](#)), spatial or temporal attention mechanisms ([Yang et al., 2024b](#); [Jeong et al., 2024](#); [Wei et al., 2024](#)), and feature injection strategies ([Li et al., 2024f](#)) to guide the generation of controllable

dynamics. However, these approaches typically lack physics-based supervision, which can result in animations that violate physical laws or fail to align with user intentions (Meng et al., 2024). Recently, PhysGen (Liu et al., 2025) integrated a rigid-body physics simulator to generate physics-consistent dynamics for foreground objects in a given image. While effective, PhysGen is limited to 2D rigid-body motions, making it unsuitable for the fluid, elastic effects commonly seen in anime, such as the waving of cloth or hair, which do not conform to rigid-body dynamics. In this work, we address these limitations by leveraging deformable-body dynamics and a sketch-guided video diffusion model to create high-quality, physically consistent animation sequences that capture the fluidity and elasticity characteristic of anime.

5.1.3 Method

Given a reference anime illustration I_0 , our goal is to generate a sequence of stylized video frames $\{\hat{I}_1, \hat{I}_2, \dots, \hat{I}_T\}$ with natural, user-guided motions. To achieve this, we introduce PhysAnimator, a novel framework that combines video diffusion models with physics-based animation. We start by segmenting the objects of interest in a given anime illustration and creating a 2D triangulated mesh representation. Using this geometry, we obtain dynamics through image-space deformable body simulation, generating a sequence of optical flow fields $\{\mathcal{F}_{0 \rightarrow 1}, \mathcal{F}_{0 \rightarrow 2}, \dots, \mathcal{F}_{0 \rightarrow T}\}$. To offer user control, we enable interactive inputs through customizable energy strokes for defining external forces and rigging points for specifying desired trajectories. To render dynamics as high-quality frames, we extract the sketch from the input illustration and warp it using the computed optical flow fields, yielding a sequence of dynamic sketch frames $\{S_1, S_2, \dots, S_T\}$. These sketches, together with the input illustration, are fed into a video diffusion model with a sketch-guided ControlNet (Zhang et al., 2023d), synthesizing a vivid animated sequence. Finally, an optional data-driven cartoon interpolation model (Xing et al., 2024) can be applied to enhance the anime style dynamics of the results by selecting keyframes from the animated sequence as input, yielding the final output frames $\{\hat{I}_1, \hat{I}_2, \dots, \hat{I}_T\}$. An overview of our proposed framework is shown in Figure 1.5.

5.1.3.1 Preliminaries

Latent Video Diffusion Model: Most video diffusion models build on the latent diffusion model (LDM) framework (Rombach et al., 2022), which uses Variational Autoencoder (Kingma, 2013) (VAE) to map input images into a latent space. In this space, data is transformed into Gaussian noise via a forward diffusion process, which the model learns to reverse through denoising. In the forward diffusion process, the latent code z_0 is perturbed as:

$$z_t = \sqrt{\bar{\alpha}_t}z_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \epsilon \sim \mathcal{N}(0, I), \quad (5.1)$$

where $\bar{\alpha}_t = \prod_{i=1}^t (1 - \beta_t)$ with β_t controlling the noise strength coefficient at step t . The denoising model ϵ_θ is trained to recover z_t by minimizing the objective function:

$$L_\epsilon = \|\epsilon - \epsilon_\theta(z_t; c, t)\|_2^2, \quad (5.2)$$

where θ denotes the learnable network parameters and c represents conditioning input (e.g. text prompts or images). After denoising, the VAE decoder reconstructs the latent code back into the image space. The latent video diffusion model (LVDM) extends the image LDM to videos by incorporating temporal modules to maintain temporal consistency.

Deformable Body Dynamics: Mathematically, the dynamics of a continuum deformable body is described by a time-dependent continuous deformation map $\mathbf{x} = \phi(\mathbf{X}, t)$, which maps the undeformed material space Ω^0 to the deformed world space Ω^t at time t . The deformation gradient $\mathbf{F} = \frac{\partial \phi}{\partial \mathbf{X}}$ encodes the local deformation such as scaling, rotation, and shearing. In the context of a discretized setting, e.g. 2D meshes, the deformation map for each triangle can be expressed as

$$\phi_i(\mathbf{X}) = \mathbf{F}_i \mathbf{X} + \mathbf{b}_i, \quad (5.3)$$

where $\mathbf{b}_i \in \mathbb{R}^2$ accounts for translation of i -th triangle, and the deformation gradient $\mathbf{F}_i \in \mathbb{R}^{2 \times 2}$ is assumed to be constant (Li et al., 2024b). Undergoing deformation, the

deformed body aims to recover its rest shape via resisting forces. Continuum mechanics model this behavior by first defining an energy density function $\Psi(\mathbf{F}(\mathbf{x}))$, which measures the strain energy per unit undeformed volume. The total potential energy for a deformable body is then obtained as

$$E(\mathbf{x}) = \sum_{i=1}^N \Psi(\mathbf{F}_i) V_i, \quad (5.4)$$

where V_i denotes volume of i -th triangle. The internal resisting force is then defined as the negative gradient of the potential energy with respect to the vertex position

$$\mathbf{f}_{\text{int}}(\mathbf{x}) = -\frac{\partial E(\mathbf{x})}{\partial \mathbf{x}}. \quad (5.5)$$

The deformable body dynamics is governed by Newton’s Second Law as

$$\frac{d^2 \mathbf{x}}{dt^2} = \mathbf{M}^{-1}(\mathbf{f}_{\text{int}}(\mathbf{x}) + \mathbf{f}_{\text{ext}}(\mathbf{x})), \quad (5.6)$$

which can be solved numerically. Here M is the mass matrix that represents the masses of all vertices and $\mathbf{f}_{\text{ext}}$ refers to external forces, such as gravity.

5.1.3.2 Physics-Based Animation

To generate dynamic animations from a single illustration, a straightforward approach is to use existing controllable video diffusion models (Wu et al., 2025a; Shi et al., 2024). However, these purely data-driven methods often struggle due to a lack of physical understanding, leading to unrealistic results. To overcome this limitation, we incorporate physics-based animation to generate physically consistent and plausible motions.

Geometry Registration: The first step to animate objects of interest in a given illustration is to establish a geometric basis. While previous works (Hong et al., 2023; Tang et al., 2025) have explored predicting 3D geometry from single-view images, these methods are typically constrained to reconstructing real-world objects and do not generalize to

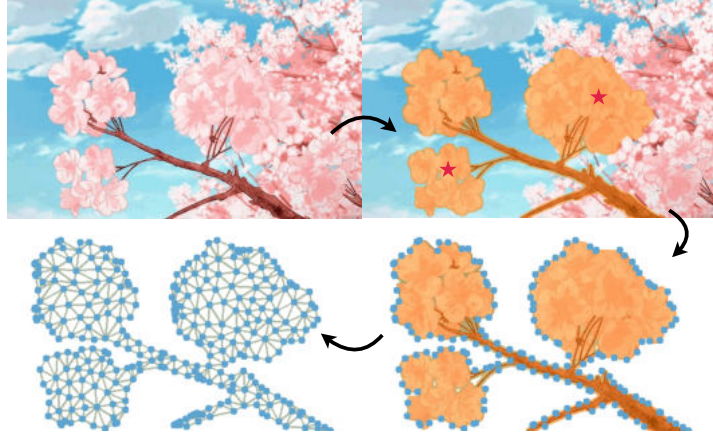


Figure 5.1: Geometry registration. SAM-based segmentation masks are used to sample boundary points, which are triangulated via conforming Delaunay triangulation into simulation-ready 2D meshes.

in-the-wild anime images, since characters and objects in anime often lack an inherent 3D representation due to their stylized and flat nature. As shown in prior works on cartoon animation (Xing et al., 2016; Kazi et al., 2016), leveraging 2D animation techniques proves effective for generating anime-style dynamic effects. Inspired by this, we focus on extracting 2D meshes for the objects of interest (Figure 5.1). To achieve this, we first utilize the Segment Anything Model (SAM) (Kirillov et al., 2023; Ravi et al., 2024) to obtain segmentation masks for each target object, guided by user-specified query points. Along the contours of each segmentation mask, we uniformly sample boundary points, defining the outline of the intended mesh. We then employ conforming Delaunay triangulation (Lee and Schachter, 1980) with these boundary constraints to generate well-structured triangular meshes, creating a simulation-ready format for subsequent animation.

Deformable Body Model: Anime scenes often feature dynamic environmental effects such as external forces that interact with characters’ clothing, hair, or other elements. Additionally, techniques like “squash” and “stretch” are commonly used to convey motion and energy, adding expressiveness to animated objects. Motivated by these stylized dynamics, we model anime objects as deformable bodies, allowing them to capture the fluidity and exaggerated motion characteristic of anime. A key property of deformable

bodies is their ability to change shape in response to external forces and attempt to return to their original rest shape upon deformation. To represent this physical behavior, we employ the Fixed Corotated constitutive model (Stomakhin et al., 2012), which defines the energy density as:

$$\Psi(\mathbf{F}) = \mu \|\mathbf{F} - \mathbf{R}\|_F^2 + \frac{\lambda}{2} (\det(\mathbf{F}) - 1)^2, \quad (5.7)$$

where μ, λ are the Lamé parameters, and $\mathbf{R}$ is the rotational part of $\mathbf{F}$ computed via polar decomposition. The first term models the stretching and compression resistance for the individual spatial directions and the second term describes resistance to volume change. The resulting internal forces $\mathbf{f}_{\text{int}}$ can then be derived via (5.5).

Interactive Animation: While $\mathbf{f}_{\text{int}}$ governs the inherent physical behavior, $\mathbf{f}_{\text{ext}}$ allows for user-defined interactions. Inspired by the concept of energy strokes (Gilland, 2012; Xing et al., 2016), we introduce customizable energy strokes that carry flow particles. These particles move along the user-specified strokes, propagating external forces to nearby vertices of the deformable mesh. This enables the creation of tailored animation effects, such as simulating wind interactions. Details of our flow particles can be found in the supplementary material. Additionally, we incorporate rigging point support, allowing animators to anchor specific regions or guide them along predefined trajectories, offering enhanced control and flexibility. Using the specified energy strokes and deformable modeling, we solve the motion equation (5.6) to evolve the dynamics. We employ the semi-implicit Euler method to compute the deformation map $\phi_i(X, t)$ at each time step t for every triangle. We refer to the supplementary materials for detailed steps on solving the motion equation. For each pixel $\mathbf{X}_p$ in the reference image, if it lies within a triangle $\mathcal{T}_i$, we assign the displacement vector as $\mathbf{d}(\mathbf{p}) = \phi_i(\mathbf{X}_p, t) - \mathbf{X}_p$; otherwise, we assign a zero vector. Collecting displacement vector $\mathbf{d}(\mathbf{p})$ of all pixels yields the optical flow $\mathcal{F}_{0 \rightarrow t}$, which defines the displacement fields for the reference illustration I_0 at time t .

5.1.3.3 Generative Rendering

In this section, we describe how to render the video sequence $\{\hat{I}_1, \hat{I}_2, \dots, \hat{I}_T\}$ given the reference image I_0 and the optical flow sequence $\{\mathcal{F}_{0 \rightarrow 1}, \mathcal{F}_{0 \rightarrow 2}, \dots, \mathcal{F}_{0 \rightarrow T}\}$ derived from simulation.

Sketch-Guided Rendering: While the video sequence can be directly generated by warping the reference image I_0 using optical flow fields, the resulting frames often exhibit black hole artifacts caused by occlusions. To address this problem, we extract the sketch of I_0 , denoted as S_0 , and obtain the sketch at time t as follows:

$$S_t = \mathcal{W}(S_0, \mathcal{F}_{0 \rightarrow t}, w_{0 \rightarrow t}), \quad (5.8)$$

where $\mathcal{W}$ represents the forward-warping operator, and $w_{0 \rightarrow t}$ denotes the warping weights for each pixel in I_0 . In forward warping, multiple pixels may map to the same 2D location in the output frame (Niklaus and Liu, 2020), potentially leading to artifacts or distortions if the weights $w_{0 \rightarrow t}$ are not properly defined. Inspired by (Li et al., 2024f), we set the pixel weight as $w_{0 \rightarrow t}(\mathbf{p}) = \|\mathcal{F}_{0 \rightarrow t}(\mathbf{p})\|_2$, giving higher importance to pixels with larger motion, typically corresponding to foreground objects.

Next, we leverage a video diffusion model to generate the rendered frames $\{I_1^r, I_2^r, \dots, I_T^r\}$ using the obtained sketch sequence $\{S_1, S_2, \dots, S_T\}$. The diffusion model uses the reference image I_0 as an input, and we employ a ControlNet (Zhang et al., 2023d) with the sketches as control signals, guiding the generation process to ensure that the results align with the sketch inputs. We observed that during the inference time, due to segmentation inaccuracies, the optical flow warping may introduce unintended distortions, or miss parts of the object contour, generating imperfect sketches. To address this, we apply Gaussian blur to the input sketches at both training and inference time, which smooths out inconsistencies. The video diffusion model is then capable of refining the results, leveraging its generative capabilities to handle imperfections and produce coherent outputs.

Complementary Dynamics: Unlike motions in the real world, dynamic effects in animation do not strictly adhere to physical laws and can not be fully captured by 2D animation methods. In the industrial animation pipeline, artists typically begin by creating a series of keyframes that define the primary motion trajectory, followed by drawing in-between frames to ensure smooth and fluid transitions. Inspired by this workflow, we leverage a data-driven module to enhance the physics-based animated results. Specifically, we select keyframes from the sketch-guided rendering results, forming the keyframe sequence $\{I_0^r, I_n^r, I_{2n}^r, \dots, I_{in}^r\}$, where n denotes the gap between keyframes. We then employ a cartoon interpolation video diffusion model (Xing et al., 2024), which synthesizes intermediate frames $\{\hat{I}_{in+1}, \hat{I}_{in+2}, \dots, \hat{I}_{i(n+1)-1}\}$ between each adjacent keyframe pair $(I_{in}^r, I_{i(n+1)}^r)$. For keyframes, we simply assign $\hat{I}_{in} = I_{in}^r$. This approach allows us to introduce expressive, data-driven complementary dynamics that go beyond what can be achieved through physics-based animation alone.

5.1.4 Experiments

In this section, we conduct a comprehensive comparison of our method against existing video diffusion models and demonstrate that our approach generates high-quality and physically plausible animations.

Implementation Details: We implement our deformable body simulator using Taichi (Hu et al., 2019). During interactive animation, users can adjust the Lamé parameters μ and λ to control the characteristics of the objects based on their specific needs. For generative rendering, we follow LVCD (Huang et al., 2024b) to train a sketch-guided ControlNet for the stable video diffusion model (Chai et al., 2023), using blurred sketches as control signals to address potential imperfections in the predicted sketch sequence. When utilizing ToonCrafter (Xing et al., 2024) for generating in-between frames, similar to its original implementation, we also train an additional sketch-guided ControlNet, but set the control scale to 0.1 during inference. We also set $n = 15$ for the in-betweening step. This configuration helps the generated frames follow the rough motion indicated by

the sketches while introducing extra stylized animation details.

Dataset: We build our training dataset using the Sakuga-42M Dataset (Pan et al., 2024), which consists of 1.4 million animation video clips. To ensure high-quality training samples, we filter out clips with fewer than 24 frames and dynamics scores outside the range of 0.05 to 0.7. Next, we extract sketches from the selected clips using the method from (Chan et al., 2022), resulting in a final dataset of 380,000 pairs of sketches and corresponding video sequences.

Baseline: We compare our method against two categories of video generation approaches. The first category includes state-of-the-art Image-to-Video (I2V) models, such as Cinemo (Ma et al., 2024) and DynamiCrafter (Xing et al., 2025), which take an input image and use text prompts to guide the motion dynamics of the generated videos. For these models, we generate prompts describing the image content and expected motion using ChatGPT-4V. The second category, including Motion-I2V (Shi et al., 2024) and Drag Anything (Wu et al., 2025a), usually trains an additional motion-control module for video generative models such that, given a single image input, a user-specified trajectory can be provided to control the movement of objects in the generated frames. To ensure a fair comparison, we extract trajectories from our animated results and use them as input for these methods.

5.1.4.1 Quantitative Evaluation

Given the absence of established benchmarks for anime-style image-to-video generation, we follow Motion-I2V (Shi et al., 2024) and construct a test set comprising 20 anime images with stylized dynamic elements such as swaying hair, clothing, and plants. For each method, we generate 10 video samples per image, resulting in a total of 200 videos per method. To quantitatively evaluate the generated results, we use the Fréchet Inception Distance (FID) (Heusel et al., 2017) to measure the similarity between the generated frames and the reference images. Additionally, we employ VideoScore (He et al., 2024b)

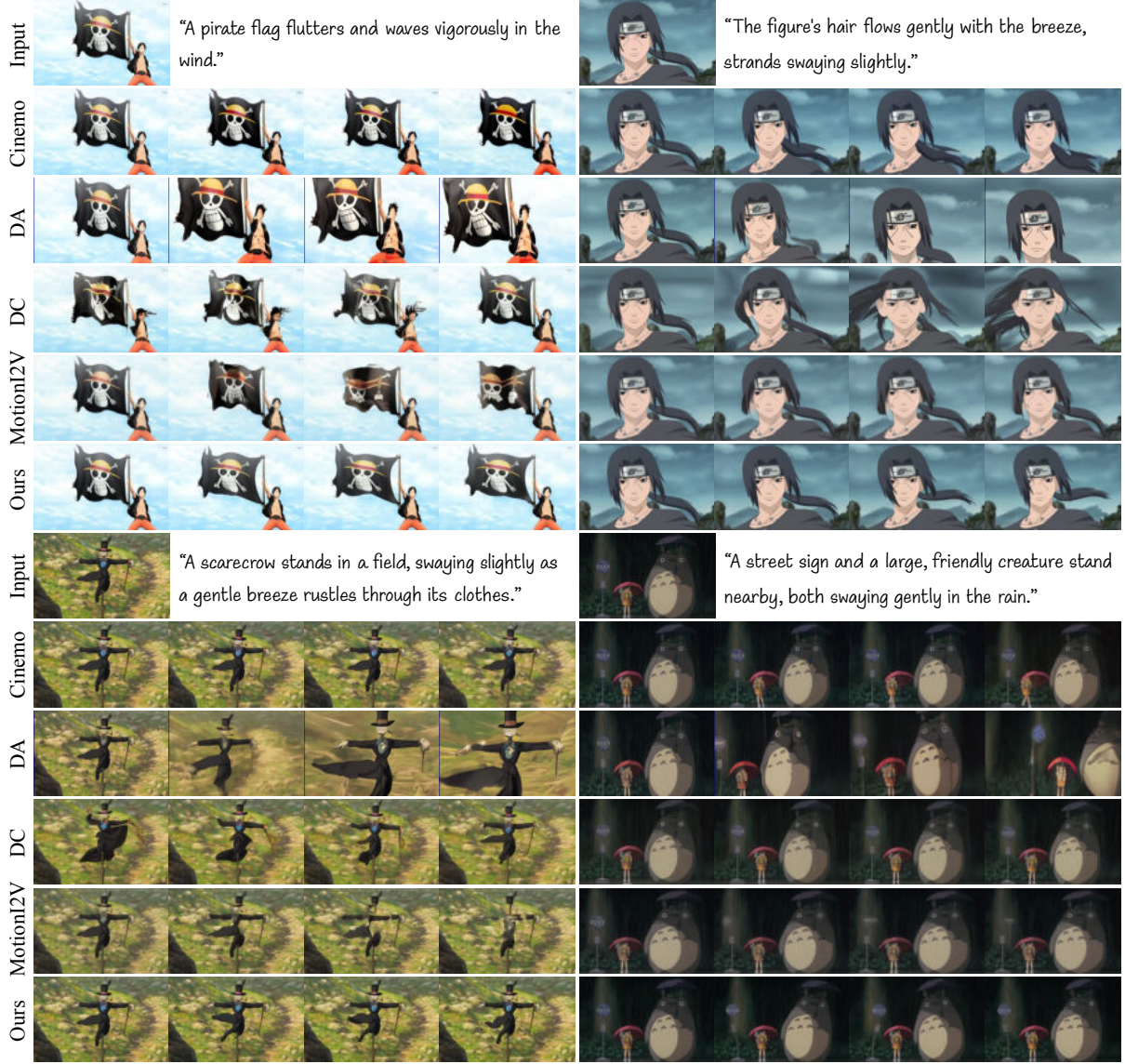


Figure 5.2: Qualitative Comparison. We compare our results against Cinemo (Ma et al., 2024), Drag Anything (Wu et al., 2025a), DynamiCrafter (Xing et al., 2025) and MotionI2V (Shi et al., 2024). Text prompts for Cinemo and DynamiCrafter are generated using ChatGPT-4V, while trajectories for Drag Anything and Motion-I2V are extracted from our animated results.

Methods	FID↓	VSVQ↑	VSTC↑	VSDD↑	VSFC↑
Cinemo	49.5	2.85	2.80	2.42	2.58
DragAnything	148.9	2.77	2.45	2.97	2.52
DynamiCrafter	94.9	2.78	2.68	2.53	2.51
Motion-I2V	121.8	2.70	2.50	2.66	2.39
Ours	90.4	2.89	2.86	2.48	2.64

Table 5.1: Quantitative Comparisons. We report FID to evaluate the similarity between the reference image and generated frames. VSVQ, VSTC, VSDD, and VSFC represent scores for visual quality, temporal consistency, dynamic degree, and factual consistency, respectively, as measured by VideoScore (He et al., 2024b).

to assess multiple aspects of video quality, including visual quality, temporal consistency, dynamic degree, and factual consistency. The factual consistency evaluates the consistency of the video content with common sense and factual knowledge. We exclude the text alignment score as our method and the trajectory-controlled approach do not involve text input.

The quantitative evaluation results, presented in Table 5.1, demonstrate that our method outperforms baseline approaches across most metrics, including visual quality, temporal consistency, and factual consistency, demonstrating the overall high quality of our generated videos. As shown in Figure 5.2, Cinema tends to generate static videos that closely resemble the reference image, leading to better FID scores. While the dynamic degree score for DragAnything is notably higher than other methods, this is due to its tendency to misinterpret motion control as camera movement, causing shifts in the entire image space and resulting in an inflated dynamics score. Although our method’s dynamic degree score is slightly lower than that of DynamiCrafter and Motion-I2V, these methods often exhibit implausible, distorted motions that create an illusion of increased dynamics. In contrast, our approach produces geometry-consistent motions and achieves high factual consistency, demonstrating the realistic motion dynamics enabled by our physics-based modeling. In addition, following (Kolkin et al., 2019; Geyer et al., 2023a), we conducted an user study adopting a two-alternative forced choice (2AFC) protocol, where participants are asked to choose the preferred video based on temporal consistency,

Methods	VQ	TC	MP	Overall
Cinemo	86%	83%	82%	81%
DragAnything	93%	91%	89%	91%
DynamiCrafter	84%	78%	76%	81%
Motion-I2V	95%	94%	97%	96%

Table 5.2: User Study. We show the user preference for our method over the baseline methods in terms of visual quality (VQ), temporal consistency (TC), motion plausibility (MP), and overall feeling.

visual quality, motion plausibility, and overall feeling given two videos (one from our method and one from baselines). The user preference results, presented in Table 5.2, show that our proposed method consistently outperforms the baselines across all evaluation criteria.

5.1.4.2 Qualitative Comparison

We present a qualitative comparison with baseline methods in Figure 5.2. Cinemo often produces static results with minimal motion dynamics. Drag Anything frequently misinterprets the motion trajectory as a camera movement, resulting in unintended dynamic sequences. DynamiCrafter, while generating larger motions, struggles to maintain the geometry of input objects, leading to noticeable distortions. Motion-I2V has difficulty preserving the appearance of the input content, often yielding unsatisfactory results. In contrast, our physics-guided approach ensures both physically plausible motions and high-quality rendering, preserving the geometry and visual consistency of the input.

5.1.4.3 Additional Qualitative Results

Sketch-Guided Rendering: After physics-based animation in our proposed framework, we obtain a sequence of optical flow $\{\mathcal{F}_{0 \rightarrow 1}, \mathcal{F}_{0 \rightarrow 2}, \dots, \mathcal{F}_{0 \rightarrow T}\}$. A straightforward approach would be to warp the input image I_0 using these optical flows and apply an inpainting algorithm to fill any occluded areas. However, this often results in suboptimal outputs. The main issue lies in segmentation inaccuracies, which can cause unintended regions to

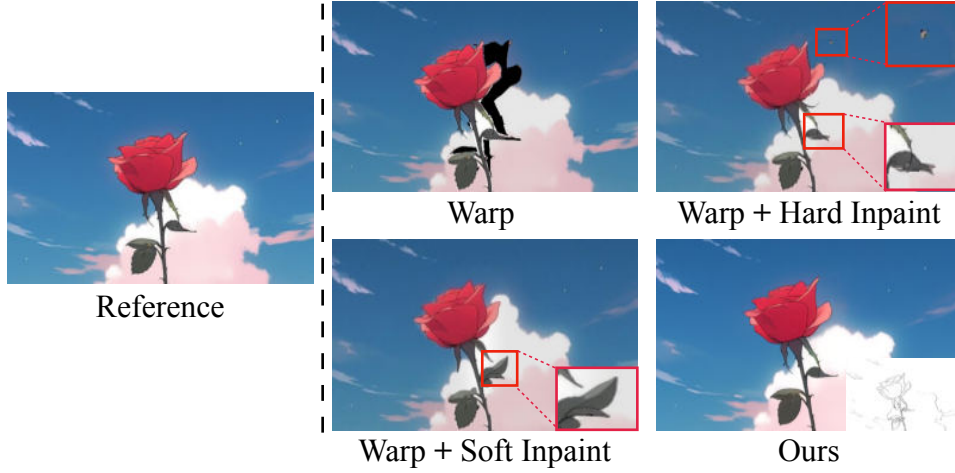


Figure 5.3: Sketch-Guided Rendering. Applying warping and inpainting introduces artifacts due to segmentation inaccuracy. Soft-inpainting (Levin and Fried, 2023) reduces these issues but can alter the content. Our sketch-guided rendering method produces high-quality results while preserving image details.

be warped or leave parts of the intended objects static, as shown in Figure 5.3. While introducing a soft mask (Levin and Fried, 2023) into the inpainting process can help smooth the transition between the newly generated region and the original figure, it may also alter the content, making it diverge from the appearance of the reference image. Additionally, applying inpainting frame by frame may also cause temporal consistency issues. To address this, our method employs a sketch-guided rendering module. We first extract a sparse geometric representation of the image and apply the animation dynamics directly to this sketch-based structure. This sparse representation is more robust to segmentation inaccuracies and helps preserve the intended motion. Our subsequent rendering module then synthesizes high-quality video sequences from the animated sketches, maintaining both temporal consistency and visual fidelity.

Complementary Dynamics Enhancement: While our physics-based animation ensures physically accurate and geometry-consistent motion, it is inherently limited to 2D representations and cannot fully capture 3D effects. Moreover, the exaggerated dynamics often seen in anime do not always conform to strict physical laws. To address these limitations, we incorporate a data-driven cartoon interpolation module (Xing et al., 2024)

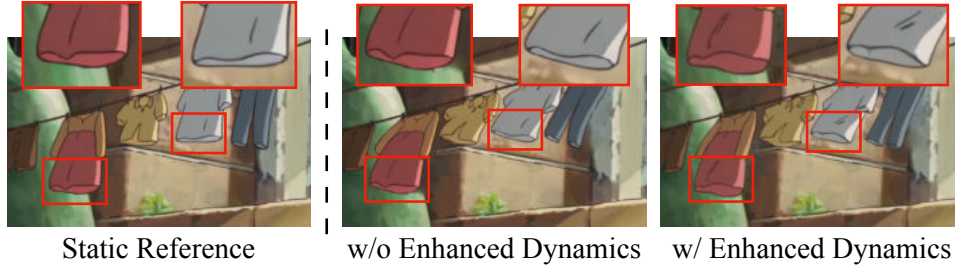


Figure 5.4: Complementary Dynamics Enhancement. While physics-based animation maintains geometric consistency, it may lack the fluidity and exaggeration commonly seen in anime. We employ a data-driven interpolation module to enhance the motion dynamics, creating more natural-looking animations that better resemble real anime.

to introduce complementary dynamics. As illustrated in Figure 5.4, this interpolation module enables fluid contour deformations of clothing during motion, rather than rigidly adhering to static geometry. It also dynamically generates new sketch lines, enhancing the expressiveness of the animation and bringing it closer to the aesthetic of traditional hand-drawn anime.

5.1.5 Conclusions

We presented PhysAnimator, a novel framework for generating dynamic and stylized animations from static anime illustrations by integrating physics-based simulations with data-driven generative models. Our approach generates physically plausible, fluid and exaggerated anime-styled motion through deformable body simulations, providing controllability to users through user-guided energy strokes. A sketch-guided video diffusion model ensures high-quality, temporally consistent frames, while a data-driven anime frame interpolation model adds expressive, non-physical dynamics. The experiments show that our proposed method outperforms existing video diffusion methods, offering a powerful tool for creating visually compelling and user-controllable anime-style animations.

5.2 AniMimic: Imitating 3D Animation From Video Priors

Creating realistic 3D animation remains a time-consuming and expertise-dependent process, requiring manual rigging, keyframing, and fine-tuning of complex motions. Meanwhile, video diffusion models have recently demonstrated remarkable motion imagination in 2D, generating dynamic and visually coherent motion from text or image prompts. However, their results lack explicit 3D structure and cannot be directly used for animation or simulation. We present AniMimic, a framework that animates static 3D meshes using motion priors learned from video diffusion models. Starting from an input mesh, AniMimic synthesizes a monocular animation video, automatically constructs a skeleton with skinning weights, and refines joint parameters through differentiable rendering and video-based supervision. To further enhance realism, we integrate a differentiable simulation module that refines mesh deformation through physically grounded soft-tissue dynamics. Our method bridges the creativity of video diffusion and the structural control of 3D rigged animation, producing physically plausible, temporally coherent, and artist-editable motion sequences that integrate seamlessly into standard animation pipelines.

5.2.1 Introduction

Creating expressive 3D animation has long been a cornerstone of computer graphics, powering lifelike digital humans, creatures, and virtual worlds. Despite decades of progress, producing high-quality animation remains an extraordinarily labor-intensive endeavor. Skilled artists must design skeletal rigs, keyframe complex motions, and manually refine subtle deformations to achieve believable results. This process demands both creative intuition and physical insight, making realistic animation difficult to automate or scale. Meanwhile, recent breakthroughs in video diffusion models have transformed motion synthesis in 2D, generating dynamic and stylistically rich sequences such as human performances (Guo et al., 2024; Zhu et al., 2024; Hu, 2024), cartoon actions (Xing et al., 2024; Li et al., 2025b), and cinematic camera movements (YU et al., 2025; Ren et al., 2025). These models exhibit remarkable motion imagination, yet their outputs reside

purely in the image plane—devoid of explicit 3D structure required for downstream applications such as rendering, simulation, or interactive editing.

To bridge this gap, emerging 4D generation approaches attempt to recover temporally coherent 3D motion from video diffusion priors. One line of work (Li et al., 2025c; Bahmani et al., 2024a,b) applies score-distillation sampling (SDS) (Poole et al., 2022) to directly optimize a 3D representation (e.g., NeRFs (Mildenhall et al., 2021) or Gaussian splats (Kerbl et al., 2023)) under video diffusion supervision. While conceptually elegant, these methods typically entangle geometry and appearance, leading to slow optimization, limited controllability, and difficulties integrating with standard animation pipelines. Another line (Jiang et al., 2023b; Wu et al., 2024c; Xie et al., 2024b) reconstructs motion from generated videos via implicit radiance-field or Gaussian representations. However, such representations are not readily compatible with modern computer graphics workflows and are difficult to edit or reuse, making integration into downstream applications cumbersome.

Observing this difficulty, we instead seek to animate an explicit 3D representation, removing the need for geometry reconstruction. Our goal thus becomes: *How can we animate an existing 3D model using the motion priors learned by video diffusion models?* Inspired by traditional animation workflows where artists begin by sketching a skeleton and iteratively refining the motion, we adopt a skeleton-based rigging animation model with a 3D mesh representation. Starting from a static 3D mesh, we first employ an automatic rigging module to extract a skeletal structure that drives mesh deformation. To provide motion guidance, we use differentiable rendering to project the mesh into 2D space and leverage animation videos generated by a video diffusion model for video-space supervision. Through gradient-based optimization, our method aligns the animated mesh poses with these motion cues, producing temporally coherent and editable animations.

To further enhance dynamic realism, we introduce a physics-aware motion-refinement module that models soft-tissue dynamics. By embedding a differentiable physics-based simulator into the animation loop, our method enforces physical plausibility and temporal smoothness while preserving the creative flexibility of data-driven motion synthesis. The resulting animations exhibit natural, physics-grounded dynamics and strong correspon-

dence to real-world motion. In summary, our main contributions include:

- A novel framework, AniMimic, that animates 3D meshes using motion priors distilled from video diffusion models.
- Differentiable rigging and pose optimization that align 3D skeletal motion with video-based cues through differentiable rendering and gradient-based optimization.
- A physics-based motion-refinement module seamlessly integrated into the optimization loop, enabling physically plausible and temporally coherent results.

5.2.2 Related Work

4D Generation: 4D generation, which incorporates the temporal dimension into 3D geometry, has become an active yet challenging area of research. The first line of research focuses on text-to-4D generation (Bahmani et al., 2024c; Xu et al., 2024a; Deng et al., 2025; Dai et al., 2025; Mou et al., 2024) which adopts diffusion-based generative models to produce reference images or videos, and employs SDS to optimize 3D geometry and motion. Another line of research focuses on image-to-4D generation (Song et al., 2023b; Yang et al., 2025; Sang et al., 2025; Li et al., 2024d; Zhou et al., 2025) and video-to-4D generation (Wang et al., 2024b; Zhu et al., 2025; Huang et al., 2025; Nag et al., 2025; Li et al., 2024e) which generates multiview video sequences for direct supervised learning, while others rely on skinning techniques such as Linear Blend Skinning (LBS) to deform 3D meshes and ensure consistent motion modeling. Multi-conditional 4D generation (Sun et al., 2025; Jiang et al., 2024b; Wu et al., 2025c; Zhao et al., 2025; Shao et al., 2024b) aims to enhance the fidelity and coherence of synthesized motion by integrating multiple modalities and diverse training strategies. However, the vast majority of 4D generation frameworks do not directly use high-quality 3D targets for generation due to their scarcity.

Recently, 3D-to-4D generation has attracted increasing attention by directly taking 3D objects as input and transforming static geometry into dynamic motion. HyperDiffusion (Erkoç et al., 2023) learns directly over neural field parameters to synthesize dynamic

meshes. ElastoGen (Feng et al., 2024b) models realistic elastic dynamics by using 3D CNN to refine deformation fields from static 3D models. Animate3D (Jiang et al., 2024b) animates static 3D models using a multi-view video diffusion model that synthesizes multi-view motion sequences. AKD (Li et al., 2025d) and Puppeteer (Song et al., 2025) both build a Skeleton-Rigging-Skinning pipeline to generate 4D animations. However, prior LBS-based methods that optimize joints via differentiable rendering yield quasi-static motion lacking physical realism. Our framework instead employs differentiable simulation to capture inertia and elasticity, producing physically consistent, realistic dynamics.

Skeleton Generation: Automatic skeleton generation is fundamental to enabling articulated 3D models to deform realistically. Conventional skeleton generation methods primarily rely on geometric analysis on mesh topology and spatial features to infer skeletal structures without the need for learning. Pinocchio (Baran and Popović, 2007) embeds skeletons into 3D meshes by optimizing medial-surface-based embeddings following a penalty-driven approach. Ju et al. (Ju et al., 2006) generates topological preserving skeletal graphs by applying iterative morphological thinning to volumetric models. However, conventional skeleton generation methods rely on geometric heuristics and lack semantic understanding and articulation awareness for realistic deformation.

Recent advances in deep learning have transformed skeleton generation from heuristic geometric analysis into a data-driven prediction problem, and a series of related works (Xu et al., 2020; Chu et al., 2024; Liao et al., 2024; Song et al., 2025) have recently emerged. RigNet (Xu et al., 2020) predicts both a skeleton structure and corresponding skinning weights directly from a 3D mesh using a deep neural architecture trained on a large dataset. HumanRig (Chu et al., 2024) applies a coarse-to-fine structure to predict and refine human’s skeletons and skinning weights. Puppeteer (Song et al., 2025) and UniRig (Zhang et al., 2025b) generate skeleton hierarchies and skinning weights using auto-regressive transformers with joint or tree structured tokenization for automatic rigging.

Physics-Based Animation: Physics-based animation aims to produce realistic motion by modeling underlying physical laws rather than relying solely on artist-defined key frames. Some of these works focus on interpolation for character animation (Zhan et al., 2024; Eisenberger et al., 2021; Zheng et al., 2022), aiming to generate smooth and natural transitions by blending existing 3D characters with different mesh and skeleton structures. NeuroMorph (Eisenberger et al., 2021) uses a graph-convolutional architecture to produce smooth shape interpolations and point-to-point correspondences between two 3D shapes. CharacterMixer (Zhan et al., 2024) builds a unified skeleton by interpolating bone distance fields to enable interpolation across diverse characters. Other works follow the idea of motion transfer across characters (Chen et al., 2025b; Li et al., 2022c; Zhang et al., 2024a), trying to transfer motions from one character to another while preserving motion style and physical constraints. Zhang et al. (Zhang et al., 2024a) leverages a vision-language model to align motions embeddings, enabling skeleton-aware and semantically consistent motion retargeting. Motion2Motion (Chen et al., 2025b) enables motion transfer across characters by matching and blending motion patches according to sparse joint correspondences. Another line of work focuses on motion generation and simulation (Yuan et al., 2023; Tevet et al., 2022; Zhang et al., 2022; Ren et al., 2023), such as PhysDiff (Yuan et al., 2023) which incorporates physical constraints into diffusion models to generate physically consistent human motions.

A closely related line of research focuses on incorporating complementary dynamics to enhance primary motion with secondary effects such as inertia, elasticity, and soft-body responses (Benchekroun et al., 2023; Bouaziz et al., 2014, 2023; Müller et al., 2007). Zhang et al. (Zhang et al., 2020b) proposes optimizing elastodynamic displacements in the subspace orthogonal to a rig’s controllable motions, thereby enriching rig animations with secondary physical effects without interfering with the original rig-driven motion. Building upon (Zhang et al., 2020b), Benchekroun et al. (Benchekroun et al., 2023) further accelerate complementary dynamics computation by parameterizing a skinning-based subspace with eigenmodes, enabling efficient, rotation-equivariant, and real-time simulation of secondary motion.

5.2.3 Method

We propose AniMimic, a novel framework that transforms a static textured mesh $\mathcal{S}$ into an animated sequence $\{\mathcal{S}^1, \mathcal{S}^2, \mathcal{S}^3, \dots, \mathcal{S}^T\}$. Given an input 3D mesh, either artist-created or generated by text/image-to-3D models, our proposed framework first renders the mesh into a canonical view image I and employs a video diffusion model to produce a monocular video sequence $\{I^1, I^2, I^3, \dots, I^T\}$ that captures rich motion priors. We then employ a feed-forward rigging network to automatically construct the skeleton with skinning weights, converting the mesh into an animation-ready representation. We further optimize the animation through a two-stage process: the first stage (Section 5.2.3.1) aims to optimize joint parameters, including rotations and translations, via differentiable rendering, tracking, and depth supervision to align the mesh motion with video-based cues; the second stage (Section 5.2.3.2) incorporates differentiable deformable-body simulation to refine the motion with physics-based deformation. This unified pipeline effectively bridges the creative motion imagination of video diffusion models with the structural controllability of 3D rigged mesh animation, producing dynamic, editable, and physically consistent motion sequences. Figure 1.6 provides an overview of our pipeline.

5.2.3.1 Rigging-Based Animation

To make the animation optimization stable and tractable, we represent mesh motion using a low-dimensional rigging formulation rather than directly optimizing all vertex positions. Directly manipulating vertex positions is computationally expensive and often leads to unstable or suboptimal results due to the high-dimensional search space. Inspired by traditional animation workflows, we model motion through a rigging system and optimize the joint parameters by aligning the animated mesh with a generated reference video.

Rigging and Skinning: A rigging system consists of a root joint J_0 and a hierarchy of joints $\{J_i\}_{i=1}^K$, each associated with a local transformation parameterized by a rotation $\mathbf{R}_i \in SO(3)$ and a translation $\mathbf{t}_i \in \mathbb{R}^3$. The global transformation of each non-root joint is

computed via forward kinematics (FK) (Denavit and Hartenberg, 1955), which recursively propagates local transformations along the kinematic chain:

$$\mathbf{T}_i = \mathbf{T}_{\text{parent}(i)} [\mathbf{R}_i \mid \mathbf{t}_i], \quad (5.9)$$

where $\mathbf{T}_i \in SE(3)$ denotes the global transformation of joint J_i . This hierarchical formulation ensures coherent motion propagation along the articulated structure.

To deform the surface mesh according to skeletal motion, we adopt Linear Blend Skinning (LBS) (Magenat-Thalmann et al., 1989), a widely used and efficient deformation model in computer animation. Each vertex with position $\mathbf{X}_i$ in the rest pose is influenced by a set of nearby joints through predefined skinning weights $\{w_{ik}\}_{k=1}^K$. The deformed vertex position $\mathbf{x}_i$ is given by:

$$\mathbf{x}_i = \sum_{k=1}^K w_{ik} \mathbf{T}_k \mathbf{X}_i, \quad \text{where} \quad \sum_k w_{ik} = 1. \quad (5.10)$$

The skinning weights determine the contribution of each joint to vertex deformation, producing smooth transitions between rigid parts while preserving local structure.

We use UniRig (Zhang et al., 2025c), a feed-forward network trained on large-scale rigged 3D models, to automatically predict both the skeletal hierarchy and skinning weights for arbitrary 3D meshes. We then parameterize the pose of the 3D model using a set of transformation parameters $\boldsymbol{\theta} = \{(\mathbf{r}_i, \mathbf{t}_i)\}_{i=0}^K$, where $\mathbf{r}_i$ is the 6D rotation representation (Zhou et al., 2019) and $\mathbf{t}_i$ is the translation vector for joint J_i . Each 6D rotation vector $\mathbf{r}_i$ can be mapped to a valid rotation matrix via the continuous 6D-to- $SO(3)$ conversion function $\mathbf{R}_i = \mathbf{R}(\mathbf{r}_i)$.

Joint Parameter Optimization: We aim to leverage the motion priors embedded in video foundation models to automatically animate a 3D mesh. To achieve this, we first render a reference image I of the input 3D model from a user-specified camera view. Given either a user-provided or VLM-generated text prompt, the reference image is then fed into

the video diffusion model to synthesize a sequence of frames $\{I^t\}_{t=1}^T = \{I^1, I^2, \dots, I^T\}$ as the image-space motion reference.

We then formulate an optimization problem to lift this 2D motion into 3D space by solving for the joint transformation parameters across all frames. Formally, we optimize:

$$\min L(\{\boldsymbol{\theta}^t\}_{t=1}^T; \mathcal{S}, \{I^t\}_{t=1}^T, \{J_i\}_{i=0}^K, w), \quad (5.11)$$

where $\boldsymbol{\theta}^t = \{(\mathbf{r}_i^t, \mathbf{t}_i^t)\}_{i=0}^K$ denotes the set of joint transformation parameters to be optimized at frame t . The overall objective function L comprises several components:

$$\begin{aligned} L = & \lambda_{\text{rgb}} L_{\text{rgb}} + \lambda_{\text{mask}} L_{\text{mask}} + \lambda_{\text{track}} L_{\text{track}} \\ & + \lambda_{\text{depth}} L_{\text{depth}} + \lambda_{\text{smooth}} L_{\text{smooth}} + \lambda_{\text{reg}} L_{\text{reg}} \end{aligned} \quad (5.12)$$

We detail the loss components below.

Differentiable Rendering Loss: The objective of this optimization is to align the rendered animation sequence with the reference video. To obtain accurate spatial supervision, we extract the ground-truth mask sequence $\{M^1, M^2, \dots, M^T\}$ using SAM2 (Ravi et al., 2024). We then employ differentiable SoftPhong and silhouette rendering from PyTorch3D (Ravi et al., 2020) to backpropagate gradients through the rendering process to optimize the joint parameters. We define two differentiable rendering losses:

$$L_{\text{rgb}} = \frac{1}{T} \sum_{t=1}^T \|\mathcal{R}^I(\mathcal{S}, \boldsymbol{\theta}^t) - I^t\|, \quad (5.13)$$

$$L_{\text{mask}} = \frac{1}{T} \sum_{t=1}^T \|\mathcal{R}^M(\mathcal{S}, \boldsymbol{\theta}^t) - M^t\|, \quad (5.14)$$

where $\mathcal{R}^I(\cdot)$ and $\mathcal{R}^M(\cdot)$ denote the differentiable SoftPhong and silhouette renderers, respectively.

Tracking Loss: The videos generated by video diffusion models can exhibit hallucinated regions and inconsistent appearances caused by lighting variations and changing shadows. To provide more stable image-space supervision, we sample a set of pixels $\mathcal{P}$ from the foreground region of the reference image I , where $\mathcal{P}_i$ denotes the coordinate of the i -th sampled pixel. Each pixel is then unprojected onto the 3D mesh to identify the corresponding triangle that contains it, and we record its barycentric weights β_i . We then employ AllTracker (Harley et al., 2025) to track the 2D trajectories of these pixels across the generated video, yielding the tracked pixel sets $\{\mathcal{P}^1, \mathcal{P}^2, \dots, \mathcal{P}^T\}$. The tracking loss is defined as:

$$L_{\text{track}} = \frac{1}{T} \sum_{t=1}^T \|\mathcal{B}(\mathcal{S}^t, \beta) - \mathcal{P}^t\|, \quad (5.15)$$

where $\mathcal{B}(\cdot)$ denotes the barycentric interpolation and camera projection function that maps 3D points on the mesh $\mathcal{S}^t$ to the corresponding 2D image-space coordinates.

Depth Loss: While differentiable rendering and pixel tracking provide effective image-space supervision, they offer limited constraint along the depth dimension. To address this, we incorporate additional depth supervision using VGGT (Wang et al., 2025a), which predicts per-frame depth maps from the generated video. For each tracked pixel, we query its corresponding depth value, obtaining a sequence of tracked-point depths $\{\mathcal{D}^1, \mathcal{D}^2, \dots, \mathcal{D}^T\}$. We then define the normalized depth loss as:

$$L_{\text{depth}} = \frac{1}{T} \sum_{t=1}^T \|\mathcal{Z}(\mathcal{S}^t, \beta) - \mathcal{N}(\mathcal{D}^t)\|, \quad (5.16)$$

where $\mathcal{Z}(\cdot)$ is similar to $\mathcal{B}(\cdot)$ but outputs the depth values of the corresponding 3D points in the camera space. A normalization operator $\mathcal{N}(\cdot)$ is applied to mitigate depth-scale ambiguity across frames.

Additional Loss: To encourage temporal coherence in the reconstructed animated mesh sequence, we introduce a smoothness loss L_{smooth} that penalizes abrupt motion changes across time. This loss incorporates both first- and second-order vertex differences

across consecutive frames to enforce stable and gradual transitions.

We further apply a regularization term to prevent excessive local transformations for non-root joints:

$$L_{\text{reg}} = \frac{1}{T} \sum_{t=1}^T \|\boldsymbol{\theta}^t - \text{clip}(\boldsymbol{\theta}^t, -\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\theta}})\|_2, \quad (5.17)$$

where $\hat{\boldsymbol{\theta}}$ is the threshold value for rotation and translation magnitudes, and $\text{clip}(\cdot)$ limits each transformation within the allowed range.

5.2.3.2 Physics-Based Dynamics

While the rigging-based animation offers a simple and compact parameterization for capturing motion sequences, it can suffer from surface artifacts caused by imperfect skinning weights predicted by automatic rigging networks. Moreover, the LBS-based deformation model is inherently quasistatic and therefore unable to capture dynamical responses such as inertia effects and elastic response. To overcome these limitations, we further refine the animation sequence using a differentiable physics-based soft-tissue simulation.

Differentiable Elastic Simulation: We model the motion as the dynamics of a continuum deformable body. Formally, the deformation is represented by a time-dependent map $\boldsymbol{x} = \phi(\boldsymbol{X}, t)$ that transforms material coordinates $\boldsymbol{X} \in \Omega^0$ in the undeformed rest space to world coordinates $\boldsymbol{x} \in \Omega^t$ in the deformed configuration at time t . For a discretized tetrahedral mesh, the deformation map of the i -th element can be expressed as $\phi_i(\boldsymbol{X}, t) = \boldsymbol{F}_i \boldsymbol{X} + \boldsymbol{b}_i$, where $\boldsymbol{F}_i \in \mathbb{R}^{3 \times 3}$ denotes the deformation gradient and $\boldsymbol{b}_i \in \mathbb{R}^3$ is the translation vector. The physics-based simulation seeks to solve the motion governed by Newton’s second law:

$$\frac{d^2 \boldsymbol{x}}{dt^2} = \boldsymbol{M}^{-1} \boldsymbol{f}(\boldsymbol{x}), \quad (5.18)$$

where $\boldsymbol{M}$ is the mass matrix and $\boldsymbol{f}$ represents the internal and external forces acting on the system.

To numerically integrate the motion equation (5.18), we employ the finite element method (FEM) and use the backward Euler scheme for numerical stability, yielding:

$$\begin{aligned}\mathbf{x}^{n+1} &= \mathbf{x}^n + \Delta t \mathbf{v}^{n+1}, \\ \mathbf{v}^{n+1} &= \mathbf{v}^n + \Delta t \mathbf{M}^{-1} \mathbf{f}(\mathbf{x}^{n+1}),\end{aligned}\tag{5.19}$$

where $\mathbf{x}^n, \mathbf{v}^n$ and $\mathbf{x}^{n+1}, \mathbf{v}^{n+1}$ denote the positions and velocities at time steps t^n and t^{n+1} , respectively. This nonlinear system can be formulated as an optimization-based time integration problem (Gast et al., 2015) to compute the next state:

$$\mathbf{x}^{n+1} = \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \tilde{\mathbf{x}}\|_{\mathbf{M}}^2 + \Psi(\mathbf{x}),\tag{5.20}$$

where $\tilde{\mathbf{x}} = \mathbf{x}^n + \mathbf{v}^n \Delta t + \mathbf{g} \Delta t^2$ is the predicted position, and $\Psi(\mathbf{x})$ denotes the total potential energy, including elastic and gravitational terms, such that $\mathbf{f} = -\frac{\partial \Psi}{\partial \mathbf{x}}$. We adopt the Fixed Corotated model (Stomakhin et al., 2012) to define the elasticity energy and solve the optimization using Newton’s method with line search for robust and stable convergence.

To enable end-to-end optimization, we follow (Li et al., 2025e) to achieve differentiability in simulation, leveraging both the adjoint method and automatic differentiation to backpropagate gradients through the time integration process (5.20). This allows the simulation to remain fully differentiable with respect to $\mathbf{x}^n$ and the material parameters.

Material Parameter Optimization: To integrate the differentiable elastic simulation into the animation optimization, we first convert the surface triangle mesh $\mathcal{S}$ into a volumetric tetrahedral mesh $\mathcal{S}^{\text{tet}}$ using TetWild (Hu et al., 2018b), making it suitable for finite element simulation. To stabilize training, for each joint J_i , we locate its nearest tetrahedron in $\mathcal{S}^{\text{tet}}$ and assign the corresponding global transformation $\mathbf{T}_i$ as the driving boundary condition for the simulation. Since TetWild does not guarantee that $\mathcal{S}$ and $\mathcal{S}^{\text{tet}}$ share identical surface topology, for each vertex on $\mathcal{S}$ we find its closest tetrahedron in $\mathcal{S}^{\text{tet}}$ and compute its barycentric weights, allowing the simulated deformation of $\mathcal{S}^{\text{tet}}$ to

drive the motion of $\mathcal{S}$.

To model the elastic stiffness of the body, we consider the Young’s modulus E_i of each tetrahedral element τ_i and optimize it so that the simulated deformation matches the reference image sequence $\{I^t\}_{t=1}^T$. This can be formulated as the following optimization problem:

$$\min_{\{E_i\}_{i=1}^N} L(\{E_i\}_{i=1}^N; \mathcal{S}^{\text{tet}}, \{I^t\}_{t=1}^T, \{\mathbf{T}_i\}_{i=0}^K). \quad (5.21)$$

Here we use the same optimization objective L as in [Section 5.2.3.1](#). $\mathbf{T}_i$ are the joint transformations serving as boundary conditions.

Directly optimizing E_i for every tetrahedral element can be inefficient and may lead to suboptimal results due to the high degrees of freedom. To accelerate convergence and enhance stability, we first cluster the tetrahedral elements based on their spatial proximity to each joint, assigning all elements within the same cluster a shared Young’s modulus during optimization. We then progressively subdivide these clusters to allow finer-grained updates, forming a coarse-to-fine optimization strategy. With the optimized material parameters, the simulation produces a physically realistic animation sequence $\{\mathcal{S}^t\}_{t=1}^T$.

5.2.4 Experiments

Datasets: To evaluate the effectiveness and generality of our framework, we curate a diverse collection of textured 3D models from multiple sources, including public datasets ([Xu et al., 2020](#)) and commercially available text/image-to-3D generative tools ([Meshy, 2025](#); [Tripo3D, 2025](#)). The resulting dataset spans a broad range of object categories, material appearances, and geometric complexities, covering both characters and various animal species.

Implementation Details: To generate the reference animation videos, we first render a single frame of the input 3D model and query an LLM to produce a descriptive motion prompt. This prompt, together with the rendered frame, is then fed into the Kling’s image-to-video model ([Technology, 2025](#)) to synthesize the reference animation sequences.

For differentiable deformable-body simulation, we implement our physics module using Warp (Macklin, 2022), which supports AutoDiff operators and is therefore well-suited for differentiable simulation. For both optimization stages, we use the Adam optimizer with a learning rate of 10^{-3} to update all parameters. In the simulation-based optimization, we additionally optimize the logarithm of Young’s modulus to improve stability and convergence.

Baselines: We compare our method with state-of-the-art 4D motion generation approaches, including SC4D (Wu et al., 2024c), DreamMesh4D (Li et al., 2024e), and Puppeteer (Song et al., 2025). DreamMesh4D and Puppeteer adopt mesh-based representations, while SC4D relies on Gaussian Splatting. Since the original DreamMesh4D requires estimating a 3D model from a single-view image, we use the ground-truth mesh as input to ensure a fair comparison.

5.2.4.1 Quantitative Evaluation

To quantitatively evaluate the generated results, we first measure how well the rendered video matches the input video. We compute SSIM (Wang et al., 2004), LPIPS (Zhang et al., 2018), and VBench (Huang et al., 2024a), which capture overall consistency (VBOC), aesthetic quality (VBAQ) and image quality (VBIQ). For each method, we generate 20 motion sequences and adopt a multi-view rendering protocol: in addition to the input camera view, we render two novel viewpoints by rotating the camera by $\pm 45^\circ$, enabling consistent multi-view comparison across all methods. Additionally, following (Geyer et al., 2023b), we conduct a user study using a two-alternative forced choice (2AFC) protocol, where participants are asked to choose the preferred video based on physical plausibility and overall satisfaction.

The quantitative results in Table 5.3 show that our method outperforms all baseline approaches in most metrics, including LPIPS, overall temporal consistency, and aesthetic quality, demonstrating the superior fidelity of our generated 4D motions. As illustrated in Figure 5.5, SC4D tends to produce coarse but roughly accurate motions, achieving

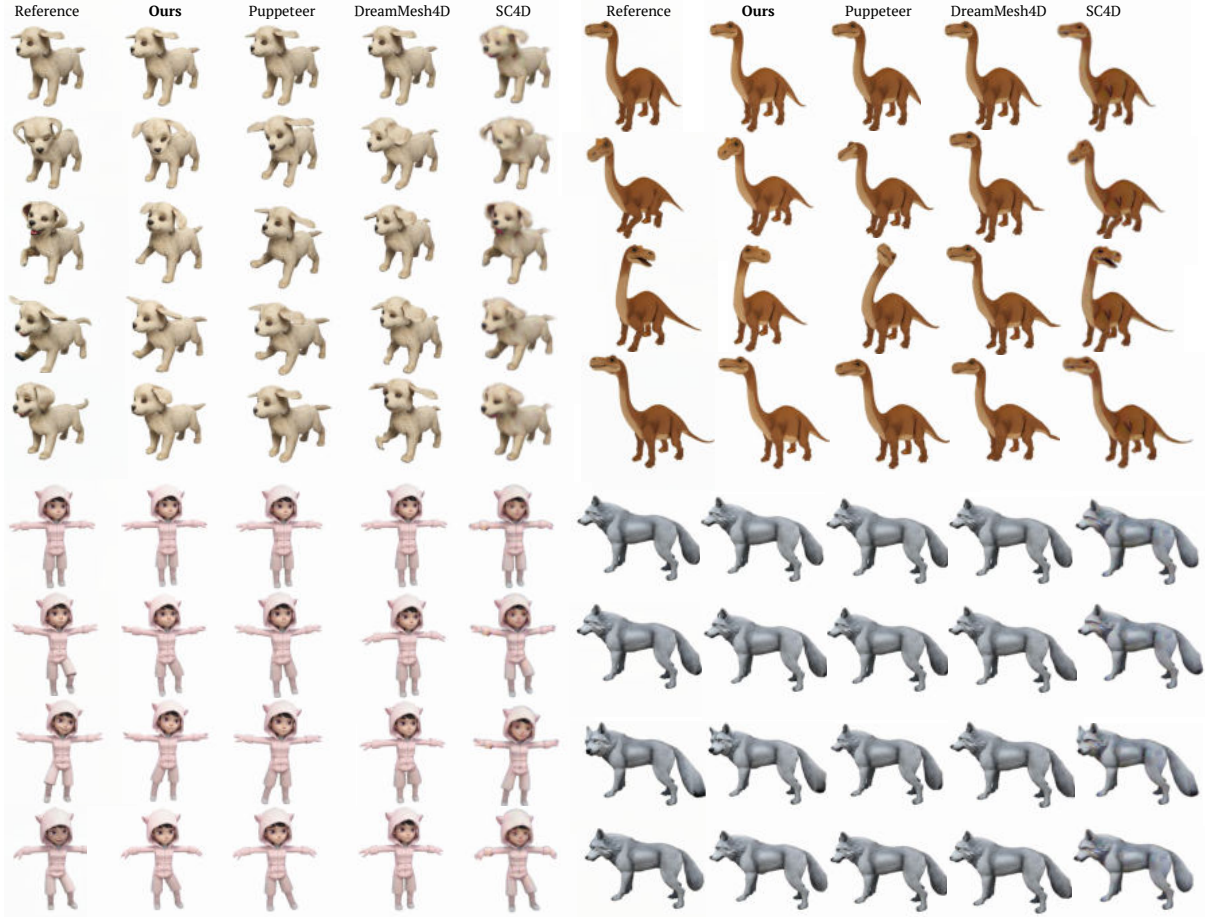


Figure 5.5: Qualitative Comparison. In comparison with SC4D (Wu et al., 2024c), DreamMesh4D (Li et al., 2024e), and Puppeteer (Song et al., 2025), our method yields more coherent motion trajectories and more accurately reflects the dynamics present in the reference videos.

high SSIM scores due to preserving low-frequency structure. However, it often generates incorrect textures and distorted geometry, resulting in poor temporal consistency and image quality scores. Puppeteer synthesizes visually sharp and high-quality frames, yet the generated motions are frequently unrealistic, leading to actions that are physically impossible and ultimately yielding lower LPIPS scores. DreamMesh4D maintains closer alignment with the input object’s appearance but is limited to small, near-rigid motions. The user preference results, presented in Table 5.4, show that our proposed method consistently outperforms the baselines across all evaluation criteria.

Methods	SSIM $\uparrow$	LPIPS $\downarrow$	VBAQ $\uparrow$	VBOC $\uparrow$	VBIQ $\uparrow$
SC4D	0.9403	0.0924	0.541	0.174	0.392
DreamMesh4D	0.8662	0.1482	0.543	0.175	0.550
Puppeteer	0.9023	0.1097	0.572	0.176	0.632
Ours	0.9318	0.0849	0.581	0.176	0.606

Table 5.3: Quantitative Comparisons. We report SSIM and LPIPS to measure the similarity between the generated 4D motion and the reference video. VBAQ, VBOC and VBIQ represent scores for aesthetic quality, overall consistency and imaging quality, as measured by VBench (Huang et al., 2024a).

Methods	VQ	TC	MP	Overall
SC4D	96%	93%	92%	91%
DreamMesh4D	88%	91%	95%	91%
Puppeteer	74%	78%	69%	71%

Table 5.4: User Study. We show the user preference for our method over the baseline methods in terms of visual quality (VQ), temporal consistency (TC), motion plausibility (MP), and overall feeling.

5.2.4.2 Qualitative Evaluation

Motion Reconstruction: We present qualitative comparisons with baseline methods in Figure 5.5. SC4D frequently produces unclear or unstable motion and, due to its point-based representation, struggles to maintain consistent topology, often resulting in holes or missing regions. DreamMesh4D tends to exhibit distorted geometry around articulated areas such as limbs and joints, due to the large number of independent control nodes to be optimized. Puppeteer generates temporally stable results, but its achievable motion range is noticeably limited. In contrast, our method produces physically plausible and visually coherent 4D motion while faithfully preserving the object’s geometry and appearance.

Novel-View Synthesis: Benefiting from our rigged generation framework, our method can synthesize novel-view renderings of the generated motions with strong cross-view consistency. As shown in Figure 5.6, Puppeteer struggles to maintain alignment between

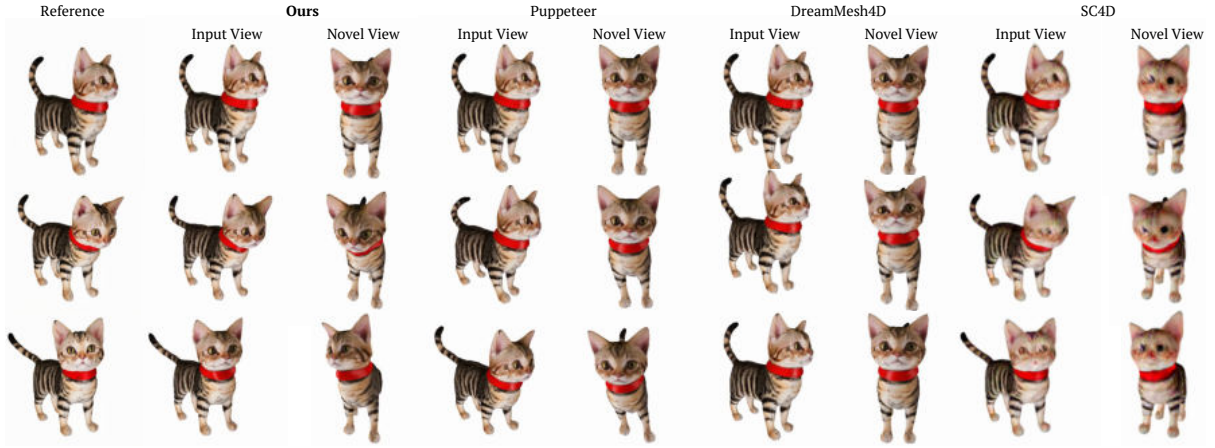


Figure 5.6: Novel View Synthesis. Our method closely aligns with the reference video in the input view and produces coherent novel views, whereas the baseline methods deviate from the ground truth.

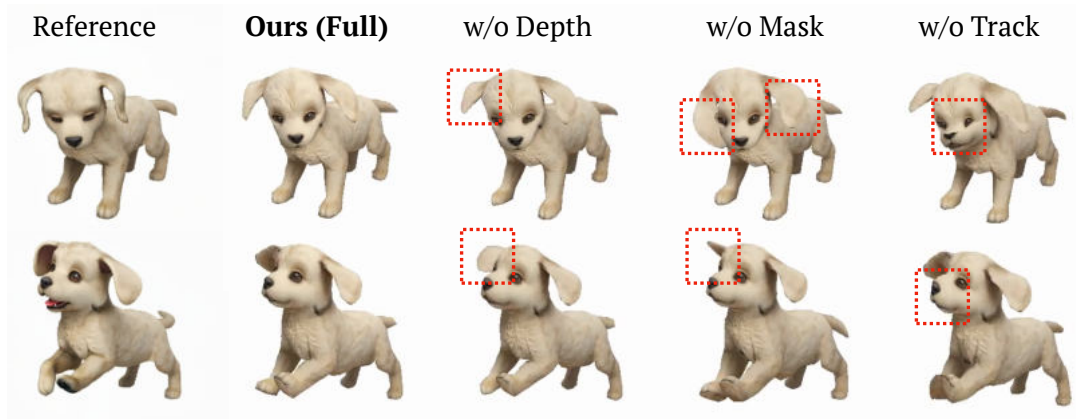


Figure 5.7: Ablation Studies. We conduct ablation studies on the proposed loss terms during optimization. Incorporating these terms leads to more plausible motion and enables the reconstructed dynamics to more faithfully adhere to the input video.

motion and viewpoint. DreamMesh4D better preserves view–motion alignment but exhibits only limited motion range. SC4D continues to suffer from low-quality textures. In contrast, our method generates coherent motions with consistent, high-quality appearance across novel views.

5.2.4.3 Ablation Study

Loss Components: To evaluate the efficacy of the proposed loss components, we conduct ablation studies on the loss terms (see Figure 5.7), including depth, mask, and

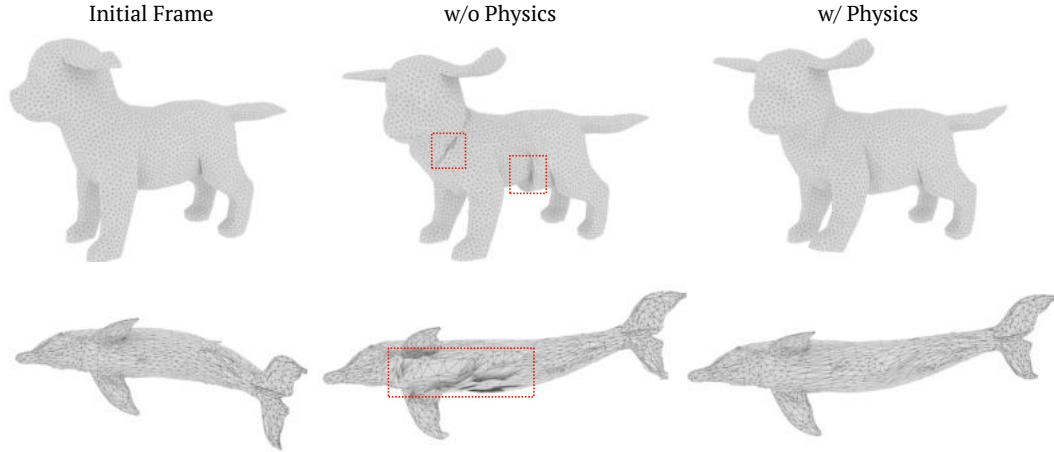


Figure 5.8: Physics-based Refinement. Rigged mesh surfaces often suffer from artifacts such as jittering or self-intersections (middle). Our physics-aware refinement stage eliminates such issues and enhances mesh smoothness and motion consistency.

point tracking. We observe that dropping any of these loss terms leads to distorted parts and deviations from the reference motion. Our full model exhibits consistent shape and motion, aligning well with the reference video.

Physics-based Refinement: We visualize mesh views of the animated results from our refinement stage in Figure 5.8. Rigging-based mesh deformation exhibits severe distortion in regions with large local deformation. Our differentiable simulation removes these surface-mesh artifacts from the rigged mesh, producing smooth geometry and motion.

5.2.5 Conclusions

We presented AniMimic, a framework for animating explicit 3D meshes using motion priors from video diffusion models. By combining differentiable rigging, video-space supervisory signals, and gradient-based pose optimization, our method transfers rich 2D motion cues into coherent 3D articulations without requiring full 4D reconstruction. A physics-aware refinement module further improves realism by enforcing temporal smoothness and physically plausible deformation. Together, these components enable controllable, editable, and simulation-ready 4D motion, offering a practical bridge between generative video models and modern animation workflows.

5.3 GRAIL: Generating Humanoid Loco-Manipulation From Video Foundation Models

Acquiring large-scale, physically plausible, and behaviorally diverse demonstration data remains a critical bottleneck for training humanoid loco-manipulation policies. Existing approaches, including teleoperation, motion capture, and in-the-wild video reconstruction, each face fundamental scalability limits. We present GRAIL, a scalable framework that generates diverse 4D human-object interaction (HOI) data for arbitrary objects by harnessing video foundation models (VFMs) and 3D object assets. Our key insight is that generating interaction videos from a fully specified 3D configuration, with known object geometry, a character model sized to the target robot, and camera parameters, makes the subsequent 4D reconstruction well-conditioned and ensures the synthesized interactions are morphologically compatible with the robot, in contrast to prior methods that reconstruct from unconstrained footage. Given a 3D object, GRAIL renders an initial frame with a robot-proportioned character, feeds it into a VFM to synthesize a reference interaction video, and applies a holistic global optimization process to recover temporally consistent and detailed 4D HOI motions without any category-specific training. We further retarget the reconstructed motions to a humanoid robot and extend a humanoid whole-body control foundation model to learn from the retargeted motions and produce physically plausible loco-manipulation skills. Extensive experiments demonstrate that GRAIL substantially outperforms existing methods in motion quality and physical plausibility, yielding a large-scale dataset of diverse 4D HOI motions and humanoid loco-manipulation sequences for skill learning.

5.3.1 Introduction

Training humanoid robots for diverse loco-manipulation, such as carrying a box across a room or operating a tool at a workbench, requires large-scale demonstration data that is physically plausible and spans the full spectrum from whole-body locomotion to fine-grained hand-object interactions. Yet acquiring such data at scale remains a

fundamental bottleneck.

The most direct data-collection approaches each face fundamental scalability limits. Robot teleoperation (Ben et al., 2025; Aldaco et al., 2024; Ze et al., 2025c,b; Khazatsky et al., 2024) is limited to a single robot morphology and focuses predominantly on tabletop manipulation. Marker-based motion capture (Lu et al., 2025; Taheri et al., 2020) yields high-quality trajectories but requires expensive, controlled studio setups that severely limit object and scenario diversity. Reconstructing human-object interactions from in-the-wild video sidesteps these constraints in principle, but jointly estimating camera parameters, object geometry, human body shape, and world-space motion from monocular footage is severely underconstrained, yielding temporally unstable and physically implausible trajectories. Learning-based reconstruction methods (Xie et al., 2024a, 2025c) reduce reconstruction ambiguity but remain category-specific and do not generalize to unseen objects.

Video foundation models (VFMs) offer a compelling path forward. Trained on internet-scale data, VFMs encode rich priors about how humans approach, grasp, and manipulate diverse objects. Several works have leveraged VFMs for robot policy learning and planning (Du et al., 2023; Ni et al., 2025; Chen et al., 2025a), while others generate interaction videos with VFMs and reconstruct 3D trajectories from the generated footage (Kim et al., 2025; Lou et al., 2025). However, these pipelines do not encode the underlying 3D scene parameters (camera, human proportions, and object geometry), leaving the downstream 4D reconstruction as ill-posed as monocular reconstruction from unconstrained footage. This fundamental limitation leads to inaccurate motion recovery, contact artifacts, and poor physical plausibility.

We present GRAIL, a scalable framework for generating physically plausible 4D human-object interaction data across diverse objects. Our key insight is that *generating interaction videos from a fully specified 3D configuration, in which the character is sized to the target robot, makes the subsequent 4D reconstruction well-conditioned and the synthesized interactions morphologically compatible with the robot*, in contrast to prior methods that reconstruct from unconstrained footage. To overcome the ill-posed nature

of prior VFM-based pipelines, GRAIL renders an initial frame from a fully specified 3D scene, where object geometry, a character model sized to the target robot, and camera parameters are all known by construction, and feeds it into a VFM to synthesize a reference interaction video suitable for robot policy learning. To recover temporally consistent 4D motions, we apply model-based 6-DoF object tracking and video-based human motion estimation, followed by a holistic global optimization that jointly refines human and object trajectories, recovering whole-body locomotion, detailed finger articulation, and object dynamics.

Beyond the purely kinematic outputs of prior work (Kim et al., 2025; Li et al., 2026), we retarget the reconstructed motions to a target humanoid (e.g., Unitree G1) and propose a latent adaptor policy that extends a whole-body control foundation model (Luo et al., 2025) with loco-manipulation skills and scene-aware locomotion capabilities via reinforcement learning in physics simulation. With GRAIL, we generate a large-scale dataset of over 20,000 diverse 4D HOI sequences spanning pick-up tasks, terrain motions, sitting motions, and complex object manipulations.

Our main contributions are as follows:

- We propose GRAIL, a scalable pipeline that generates diverse 4D HOI data for arbitrary objects by combining VFM priors with fully specified 3D configurations, transforming the ill-posed 4D reconstruction problem into a well-conditioned one without any category-specific training.
- We introduce an interaction-aware global optimization that jointly refines human and object trajectories by exploiting the known 3D configuration, achieving the best overall reconstruction quality across various metrics and significantly improving the tracking success rate from state-of-the-art baselines (89% vs 24%).
- We propose a latent adaptor policy learning approach that enables humanoid whole-body control foundation models to learn loco-manipulation skills and scene-aware humanoid control.

- We present a large-scale dataset of over 20,000 diverse 4D HOI sequences and demonstrate their utility for humanoid skill learning.

5.3.2 Related Work

Human-Object Interaction Generation: Synthesizing human-object interactions (HOI) requires reasoning about human motion, object affordances, and physical constraints. Traditional approaches rely on motion capture (Taheri et al., 2020; Bhatnagar et al., 2022; Jiang et al., 2023a; Huang et al., 2022a; Zhang et al., 2023c; Fan et al., 2023; Li et al., 2023b; Zhao et al., 2024; Kim et al., 2024b; Lu et al., 2025) or RGB-based reconstruction (Zhang et al., 2020c; Xie et al., 2022a; Wang et al., 2022; Petrov et al., 2023; Hou et al., 2023; Kim et al., 2023), but remain expensive, category-limited, or restricted to static geometry. Even recent category-agnostic methods (Xie et al., 2024a, 2025c, 2026) show that monocular HOI reconstruction remains severely underconstrained. Learning-based methods synthesize HOI using affordance-based priors (Li et al., 2023g; Ye et al., 2023c; Zheng et al., 2023a; Zhou et al., 2022; Zhang et al., 2023a; Kulkarni et al., 2024; Diller and Dai, 2024; Li et al., 2024a; Peng et al., 2025; Xu et al., 2023b; Jiang et al., 2024a; Dang et al., 2025) or vision-language priors (Zhang et al., 2025a; Xu et al., 2024c; Dwivedi et al., 2025; Li and Dai, 2024; Kim et al., 2024a; Wu et al., 2025b), but ensuring physical realism and temporal coherence remains challenging. Video foundation models (VFM) have emerged as powerful priors for scalable 4D HOI generation. DAViD (Kim et al., 2025) conditions a VFM on 2D edge renderings and reconstructs 4D HOI via PnP-based tracking; (Lou et al., 2025) generates HOI videos conditioned on human motions and recovers object trajectories through differentiable rendering; ZeroHSI (Li et al., 2026) reconstructs 4D HOI with animatable human models and known object geometry via differentiable rendering and 3D Gaussian splatting. Despite these advances, all prior methods generate and reconstruct without fully specifying the 3D scene (camera, character proportions, and object geometry), leaving the 4D reconstruction as ill-posed as that from unconstrained in-the-wild footage. In contrast, GRAIL renders the conditioning frame from a fully specified 3D scene and exploits these constraints in a holistic global

optimization to make the reconstruction well-conditioned.

Physics-based Motion Tracking: Physics-based tracking produces realistic human-object interactions by leveraging motion imitation within physics simulation (Peng et al., 2018; Liu and Hodgins, 2017; Starke et al., 2020, 2022; Liu and Hodgins, 2018; Zhang et al., 2023b; Haarnoja et al., 2024; Cui et al., 2024; Tevet et al., 2025; Luo et al., 2023; Peng et al., 2022a; Tessler et al., 2023; Luo et al., 2024; Wang et al., 2024c; Hassan et al., 2023; Peng et al., 2021b; Gao et al., 2024; Xu et al., 2025). These methods typically require clean motion capture data (Taheri et al., 2020; Bhatnagar et al., 2022) and train policies from scratch for each interaction, limiting scalability. They also often fail on noisy or physically implausible reference motions, as is common with reconstructions from RGB videos. In contrast, GRAIL extends a pre-trained whole-body control foundation model (Luo et al., 2025) with a lightweight latent adaptor that adds manipulation capability while preserving the learned locomotion prior, enabling scalable skill learning from VFM-reconstructed motions with interaction-aware rewards.

Humanoid Control and Loco-Manipulation: Recent advances in physics-based humanoid control (Peng et al., 2018, 2021b; Luo et al., 2023), whole-body imitation learning (He et al., 2024a, 2025; Fu et al., 2024; Luo et al., 2025), and unified multimodal controllers (He et al., 2026; Jiang et al., 2026) have brought end-to-end humanoid loco-manipulation within reach, making scalable, high-quality demonstration data the key bottleneck for general-purpose policies. Existing data collection approaches each face fundamental scalability limits. Teleoperation systems (Ze et al., 2025b,c; Ben et al., 2025; Khazatsky et al., 2024) provide high-quality demonstrations but are tied to a fixed morphology and require substantial per-task effort. Motion capture (Lu et al., 2025; Taheri et al., 2020) and wearable interfaces (Nai et al., 2026) collect accurate whole-body trajectories but require specialized setups and a separate retargeting step. Egocentric and in-the-wild approaches (Shi et al., 2026; Yu et al., 2025) mine video data but are limited by ill-posed monocular reconstruction. Recent work (Patel et al., 2025) explores generated

videos as robot demonstrations but represents the agent as a generic human body, creating a morphological mismatch. In contrast, GRAIL generates physically plausible 4D HOI data by rendering from a fully specified 3D scene in which the character already matches the target robot’s morphology, mitigating both the ill-posed reconstruction problem and the retargeting gap while scaling to diverse objects and tasks.

5.3.3 Method

Given a 3D object asset $\mathcal{M}^{\mathcal{O}}$, GRAIL synthesizes physically plausible humanoid loco-manipulation trajectories, comprising humanoid kinematic motion $\{\Theta_t^{\mathcal{R}}\}_{t=1}^T$, actions $\{\mathbf{a}_t^{\mathcal{R}}\}_{t=1}^T$, and object motion $\{\Theta_t^{\mathcal{O}}\}_{t=1}^T$. The pipeline proceeds in three stages (Figure 1.7). First, we assemble a fully specified 3D configuration (object, character, camera, and environment), render an initial frame, and feed it into a VFM to synthesize a reference HOI video $\{I_t\}_{t=1}^T$ (Section 5.3.3.1). Second, leveraging the known 3D configuration, we reconstruct coherent 4D human-object interaction (HOI) trajectories $\{(\tilde{\Theta}_t^{\mathcal{H}}, \tilde{\Theta}_t^{\mathcal{O}})\}_{t=1}^T$ through a holistic global optimization framework (Section 5.3.3.3). Third, we retarget the reconstructed HOI motions to a target humanoid (e.g., Unitree G1) and learn physically plausible loco-manipulation skills via interaction-aware physics-based tracking in simulation by extending a whole-body control foundation model (Section 5.3.3.4).

5.3.3.1 Robot-Centric Human Video Generation

Although it is tempting to generate robotic manipulation videos directly, VFMs have far richer priors about human manipulation, and mature reconstruction tools exist for human bodies and hands but not for robotic embodiments. We therefore generate human interaction videos using a character asset sized to the target humanoid and retarget the reconstructed motions afterward. Rather than conditioning image generative models on control signals (Kim et al., 2025; Li and Dai, 2024) (which often degrade object texture fidelity and introduce depth-scale ambiguity), we render the initial frame from a fully specified 3D configuration that preserves exact object geometry, texture, and camera

parameters.

To assemble the 3D configuration, we construct candidate environments using Infinigen (Raistrick et al., 2023) and position the human asset in a rest pose alongside the object. We use rigid body simulation (Macklin et al., 2016) to settle the object into a stable, collision-free initial configuration $\Theta_1^{\mathcal{O}}$. We then render the first frame using Blender with known camera intrinsics $\mathbf{C}_K \in \mathbb{R}^{3 \times 3}$ and extrinsics $\mathbf{C}_E = (\mathbf{r}^c, \mathbf{t}^c)$. The generated environment serves two purposes: it provides realistic visual context that improves VFM generation quality, and its known 3D geometry yields a ground-truth point cloud that enables metric-scale depth alignment during reconstruction (Section 5.3.3.3).

A VLM (OpenAI, 2024) generates the text prompt for the intended interaction from the rendered frame. The prompt and frame are passed to a VFM (e.g., Kling (Technology, 2025)), which synthesizes a reference HOI video $\{I_t\}_{t=1}^T$ capturing whole-body human locomotion, manipulation behaviors, and fine-grained hand-object interactions. We configure the video model for a static-camera setting, allowing us to exploit the known camera parameters $(\mathbf{C}_K, \mathbf{C}_E)$ during reconstruction. Since VFM generation is stochastic, we can synthesize an arbitrary number of diverse interactions per object, providing abundant demonstrations for downstream humanoid skill learning.

5.3.3.2 Initial Motion Estimation

We estimate initial human and object motion independently using off-the-shelf models, yielding world-space human motion $\{\hat{\Theta}_t^{\mathcal{H}}\}_{t=1}^T$ and object motion $\{\hat{\Theta}_t^{\mathcal{O}}\}_{t=1}^T$.

Human Motion Estimation: To recover the global human motion, we use GENMO (Li et al., 2025a) to estimate an initial sequence of SMPL-X (Pavlakos et al., 2019) parameters from the reference video $\{I_t\}_{t=1}^T$ in camera space. Since hand motion is critical for manipulation, we further refine hand poses with WiLoR (Potamias et al., 2025) to predict per-frame MANO hand parameters (Romero et al., 2017). We integrate the refined hand poses into the SMPL-X body via wrist IK alignment and rescale the prediction to match

the character asset’s known height. Using the known camera extrinsics $\mathbf{C}_E$, we transform the result to world space, obtaining the initial human motion $\{\hat{\Theta}_t^{\mathcal{H}}\}_{t=1}^T$.

Object Pose Tracking: To estimate the global object motion, we fine-tune FoundationPose (Wen et al., 2024) on its proposed synthetic dataset with RGB-only input, eliminating the need for depth. During inference, FoundationPose tracks the 6-DoF object pose over time, initialized from the first frame result $\Theta_1^{\mathcal{O}}$. Notably, accurate tracking with FoundationPose requires known object geometry, texture, and camera parameters, all of which are available by construction in our pipeline. We validate tracking quality by comparing SAM2 (Ravi et al., 2024) segmentation masks against rendered masks from the predicted poses, discarding sequences with inconsistent geometry.

5.3.3.3 Interaction-Aware Trajectory Refinement

Directly combining the independently reconstructed human and object trajectories often results in misaligned interactions. We therefore jointly refine both through a global optimization over all frames, keeping SMPL-X body shape and hand poses fixed for stability:

$$L = \lambda_{\text{kp}} L_{\text{kp}} + \lambda_{\text{proj}} L_{\text{proj}} + \lambda_{\text{depth}} L_{\text{depth}} + \lambda_{\text{cont}} L_{\text{cont}} + \lambda_{\text{reg}} L_{\text{reg}}. \quad (5.22)$$

The optimized trajectories yield the final 4D HOI reconstruction $\{(\tilde{\Theta}_t^{\mathcal{H}}, \tilde{\Theta}_t^{\mathcal{O}})\}_{t=1}^T$. We detail the key losses below. The regularization term L_{reg} is defined in [Appendix F](#).

Projection Alignment: To keep human and object projections consistent with the reference HOI video $\{I_t\}_{t=1}^T$, we impose image-space alignment constraints on both entities. For the human, we minimize the distance between projected and detected 2D keypoints via a keypoint loss:

$$L_{\text{kp}} = \frac{1}{T} \sum_{t=1}^T \|\mathcal{K}^{\mathcal{H}}(\Theta_t^{\mathcal{H}}) - p_t\|, \quad (5.23)$$

where $p_t \in \mathbb{R}^{J \times 3}$ are detected 2D keypoints from body and hand keypoint estimators (Xu et al., 2022b; Potamias et al., 2025), and $\mathcal{K}^{\mathcal{H}}(\cdot)$ computes the projected 2D keypoints based on the SMPL-X parameters. For the object, since FoundationPose (Wen et al., 2024) already yields well-aligned projections, we add a regularization term to preserve this alignment:

$$L_{\text{proj}} = \sum_{t=1}^T \left\| \mathcal{P}(V_t^{\mathcal{O}}) - \mathcal{P}(\hat{V}_t^{\mathcal{O}}) \right\|, \quad (5.24)$$

where $\mathcal{P}(\cdot)$ is the camera projection function, and $V^{\mathcal{O}}, \hat{V}^{\mathcal{O}}$ denote object vertices under the optimized and initial poses, respectively.

Depth Alignment: Leveraging the known 3D configuration, we estimate a depth map with MoGe-2 (Wang et al., 2025b) and align it to the ground-truth background depth rendered from the environment, obtaining metric-scale depth. We then segment human and object regions with SAM2 (Ravi et al., 2024) and unproject them using the known camera parameters to obtain per-frame point clouds $\mathbf{P}_t^{\mathcal{H}}$ and $\mathbf{P}_t^{\mathcal{O}}$. The depth alignment loss encourages the reconstructed meshes to match these point clouds:

$$L_{\text{depth}} = \frac{1}{T} \sum_{t=1}^T \mathcal{CD}(V_t^{\mathcal{H}_v}, \mathbf{P}_t^{\mathcal{H}}) + \mathcal{CD}(V_t^{\mathcal{O}_v}, \mathbf{P}_t^{\mathcal{O}}), \quad (5.25)$$

where $V_t^{\mathcal{H}_v}$ and $V_t^{\mathcal{O}_v}$ denote the visible vertices of the SMPL-X body and object mesh, and $\mathcal{CD}(\cdot, \cdot)$ is the bidirectional Chamfer distance.

Contact Alignment: To encourage physically plausible contact between the human and the object, we query an LLM (OpenAI, 2024) on uniformly sampled video frames to predict per-frame contact labels (e.g., left or right hand), propagating each label to its surrounding interval. Using the predicted labels, we identify the relevant SMPL-X vertices $V_t^{\mathcal{H}_c}$ via SMPL-X part segmentation and apply the contact loss only to frames where contact is detected. Since the image-space losses already enforce projection consistency, the contact loss only needs to resolve depth discrepancies. We therefore restrict it to object vertices whose projected positions overlap with the contact body region, penalizing

only their depth offset from the nearest neighbor:

$$L_{\text{cont}} = \frac{1}{|\mathcal{T}_c|} \sum_{t \in \mathcal{T}_c} \mathcal{CD}_z(V_t^{\mathcal{H}_C}, V_t^{\mathcal{O}_C}), \quad V_t^{\mathcal{O}_C} = \mathcal{F}(V_t^{\mathcal{O}}, V_t^{\mathcal{H}_C}), \quad (5.26)$$

where $\mathcal{T}_c$ denotes the set of frames where contact is detected. The filtering function $\mathcal{F}$ projects both $V_t^{\mathcal{O}}$ and $V_t^{\mathcal{H}_C}$ to screen space and selects the subset of object vertices $V_t^{\mathcal{O}_C} \subseteq V_t^{\mathcal{O}}$ whose projections fall within a distance threshold of those of the contact body vertices. $\mathcal{CD}_z(\cdot, \cdot)$ is a depth-only bidirectional Chamfer distance that penalizes only the positional difference along the viewing direction.

5.3.3.4 Humanoid Loco-Manipulation Skill Learning

This stage retargets the reconstructed motions to the target humanoid and learns physically plausible loco-manipulation skills via interaction-aware physics-based tracking in simulation. We propose a latent adaptor policy learning approach to endow SONIC (Luo et al., 2025), a whole-body humanoid control foundation model, with loco-manipulation ability. The learned policy produces the final humanoid kinematic motion $\{\Theta_t^{\mathcal{R}}\}_{t=1}^T$, actions $\{a_t^{\mathcal{R}}\}_{t=1}^T$, and object motion $\{\Theta_t^{\mathcal{O}}\}_{t=1}^T$.

Motion Retargeting: We retarget the reconstructed SMPL-X motions $\{\tilde{\Theta}_t^{\mathcal{H}}\}_{t=1}^T$ to the humanoid using GMR (Ze et al., 2025a), a general motion retargeting framework that transfers human motions across embodiments. This yields kinematic reference trajectories $\{\tilde{q}_t\}_{t=1}^T$ in the robot’s joint configuration space. The reconstructed object motion $\tilde{\Theta}_t^{\mathcal{O}} = (\tilde{p}_t^{\mathcal{O}}, \tilde{r}_t^{\mathcal{O}})$ provides the reference object position and orientation for the skill learning stage.

Whole-Body Control with Latent Adaptors: We build upon SONIC (Luo et al., 2025), a foundation model for humanoid whole-body control that operates through a learned latent motion space. SONIC encodes kinematic motion targets into a discrete latent token $z_t = \mathcal{E}(\tilde{q}_t)$ via finite scalar quantization and decodes them into joint-level

actions via $\mathcal{G}(\mathbf{z}_t)$. While SONIC excels at tracking whole-body locomotion, it does not account for object interactions or dexterous hand control.

To enable manipulation while preserving locomotion capabilities, we introduce a lightweight adaptor policy π_ϕ that modulates the latent space and directly controls the hands. The policy observes the robot’s proprioceptive state $\mathbf{s}_t$ along with object information $\mathbf{o}_t$. The proprioceptive state $\mathbf{s}_t$ includes joint positions, velocities, root angular velocity, gravity direction, and previous actions. The object observation $\mathbf{o}_t$ includes the current object pose relative to the robot, hand-to-object transforms, finger contact forces, and, critically, delta observations encoding the difference between the reference future object pose and the current simulated pose. The policy outputs a latent residual $\Delta\mathbf{z}_t \in \mathbb{R}^{64}$ along with hand primitive actions $\mathbf{a}_t^{\text{hand}} \in \mathbb{R}^2$:

$$(\Delta\mathbf{z}_t, \mathbf{a}_t^{\text{hand}}) = \pi_\phi(\mathbf{s}_t, \mathbf{o}_t). \quad (5.27)$$

The residual is added prior to quantization: $\tilde{\mathbf{z}}_t = \mathbf{z}_t + \lambda\Delta\mathbf{z}_t$, and decoded to produce whole-body joint targets $\mathbf{a}_t^{\text{body}} = \mathcal{G}(\tilde{\mathbf{z}}_t)$. Each hand primitive produces a binary open/close grasp signal, which is mapped to 7 finger joint positions per hand via predefined grasp configurations. An auxiliary ℓ_2 regularization $\mathcal{L}_{\text{aux}} = \beta\|\Delta\mathbf{z}_t\|^2$ encourages the adaptor to stay close to the pre-trained behavior.

Reward Design: We train the adaptor policy with three reward groups: $R_t = R_t^{\text{motion}} + R_t^{\text{obj}} + \mathbf{1}_{\text{contact}} \cdot R_t^{\text{grasp}}$, where $\mathbf{1}_{\text{contact}}$ is a contact indicator derived from the reconstructed trajectories by thresholding the hand-to-object distance, so that the grasp reward is active only during contact phases. The *motion tracking reward* encourages the simulated robot to follow the retargeted reference:

$$R_t^{\text{motion}} = \sum_i w_i \exp\left(-\frac{\|\tilde{\mathbf{x}}_{i,t} - \mathbf{x}_{i,t}\|^2}{\sigma_i^2}\right), \quad (5.28)$$

where $\tilde{\mathbf{x}}_{i,t}$ and $\mathbf{x}_{i,t}$ are reference and simulated quantities spanning root pose, per-body positions and orientations, and linear and angular velocities. We additionally boost the

tracking weight on the wrist links to encourage accurate hand placement. The *object tracking reward* penalizes deviation from the reference object pose:

$$R_t^{\text{obj}} = w_p \exp(-\alpha_p \|\tilde{\mathbf{p}}_t^{\mathcal{O}} - \mathbf{p}_t^{\mathcal{O}}\|_2) + w_r \exp(-\alpha_r \|\tilde{\mathbf{r}}_t^{\mathcal{O}} \ominus \mathbf{r}_t^{\mathcal{O}}\|_2), \quad (5.29)$$

where $\alpha_p, \alpha_r > 0$ are scaling coefficients. The *grasp reward* combines three terms per hand:

$$\begin{aligned} R_t^{\text{grasp}} = & \underbrace{w_c \min\left(\frac{N_t^{\text{contact}}}{N_{\min}}, 1\right)}_{\text{contact time}} + \underbrace{w_d [-\cos(\mathbf{d}_t^{\text{thumb}}, \mathbf{d}_t^{\text{index}})]^+}_{\text{grasp pose}} \\ & + \underbrace{w_f \exp\left(-\gamma \sum_j \|\mathbf{f}_{j,t} - \mathbf{c}_t\|\right)}_{\text{contact proximity}}, \end{aligned} \quad (5.30)$$

The first term rewards sustained finger contact with the object (saturating at $N_{\min}$ contacts), the second encourages the thumb and index finger to approach from opposing sides for a stable pinch grasp ($\mathbf{d}_t^{\text{thumb}}, \mathbf{d}_t^{\text{index}}$ are vectors from the object center to each fingertip), and the third draws all fingertips $\mathbf{f}_{j,t}$ toward the object contact centroid $\mathbf{c}_t$.

Training: We train π_ϕ with Proximal Policy Optimization (PPO) (Schulman et al., 2017) in Isaac Lab, keeping the pre-trained SONIC encoder and decoder frozen. Each motion sequence trains independently with 4,096 parallel environments. Domain randomization is disabled during manipulation training to isolate contact-sensitive behaviors. This lightweight design enables the adaptor to acquire dexterous manipulation behaviors without compromising the pre-trained locomotion quality.

Scene-Aware Humanoid Control: Beyond manipulation, our approach can also benefit other tasks such as scene-aware locomotion. The latent adaptor assumes that SONIC already possesses the necessary locomotion primitives and only needs to be steered for manipulation. However, for tasks involving scene-level interactions, such as navigating stairs, stepping over curbs, or sitting on furniture, SONIC’s flat-ground

prior is insufficient. By generating diverse scene-aware motion data through our pipeline (Section 5.3.3.1 through Section 5.3.3.3), we can use these reconstructed trajectories to finetune the foundation model itself, augmenting it with a local height map that provides 3D perception of the surrounding scene geometry. This extends SONIC into a scene-aware whole-body controller capable of adapting its locomotion to the environment. Since these tasks do not involve hand-object interaction, we train with only the motion tracking reward R_t^{motion} . Architectural details are provided in Appendix B.

5.3.4 Experiments

We evaluate GRAIL across two axes: (1) 4D HOI generation quality, compared against state-of-the-art baselines; (2) physics-based humanoid skill learning demonstrated on the Unitree G1. Implementation details are provided in Appendix B.

5.3.4.1 Scaling Humanoid Loco-Manipulation Data

To demonstrate the generality and scalability of GRAIL, we generate a large-scale loco-manipulation dataset for the Unitree G1. Object assets are sourced from RoboCasa (Nasiriany et al., 2024), ComAsset (Kim et al., 2024a), OMOMO (Li et al., 2023b), or generated from Hunyuan3D (Team, 2025). In total, we generate over 20,000 sequences across 1,000 objects and 1,000 terrain configurations, spanning various task categories (Figure 5.9).

Pick-up tasks require the robot to approach, grasp, and lift household objects. We cover both tabletop and ground scenarios, exercising diverse grasp strategies across varying object shapes and placement heights. As shown in Figure 5.9, the learned policies achieve accurate grasp poses tailored to each object’s geometry. *Dexterous manipulation* spans tabletop manipulation, where the robot performs precise grasping and handling of objects on a surface, and mobile manipulation, where the robot carries or moves larger items (e.g., boxes, carts, chairs) while walking. These tasks go beyond simple pick-up and highlight the ability of GRAIL to capture nuanced grasp strategies and coordinated whole-body



Figure 5.9: Diverse Loco-Manipulation Skills on the Unitree G1. We show representative results across pick-up (tabletop and ground), dexterous manipulation (tabletop and mobile), and generalized locomotion (sitting, curbs, slopes, and stairs).

motions from VFM-generated videos. *Sitting* involves approaching a piece of furniture, adjusting lower-body locomotion, and settling into a seated posture across diverse chair styles, demonstrating that the pipeline generalizes beyond hand-centric manipulation to full-body scene interactions. *Terrain traversal* targets robust locomotion over uneven surfaces, which is essential for real-world deployment but underrepresented in existing datasets. We procedurally generate environments featuring curbs, slopes, and stairs.

5.3.4.2 Comparison With HOI Generation Approaches

We compare the HOI generation component of GRAIL against training-based (CHOIS (Li et al., 2024a) and HOIDiff (Peng et al., 2025)) and training-free (DAViD (Kim et al., 2025)) 4D HOI generation approaches on a shared evaluation set of 20 everyday objects sourced from ComAsset (Kim et al., 2024a), OMOMO (Li et al., 2023b), and InterCap (Huang et al., 2022a).

Quantitative Comparison: We assess both geometric and perceptual aspects of the generated 4D HOI motions. Geometrically, we compute contact distance, defined as the average top- k vertex-to-vertex distance between the SMPL-X human surface and the object surface, and penetration ratio, which measures the percentage of SMPL-X vertices that interpenetrate the object mesh. For perceptual realism, we introduce an Interaction Score where a VLM (OpenAI, 2024) rates sampled keyframes (1-5) based on physical plausibility and affordance correctness. We additionally report motion smoothness for both the human and the object, computed as the second-order temporal derivative of their vertex trajectories. As shown in Table 5.5, our method achieves the strongest performance across nearly all metrics. The generated sequences exhibit natural and smooth motion, significantly reduced contact distance, and the lowest penetration rate among all methods. The high Interaction Score further demonstrates that our method better captures object affordances and everyday human-object interaction patterns.

Methods	Contact ↓	Penetration ↓	Inter. Score ↑	Human Smo. ↓	Obj Smo. ↓
HOIDiff	0.012	2.07%	1.79	0.0043	0.0118
CHOIS	0.034	3.74%	2.47	0.0055	0.0062
DAViD	0.246	1.46%	2.74	0.0024	0.0605
Ours	0.008	0.90%	3.58	0.0033	0.0022

Table 5.5: Quantitative Comparison of 4D HOI Generation. We compare against baseline methods on geometric quality (contact distance, penetration ratio), perceptual realism (Interaction Score, 1-5), and motion smoothness.

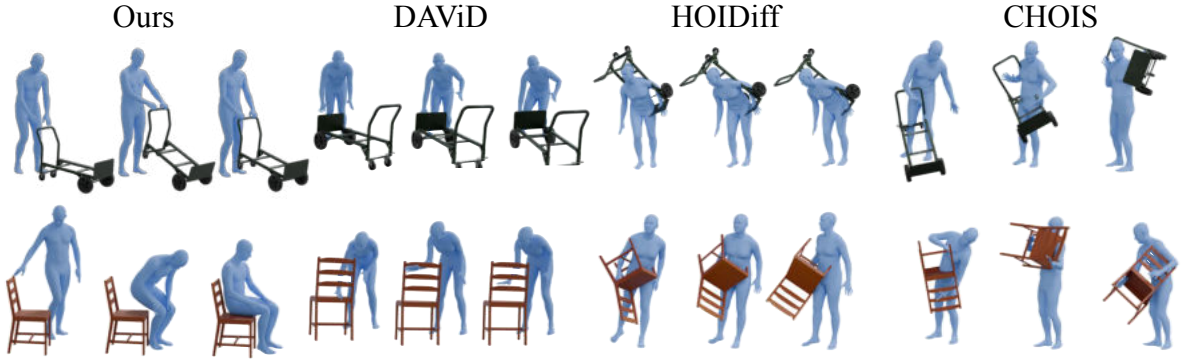


Figure 5.10: Qualitative Comparison of 4D HOI Generation. Our method produces more coherent motions with accurate contact and natural hand poses.

Qualitative Comparison: Figure 5.10 provides the visualized comparison results. Our approach produces more coherent and physically plausible motions. Notably, it generates natural hand poses, which is critical for downstream humanoid skill learning. In contrast, baseline methods often produce unrealistically flat or static hand configurations unsuitable for dexterous manipulation.

Physical Plausibility Evaluation: We further evaluate physical plausibility using InterMimic (Xu et al., 2025), a physics-based HOI tracking framework, which directly measures each method’s suitability for downstream humanoid skill learning. As shown in Table 5.6, GRAIL achieves an 89% tracking success rate, substantially exceeding all baselines, while maintaining the lowest trajectory deviation for the full body, hands, and object. These results confirm that our generated data is physically consistent and directly suitable for downstream imitation learning.

Methods	Succ. Rate $\uparrow$	Full-body Dev. $\downarrow$	Hand Dev. $\downarrow$	Obj Dev. $\downarrow$
HOIDiff	15.8%	0.2120	0.2239	0.3352
CHOIS	10.5%	0.2564	0.2626	0.3642
DAViD	24.0%	0.4723	0.5240	0.5826
Ours	88.9%	0.0913	0.0963	0.0851

Table 5.6: Physical Plausibility Comparison. We apply InterMimic (Xu et al., 2025) to reproduce 4D HOI sequences from each method in physics simulation, reporting tracking success rate and trajectory deviation for the full body, hands, and object.

Methods	Affordance Real.	Physical Real.
HOIDiff	2.0%	1.9%
CHOIS	12.2%	16.8%
DAViD	11.2%	10.4%
Ours	74.7%	70.9%

Table 5.7: User Study on 4D HOI Quality. Pick rates across affordance realism and physical realism. In each trial, participants select the best result from 3 of the 4 methods (theoretical upper bound 75%).

User Study: We conduct a user study with 30 participants to assess perceptual quality of the generated 4D HOI results. In each trial, participants view sequences from three of the four methods, drawn at random, and select the result with (1) the most physically plausible motion and (2) the most appropriate object affordances. As shown in Table 5.7, our method is consistently preferred, with pick rates of 74.7% for affordance realism and 70.9% for physical realism, confirming that the generated interactions are both physically convincing and semantically meaningful.

Methods	Contact $\downarrow$	Pen. $\downarrow$	MPJPE $\downarrow$	Human Smo. $\downarrow$	Obj Smo. $\downarrow$
w/o Projection Align.	0.016	1.93%	16.70	0.0023	0.0011
w/o Depth Align.	0.017	1.97%	4.35	0.0020	0.0011
w/o Contact Align.	0.024	1.52%	4.81	0.0022	0.0009
w/o Contact Filtering	0.014	2.18%	5.13	0.0023	0.0009
Ours (Full)	0.015	1.81%	4.89	0.0020	0.0009

Table 5.8: Ablation Study on Reconstruction Quality. Combining all loss components achieves the best overall balance across metrics.

5.3.4.3 Ablation Study

We ablate key design choices across two stages of GRAIL: interaction-aware trajectory refinement (Table 5.8) and physics-based humanoid skill learning (Table 5.9). Both studies are evaluated on the same benchmark of 124 motions across 43 objects. For reconstruction quality, we report contact distance, penetration ratio (Pen.), motion smoothness, and Mean Per Joint Position Error (MPJPE) in pixel space, which measures alignment between the reconstructed human motion and detected 2D keypoints. For skill learning, we report manipulation success rate (SR@ k : fraction of episodes where the average object position error falls below k cm), object position and orientation error (ObjPos/ObjOri), and local mean per-joint position error (MPJPE-L) of the humanoid.

Ablations for 4D HOI Reconstruction: As shown in Table 5.8, we ablate the three key losses in our interaction-aware trajectory refinement (projection alignment, depth alignment, and contact alignment) along with the 2D contact filtering strategy. Since all variants share the same generated videos, MPJPE isolates the effect of each loss on reconstruction fidelity. Removing projection alignment degrades image-space consistency, yielding the highest MPJPE (16.70). Without depth alignment, the coarse spatial relationship between human and object is lost, causing inaccurate contact estimation and severe penetration artifacts. Omitting contact alignment leads to substantially higher contact distance (0.024), as fine-grained hand-object interactions are no longer explicitly encouraged. Without contact filtering, a naive Chamfer distance loss constructs erroneous contact pairs, resulting in abrupt body deformations and incorrect human-object alignment. Combining all components achieves the best overall balance, as each targets a distinct failure mode.

Full-Pipeline Ablation on Skill Learning: As shown in Table 5.9, replacing the robot-proportioned character (matched to the G1’s 1.3 m height) with a standard 1.8 m human model yields the worst upstream quality (SR@20: 44.8%), confirming that morphological compatibility is critical for downstream humanoid skill learning. Among the reconstruction-

Method	SR@20 $\uparrow$	SR@10 $\uparrow$	ObjPos $\downarrow$	ObjOri $\downarrow$	MPJPE-L $\downarrow$
w/ Non Robot-prop. Char.	44.8%	34.0%	0.417	0.799	55.1
w/o Projection Align.	41.6%	29.6%	0.374	0.774	47.1
w/o Depth Align.	42.6%	34.6%	0.372	0.761	49.3
w/o Contact Align.	53.3%	42.9%	0.332	0.738	52.4
w/o Contact Filtering	51.1%	40.0%	0.336	0.757	51.6
w/o SONIC	45.0%	33.7%	0.395	0.766	243.5
w/o Residual (Vanilla SONIC)	39.7%	19.4%	0.303	0.570	37.1
w/o Delta Obs.	57.9%	41.0%	0.257	0.635	43.0
Ours (Full)	81.4%	74.3%	0.135	0.436	41.8

Table 5.9: Ablation Study on Humanoid Skill Learning. We ablate design choices in reconstruction losses, character proportions, and RL architecture, assessed by humanoid skill learning performance.

loss ablations, removing depth or projection alignment causes the largest drops (SR@20: 42.6% and 41.6%), while removing contact alignment or filtering also degrades performance to a lesser extent (SR@20: 53.3% and 51.1%), indicating that contact-aware losses are most critical for grasp quality. For the RL architecture, removing SONIC and training from scratch degrades body tracking and success rate significantly. Disabling the latent residual (Vanilla SONIC) yields the lowest manipulation success (SR@10: 19.4%) despite having the best body tracking performance. Finally, replacing delta observations with absolute positions also degrades performance, confirming the importance of each architectural component.

5.3.5 Discussion

Conclusions: We introduce GRAIL, a scalable framework for generating physically plausible 4D human-object interaction data for arbitrary objects. By rendering interaction videos from a fully specified 3D scene, in which the character is sized to the target robot and object geometry, camera, and scale are all known by construction, GRAIL transforms the ill-posed HOI reconstruction problem into a well-conditioned one and ensures the synthesized interactions are morphologically compatible with the target robot. A holistic global optimization then recovers temporally consistent full-body motion, detailed hand

articulation, and coherent object dynamics directly from the generated video. The resulting motions are retargeted to a target humanoid and refined via a lightweight latent adaptor that extends a whole-body control foundation model for loco-manipulation in physics simulation. Across geometric, perceptual, and physics-based evaluations, GRAIL substantially outperforms prior approaches and more faithfully captures object affordances and interaction patterns.

Limitations: Our pipeline requires some manual effort to set up the 3D scene (e.g., camera angle, character placement), and reconstruction quality degrades under severe occlusion. Future work can consider adopting multi-view video generation to reduce the occlusion ambiguity.

CHAPTER 6

Conclusions

In this dissertation, we explored a range of physics-informed methods for generative modeling and dynamics, addressing the fundamental disconnect between data-driven content synthesis and the physical laws that govern the real world. We developed techniques that embed physical constraints into generative pipelines, enable physically grounded 4D dynamics from real-world reconstructions, and scale physics-based simulation to humanoid loco-manipulation and large-scale interaction generation. Our key contributions are as follows:

Efficient and Robust Solid-Fluid Coupling (Chapter 2): We introduced a contact-proxy-based operator-splitting method for Lagrangian solid-fluid coupling that provides a penetration-free guarantee through a variational interior-point barrier formulation. By separating the simulation into distinct fluid and solid-coupling phases augmented by contact proxies, the method achieves significant computational speedups over monolithic approaches while accurately handling hyperelastic objects of varying geometries and codimensions. This work established the simulation foundations upon which subsequent chapters build.

Self-Supporting Text-to-3D Generation (Chapter 3): We introduced Atlas3D, a physics-aware refinement framework that embeds differentiable simulation constraints into text-to-3D optimization, ensuring that generated objects remain structurally stable under gravity, contact, and friction. By formulating physics-informed stability losses within Score Distillation Sampling pipelines, Atlas3D enforces physical plausibility without

sacrificing visual fidelity. It was validated through extensive generation tasks and real-world 3D-printed fabrication experiments.

Physics-Integrated 4D Generative Dynamics (Chapter 4): We presented Phys-Gaussian, which unifies visual and physical representations by integrating Newtonian dynamics into 3D Gaussian Splatting via a custom Material Point Method, realizing the principle of “what you see is what you simulate” (WS²). Building on this, VR-GS extends the framework to real-time interactive editing in virtual reality through a novel two-level embedding strategy, deformable body simulation, and dynamic shadow casting. Together, these works demonstrated that multi-view reconstructions can be directly endowed with physically grounded 4D motion for both offline synthesis and real-time interaction.

Physics-Guided Animation and Humanoid Loco-Manipulation (Chapter 5): We introduced three complementary contributions spanning 2D animation, 3D character dynamics, and 4D human-object interaction. PhysAnimator combines image-space deformable-body simulation with sketch-guided video diffusion to produce physically plausible cartoon animations from static anime illustrations. AnimaMimic closes the loop between video generation and physics by recovering material parameters and external forces from generated videos via differentiable simulation-based tracking, enabling faithful reproduction of complex 3D character dynamics. GRAIL scales this paradigm to humanoid loco-manipulation, generating over 20,000 diverse 4D human-object interaction sequences from video foundation models and learning physically plausible humanoid skills through latent adaptor policy learning in physics simulation.

APPENDIX A

Lagrangian Solid Fluid Coupling Technical Details

A.1 Derivatives of Fluid Potentials

The gradient and Hessian of the incompressibility potential with respect to the fluid particle position are

$$\begin{aligned}\frac{\partial P_I(\mathbf{x})}{\partial \mathbf{x}} &= \sum_i k_I V_0 (J_i - 1) J_i^n h \frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}}, \\ \frac{\partial^2 P_I(\mathbf{x})}{\partial \mathbf{x}^2} &= \sum_i k_I V_0 (J_i^n)^2 h^2 \frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}} \left(\frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}} \right)^T,\end{aligned}\tag{A.1}$$

where

$$\frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}_k} = \begin{cases} \sum_j -\frac{m_j}{h\rho_j^n} \nabla W_{ij} + \sum_b -\frac{m_b}{h\rho_b^n} \nabla_i W_{ib} & \text{for } k = i, \\ \frac{m_j}{h\rho_j^n} \nabla W_{ij} & \text{for } k = j, \\ 0 & \text{otherwise,} \end{cases}\tag{A.2}$$

where $\mathbf{x}_k$ denotes the position of particle k , and $\frac{\partial(\nabla \cdot \mathbf{v}_i^{n+1})}{\partial \mathbf{x}_k}$ can be nonzero if particle k is a neighbor of particle i or $k = i$. According to (A.1), the constant Hessian matrix is obviously positive semi-definite (PSD) since it is simply the sum of the outer product of a vector with positive coefficients.

Similarly, the gradient and Hessian of the viscosity potential with respect to the fluid

particle position are

$$\begin{aligned} \frac{\partial P_V(\mathbf{x})}{\partial \mathbf{x}_i} &= \nu \sum_j \mathbf{V}_{ij} \mathbf{v}_{ij}^{n+1}, \\ \frac{\partial^2 P_V(\mathbf{x})}{\partial \mathbf{x}_i \partial \mathbf{x}_k} &= \begin{cases} \frac{\nu}{h} \sum_j \mathbf{V}_{ij} & \text{for } k = i, \\ -\frac{\nu}{h} \mathbf{V}_{ij} & \text{for } k = j, \\ 0 & \text{otherwise,} \end{cases} \end{aligned} \quad (\text{A.3})$$

where $\mathbf{x}_i$ and $\mathbf{x}_k$ denote the positions of particle i and k , respectively. Since $\mathbf{V}_{ij} = -4(d+2) \frac{m_i m_j}{\rho_i + \rho_j} \frac{\nabla_i W_{ij}(\mathbf{x}_{ij}^n)^T}{\|\mathbf{x}_{ij}^n\|^2 + 0.01h^2} \propto \mathbf{x}_{ij}^n (\mathbf{x}_{ij}^n)^T$ is a 3×3 constant PSD matrix for any particle pair within a time step, the Hessian of the viscosity potential is a constant PSD matrix as well.

A.2 Time Splitting

A.2.1 Baseline Time Splitting

Applying time splitting, we can split the original time integration into a fluid phase

$$\begin{cases} \mathbf{v}_f^{n+1/2} = \mathbf{v}_f^n + h \mathbf{M}_f^{-1} \left(-\nabla_f P \left([(\mathbf{x}_f^{n+1/2})^T, (\mathbf{x}_s^n)^T]^T \right) + \mathbf{f}_f \right), \\ \mathbf{x}_f^{n+1/2} = \mathbf{x}_f^n + h \mathbf{v}_f^{n+1/2}, \end{cases} \quad (\text{A.4})$$

and a solid-coupling phase

$$\begin{cases} \mathbf{v}^{n+1} = \begin{bmatrix} \mathbf{v}_f^{n+1/2} \\ \mathbf{v}_s^n \end{bmatrix} + h \mathbf{M}^{-1} \left(-\nabla \Psi(\mathbf{x}^{n+1}) - \nabla C(\mathbf{x}^{n+1}) + \begin{bmatrix} \mathbf{0} \\ \mathbf{f}_s \end{bmatrix} \right), \\ \mathbf{x}^{n+1} = \mathbf{x}^n + h \mathbf{v}^{n+1}, \end{cases} \quad (\text{A.5})$$

Algorithm 2: Baseline Time Splitting

```

1:  $\mathbf{x} \leftarrow \mathbf{x}^n, \hat{\mathbf{x}}_f^n \leftarrow \mathbf{x}_f^n + h\mathbf{v}_f^n + h^2\mathbf{M}_f^{-1}\mathbf{f}_f$ 
2: SPH Neighbor Search & Density Update
3: // Fluid Phase
4:  $\mathbf{H}_f \leftarrow h^2\nabla_f^2 P([\mathbf{x}_f^T, (\mathbf{x}_s^n)^T]) + \mathbf{M}_f$ 
5:  $\mathbf{p}_f \leftarrow -\mathbf{H}_f^{-1} (h^2\nabla_f P([\mathbf{x}_f^T, (\mathbf{x}_s^n)^T]^T) + \mathbf{M}_f(\mathbf{x}_f - \hat{\mathbf{x}}_f^n))$ 
6:  $\mathbf{x}_f \leftarrow \mathbf{x}_f + \mathbf{p}_f$ 
7:  $\mathbf{x}_f^{n+1/2} \leftarrow \mathbf{x}_f, \mathbf{v}_f^{n+1/2} = (\mathbf{x}_f - \mathbf{x}_f^n)/h$ 
8: // Solid Coupling Phase
9:  $\hat{\mathbf{x}}^{n+1/2} \leftarrow \mathbf{x}^n + h[(\mathbf{v}_f^{n+1/2})^T, (\mathbf{v}_s^n)^T]^T + h^2\mathbf{M}^{-1}[\mathbf{0}^T, \mathbf{f}_s^T]^T$ 
10: do
11:    $\mathbf{H} \leftarrow h^2(\nabla^2\Psi(\mathbf{x}) + \nabla^2C(\mathbf{x})) + \mathbf{M}$ 
12:    $\mathbf{p} \leftarrow -\mathbf{H}^{-1} (h^2(\nabla\Psi(\mathbf{x}) + \nabla C(\mathbf{x})) + \mathbf{M}(\mathbf{x} - \hat{\mathbf{x}}^{n+1/2}))$ 
13:    $\alpha \leftarrow$  Backtracking Line Search with CCD
14:    $\mathbf{x} \leftarrow \mathbf{x} + \alpha\mathbf{p}$ 
15: while  $\frac{1}{h}\|\mathbf{p}\| > \epsilon$ 
16:  $\mathbf{x}^{n+1} \leftarrow \mathbf{x}, \mathbf{v}^{n+1} \leftarrow (\mathbf{x} - \mathbf{x}^n)/h$ 
17: return  $\mathbf{x}^{n+1}, \mathbf{v}^{n+1}$ 

```

where $\mathbf{f}_f$ and $\mathbf{f}_s$ are the external forces on the fluids and the solids, respectively. The two phases of this baseline time splitting scheme have equivalent optimization forms

$$\begin{aligned}
\mathbf{x}_f^{n+1/2} &= \arg \min_{\mathbf{x}_f} \frac{1}{2} \|\mathbf{x}_f - \hat{\mathbf{x}}_f^n\|_{\mathbf{M}_f}^2 + h^2 P([\mathbf{x}_f^T, (\mathbf{x}_s^n)^T]^T), \\
\mathbf{x}^{n+1} &= \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \hat{\mathbf{x}}^{n+1/2}\|_{\mathbf{M}}^2 + h^2 (\Psi(\mathbf{x}) + C(\mathbf{x})),
\end{aligned} \tag{A.6}$$

where $\hat{\mathbf{x}}_f^n = \mathbf{x}_f^n + h\mathbf{v}_f^n + h^2\mathbf{M}_f^{-1}\mathbf{f}_f$ and $\hat{\mathbf{x}}^{n+1/2} = \mathbf{x}^n + h[(\mathbf{v}_f^{n+1/2})^T, (\mathbf{v}_s^n)^T]^T + h^2\mathbf{M}^{-1}[\mathbf{0}^T, \mathbf{f}_s^T]^T$ with $\mathbf{v}_f^{n+1/2} = (\mathbf{x}_f^{n+1/2} - \mathbf{x}_f^n)/h$. The details of the optimization algorithm can be found in [Algorithm 2](#).

A.2.2 Error Analysis

The position update of the implicit Euler step and our proxy-enhanced time splitting scheme can be respectively expressed as

$$\begin{aligned}
\mathbf{x}_{\text{IE}}^{n+1} &= \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\left(-\nabla P(\mathbf{x}_{\text{IE}}^{n+1}) \right. \\
&\quad \left. - \nabla\Psi(\mathbf{x}_{\text{IE}}^{n+1}) - \nabla C_{\text{sf}}(\mathbf{x}_{\text{IE}}^{n+1}) - \nabla C_{\text{ss}}(\mathbf{x}_{\text{IE}}^{n+1}) + \mathbf{f}_{\text{ext}}\right), \\
\mathbf{x}^{n+1} &= \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\left(-\nabla P(\mathbf{x}^{n+1/2}) - \nabla\Psi(\mathbf{x}^{n+1}) \right. \\
&\quad \left. - \nabla\hat{C}_{\text{sf}}(\mathbf{x}^{n+1/2}) - \frac{1}{2}\nabla C_{\text{sf}}(\mathbf{x}^{n+1}) - \nabla C_{\text{ss}}(\mathbf{x}^{n+1}) + \mathbf{f}_{\text{ext}}\right).
\end{aligned} \tag{A.7}$$

If we define

$$e(\mathbf{x}) = \left\| \mathbf{x}^n + h\mathbf{v}^n + h^2\mathbf{M}^{-1}\left(-\nabla P(\mathbf{x}) - \nabla\Psi(\mathbf{x}) - \nabla C_{\text{sf}}(\mathbf{x}) - \nabla C_{\text{ss}}(\mathbf{x}) + \mathbf{f}_{\text{ext}}\right) - \mathbf{x} \right\|, \tag{A.8}$$

$\mathbf{x}_{\text{IE}}^{n+1}$ given by implicit Euler satisfies $e(\mathbf{x}_{\text{IE}}^{n+1}) = 0$, while for $\mathbf{x}^{n+1}$ in our scheme, we have $e(\mathbf{x}^{n+1}) = \mathcal{O}(h^4)$. Specifically,

$$\begin{aligned}
e(\mathbf{x}^{n+1}) &= h^2 \left\| \mathbf{M}^{-1}(\nabla P(\mathbf{x}^{n+1/2}) - \nabla P(\mathbf{x}^{n+1}) + \nabla\hat{C}_{\text{sf}}(\mathbf{x}^{n+1/2}) - \frac{1}{2}\nabla C_{\text{sf}}(\mathbf{x}^{n+1})) \right\| \\
&= \mathcal{O}\left(h^3 \left\| \mathbf{M}^{-1}\left(\nabla^2 P(\mathbf{x}^n)(\mathbf{v}^{n+1/2} - \mathbf{v}^{n+1}) + \frac{1}{2}\nabla^2 C_{\text{sf}}(\mathbf{x}^n)(\mathbf{v}^{n+1/2} - \mathbf{v}^{n+1})\right) \right\| \right) \\
&= \mathcal{O}\left(h^4 \left\| \mathbf{M}^{-1}\left(\nabla^2 P(\mathbf{x}^n) + \frac{1}{2}\nabla^2 C_{\text{sf}}(\mathbf{x}^n)\right) \mathbf{M}^{-1}\left(\nabla\Psi(\mathbf{x}^n) + \frac{1}{2}\nabla C_{\text{sf}}(\mathbf{x}^n) + \nabla C_{\text{ss}}(\mathbf{x}^n)\right) \right\| \right) \\
&= \mathcal{O}(h^4).
\end{aligned} \tag{A.9}$$

Here we assume that, in our discretized domain, the distance between any pair of primitives (particle-particle pair, particle-triangle pair, and triangle-triangle pair) has a lower bound ϵ . Thus, $\nabla^2 P(\mathbf{x}^n)$, $\nabla^2 C_{\text{sf}}(\mathbf{x}^n)$, $\nabla\Psi(\mathbf{x}^n)$, $\nabla C_{\text{sf}}(\mathbf{x}^n)$, and $\nabla C_{\text{ss}}(\mathbf{x}^n)$ are all bounded, and this indicates that our method has an $\mathcal{O}(h^4)$ difference compared to the implicit Euler solution. Since implicit Euler has an $\mathcal{O}(h^2)$ error compared to the PDE solution, our proposed time splitting scheme shares the same order of accuracy with implicit Euler

when it is stable.

APPENDIX B

Atlas3D Additional Experiments

B.1 Implementation and Training Details

We implement our pipeline in PyTorch with Adam optimizer. For differentiable simulation, we adopt the semi-implicit Euler simulator in Warp (Macklin, 2022), where gradients of physical states are computed via auto-differentiation and backpropagated to the parameters of 3D representations (Macklin, 2022; Paszke et al., 2017). We use a two-stage training strategy and leave the choice of the first stage open for various SDS-based methods as baselines. For a two-stage baseline method, we implement our physics-inspired losses as submodules that can be seamlessly integrated into the refinement stage. For a one-stage method, our approach can be used as a standalone refinement stage to improve the physical quality of the baseline. For baseline methods that are not publicly available, we use the reimplementation from threestudio (Guo et al., 2023).

For the rigid body simulator in Warp, we set $dt = 10^{-3}$ s. Contact stiffness and damping are set to 10^3 and 2.0; friction coefficient is set to 0.5; stiffness of friction force is set to 10^3 . Density of the 3D objects is set to 10^3 .

We train our models using a single NVIDIA RTX 3090 GPU. During our refinement stage, we implement a skipping strategy, incorporating standability loss only once every 10 iterations. In our quantitative evaluation of a batch of prompts, we observed an average refinement time of 36 minutes for each training step, with a default setting of 5,000 iterations.

Figure	<i>Broccoli</i>	<i>Egg</i>	<i>Horse</i>	<i>Kang.</i>	<i>Konan</i>	<i>Manneq.</i>	<i>Snowboard</i>	<i>Ultraman</i>
Baseline	0	0	1	0	0	0	0	1
Ours	3	4	3	4	4	4	3	4

Table B.1: Number of successes in robotic trials.

B.2 Additional Results

B.2.1 Qualitative Comparison With the Magic3D Baseline

Refer to [Figure B.1](#), [Figure B.2](#), and [Figure B.3](#).

B.2.2 Additional Comparison With the MVDream Baseline

Refer to [Figure B.4](#).

B.2.3 Quantitative Comparison With the Magic3D Baseline

The results of the quantitative experiments are shown in [Table 3.2](#). For CLIP score calculation, we specifically employ *openai/clip-vit-large-patch14-336* as the check point of CLIP model. We rendered images from various angles (0-360 degree, 3 degree as the interval). For Elo (GPT-4o) benchmark, we made use of the newly released GPT-4o model instead of the one in ([Wu et al., 2024b](#)) due to capacity limitation. Additionally, we reduce the number of views to 6 because of token length limitation.

B.3 Real World Experiments

B.3.1 Robot Manipulation Results

We record our robot manipulation experiment in [Figure B.5](#) and summarize quantitative results in [Table B.1](#).

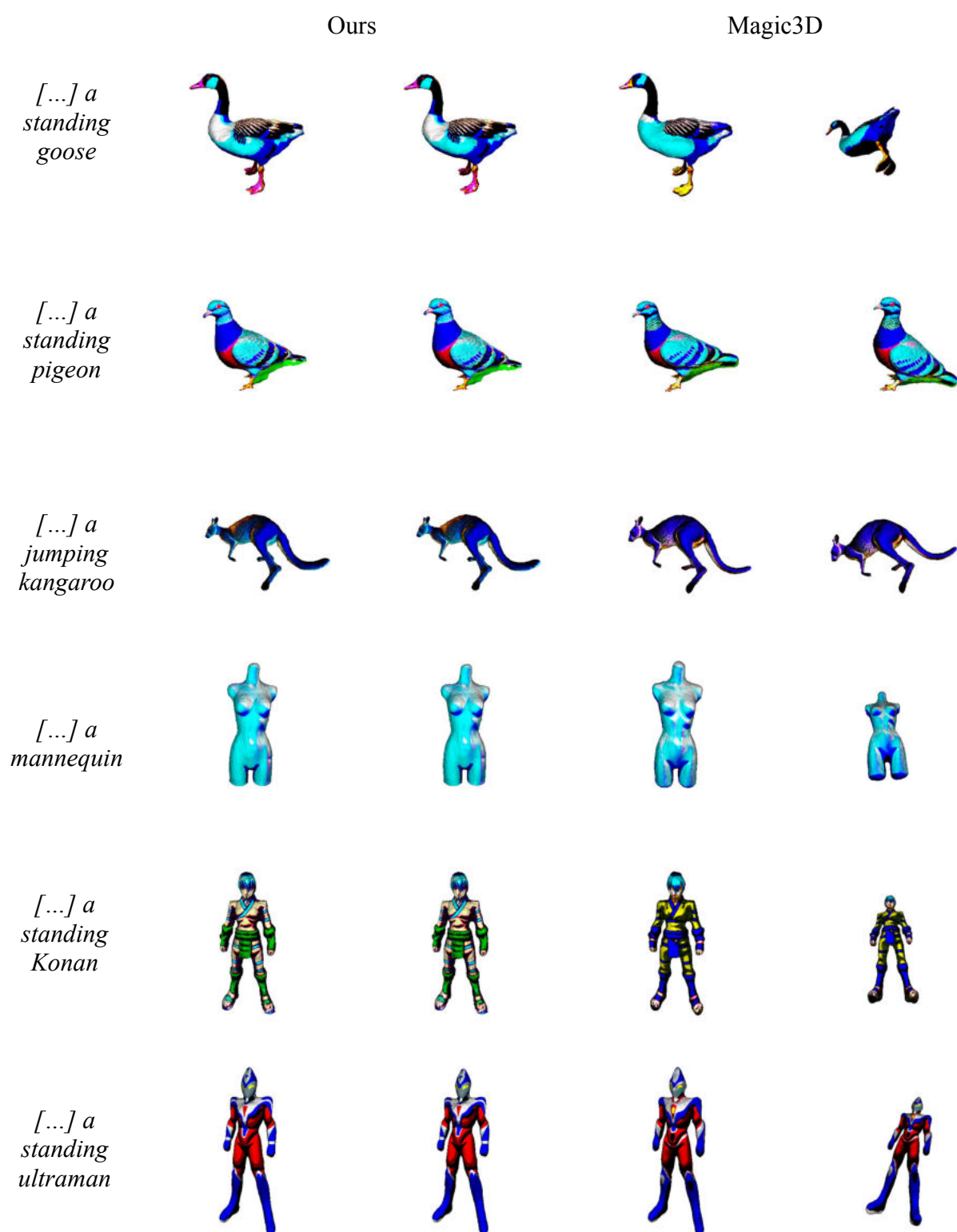


Figure B.1: More comparison with Magic3D baseline

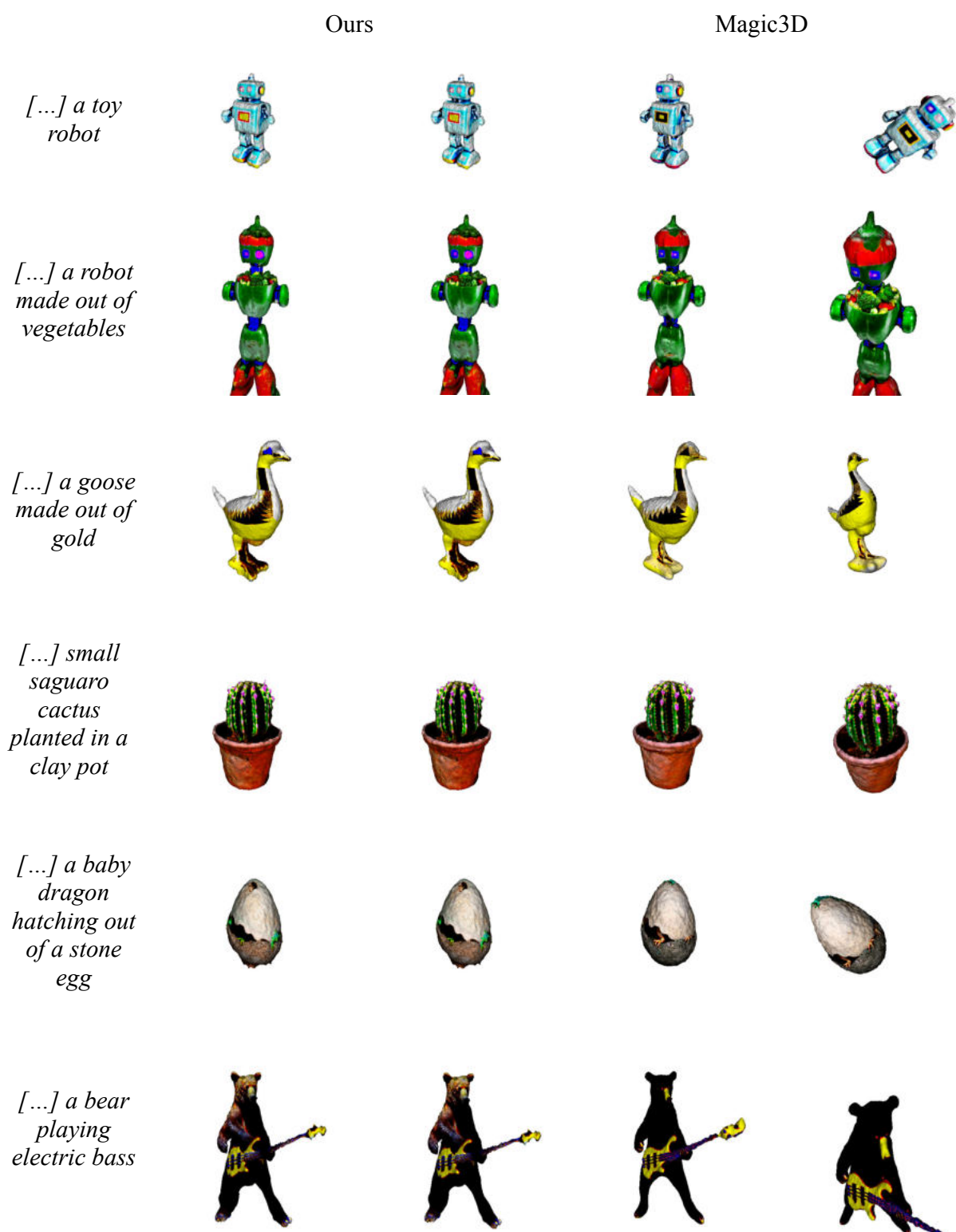


Figure B.2: More comparison with Magic3D baseline

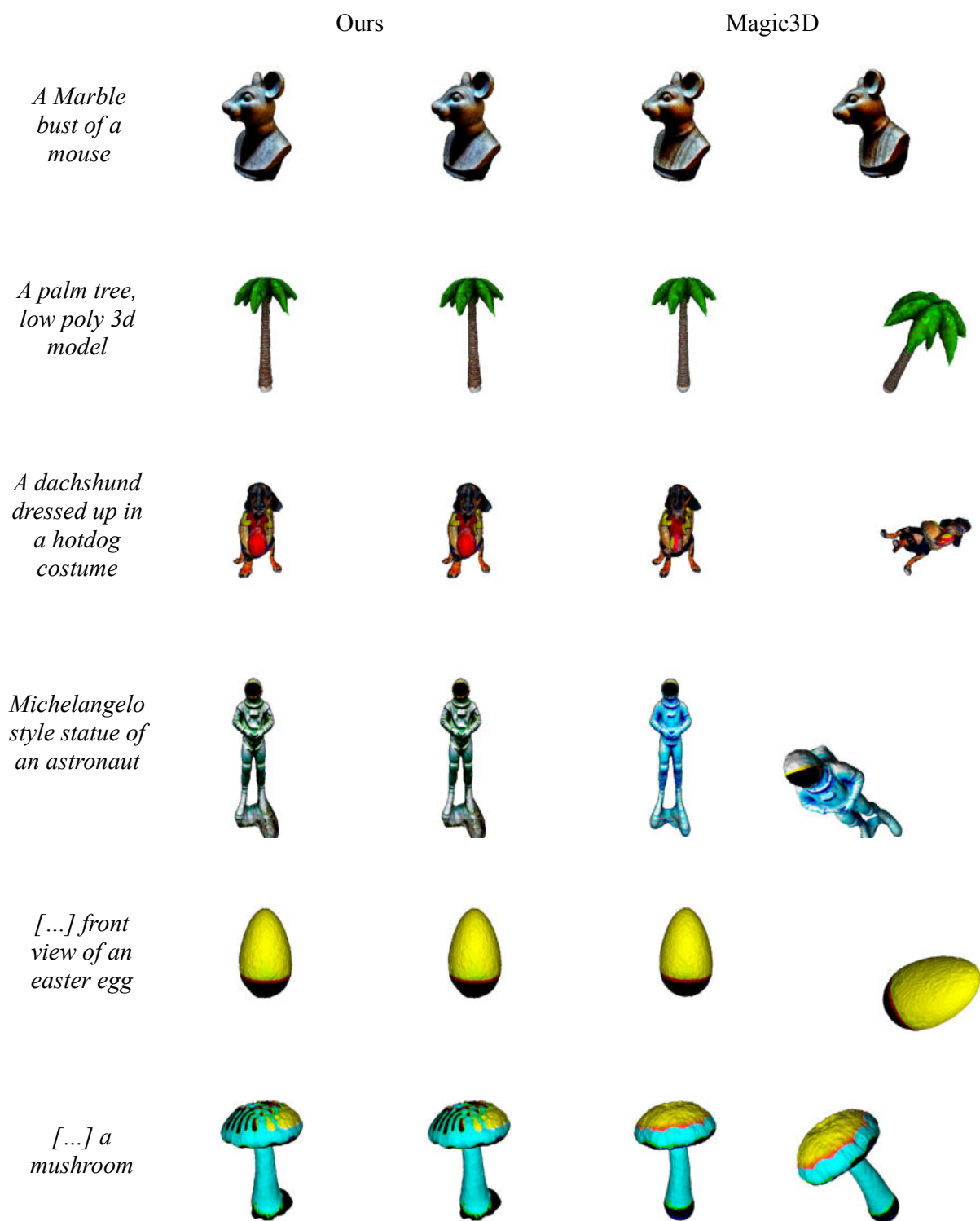


Figure B.3: More comparison with Magic3D baseline

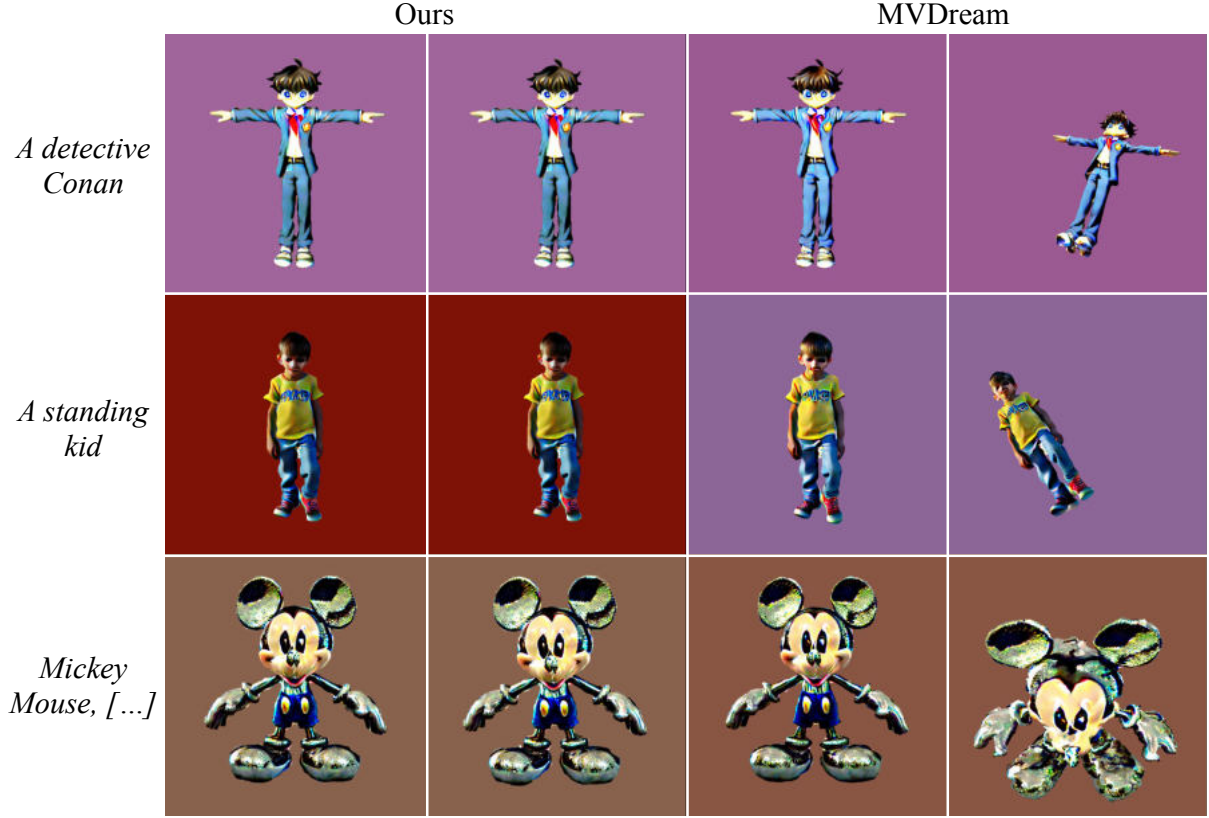


Figure B.4: More comparison with MVDream baseline

Figure	<i>Broccoli</i>	<i>Egg</i>	<i>Horse</i>	<i>Kang.</i>	<i>Konan</i>	<i>Manneq.</i>	<i>Snowboard</i>	<i>Ultraman</i>
Baseline	0	0	8	4	6	2	1	7
Ours	44	46	40	48	47	50	45	49

Table B.2: Number of successes in user studies.

B.3.2 User Study Results

We report the detailed results of our user studies in [Table B.2](#).

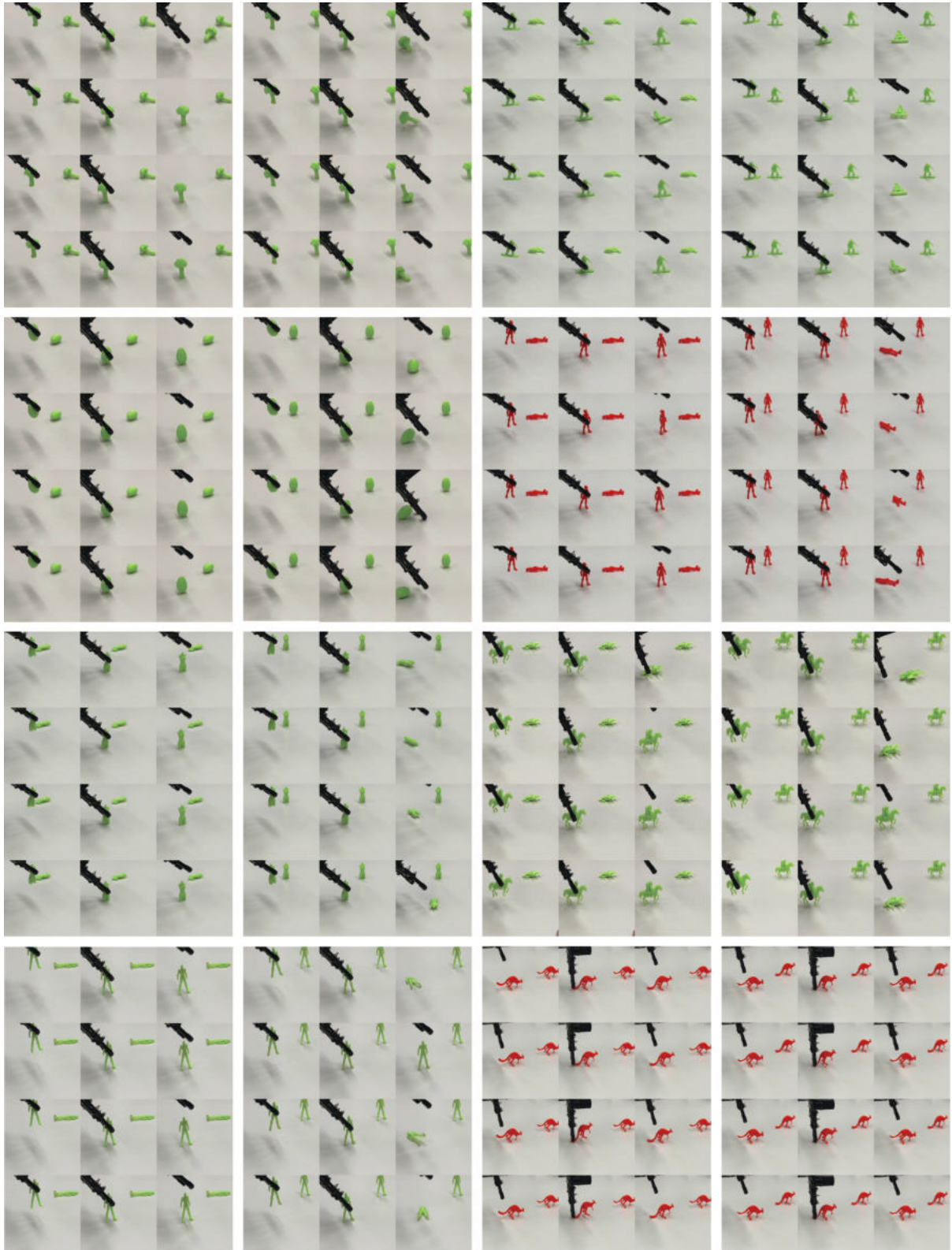


Figure B.5: Robot manipulation experiment

APPENDIX C

PhysGaussian Technical Details and Additional Evaluations

C.1 MPM Algorithm

In MPM, a continuum body is discretized into a set of Lagrangian particles p , and time is discretized into a sequence of time steps $t = 0, t^1, t^2, \dots$. Here we take a fixed stepsize Δt , so $t^n = n\Delta t$.

At each time step, masses and momentums on particles are first transferred to grid nodes. Grid velocities are then updated using forward Euler's method and transferred back to particles for subsequent advection. Let m_p , $\mathbf{x}_p^n$, $\mathbf{v}_p^n$, $\mathbf{F}_p^n$, $\boldsymbol{\tau}_p^n$, and $\mathbf{C}_p^n$ denote the mass, position, velocity, deformation gradient, Kirchhoff stress, and affine momentum on particle p at time t_n . Here, particle masses are invariant due to mass conservation law. Let m_i^n , $\mathbf{x}_i^n$ and $\mathbf{v}_i^n$ denote the mass, position, and velocity on grid node i at time t^n . We summarize the explicit MPM algorithm as follows:

1. **Transfer Particles to Grid.** Transfer mass and momentum from particles to grids as

$$\begin{aligned} m_i^n &= \sum_p w_{ip}^n m_p, \\ m_i^n \mathbf{v}_i^n &= \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + \mathbf{C}_p^n (\mathbf{x}_i - \mathbf{x}_p^n)). \end{aligned} \tag{C.1}$$

We adopt the APIC scheme ([Jiang et al., 2015](#)) for momentum transfer.

2. **Grid Update.** Update grid velocities based on forces at the next timestep by

$$\mathbf{v}_i^{n+1} = \mathbf{v}_i^n - \frac{\Delta t}{m_i} \sum_p \boldsymbol{\tau}_p^n \nabla w_{ip}^n V_p^0 + \Delta t \mathbf{g}. \quad (\text{C.2})$$

3. **Transfer Grid to Particles.** Transfer velocities back to particles and update particle states.

$$\begin{aligned} \mathbf{v}_p^{n+1} &= \sum_i \mathbf{v}_i^{n+1} w_{ip}^n, \\ \mathbf{x}_p^{n+1} &= \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1}, \\ \mathbf{C}_p^{n+1} &= \frac{12}{\Delta x^2 (b+1)} \sum_i w_{ip}^n \mathbf{v}_i^{n+1} (\mathbf{x}_i^n - \mathbf{x}_p^n)^T, \\ \nabla \mathbf{v}_p^{n+1} &= \sum_i \mathbf{v}_i^{n+1} \nabla w_{ip}^n, \\ \mathbf{F}_p^{\text{E, tr}} &= (\mathbf{I} + \nabla \mathbf{v}_p^{n+1}) \mathbf{F}^{E,n}, \\ \mathbf{F}_p^{E,n+1} &= \mathcal{Z}(\mathbf{F}_p^{\text{E, tr}}), \\ \boldsymbol{\tau}_p^{n+1} &= \boldsymbol{\tau}(\mathbf{F}_p^{E,n+1}). \end{aligned} \quad (\text{C.3})$$

Here b is the B-spline degree, and Δx is the Eulerian grid spacing. The computation of the return map $\mathcal{Z}$ and the Kirchhoff stress $\boldsymbol{\tau}$ is outlined in [Section C.2](#). We refer the readers to ([Jiang et al., 2016](#)) for the detailed derivations from the continuous conservation law to its MPM discretization.

C.2 Elasticity and Plasticity Models

We adopt the constitutive models used in ([Zong et al., 2023](#)). We list the models used for each scene in [Table C.1](#) and summarize all the parameters needed in discussing the constitutive models in [Table C.2](#).

In all plasticity models used in our work, the deformation gradient is multiplicatively decomposed into $\mathbf{F} = \mathbf{F}^E \mathbf{F}^P$ following some yield stress condition. A hyperelastic constitutive model is applied to $\mathbf{F}^E$ to compute the Kirchhoff stress $\boldsymbol{\tau}$. For a pure elastic

Scene	Figure	Constitutive Model
Vasdeck	Figure 4.1	Fixed corotated
Ficus	Figure 1.3	Fixed corotated
Fox	Figure 4.5	Fixed corotated
Plane	Figure 4.5	von Mises
Toast	Figure 4.5	Fixed corotated
Ruins	Figure 4.5	Drucker-Prager
Jam	Figure 4.5	Herschel-Bulkley
Sofa Suite	Figure 4.5	Fixed corotated
Materials	Figure 4.8	Fixed corotated
Microphone	Figure 4.9	Neo-Hookean
Bread	Figure C.1	Fixed corotated
Cake	Figure C.1	Herschel-Bulkley
Can	Figure C.1	von Mises
Wolf	Figure C.1	Drucker-Prager

Table C.1: Model Settings.

Notation	Meaning	Relation to E, ν
E	Young's modulus	/
ν	Poisson's ratio	/
μ	Shear modulus	$\mu = \frac{E}{2(1+\nu)}$
λ	Lamé modulus	$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}$
κ	Bulk modulus	$\kappa = \frac{E}{3(1-2\nu)}$

Table C.2: Material Parameters.

continuum, we simply take $\mathbf{F}^E = \mathbf{F}$.

C.2.1 Fixed Corotated Elasticity

The Kirchhoff stress $\boldsymbol{\tau}$ is defined as

$$\boldsymbol{\tau} = 2\mu(\mathbf{F}^E - \mathbf{R})\mathbf{F}^{E^T} + \lambda(J - 1)\mathbf{J}, \quad (\text{C.4})$$

where $\mathbf{R} = \mathbf{U}\mathbf{V}^T$ and $\mathbf{F}^E = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ is the singular value decomposition of elastic deformation gradient. J is the determinant of $\mathbf{F}^E$ ([Jiang et al., 2015](#)).

C.2.2 StVK Elasticity

The Kirchhoff stress $\boldsymbol{\tau}$ is defined as

$$\boldsymbol{\tau} = \mathbf{U} (2\mu\boldsymbol{\epsilon} + \lambda \text{sum}(\boldsymbol{\epsilon})\mathbf{1}) \mathbf{V}^T, \quad (\text{C.5})$$

where $\boldsymbol{\epsilon} = \log(\boldsymbol{\Sigma})$ and $\mathbf{F}^E = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ (Klár et al., 2016).

C.2.3 Neo-Hookean Elasticity

The Kirchhoff stress $\boldsymbol{\tau}$ is defined as

$$\boldsymbol{\tau} = \mu(\mathbf{F}^E \mathbf{F}^{E^T} - \mathbf{I}) + \log(J)\mathbf{I}, \quad (\text{C.6})$$

where J is the determinant of $\mathbf{F}^E$ (Jiang et al., 2015).

C.2.4 Drucker-Prager Plasticity

The return mapping of Drucker-Prager plasticity for sand (Klár et al., 2016) is, given $\mathbf{F} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ and $\boldsymbol{\epsilon} = \log(\boldsymbol{\Sigma})$,

$$\mathbf{F}^E = \mathbf{U}\mathcal{Z}(\boldsymbol{\Sigma})\mathbf{V}^T, \quad (\text{C.7})$$

$$\mathcal{Z}(\boldsymbol{\Sigma}) = \begin{cases} \mathbf{1}, & \text{sum}(\boldsymbol{\epsilon}) > 0, \\ \boldsymbol{\Sigma}, & \delta\gamma \leq 0, \text{ and } \text{sum}(\boldsymbol{\epsilon}) \leq 0, \\ \exp\left(\boldsymbol{\epsilon} - \delta\gamma \frac{\hat{\boldsymbol{\epsilon}}}{\|\hat{\boldsymbol{\epsilon}}\|}\right), & \text{otherwise.} \end{cases} \quad (\text{C.8})$$

Here $\delta\gamma = \|\hat{\boldsymbol{\epsilon}}\| + \alpha \frac{(d\lambda + 2\mu)\text{sum}(\boldsymbol{\epsilon})}{2\mu}$, $\alpha = \sqrt{\frac{2}{3} \frac{2\sin\phi_f}{3-\sin\phi_f}}$, and ϕ_f is the friction angle. $\hat{\boldsymbol{\epsilon}} = \text{dev}(\boldsymbol{\epsilon})$.

C.2.5 Von-Mises Plasticity

Similar to Drucker-Prager plasticity, given $\mathbf{F} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ and $\boldsymbol{\epsilon} = \log(\boldsymbol{\Sigma})$,

$$\mathbf{F}^E = \mathbf{U}\mathcal{Z}(\boldsymbol{\Sigma})\mathbf{V}^T,$$

where

$$\mathcal{Z}(\boldsymbol{\Sigma}) = \begin{cases} \boldsymbol{\Sigma}, & \delta\gamma \leq 0, \\ \exp\left(\boldsymbol{\epsilon} - \delta\gamma \frac{\hat{\boldsymbol{\epsilon}}}{\|\hat{\boldsymbol{\epsilon}}\|}\right), & \text{otherwise,} \end{cases} \quad (\text{C.9})$$

and $\delta\gamma = \|\hat{\boldsymbol{\epsilon}}\|_F - \frac{\tau_Y}{2\mu}$. Here τ_Y is the yield stress.

C.2.6 Herschel-Bulkley Plasticity

We follow [Yue et al. \(2015\)](#) and take the simple case where $h = 1$. Denote $\mathbf{s}^{\text{trial}} = \text{dev}(\boldsymbol{\tau}^{\text{trial}})$, and $s^{\text{trial}} = \|\mathbf{s}^{\text{trial}}\|$. The yield condition is $\Phi(s) = s - \sqrt{\frac{2}{3}}\sigma_Y \leq 0$. If it is violated, we modify s^{trial} by

$$s = s^{\text{trial}} - \left(s^{\text{trial}} - \sqrt{\frac{2}{3}}\sigma_Y \right) / \left(1 + \frac{\eta}{2\mu\Delta t} \right).$$

$\mathbf{s}$ can then be recovered as $\mathbf{s} = s \cdot \frac{\mathbf{s}^{\text{trial}}}{\|\mathbf{s}^{\text{trial}}\|}$. Define $\mathbf{b}^E = \mathbf{F}^E \mathbf{F}^{E^T}$. The Kirchhoff stress $\boldsymbol{\tau}$ is computed as

$$\boldsymbol{\tau} = \frac{\kappa}{2} (J^2 - 1) \mathbf{I} + \mu \text{dev} \left[\det(\mathbf{b}^E)^{-\frac{1}{3}} \mathbf{b}^E \right].$$

C.3 Additional Evaluations

We present additional evaluations of our method in [Figure C.1](#). The *vasedeck* data is from the Nerf dataset ([Mildenhall et al., 2021](#)) and the others are synthetic data, generated using BlenderNeRF ([Raafat, 2023](#)).



Figure C.1: Additional Evaluation. Examples from top to bottom are: *vasedeck* (elastic entity), *bread* (fracture), *cake* (viscoplastic material), *can* (plastic metal) and *wolf* (granular material).

APPENDIX D

PhysAnimator Technical Details and Additional Experiments

D.1 Motion Equation

To model the fluidity and exaggerated motions typical of anime, we represent anime objects as deformable bodies and adopt the Fixed Corotated model (Stomakhin et al., 2012) as the constitutive model. The dynamics of the deformable bodies are governed by Newton’s Second Law, expressed as:

$$\frac{d^2 \mathbf{x}}{dt^2} = \mathbf{M}^{-1}(\mathbf{f}_{\text{int}}(\mathbf{x}) + \mathbf{f}_{\text{ext}}(\mathbf{x})), \quad (\text{D.1})$$

where $\mathbf{M}$ is the mass matrix, representing the masses of all vertices:

$$\mathbf{M} = \begin{pmatrix} m_1 & 0 & \cdots & 0 \\ 0 & m_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & m_N \end{pmatrix}. \quad (\text{D.2})$$

The mass of i -th vertex is calculated as

$$m_i = \sum_{\mathcal{T}_j \in S_i} \frac{\rho_j V_j}{3} \quad (\text{D.3})$$

where S_i is the set of triangles containing the i -th vertices. ρ_j and V_j denote the mass density and volume of j -th triangle, respectively. The internal force for the $\mathbf{f}_{\text{int}}(\mathbf{x})$ of

Fixed Corotated deformable model can be derived as

$$\begin{aligned}
\mathbf{f}_{\text{int}}(\mathbf{x}) &= -\frac{\partial E(\mathbf{x})}{\partial \mathbf{x}} \\
&= -\frac{\partial \sum_{i=1}^N \Psi(\mathbf{F}_i) V_i}{\partial \mathbf{x}} \\
&= -\sum_{i=1}^N \left(\frac{\partial \Psi(\mathbf{F}_i)}{\partial \mathbf{F}_i} \frac{\partial \mathbf{F}_i}{\partial \mathbf{x}} V_i \right),
\end{aligned} \tag{D.4}$$

where

$$\begin{aligned}
\frac{\partial \Psi(\mathbf{F}_i)}{\partial \mathbf{F}_i} &= \frac{\partial (\mu \|\mathbf{F}_i - \mathbf{R}_i\|_F^2 + \frac{\lambda}{2} (\det(\mathbf{F}_i) - 1)^2)}{\partial \mathbf{F}_i} \\
&= 2\mu(\mathbf{F}_i - \mathbf{R}) + \lambda (\det(\mathbf{F}_i) - 1) \det(\mathbf{F}_i) \mathbf{F}_i^{-T},
\end{aligned} \tag{D.5}$$

The deformation gradient $\mathbf{F}_i$ for a 2D triangle $\mathcal{T}_i$ is computed as:

$$\mathbf{F}_i = [\mathbf{x}_1 - \mathbf{x}_0, \mathbf{x}_2 - \mathbf{x}_0][\mathbf{X}_1 - \mathbf{X}_0, \mathbf{X}_2 - \mathbf{X}_0]^{-1}. \tag{D.6}$$

Here $\{\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_2\}$ and $\{\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2\}$ represent the positions of vertices of triangle $\mathcal{T}_i$ in the undeformed (material) space and the deformed (world) space respectively.

To solve the motion equation (D.1), we apply a semi-implicit Euler method, discretizing it as:

$$\begin{aligned}
\mathbf{x}^{n+1} &= \mathbf{x}^n + \Delta t \mathbf{v}^{n+1}, \\
\mathbf{v}^{n+1} &= \mathbf{v}^n + \Delta t \mathbf{M}^{-1}(\mathbf{f}_{\text{int}}(\mathbf{x}^n) + \mathbf{f}_{\text{ext}}(\mathbf{x}^n)),
\end{aligned} \tag{D.7}$$

where $\mathbf{x}^n, \mathbf{v}^n$ and $\mathbf{x}^{n+1}, \mathbf{v}^{n+1}$ denote the positions and velocities at time steps t^n and t^{n+1} , respectively. The timestep size Δt is set to 0.001 in our experiments. By iteratively applying (D.7) from $t = 0$, we obtain the physical states (positions and velocities) at any future time step.

D.2 Details of Interactive Animation

To enable interactive animation, we introduce energy strokes that carry flow particles, which exert forces on nearby vertices to drive motion. As illustrated in Figure D.1, we

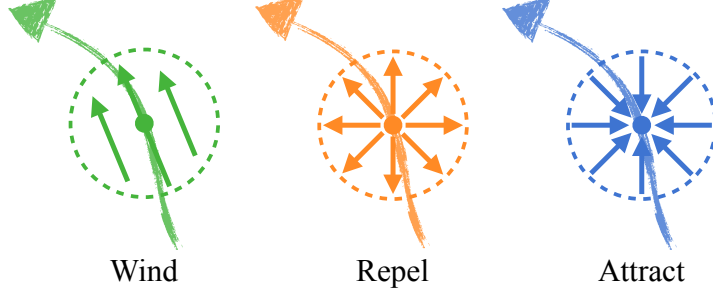


Figure D.1: Flow Particles. Wind particles can simulate wind effects, creating natural swaying motions; repel particles act as collision barriers, adjusting object trajectories; attract particles can pull objects toward specific points as desired.

provide three types of customizable energy strokes: wind, repel, and attract.

For a wind flow particle located at position $\mathbf{p}$ and a nearby vertex at position $\mathbf{q}$, the exerted force is defined as:

$$\begin{aligned} \mathbf{f}_{\text{wind}}(\mathbf{p}, \mathbf{q}) &= s(1 - w(\mathbf{p}, \mathbf{q}))\mathbf{d}, \\ w(\mathbf{p}, \mathbf{q}) &= \frac{\|\mathbf{p} - \mathbf{q}\|}{r}, \end{aligned} \tag{D.8}$$

where r is the influence range, s represents the particle's strength, and $\mathbf{d}$ denotes the particle's movement direction. Similarly, the forces exerted by repel and attract flow particles are given as:

$$\begin{aligned} \mathbf{f}_{\text{repel}} &= s \frac{\mathbf{q} - \mathbf{p}}{r}, \\ \mathbf{f}_{\text{attract}} &= s \left(1 - \frac{\|\mathbf{q} - \mathbf{p}\|}{r} \right) \frac{\mathbf{p} - \mathbf{q}}{\|\mathbf{p} - \mathbf{q}\|}. \end{aligned} \tag{D.9}$$

In addition to flow particles, we incorporate rigging point support to allow animators to fix specific regions or define desired motion trajectories. Inspired by the swaying effects commonly seen in anime, we design a periodic wavy rigging point. The position of a wavy rigging point at time t is expressed as:

$$\mathbf{x}_r(t) = \mathbf{X}_r + s \sin(ft)\mathbf{d}_r, \tag{D.10}$$

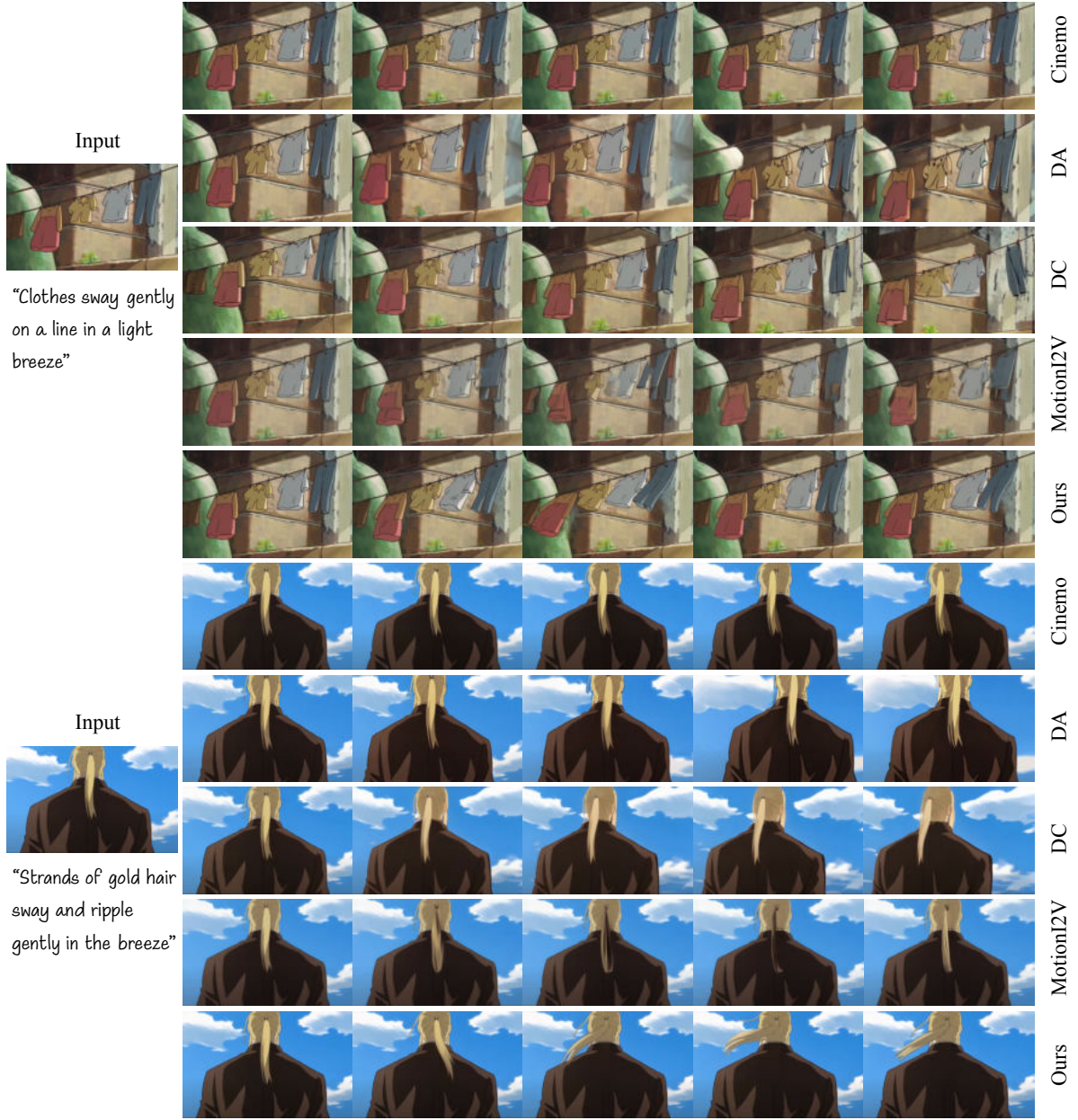


Figure D.2: Additional Qualitative Comparison. We provide additional comparison results against Cinemo (Ma et al., 2024), Drag Anything (DA) (Wu et al., 2025a), DynamiCrafter (DC) (Xing et al., 2025) and Motion-I2V (Shi et al., 2024).

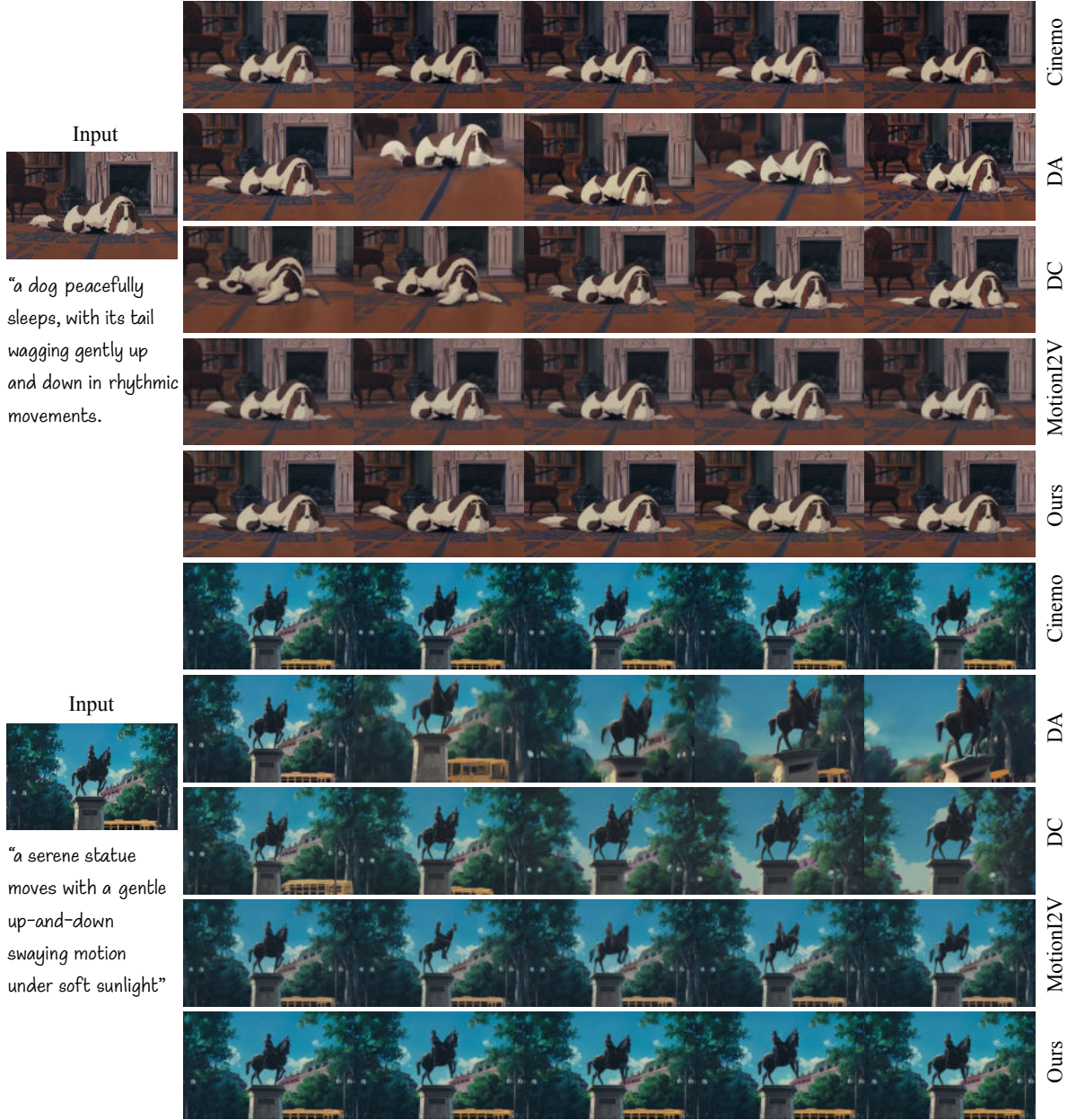


Figure D.3: Additional Qualitative Comparison. We provide additional comparison results against Cinemo (Ma et al., 2024), Drag Anything (DA) (Wu et al., 2025a), DynamiCrafter (DC) (Xing et al., 2025) and Motion-I2V (Shi et al., 2024).

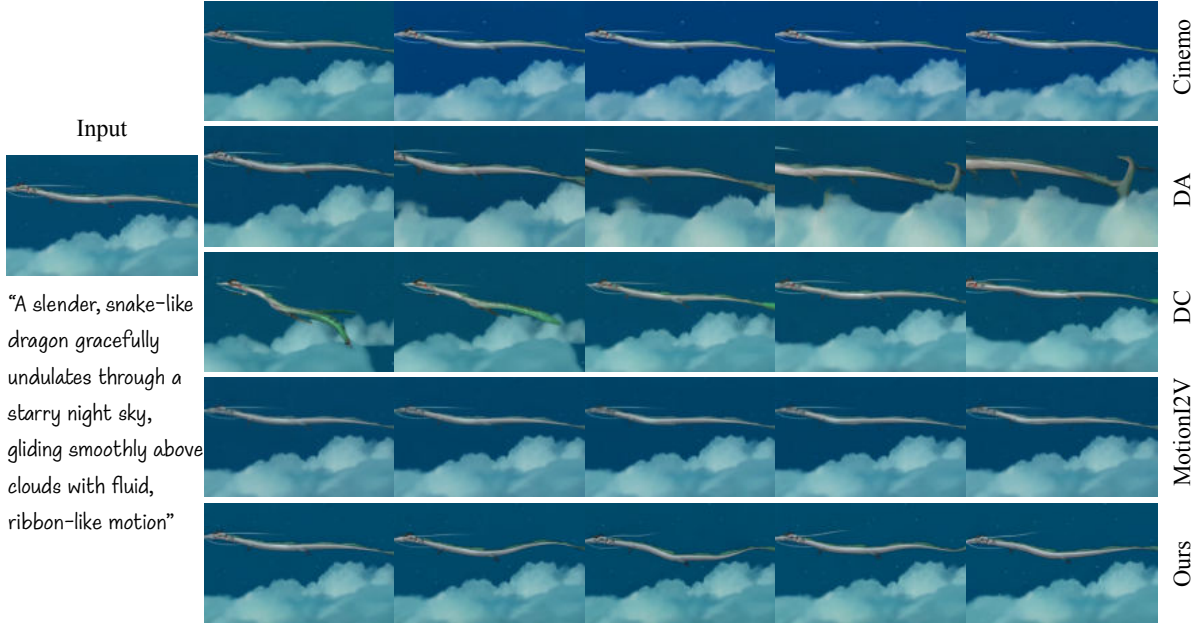


Figure D.4: Additional Qualitative Comparison. We provide additional comparison results against Cinemo (Ma et al., 2024), Drag Anything (DA) (Wu et al., 2025a), DynamiCrafter (DC) (Xing et al., 2025) and Motion-I2V (Shi et al., 2024).

Methods	User Pref.↑	VSVQ↑	VSTC↑	VSDD↑	VSFC↑
Ours w/o DE	29.6%	2.94	2.90	2.48	2.68
Ours w/ DE	70.4%	2.89	2.86	2.48	2.64

Table D.1: Ablation of Dynamics Enhancement. We present the evaluation results for our Dynamics Enhancement (DE) module. The metrics VSVQ, VSTC, VSDD, and VSFC represent VideoScore (He et al., 2024b) assessments of visual quality, temporal consistency, dynamic degree, and factual consistency, respectively. User preferences are determined based on which video resembles real anime more.

where $\mathbf{X}_r$ is the position in the undeformed state, f is the swaying frequency, and $\mathbf{d}_r$ represents the swaying direction. The selected region will follow the rigging point’s motion, creating desired animation effects.

D.3 Controllable Generation

Our physics-based modeling approach offers flexibility in adjusting the properties of animated objects, such as stiffness, as shown in Figure D.5. Additionally, Figure D.6

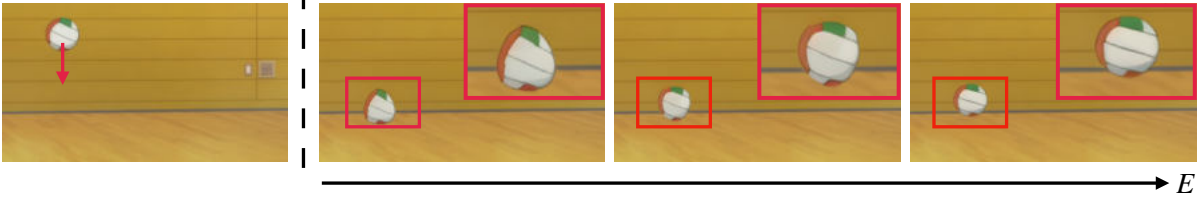


Figure D.5: Controllable Generation with Material Property. As the Young’s modulus E of the ball increases from left to right, the ball becomes more capable of maintaining its original shape under external forces.

demonstrates how our method enables the convenient creation of diverse dynamic effects by applying energy brushes with varying settings.

D.4 Ablation Study

We perform an ablation study to evaluate the effectiveness of our complementary dynamics enhancement module. As part of this evaluation, we conducted a user study where participants were asked, ”Which video resembles real anime more?” We present the user preferences alongside the VideoScore (He et al., 2024b) metrics in Table D.1. Although applying the dynamics enhancement results in a slight decrease in VideoScore metrics, the user study shows a clear preference (70.4%) for the videos with dynamics enhancement, demonstrating this module’s contribution to improved anime stylization.

D.5 More Comparison Results

In Figure D.2, Figure D.3, and Figure D.4, we provide additional qualitative comparison results with baseline methods, including Cinemo (Ma et al., 2024), Drag Anything (Wu et al., 2025a), DynamiCrafter (Xing et al., 2025) and Motion-I2V (Shi et al., 2024).



Figure D.6: Controllable Generation with Energy Stroke. By applying energy strokes with varying directions and strengths, our method enables the convenient generation of diverse dynamic effects.

APPENDIX E

AniMimic Technical Details and Additional Results

E.1 Differentiable Simulation Details

Forward Simulation: The dynamics of deformable bodies follow Newton’s Second Law, formulated as

$$\frac{d^2 \mathbf{x}}{dt^2} = \mathbf{M}^{-1} \mathbf{f}(\mathbf{x}) \quad (\text{E.1})$$

where $\mathbf{M}$ is the mass matrix, whose diagonal entries contain the masses of all vertices. The total force $\mathbf{f}(\mathbf{x})$ consists of the external forces $\mathbf{f}_{\text{ext}}(\mathbf{x})$ (e.g., gravity) and the internal elastic forces $\mathbf{f}_{\text{int}}(\mathbf{x})$ arising from deformation. To ensure consistency with skinning-based animation, we omit external forces and simulate only the internal elastic response. The internal force $\mathbf{f}_{\text{int}}(\mathbf{x})$ captures how the deformable object attempts to restore its rest shape by resisting deformation.

In continuum mechanics, this elastic behavior is characterized by defining an energy density function $\psi(\mathbf{F}(\mathbf{x}))$, which measures the strain energy per unit undeformed volume. We employ the Fixed Corotated constitutive model ([Stomakhin et al., 2012](#)) to describe the material behavior. Its energy density is defined as

$$\psi(\mathbf{x}) = \mu \|\mathbf{F}_i - \mathbf{R}_i\|_F^2 + \frac{\lambda}{2} (\det(\mathbf{F}_i) - 1)^2, \quad (\text{E.2})$$

where μ and λ are the Lamé parameters, $\mathbf{F}_i$ is the deformation gradient, and $\mathbf{R}_i$ is the rotational component extracted from the polar decomposition of $\mathbf{F}_i$. The total potential

energy of the deformable body is computed as

$$\Psi(\mathbf{x}) = \sum_{i=1}^N \psi(\mathbf{F}_i) V_i, \quad (\text{E.3})$$

where V_i denotes volume of the i -th tetrahedron. The internal resisting force is then defined as the negative gradient of the potential energy with respect to the vertex position

$$\mathbf{f}_{\text{int}}(\mathbf{x}) = -\frac{\partial \Psi(\mathbf{x})}{\partial \mathbf{x}}. \quad (\text{E.4})$$

To advance the simulation, we use an optimization-based implicit integrator:

$$\mathbf{x}^{n+1} = \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \tilde{\mathbf{x}}\|_M^2 + \Psi(\mathbf{x}), \quad (\text{E.5})$$

and solve this minimization via Newton's method with line search. This yields the updated configuration $\mathbf{x}^{n+1}$ and allows us to compute the simulated trajectory of every vertex over time.

Backward Propagation: Following [Li et al. \(2025e\)](#), once the optimization problem in [\(E.5\)](#) is solved at each time step, we can backpropagate through the implicit integrator. For any loss function L , let $\mathbf{G}$ denote the gradient of the objective $\frac{1}{2} \|\mathbf{x} - \tilde{\mathbf{x}}\|_M^2 + \Psi(\mathbf{x})$. The gradient can be propagated from time step $n+1$ to time step n as:

$$\frac{dL}{d\mathbf{x}^n} = -\mathcal{A} \frac{\partial \mathbf{G}}{\partial \mathbf{x}^n} - \frac{1}{h} \frac{dL}{d\mathbf{v}^{n+1}} \quad (\text{E.6})$$

$$\left[\frac{dL}{d\mathbf{v}^n}, \frac{dL}{dE} \right] = -\mathcal{A} \left[\frac{\partial \mathbf{G}}{\partial \mathbf{v}^n}, \frac{\partial \mathbf{G}}{\partial E} \right], \quad (\text{E.7})$$

where the velocity update is $\mathbf{v}^{n+1} = \frac{\mathbf{x}^{n+1} - \mathbf{x}^n}{\Delta t}$ and E is the Young's modulus. $\mathcal{A}$ is obtained by solving a linear system:

$$\mathcal{A} = \left[\frac{dL}{d\mathbf{x}^{n+1}} + \frac{1}{h} \frac{dL}{d\mathbf{v}^{n+1}} \right] \left[\frac{\partial \mathbf{G}}{\partial \mathbf{x}^{n+1}} \right]^{-1} \quad (\text{E.8})$$

By combining these analytical gradients with Warp’s differentiation operator (Macklin, 2022), our simulation becomes fully differentiable with respect to material parameters. This enables direct optimization of material properties using losses computed at arbitrary time steps.

E.2 Additional Results

Additional Qualitative Comparison: In Figure E.1, we provide additional comparison results with the baseline methods, including Puppeteer (Song et al., 2025), DreamMesh4D (Li et al., 2024e) and SC4D (Wu et al., 2024c).

Diverse Motion: By leveraging the capability of video models, our method naturally supports generating diverse motions for the input 3D assets by producing multiple videos and optimizing the rigging model for each, as demonstrated in Figure E.2.

Novel Views: We present additional novel-view results of our generated motions in Figure E.3. Since the motions are reconstructed on a 3D model, our method naturally maintains cross-view consistency, removing the need to generate multi-view videos.

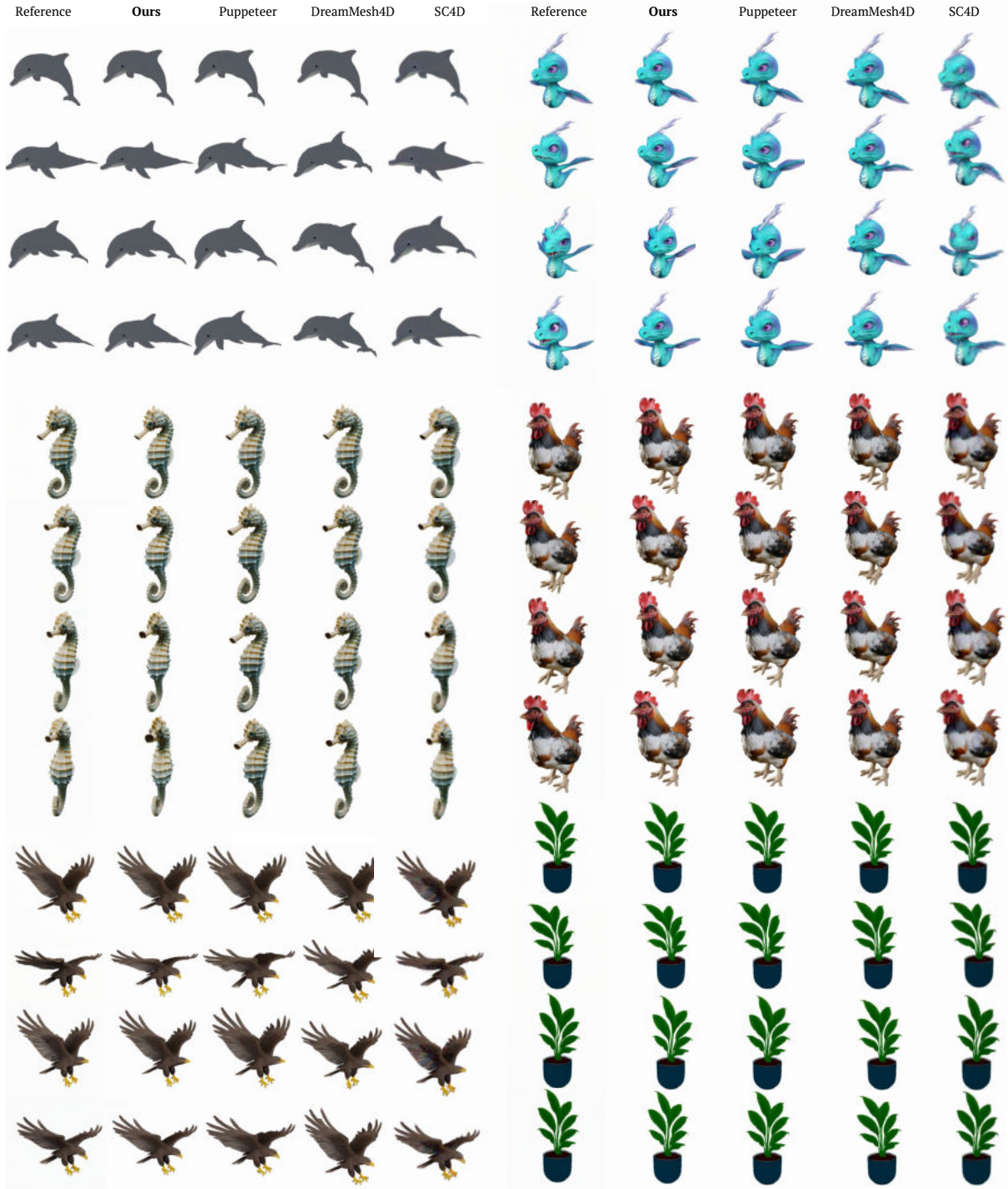


Figure E.1: Qualitative Comparison. We provide additional qualitative comparison results with Puppeteer (Song et al., 2025), DreamMesh4D (Li et al., 2024e), and SC4D (Wu et al., 2024c).



Figure E.2: Varying Motion Generation. With different guiding videos of the same character, our method generates the corresponding 4D trajectories containing varied motion sequences.

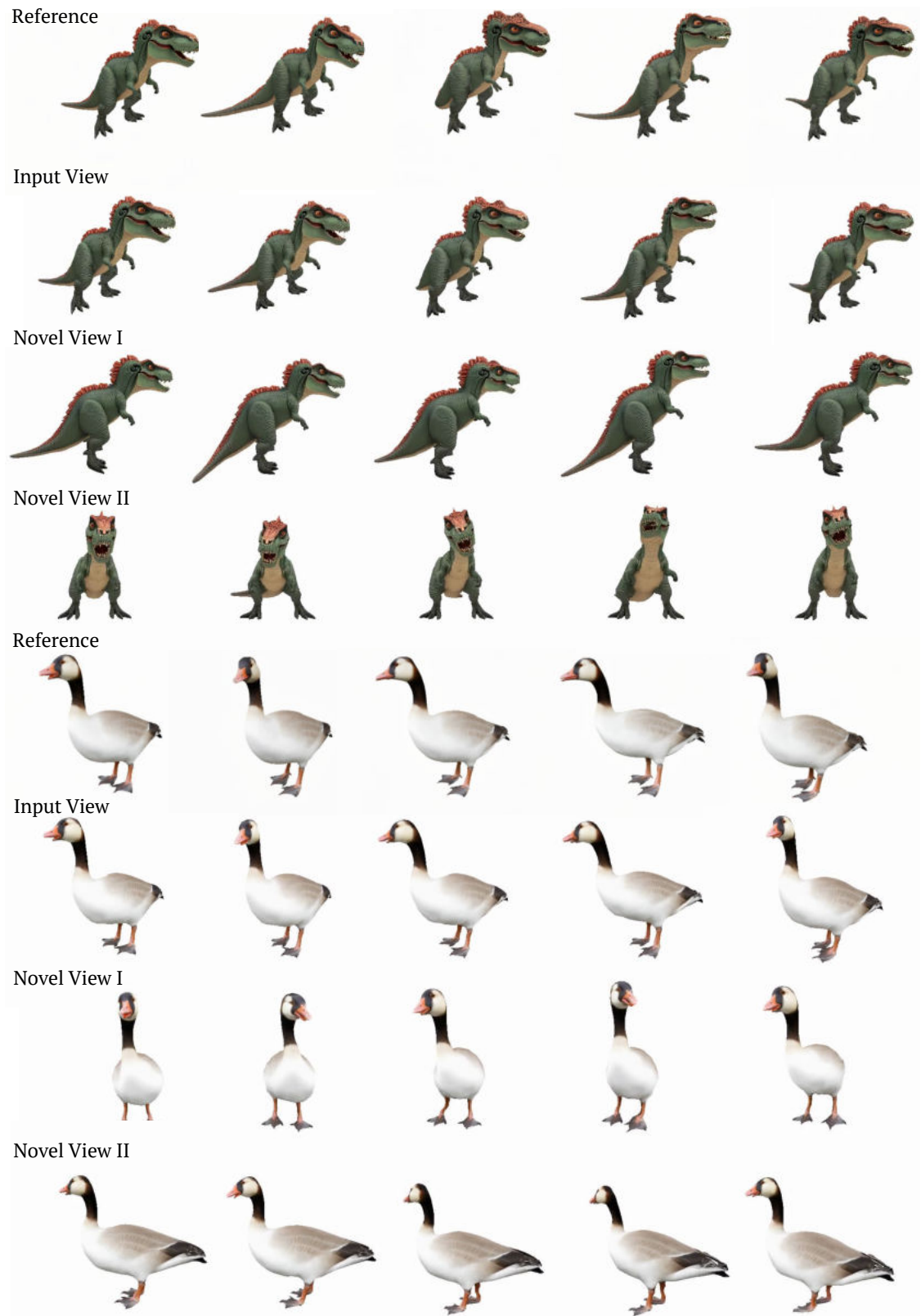


Figure E.3: Novel Views. In addition to the input-view results, we render the generated animations from two novel viewpoints.

APPENDIX F

GRAIL Technical Details and Additional Results

F.1 Qualitative Results

We provide a supplementary video showcasing extensive and diverse results generated by GRAIL. [Figure F.1](#) provides a snapshot of the results. The video covers a wide range of skills, including pick-up (approaching and grasping objects), dexterous manipulation (carrying, pushing, hoisting, and pulling), sitting (whole-body interaction with furniture), and terrain traversal (stairs, slopes, and curbs). We encourage readers to view the video for a comprehensive overview of our method’s capabilities.

F.2 Implementation Details

F.2.1 Runtime Performance

[Table F.1](#) reports the wall-clock time for each stage of the 4D HOI generation pipeline per sequence (5 second video with 121 frames), measured on a single NVIDIA A100 GPU. The overall pipeline takes approximately 14 minutes per sequence. Video generation and initial motion estimation (human and object) together account for about 4 minutes. The optimization preprocessing stage, which runs MoGe-2 ([Wang et al., 2025b](#)) for metric depth estimation and SAM2 ([Ravi et al., 2024](#)) for human and object segmentation to produce per-frame point clouds as optimization targets, takes roughly 2 minutes. The trajectory refinement stage dominates at approximately 8 minutes, as it jointly optimizes human and object trajectories across all frames.

For the RL stage, fine-tuning the latent adaptor takes approximately 12–24 hours



Figure F.1: GRAIL Results Overview. A snapshot of the diverse loco-manipulation skills generated by GRAIL, spanning pick-up, dexterous manipulation, sitting, and terrain traversal tasks.

Stage	Time
Video Generation (Kling API)	~ 1 min
Human Motion Estimation	~ 2 min
Object Pose Tracking	~ 1 min
Optimization Preprocessing	~ 2 min
Trajectory Refinement	~ 8 min
Physics-based Skill Learning	~ 10 min (amortized)
Total	~ 24 min

Table F.1: Runtime breakdown of the GRAIL pipeline per sequence on a single GPU.

on one NVIDIA A100 GPU. However, by grouping objects with similar geometry and interaction patterns (e.g., mugs, bottles) and co-training on shared terrain configurations, a single policy can cover 200+ objects. The amortized training cost is therefore under 10 GPU-minutes per object.

F.2.2 Video Generation

We construct two candidate scene configurations using Infinigen (Raistrick et al., 2023): an indoor floor-only environment and a furnished room with a table (Figure F.2). For each input object, a VLM (OpenAI, 2024) determines whether it should be placed on the floor (e.g., a sofa) or on the table (e.g., a frypan) based on its typical affordances. We pair the object with a human asset sized to match the Unitree G1 (1.3m height), render

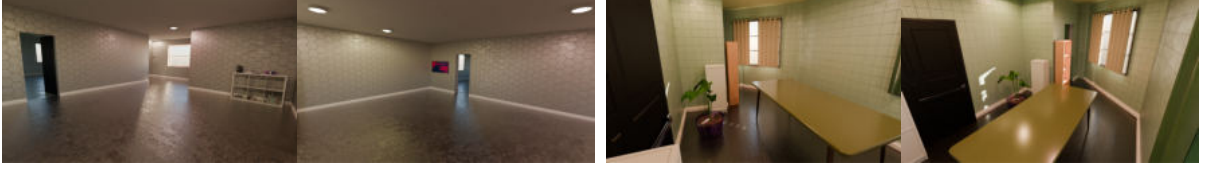


Figure F.2: Candidate Scene Configurations.

the initial frame, and use a VLM (OpenAI, 2024) to generate a text prompt describing the intended interaction. The rendered frame and prompt are then passed to Kling 2.5 Turbo Pro (Technology, 2025), which supports generating 5 or 10 second videos at 24 fps with resolution of 1920×1080 . We can optionally render an ending frame to control the final positions of the human and object, and produce longer sequences auto-regressively by feeding the last frame of each segment as the initial frame of the next.

F.2.3 Initial Motion Estimation

Hand Pose Reconstruction: We apply WiLoR (Potamias et al., 2025) to estimate per-frame MANO (Romero et al., 2017) hand parameters. Missing detections due to occlusion are filled via temporal interpolation, and the resulting sequence is smoothed with a Savitzky-Golay filter (Savitzky and Golay, 1964) to suppress per-frame jitter.

Object Pose Tracking: We fine-tune FoundationPose (Wen et al., 2024) on its synthetic dataset for 5 epochs. To adapt to our RGB-only setup, we zero out the input depth channels while retaining the original architecture.

F.2.4 Interaction-Aware Trajectory Refinement

We provide additional details on the interaction-aware trajectory refinement. Rather than optimizing full trajectories directly, we optimize residual motion parameters $\{\Delta\Theta_t^{\mathcal{H}}\}_{t=1}^T$ and $\{\Delta\Theta_t^{\mathcal{O}}\}_{t=1}^T$. The final poses are computed as $\Theta_t^{\mathcal{H}} = \hat{\Theta}_t^{\mathcal{H}} \oplus \Delta\Theta_t^{\mathcal{H}}$ and $\Theta_t^{\mathcal{O}} = \hat{\Theta}_t^{\mathcal{O}} \oplus \Delta\Theta_t^{\mathcal{O}}$, with $\oplus$ denoting residual translation and rotation updates. We adopt the 6D rotation representation (Zhou et al., 2019) for continuous parameterization.

The regularization term L_{reg} referenced in the main text is:

$$L_{\text{reg}} = L_{\text{foot}} + L_{\text{vel}} + L_{\text{smooth}}. \quad (\text{F.1})$$

Foot contact loss (L_{foot}). Initial motion estimates often exhibit foot skating artifacts. We leverage per-frame foot contact labels from GENMO (Li et al., 2025a) and penalize foot vertex displacement during detected contact frames to enforce stationary foot constraints.

Velocity regularization (L_{vel}). While the camera-space motion from GENMO aligns well in image space, it oscillates along the depth direction due to depth-scale ambiguity. We regularize the optimized pelvis velocity to match GENMO’s global-space velocity estimate, suppressing depth-direction drift.

Smoothness loss (L_{smooth}). We penalize the first- and second-order temporal finite differences of the positions of the human and object mesh vertices, ensuring temporally coherent trajectory.

F.2.5 Physics-based Humanoid Skill Learning

Adaptor Architecture: The adaptor policy π_ϕ is a 3-layer MLP with hidden dimensions [512, 256, 128] and SiLU activations. The policy outputs a 66-dimensional meta-action: 64 dimensions for the latent residual (matching SONIC’s token dimension of 2×32) and 2 dimensions for left/right hand primitives. The residual is scaled by $\lambda = 0.1$ and added to SONIC’s encoder output before finite scalar quantization (pre-quantization mode), then decoded to 29 body joint position targets. Each hand primitive is passed through a sigmoid and thresholded to produce a binary open/close signal, which is mapped to 7 finger joint positions per hand via predefined grasp configurations. The critic is a separate 3-layer MLP [512, 256, 128] that receives privileged observations including full body state and ground-truth contact flags.

Observations: The policy observes the robot’s proprioceptive state (joint positions, velocities, base angular velocity, gravity direction, and previous actions, each with 10-

frame history) concatenated with object information: the current object pose (and table pose) in body frame, hand-to-object transforms (3D position + 6D rotation), finger tip contact forces, and delta observations encoding the difference between reference future object poses and the current simulated pose. The delta observation formulation ($\text{ref}_{\text{future}} - \text{sim}_{\text{current}}$) provides the policy with an error signal that directly encodes what needs to be corrected.

Training: Each motion sequence trains its own adaptor independently using PPO (Schulman et al., 2017) with 4,096 parallel environments per GPU in Isaac Lab. We run 20,000 PPO iterations per sequence with actor learning rate 2×10^{-5} (adaptive schedule with target $\text{KL} = 0.01$), critic learning rate 10^{-3} , discount factor $\gamma = 0.99$, GAE parameter $\lambda_{\text{GAE}} = 0.95$, clipping parameter $\epsilon = 0.2$, entropy coefficient 0.01, 24 steps per environment, 5 learning epochs with 4 mini-batches, and maximum gradient norm 0.1. The pre-trained SONIC encoder and decoder remain frozen throughout training. Domain randomization is disabled to isolate contact-sensitive behaviors.

Reward Weights and Termination: Object tracking weight $w_{\text{obj}} = 20.0$ (with position sub-weight $w_{\text{op}} = 0.5$ and orientation sub-weight $w_{\text{or}} \in \{0.3, 0.8\}$, contact-gated), grasp contact count weight $w_{\text{grasp}} = 40.0$ (left hand scaled by $0.5\times$), finger direction weight 10.0 (left hand 5.0), contact center weight 0.1 (left hand 0.05), anchor orientation weight 2.5, relative body orientation weight 5.0 with wrist tracking boost 1.0, undesired contact penalty -0.1 , hand-table contact penalty -0.5 , and action smoothness penalties (-0.01 for both meta-action rate and full latent rate). An auxiliary ℓ_2 loss on the latent residual (weight 0.1) regularizes the adaptor toward the pre-trained SONIC behavior. Episodes terminate if the object’s z-position deviates by more than 0.4 m from the reference, the root height deviates by more than 0.25 m, or the root orientation error exceeds 1.0 rad.

F.2.6 Scene-Aware Humanoid Control

For scene-level interactions such as stepping over curbs or traversing slopes, SONIC’s flat-ground prior is insufficient. Instead of attaching a latent adaptor, we directly fine-tune the SONIC foundation model on reconstructed scene-aware trajectories and augment its observation with a local height map that provides 3D terrain perception.

Height Map Observation: We construct an 11×11 height map grid centered on the robot with a total extent of 1.5 m and a resolution of 0.15 m. At each grid point, a downward ray is cast against the scene mesh to obtain the terrain hit position. The resulting 3D hit positions are transformed into the robot’s yaw-aligned local frame, yielding a $(11, 11, 3)$ tensor that encodes the local terrain geometry around the robot.

Height Map Encoder: The height map is processed by a 3-layer CNN with channels $[64, 128, 256]$, kernel size 3×3 , stride 2, and LeakyReLU activations. The spatial dimensions reduce as $11 \rightarrow 6 \rightarrow 3 \rightarrow 2$, and the output is flattened to a 1,024-dimensional feature vector. This vector is concatenated with the proprioceptive observation and the SONIC tokenizer features, and passed through a fusion MLP ($[256]$, SiLU) that produces the latent input to SONIC’s motion decoder.

Training: We fine-tune using the SONIC universal tokenizer configuration with 4,096 parallel environments per GPU in Isaac Lab for 20,000 PPO iterations. The robot is modeled as the G1 humanoid with mesh colliders. Since these tasks do not involve hand-object interaction, we train with only the motion tracking reward R_t^{motion} .

F.2.7 Failure Case Filtering

While image-to-video models produce realistic HOI sequences, they may introduce artifacts such as texture inconsistencies or geometry mismatches across frames, causing Foundation-Pose to lose tracking. To automatically filter such failures, we compare SAM2 (Ravi et al.,

2024) object masks $\{\widehat{\mathcal{M}}_t\}_{t=1}^T$ against rendered silhouettes $\{\mathcal{M}_t\}_{t=1}^T$ from the predicted poses, and compute the mask tracking error:

$$e^{\mathcal{M}} = \sum_{t=1}^T \frac{\text{Sum}\left((1 - \mathcal{M}_t) \cdot \widehat{\mathcal{M}}_t\right)}{\text{Sum}(\mathcal{M}_t)}, \quad (\text{F.2})$$

where $\text{Sum}(\cdot)$ counts non-zero pixels. This measures the fraction of SAM2-tracked mask pixels not covered by the predicted mask. We discard sequences where $e^{\mathcal{M}}$ exceeds $\tau = 0.2$, effectively removing cases caused by fast motion, blurry frames, or inconsistent object appearance.

F.3 Experiment Details

F.3.1 Experiment Settings

Baseline Settings: We compare against training-based and training-free 4D HOI generation approaches. Training-based baselines include CHOIS (Li et al., 2024a), a controllable motion generation framework guided by language and sparse waypoints, and HOIDiff (Peng et al., 2025), a diffusion-based model for affordance-conditioned HOI synthesis. For training-free comparison, we evaluate DAViD (Kim et al., 2025), which generates the first frame using an image generative model (Labs et al., 2025) and produces the video from that frame. Since image generation often fails under partial control signals (e.g., Canny edge maps), we generate 24 images per object and manually select a successful result as the starting frame. To ensure fairness, DAViD uses Kling’s image-to-video model under the same setting as our approach.

Metrics for Physical Plausibility Evaluation: For the physical plausibility evaluation, we apply InterMimic (Xu et al., 2025) to reproduce each method’s 4D HOI sequences in physics simulation and report trajectory deviations between the simulated and reconstructed trajectories. We generate humanoids using capsule primitives that conform to the body shape of the input SMPL-X motion, without the need for motion

Methods	Affordance Real.	Physical Real.
HOIDiff	2.0%	1.9%
CHOIS	12.2%	16.8%
DAViD	11.2%	10.4%
Ours	74.7%	70.9%

Table F.2: User Study on 4D HOI Quality. Pick rates across affordance realism and physical realism. In each trial, participants select the best result from 3 of the 4 methods (theoretical upper bound 75%).

retargeting. For the humanoid, we report the mean per-joint position deviation across all frames for the full body (Full-body Dev.) and hands only (Hand Dev.). For the object, we report the mean surface point deviation (Obj Dev.). We define the tracking success rate (Succ. Rate) as the fraction of the sequence where the normalized full-body and object deviations (divided by the object’s maximum dimension) are both below 0.25.

F.3.2 Additional Results

Visualization for Physical Plausibility Evaluation: Figure F.3 presents the results for the physical plausibility evaluation of our method and all baselines. Our generated sequences are visually more plausible and also achieve substantially higher tracking success rates, reflecting their superior physical plausibility.

User Study: We conducted a user study with 30 participants to assess perceptual quality of the generated 4D HOI results. In each trial, participants viewed sequences from three of the four methods, drawn at random, and select the result with (1) the most physically plausible motion and (2) the most appropriate object affordances. As shown in Table F.2, our method was consistently preferred, with pick rates of 74.7% for affordance realism and 70.9% for physical realism, confirming that the generated interactions are both physically convincing and semantically meaningful.

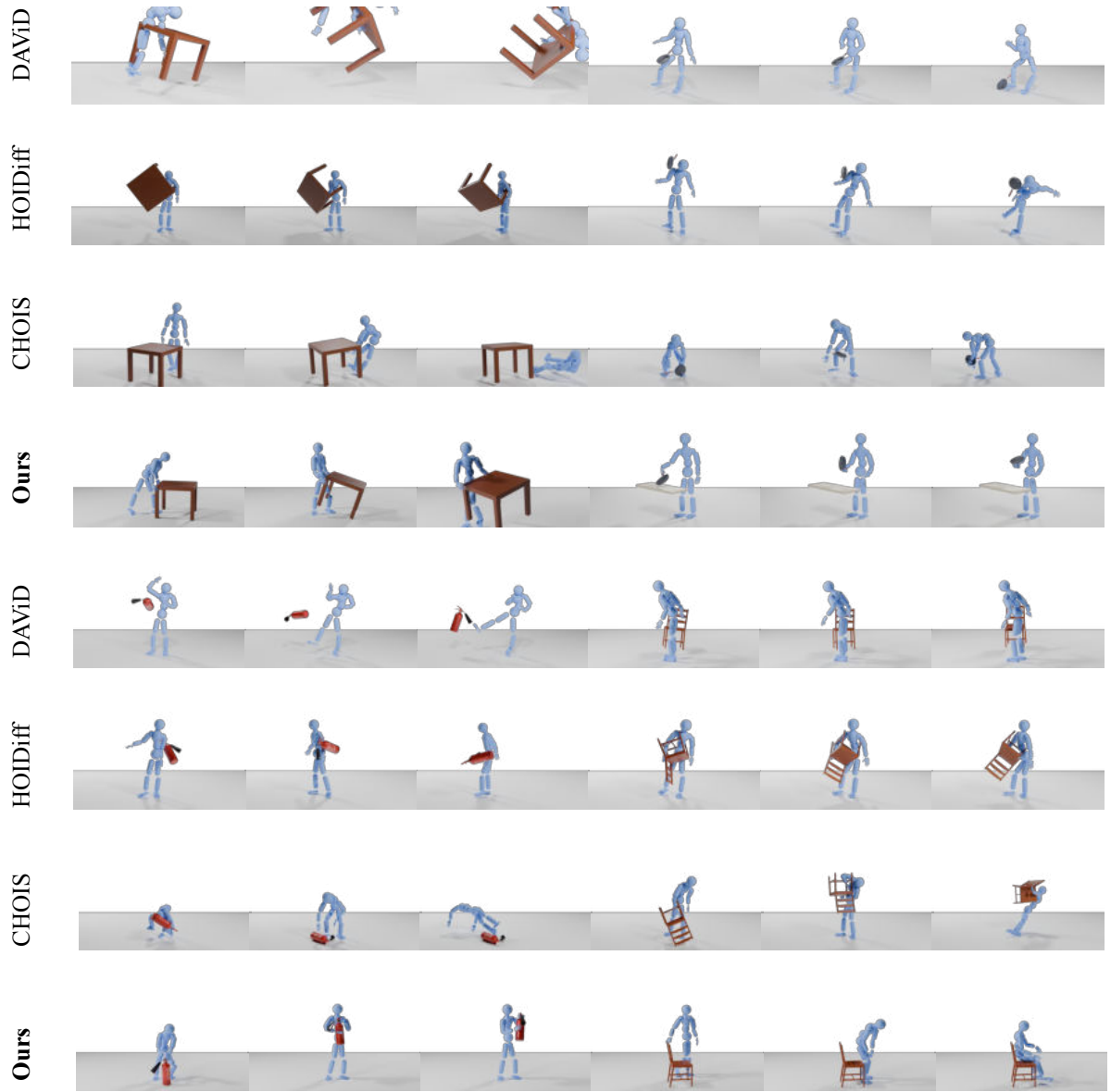


Figure F.3: Visualization for Physical Plausibility Evaluation.

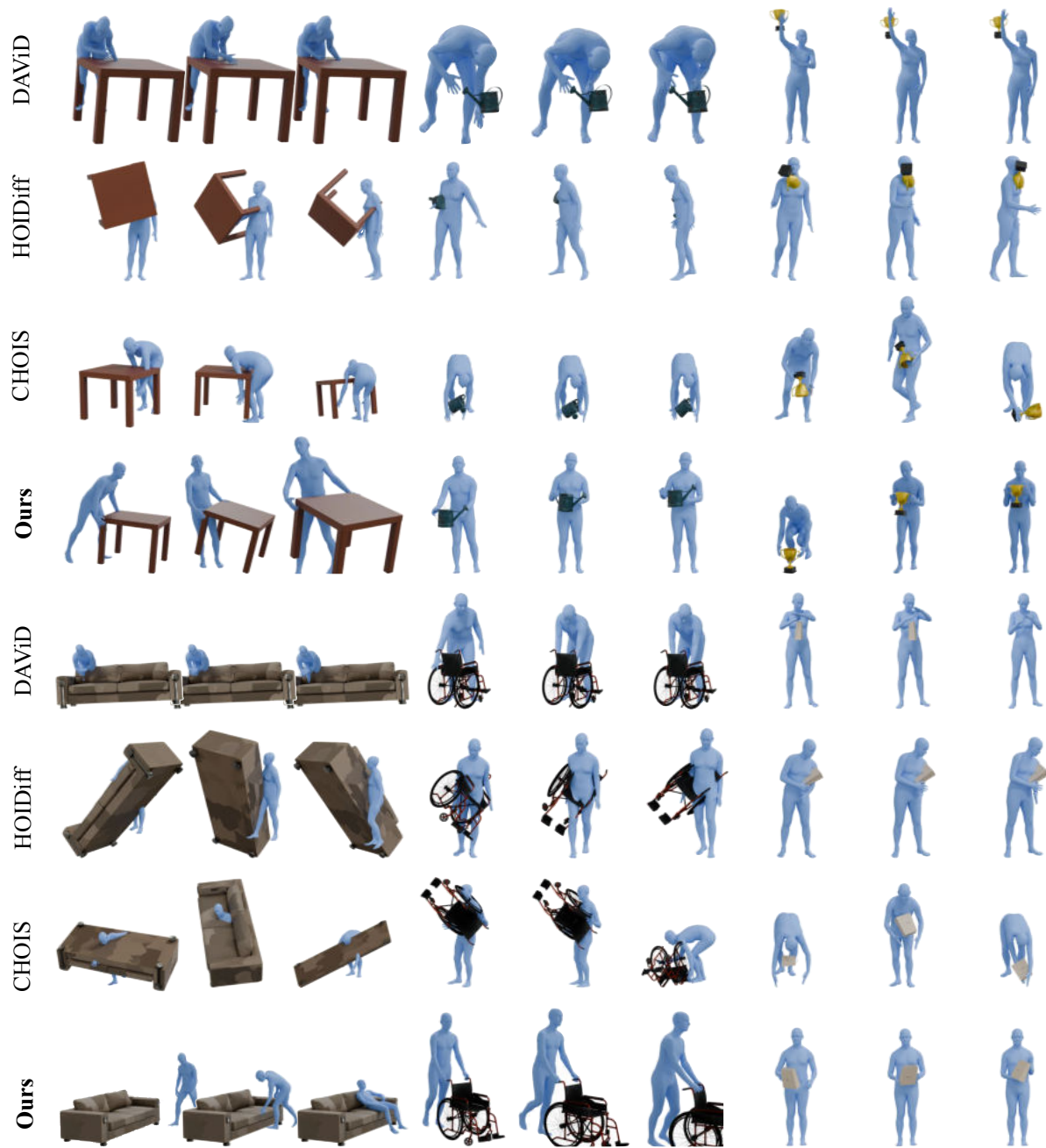


Figure F.4: Additional Qualitative Comparison. We provide additional comparison results against DAViD (Kim et al., 2025), HOIDiff (Peng et al., 2025) and CHOIS (Li et al., 2024a).

Additional Qualitative Comparison: Figure F.4 presents additional qualitative comparisons between our generated 4D HOI sequences and those produced by baseline methods.

REFERENCES

- Akinci, N., Cornelis, J., Akinci, G., and Teschner, M. (2013). Coupling elastic solids with smoothed particle hydrodynamics fluids. *Computer Animation and Virtual Worlds*, 24(3-4):195–203. [12](#), [31](#)
- Akinci, N., Ihmsen, M., Akinci, G., Solenthaler, B., and Teschner, M. (2012). Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics (ToG)*, 31(4):1–8. [8](#), [12](#), [21](#), [31](#)
- Aldaco, J., Armstrong, T., Baruch, R., Bingham, J., Chan, S., Draper, K., Dwibedi, D., Finn, C., Florence, P., Goodrich, S., et al. (2024). ALOHA 2: An enhanced low-cost hardware for bimanual teleoperation. *arXiv preprint arXiv:2405.02292*. [140](#)
- Anciukevičius, T., Xu, Z., Fisher, M., Henderson, P., Bilen, H., Mitra, N. J., and Guerrero, P. (2023). RenderDiffusion: Image diffusion for 3D reconstruction, inpainting and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12608–12618. [43](#)
- Bahmani, S., Liu, X., Yifan, W., Skorokhodov, I., Rong, V., Liu, Z., Liu, X., Park, J. J., Tulyakov, S., Wetzstein, G., et al. (2024a). TC4D: Trajectory-conditioned text-to-4D generation. In *European Conference on Computer Vision*, pages 53–72. Springer. [123](#)
- Bahmani, S., Skorokhodov, I., Rong, V., Wetzstein, G., Guibas, L., Wonka, P., Tulyakov, S., Park, J. J., Tagliasacchi, A., and Lindell, D. B. (2024b). 4D-fy: Text-to-4D generation using hybrid score distillation sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7996–8006. [123](#)
- Bahmani, S., Skorokhodov, I., Rong, V., Wetzstein, G., Guibas, L., Wonka, P., Tulyakov, S., Park, J. J., Tagliasacchi, A., and Lindell, D. B. (2024c). 4D-fy: Text-to-4D generation using hybrid score distillation sampling. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [124](#)
- Bangor, A., Kortum, P., and Miller, J. (2009). Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of Usability Studies*, 4(3):114–123. [101](#), [102](#)
- Bao, C., Zhang, Y., Yang, B., Fan, T., Yang, Z., Bao, H., Zhang, G., and Cui, Z. (2023). Sine: Semantic-driven image-based nerf editing with prior-guided editing field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20919–20929. [84](#)
- Baraff, D. (2001). Physically based modeling: Rigid body simulation. *ACM SIG-GRAPH Course Notes*, 2(1):2–1. [44](#)
- Baran, I. and Popović, J. (2007). Automatic rigging and animation of 3D characters. *ACM Transactions on Graphics (ToG)*, 26(3):72–es. [125](#)

- Barbič, J., da Silva, M., and Popović, J. (2009). Deformable object animation using reduced optimal control. In *ACM SIGGRAPH 2009 papers*, pages 1–9. 108
- Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., and Hedman, P. (2022). Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479. 63
- Batty, C., Bertails, F., and Bridson, R. (2007). A fast variational framework for accurate solid-fluid coupling. *ACM Transactions on Graphics (ToG)*, 26(3):100–es. 10, 13
- Batty, C., Uribe, A., Audoly, B., and Grinspun, E. (2012). Discrete viscous sheets. *ACM Transactions on Graphics (ToG)*, 31(4):1–7. 11
- Bautista, M. A., Guo, P., Abnar, S., Talbott, W., Toshev, A., Chen, Z., Dinh, L., Zhai, S., Goh, H., Ulbricht, D., et al. (2022). Gaudi: A neural architect for immersive 3D scene generation. *Advances in Neural Information Processing Systems*, 35:25102–25116. 43
- Becker, M., Ihmsen, M., and Teschner, M. (2009a). Corotated SPH for deformable solids. In *NPH*, pages 27–34. 12
- Becker, M. and Teschner, M. (2007). Weakly compressible SPH for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 209–217. 11, 14
- Becker, M., Tessendorf, H., and Teschner, M. (2009b). Direct forcing for Lagrangian rigid-fluid coupling. *IEEE Transactions on Visualization and Computer Graphics*, 15(3):493–503. 11, 21
- Ben, Q., Jia, F., Zeng, J., Dong, J., Lin, D., and Pang, J. (2025). HOMIE: Humanoid loco-manipulation with isomorphic exoskeleton cockpit. *arXiv preprint arXiv:2502.13013*. 140, 143
- Bencheikroun, O., Zhang, J. E., Chaudhuri, S., Grinspun, E., Zhou, Y., and Jacobson, A. (2023). Fast complementary dynamics via skinning eigenmodes. *arXiv preprint arXiv:2303.11886*. 126
- Bender, J. and Koschier, D. (2015). Divergence-free smoothed particle hydrodynamics. In *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 147–155. 11, 14, 31, 32
- Bender, J. and Koschier, D. (2016). Divergence-free SPH for incompressible and viscous fluids. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1193–1206. 16
- Bender, J., Kugelstadt, T., Weiler, M., and Koschier, D. (2019). Volume maps: An implicit boundary representation for SPH. In *Motion, Interaction and Games*, pages 1–10. 12, 31

- Bhatnagar, B. L., Xie, X., Petrov, I., Sminchisescu, C., Theobalt, C., and Pons-Moll, G. (2022). BEHAVE: Dataset and method for tracking human object interactions. In *CVPR*. 142, 143
- Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al. (2023a). Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv:2311.15127*. 105
- Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., Kim, S. W., Fidler, S., and Kreis, K. (2023b). Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575. 107
- Blinn, B., Ding, A., Jones, R. K., Savva, M., Sridhar, S., and Ritchie, D. (2021). Learning body-aware 3D shape generative models. *arXiv preprint arXiv:2112.07022*. 43
- Bonet, J. and Lok, T.-S. (1999). Variational and momentum preservation aspects of smooth particle hydrodynamic formulations. *Computer Methods in Applied Mechanics and Engineering*, 180(1-2):97–115. 12
- Bonet, J. and Wood, R. D. (1997). *Nonlinear continuum mechanics for finite element analysis*. Cambridge university press. 66
- Bouaziz, S., Martin, S., Liu, T., Kavan, L., and Pauly, M. (2014). Projective dynamics: Fusing constraint projections for fast simulation. *ACM Transactions on Graphics (ToG)*, 33(4). 126
- Bouaziz, S., Martin, S., Liu, T., Kavan, L., and Pauly, M. (2023). Projective dynamics: Fusing constraint projections for fast simulation. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 1–11, New York, NY, USA. Association for Computing Machinery. 126
- Brandt, C., Scandolo, L., Eisemann, E., and Hildebrandt, K. (2019). The reduced immersed method for real-time fluid-elastic solid interaction and contact simulation. *ACM Transactions on Graphics (ToG)*, 38(6):1–16. 11
- Bridson, R. (2015). *Fluid Simulation for Computer Graphics*. AK Peters/CRC Press. 13
- Cao, Z., Hong, F., Wu, T., Pan, L., and Liu, Z. (2023). Large-vocabulary 3D diffusion model with transformer. *arXiv preprint arXiv:2309.07920*. 43
- Cen, J., Fang, J., Yang, C., Xie, L., Zhang, X., Shen, W., and Tian, Q. (2023). Segment any 3D Gaussians. *arXiv preprint arXiv:2312.00860*. 80
- Chai, W., Guo, X., Wang, G., and Lu, Y. (2023). StableVideo: Text-driven consistency-aware diffusion video editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23040–23050. 115

- Chan, C., Durand, F., and Isola, P. (2022). Learning to generate line drawings that convey geometry and semantics. In *CVPR*. 116
- Chandrasekhar, S. (1967). Ellipsoidal figures of equilibrium—an historical account. *Communications on Pure and Applied Mathematics*, 20(2):251–265. 70
- Chen, B., Zhang, T., Geng, H., Song, K., Zhang, C., Li, P., Freeman, W. T., Malik, J., Abbeel, P., Tedrake, R., et al. (2025a). Large video planner enables generalizable robot control. *arXiv preprint arXiv:2512.15840*. 140
- Chen, H., Gu, J., Chen, A., Tian, W., Tu, Z., Liu, L., and Su, H. (2023a). Single-stage diffusion NeRF: A unified approach to 3D generation and reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2416–2425. 43
- Chen, H., Zhang, Y., Cun, X., Xia, M., Wang, X., Weng, C., and Shan, Y. (2024a). VideoCrafter2: Overcoming data limitations for high-quality video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7310–7320. 1
- Chen, H.-y., Tretschk, E., Stuyck, T., Kadlecsek, P., Kavan, L., Vouga, E., and Lassner, C. (2022). Virtual elastic objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15827–15837. 44, 64
- Chen, J., Zhang, Y., Kang, D., Zhe, X., Bao, L., Jia, X., and Lu, H. (2021). Animatable neural radiance fields from monocular rgb videos. *arXiv preprint arXiv:2106.13629*. 83
- Chen, J.-K., Lyu, J., and Wang, Y.-X. (2023b). NeuralEditor: Editing neural radiance fields via manipulating point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12439–12448. 64, 69
- Chen, L.-H., Zhang, Y., Yin, Z., Dou, Z., Chen, X., Wang, J., Komura, T., and Zhang, L. (2025b). Motion2Motion: Cross-topology motion transfer with sparse correspondence. 126
- Chen, R., Chen, Y., Jiao, N., and Jia, K. (2023c). Fantasia3D: Disentangling geometry and appearance for high-quality text-to-3D content creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22246–22256. 41, 43
- Chen, W., Ji, Y., Wu, J., Wu, H., Xie, P., Li, J., Xia, X., Xiao, X., and Lin, L. (2023d). Control-a-Video: Controllable text-to-video generation with diffusion models. *arXiv:2305.13840*. 107
- Chen, X., Wang, Y., Zhang, L., Zhuang, S., Ma, X., Yu, J., Wang, Y., Lin, D., Qiao, Y., and Liu, Z. (2023e). SEINE: Short-to-long video diffusion model for generative transition and prediction. In *The Twelfth International Conference on Learning Representations*. 108
- Chen, Y., Chen, Z., Zhang, C., Wang, F., Yang, X., Wang, Y., Cai, Z., Yang, L., Liu, H., and Lin, G. (2023f). GaussianEditor: Swift and controllable 3D editing with Gaussian splatting. *arXiv preprint arXiv:2311.14521*. 84

- Chen, Y., Davis, T. A., Hager, W. W., and Rajamanickam, S. (2008). Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software (TOMS)*, 35(3):1–14. [25](#)
- Chen, Y., Xie, T., Zong, Z., Li, X., Gao, F., Yang, Y., Wu, Y. N., and Jiang, C. (2024b). Atlas3D: Physically constrained self-supporting text-to-3D for simulation and fabrication. *Advances in Neural Information Processing Systems*, 37:102501–102530. [2](#)
- Chen, Z., Wang, Y., Wang, F., Wang, Z., and Liu, H. (2024c). V3D: Video diffusion models are effective 3D generators. *arXiv preprint arXiv:2403.06738*. [42](#), [107](#)
- Cheng, H. K., Oh, S. W., Price, B., Lee, J.-Y., and Schwing, A. (2023). Putting the object back into video object segmentation. *arXiv preprint arXiv:2310.12982*. [88](#), [100](#)
- Chu, Z., Xiong, F., Liu, M., Zhang, J., Shao, M., Sun, Z., Wang, D., and Xu, M. (2024). HumanRig: Learning automatic rigging for humanoid character in a large scale dataset. [125](#)
- Clausen, P., Wicke, M., Shewchuk, J. R., and O’Brien, J. F. (2013). Simulating liquids and solid-liquid interactions with Lagrangian meshes. *ACM Transactions on Graphics (ToG)*, 32(2):1–15. [11](#)
- Coros, S., Martin, S., Thomaszewski, B., Schumacher, C., Sumner, R., and Gross, M. (2012). Deformable objects alive! *ACM Transactions on Graphics (ToG)*, 31(4):1–9. [108](#)
- Cui, J., Liu, T., Liu, N., Yang, Y., Zhu, Y., and Huang, S. (2024). AnySkill: Learning open-vocabulary physical skill for interactive agents. In *CVPR*. [143](#)
- Dai, S., Su, X., Wan, B., Hu, R., and Xu, K. (2025). TextMesh4D: High-quality text-to-4D mesh generation. [124](#)
- Dang, L., Shao, R., Zhang, H., Min, W., Liu, Y., and Wu, Q. (2025). SViMo: Synchronized diffusion for video and motion generation in hand-object interaction scenarios. In *NeurIPS*. [142](#)
- Denavit, J. and Hartenberg, R. S. (1955). A kinematic notation for lower-pair mechanisms based on matrices. *American Society of Mechanical Engineers*. [128](#)
- Deng, N., He, Z., Ye, J., Duinkharjav, B., Chakravarthula, P., Yang, X., and Sun, Q. (2022). FoV-NeRF: Foveated neural radiance fields for virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, 28(11):3854–3864. [85](#), [86](#)
- Deng, Y., Xiong, H., Feng, B., Wang, X., and Liu, W. (2025). STP4D: Spatio-temporal prompt consistent modeling for text-to-4D Gaussian splatting. [124](#)
- Deul, C., Charrier, P., and Bender, J. (2016). Position-based rigid-body dynamics. *Computer Animation and Virtual Worlds*, 27(2):103–112. [98](#)

- Dhariwal, P. and Nichol, A. (2021). Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794. 106
- Diller, C. and Dai, A. (2024). CG-HOI: Contact-guided 3D human-object interaction generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 142
- Dong, Y., Zuo, Q., Gu, X., Yuan, W., Zhao, Z., Dong, Z., Bo, L., and Huang, Q. (2024). GPLD3D: Latent diffusion of 3D shape generative models by edupont2022datanforcing geometric and physical priors. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2024)*. IEEE/CVF. 43
- Du, Y., Yang, M., Dai, B., Dai, H., Nachum, O., Tenenbaum, J. B., Schuurmans, D., and Abbeel, P. (2023). Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 36:9156–9172. 140
- Duisterhof, B. P., Mandi, Z., Yao, Y., Liu, J.-W., Shou, M. Z., Song, S., and Ichnowski, J. (2023). MD-Splatting: Learning metric deformation from 4D Gaussians in highly deformable scenes. *arXiv preprint arXiv:2312.00583*. 84
- Dupont, E., Kim, H., Eslami, S., Rezende, D., and Rosenbaum, D. (2022). From data to functa: Your data point is a function and you can treat it like one. *arXiv preprint arXiv:2201.12204*. 43
- Dwivedi, S. K., Antić, D., Tripathi, S., Taheri, O., Schmid, C., Black, M. J., and Tzionas, D. (2025). InteractVLM: 3D interaction reasoning from 2D foundational models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22605–22615. 142
- Eisenberger, M., Novotny, D., Kerchenbaum, G., Labatut, P., Neverova, N., Cremers, D., and Vedaldi, A. (2021). NeuroMorph: Unsupervised shape interpolation and correspondence in one go. 126
- Erkoç, Z., Ma, F., Shan, Q., Nießner, M., and Dai, A. (2023). HyperDiffusion: Generating implicit neural fields with weight-space diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14300–14310. 43
- Erkoç, Z., Ma, F., Shan, Q., Nießner, M., and Dai, A. (2023). HyperDiffusion: Generating implicit neural fields with weight-space diffusion. 124
- Esser, P., Chiu, J., Atighehchian, P., Granskog, J., and Germanidis, A. (2023). Structure and content-guided video synthesis with diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7346–7356. 107
- Fan, Z., Taheri, O., Tzionas, D., Kocabas, M., Kaufmann, M., Black, M. J., and Hilliges, O. (2023). ARCTIC: A dataset for dexterous bimanual hand-object manipulation. In *CVPR*. 142

- Fang, J., Wang, J., Zhang, X., Xie, L., and Tian, Q. (2023). GaussianEditor: Editing 3D Gaussians delicately with text instructions. *arXiv preprint arXiv:2311.16037*. 84
- Fang, Y., Qu, Z., Li, M., Zhang, X., Zhu, Y., Aanjaneya, M., and Jiang, C. (2020). IQ-MPM: An interface quadrature material point method for non-sticky strongly two-way coupled nonlinear solids and fluids. *ACM Transactions on Graphics (ToG)*, 39(4):51–1. 9, 13
- Fedkiw, R. P. (2002). Coupling an Eulerian fluid calculation to a Lagrangian solid calculation with the ghost fluid method. *Journal of Computational Physics*, 175(1):200–224. 10
- Fedkiw, R. P., Aslam, T., Merriman, B., and Osher, S. (1999). A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *Journal of Computational Physics*, 152(2):457–492. 10
- Fei, Y., Batty, C., Grinspun, E., and Zheng, C. (2018). A multi-scale model for simulating liquid-fabric interactions. *ACM Transactions on Graphics (ToG)*, 37(4):1–16. 12
- Feng, Y., Feng, X., Shang, Y., Jiang, Y., Yu, C., Zong, Z., Shao, T., Wu, H., Zhou, K., Jiang, C., et al. (2024a). Gaussian splashing: Dynamic fluid synthesis with Gaussian splatting. *arXiv preprint arXiv:2401.15318*. 103
- Feng, Y., Shang, Y., Feng, X., Lan, L., Zhe, S., Shao, T., Wu, H., Zhou, K., Su, H., Jiang, C., and Yang, Y. (2024b). ElastoGen: 4D generative elastodynamics. 125
- Feng, Y., Shang, Y., Li, X., Shao, T., Jiang, C., and Yang, Y. (2024c). PIE-NeRF: Physics-based interactive elastodynamics with nerf. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 82, 84
- Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., and Kanazawa, A. (2022). Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510. 63
- Fu, Z., Zhao, Q., Wu, Q., Wetzstein, G., and Finn, C. (2024). HumanPlus: Humanoid shadowing and imitation from humans. In *Conference on Robot Learning (CoRL)*. 143
- Gabbard, J. L., Hix, D., and Swan, J. E. (1999). User-centered design and evaluation of virtual environments. *IEEE computer Graphics and Applications*, 19(6):51–59. 87
- Gal, R., Vinker, Y., Alaluf, Y., Bermano, A., Cohen-Or, D., Shamir, A., and Chechik, G. (2024). Breathing life into sketches using text-to-video priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4325–4336. 107
- Ganzenmüller, G. C. (2015). An hourglass control algorithm for Lagrangian smooth particle hydrodynamics. *Computer Methods in Applied Mechanics and Engineering*, 286:87–106. 12

- Gao, J., Wang, Z., Xiao, Z., Wang, J., Wang, T., Cao, J., Hu, X., Liu, S., Dai, J., and Pang, J. (2024). CooHOI: Learning cooperative human-object interaction with manipulated object dynamics. In *NeurIPS*. 143
- Gao, L., Sun, J.-M., Mo, K., Lai, Y.-K., Guibas, L. J., and Yang, J. (2023). SceneHGN: Hierarchical graph networks for 3D indoor scene generation with fine-grained geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 83
- Gao, M., Pradhana, A., Han, X., Guo, Q., Kot, G., Sifakis, E., and Jiang, C. (2018a). Animating fluid sediment mixture in particle-laden flows. *ACM Transactions on Graphics (ToG)*, 37(4):1–11. 12
- Gao, M., Wang, X., Wu, K., Pradhana, A., Sifakis, E., Yuksel, C., and Jiang, C. (2018b). GPU optimization of material point methods. *ACM Transactions on Graphics (ToG)*, 37(6):1–12. 64
- Gast, T. F., Schroeder, C., Stomakhin, A., Jiang, C., and Teran, J. M. (2015). Optimization integrator for large time steps. *IEEE Transactions on Visualization and Computer Graphics*, 21(10):1103–1115. 132
- Ge, S., Nah, S., Liu, G., Poon, T., Tao, A., Catanzaro, B., Jacobs, D., Huang, J.-B., Liu, M.-Y., and Balaji, Y. (2023). Preserve your own correlation: A noise prior for video diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22930–22941. 107
- Geyer, M., Bar-Tal, O., Bagon, S., and Dekel, T. (2023a). TokenFlow: Consistent diffusion features for consistent video editing. *arxiv:2307.10373*. 118
- Geyer, M., Bar-Tal, O., Bagon, S., and Dekel, T. (2023b). TokenFlow: Consistent diffusion features for consistent video editing. *arXiv:2307.10373*. 134
- Gilland, J. (2012). *Elemental Magic, Volume 2: The Technique of Special Effects Animation*. Routledge. 108, 113
- Gissler, C., Peer, A., Band, S., Bender, J., and Teschner, M. (2019). Interlinked SPH pressure solvers for strong fluid-rigid coupling. *ACM Transactions on Graphics (ToG)*, 38(1):1–13. 8, 12
- Griffiths, D. J. and Schroeter, D. F. (2018). *Introduction to quantum mechanics*. Cambridge university press. 48
- Gu, Y., Zhou, Y., Wu, B., Yu, L., Liu, J.-W., Zhao, R., Wu, J. Z., Zhang, D. J., Shou, M. Z., and Tang, K. (2024). VideoSwap: Customized video subject swapping with interactive semantic point correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7621–7630. 107
- Guédon, A. and Lepetit, V. (2023). Sugar: Surface-aligned Gaussian splatting for efficient 3D mesh reconstruction and high-quality mesh rendering. *arXiv preprint arXiv:2311.12775*. 80

- Guendelman, E., Selle, A., Losasso, F., and Fedkiw, R. (2005). Coupling water and smoke to thin deformable and rigid shells. *ACM Transactions on Graphics (ToG)*, 24(3):973–981. 10
- Guo, J., Zhang, D., Liu, X., Zhong, Z., Zhang, Y., Wan, P., and Zhang, D. (2024). LivePortrait: Efficient portrait animation with stitching and retargeting control. *arXiv preprint arXiv:2407.03168*. 122
- Guo, Y., Yang, C., Rao, A., Agrawala, M., Lin, D., and Dai, B. (2025). SparseCtrl: Adding sparse controls to text-to-video diffusion models. In *European Conference on Computer Vision*, pages 330–348. Springer. 107, 108
- Guo, Y.-C., Liu, Y.-T., Shao, R., Laforte, C., Voleti, V., Luo, G., Chen, C.-H., Zou, Z.-X., Wang, C., Cao, Y.-P., and Zhang, S.-H. (2023). threestudio: A unified framework for 3D content generation. <https://github.com/threestudio-project/threestudio>. 166
- Gupta, A., Xiong, W., Nie, Y., Jones, I., and Oğuz, B. (2023). 3DGen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*. 43
- Haarnoja, T., Moran, B., Lever, G., Huang, S. H., Tirumala, D., Humplik, J., Wulfmeier, M., Tunyasuvunakool, S., Siegel, N. Y., Hafner, R., et al. (2024). Learning agile soccer skills for a bipedal robot with deep reinforcement learning. *Science Robotics*, 9(89):eadi8022. 143
- Hale, K. S. and Stanney, K. M. (2014). *Handbook of virtual environments: Design, implementation, and applications*. CRC Press. 87
- Han, J., Kokkinos, F., and Torr, P. (2025). VFusion3D: Learning scalable 3D generative models from video diffusion models. In *European Conference on Computer Vision*, pages 333–350. Springer. 107
- Haque, A., Tancik, M., Efros, A. A., Holynski, A., and Kanazawa, A. (2023). Instruct-NeRF2NeRF: Editing 3D scenes with instructions. *arXiv preprint arXiv:2303.12789*. 97
- Harley, A. W., You, Y., Sun, X., Zheng, Y., Raghuraman, N., Gu, Y., Liang, S., Chu, W.-H., Dave, A., Tokmakov, P., et al. (2025). AllTracker: Efficient dense point tracking at high resolution. *arXiv preprint arXiv:2506.07310*. 130
- Hassan, M., Guo, Y., Wang, T., Black, M., Fidler, S., and Peng, X. B. (2023). Synthesizing physical character-scene interactions. In *SIGGRAPH*. 143
- He, T., Luo, Z., He, X., Xiao, W., Zhang, C., Zhang, W., Kitani, K., Liu, C., and Shi, G. (2024a). OmniH2O: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. In *Conference on Robot Learning (CoRL)*. 143

- He, T., Xiao, W., Lin, T., Luo, Z., Xu, Z., Jiang, Z., Kautz, J., Liu, C., Shi, G., Wang, X., Fan, L., and Zhu, Y. (2025). HOVER: Versatile neural whole-body controller for humanoid robots. In *IEEE International Conference on Robotics and Automation (ICRA)*. 143
- He, X., Jiang, D., Zhang, G., Ku, M., Soni, A., Siu, S., Chen, H., Chandra, A., Jiang, Z., Arulraj, A., et al. (2024b). VideoScore: Building automatic metrics to simulate fine-grained human feedback for video generation. *arXiv:2406.15252*. 116, 118, 184, 185
- He, X., Xu, S., Li, X., Dong, R., Bian, L., Wang, Y.-X., and Gui, L.-Y. (2026). ULTRA: Unified multimodal control for autonomous humanoid whole-body loco-manipulation. 143
- He, Y., Xia, M., Chen, H., Cun, X., Gong, Y., Xing, J., Zhang, Y., Wang, X., Weng, C., Shan, Y., et al. (2023). Animate-a-story: Storytelling with retrieval-augmented video generation. *arXiv:2307.06940*. 107
- He, Y., Yang, T., Zhang, Y., Shan, Y., and Chen, Q. (2022). Latent video diffusion models for high-fidelity long video generation. *arXiv:2211.13221*. 107
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). GANs trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in Neural Information Processing Systems*, 30. 116
- Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. (2022). Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646. 106
- Hong, S., Ahn, D., and Kim, S. (2024). Debiasing scores and prompts of 2D diffusion for view-consistent text-to-3D generation. *Advances in Neural Information Processing Systems*, 36. 43
- Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., and Tan, H. (2023). LRM: Large reconstruction model for single image to 3D. *arXiv:2311.04400*. 1, 111
- Hou, Z., Yu, B., and Tao, D. (2023). Compositional 3D human-object neural animation. *arXiv preprint arXiv:2304.14070*. 142
- Hu, L. (2024). Animate anyone: Consistent and controllable image-to-video synthesis for character animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8153–8163. 122
- Hu, Y., Fang, Y., Ge, Z., Qu, Z., Zhu, Y., Pradhana, A., and Jiang, C. (2018a). A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (ToG)*, 37(4):1–14. 13, 64

- Hu, Y., Li, T.-M., Anderson, L., Ragan-Kelley, J., and Durand, F. (2019). Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (ToG)*, 38(6):201. 64, 115
- Hu, Y., Zhou, Q., Gao, X., Jacobson, A., Zorin, D., and Panozzo, D. (2018b). Tetrahedral meshing in the wild. *ACM Transactions on Graphics (ToG)*, 37(4):60. 132
- Huang, H., Liu, Y., Zheng, G., Wang, J., Dou, Z., and Yang, S. (2025). MVTokenFlow: High-quality 4D content generation using multiview token flow. *arXiv preprint arXiv:2502.11697*. 124
- Huang, S., Wang, Z., Li, P., Jia, B., Liu, T., Zhu, Y., Liang, W., and Zhu, S.-C. (2023a). Diffusion-based generation, optimization, and planning in 3D scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16750–16761. 43
- Huang, Y., Taheri, O., Black, M. J., and Tzionas, D. (2022a). InterCap: Joint markerless 3D tracking of humans and objects in interaction. In *GCPR*. 142, 153
- Huang, Y., Wang, J., Shi, Y., Tang, B., Qi, X., and Zhang, L. (2023b). DreamTime: An improved optimization strategy for diffusion-guided 3D generation. In *The Twelfth International Conference on Learning Representations*. 43
- Huang, Y.-H., Cao, Y.-P., Lai, Y.-K., Shan, Y., and Gao, L. (2023c). NeRF-Texture: Texture synthesis with neural radiance fields. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–10. 83, 84
- Huang, Y.-H., Sun, Y.-T., Yang, Z., Lyu, X., Cao, Y.-P., and Qi, X. (2023d). SC-GS: Sparse-controlled Gaussian splatting for editable dynamic scenes. *arXiv preprint arXiv:2312.14937*. 84
- Huang, Z., He, Y., Yu, J., Zhang, F., Si, C., Jiang, Y., Zhang, Y., Wu, T., Jin, Q., Chanpaisit, N., et al. (2024a). VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21807–21818. 134, 136
- Huang, Z., Hu, Y., Du, T., Zhou, S., Su, H., Tenenbaum, J. B., and Gan, C. (2021). PlasticineLab: A soft-body manipulation benchmark with differentiable physics. *arXiv preprint arXiv:2104.03311*. 44
- Huang, Z., Tozoni, D. C., Gjoka, A., Ferguson, Z., Schneider, T., Panozzo, D., and Zorin, D. (2022b). Differentiable solver for time-dependent deformation problems with contact. *ACM Transactions on Graphics (ToG)*. 44
- Huang, Z., Zhang, M., and Liao, J. (2024b). LVCD: Reference-based lineart video colorization with diffusion models. *arXiv:2409.12960*. 115
- Hyde, D. A., Gagniere, S. W., Marquez-Razon, A., and Teran, J. (2020). An implicit updated Lagrangian formulation for liquids with large surface energy. *ACM Transactions on Graphics (ToG)*, 39(6):1–13. 15

- Ihmsen, M., Akinci, N., Gissler, M., and Teschner, M. (2010). Boundary handling and adaptive time-stepping for PCISPH. In *VRIPHYS'10*, pages 79–88. 11, 21
- Ihmsen, M., Cornelis, J., Solenthaler, B., Horvath, C., and Teschner, M. (2013). Implicit incompressible SPH. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):426–435. 11, 14, 31, 32
- Jain, S., Watson, D., Tabellion, E., Poole, B., Kontkanen, J., et al. (2024). Video interpolation with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7341–7351. 107
- Jambon, C., Kerbl, B., Kopanas, G., Diolatzis, S., Leimkühler, T., and Drettakis, G. (2023). NeRFshop: Interactive editing of neural radiance fields”. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 6(1). 61
- Jeong, H., Park, G. Y., and Ye, J. C. (2024). VMC: Video motion customization using temporal attention adaption for text-to-video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9212–9221. 107, 108
- Jiang, C., Gast, T., and Teran, J. (2017). Anisotropic elastoplasticity for cloth, knit and hair frictional contact. *ACM Transactions on Graphics (ToG)*, 36(4):1–14. 64
- Jiang, C., Schroeder, C., Selle, A., Teran, J., and Stomakhin, A. (2015). The affine particle-in-cell method. *ACM Transactions on Graphics (ToG)*, 34(4):1–10. 64, 173, 175, 176
- Jiang, C., Schroeder, C., Teran, J., Stomakhin, A., and Selle, A. (2016). The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses*, pages 1–52. 9, 12, 13, 66, 174
- Jiang, H., Chen, J., Bu, Q., Chen, L., Shi, M., Zhang, Y., Li, D., Suo, C., Wang, C., Peng, Z., and Li, H. (2026). WholeBodyVLA: Towards unified latent VLA for whole-body loco-manipulation control. In *ICLR*. 143
- Jiang, N., Liu, T., Cao, Z., Cui, J., Chen, Y., Wang, H., Zhu, Y., and Huang, S. (2023a). CHAIRS: Towards full-body articulated human-object interaction. In *ICCV*. 142
- Jiang, N., Zhang, Z., Li, H., Ma, X., Wang, Z., Chen, Y., Liu, T., Zhu, Y., and Huang, S. (2024a). Scaling up dynamic human-scene interaction modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1737–1747. 142
- Jiang, Y., Yu, C., Cao, C., Wang, F., Hu, W., and Gao, J. (2024b). Animate3D: Animating any 3D model with multi-view video diffusion. *arXiv*. 124, 125
- Jiang, Y., Yu, C., Xie, T., Li, X., Feng, Y., Wang, H., Li, M., Lau, H., Gao, F., Yang, Y., et al. (2024c). VR-GS: A physical dynamics-aware interactive Gaussian splatting system in virtual reality. *arXiv preprint arXiv:2401.16663*. 4, 43

- Jiang, Y., Zhang, L., Gao, J., Hu, W., and Yao, Y. (2023b). Consistent4D: Consistent 360 $\{\backslash\deg\}$ dynamic object generation from monocular video. *arXiv preprint arXiv:2311.02848*. 123
- Jones, B., Popovic, J., McCann, J., Li, W., and Bargteil, A. (2015). Dynamic sprites: artistic authoring of interactive animations. *Computer Animation and Virtual Worlds*, 26(2):97–108. 108
- Jones, B., Thuerey, N., Shinar, T., and Bargteil, A. W. (2016). Example-based plastic deformation of rigid bodies. *ACM Transactions on Graphics (ToG)*, 35(4):1–11. 108
- Ju, T., Baker, M. L., and Chiu, W. (2006). Computing a family of skeletons of volumetric models for shape description. In *Proceedings of the 4th International Conference on Geometric Modeling and Processing*, GMP’06, page 235–247, Berlin, Heidelberg. Springer-Verlag. 125
- Jun, H. and Nichol, A. (2023). Shap-E: Generating conditional 3D implicit functions. *arXiv preprint arXiv:2305.02463*. 59
- Kalawsky, R. S. (1999). VRUSE—a computerised diagnostic tool: For usability evaluation of virtual/synthetic environment systems. *Applied Ergonomics*, 30(1):11–25. 85
- Karnewar, A., Vedaldi, A., Novotny, D., and Mitra, N. J. (2023). HoloDiffusion: Training a 3D diffusion model using 2D images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18423–18433. 43
- Katzir, O., Patashnik, O., Cohen-Or, D., and Lischinski, D. (2024). Noise-free score distillation. In *The Twelfth International Conference on Learning Representations*. 43
- Kazi, R. H., Grossman, T., Umetani, N., and Fitzmaurice, G. (2016). Motion amplifiers: sketching dynamic illustrations using the principles of 2D animation. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 4599–4609. 105, 107, 112
- Kerbl, B., Kopanas, G., Leimkühler, T., and Drettakis, G. (2023). 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4):139–1. 61, 63, 64, 65, 68, 72, 81, 83, 87, 100, 123
- Khazatsky, A., Pertsch, K., Nair, S., Balakrishna, A., Dasari, S., Karamcheti, S., Nasiriany, S., Srirama, M. K., Chen, L. Y., Ellis, K., et al. (2024). DROID: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*. 140, 143
- Kim, H., Baik, S., and Joo, H. (2025). DAViD: Modeling dynamic affordance of 3D objects using pre-trained video diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10330–10341. 140, 141, 142, 144, 153, 199, 202

- Kim, H., Han, S., Kwon, P., and Joo, H. (2024a). Beyond the contact: Discovering comprehensive affordance for 3D objects from pre-trained 2D diffusion models. In *European Conference on Computer Vision*, pages 400–419. Springer. 142, 151, 153
- Kim, J., Kim, J., Na, J., and Joo, H. (2024b). ParaHome: Parameterizing everyday home activities towards 3D generative modeling of human-object interactions. *arXiv preprint arXiv:2401.10232*. 142
- Kim, T., Saito, S., and Joo, H. (2023). NCHO: Unsupervised learning for neural 3D composition of humans and objects. In *ICCV*. 142
- Kingma, D. P. (2013). Auto-encoding variational bayes. *arXiv:1312.6114*. 110
- Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., et al. (2023). Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026. 112
- Klár, G., Gast, T., Pradhana, A., Fu, C., Schroeder, C., Jiang, C., and Teran, J. (2016). Drucker-prager elastoplasticity for sand animation. *ACM Transactions on Graphics (ToG)*, 35(4):1–12. 64, 74, 176
- Klingner, B. M., Feldman, B. E., Chentanez, N., and O’Brien, J. F. (2006). Fluid animation with dynamic meshes. In *ACM SIGGRAPH 2006 Papers*, pages 820–825. 10, 11
- Knapitsch, A., Park, J., Zhou, Q.-Y., and Koltun, V. (2017). Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13. 97
- Kolkin, N., Salavon, J., and Shakhnarovich, G. (2019). Style transfer by relaxed optimal transport and self-similarity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10051–10060. 118
- Kong, X., Liu, S., Lyu, X., Taher, M., Qi, X., and Davison, A. J. (2024). EscherNet: A generative model for scalable view synthesis. *arXiv preprint arXiv:2402.03908*. 42
- Koschier, D. and Bender, J. (2017). Density maps for improved SPH boundary handling. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 1–10. 12, 31
- Koschier, D., Bender, J., Solenthaler, B., and Teschner, M. (2022). A survey on SPH methods in computer graphics. *Computer Graphics Forum*, 41(2):737–760. 11
- Ku, M., Wei, C., Ren, W., Yang, H., and Chen, W. (2024). AnyV2V: A tuning-free framework for any video-to-video editing tasks. *arXiv:2403.14468*. 107
- Kugelstadt, T., Bender, J., Fernández-Fernández, J. A., Jeske, S. R., Löschner, F., and Longva, A. (2021). Fast corotated elastic SPH solids with implicit zero-energy mode control. *Proceedings of the ACM Conference on Computer Graphics and Interactive Techniques*, 4(3):1–21. 12, 32, 37

- Kulkarni, N., Rempe, D., Genova, K., Kundu, A., Johnson, J., Fouhey, D., and Guibas, L. (2024). NIFTY: Neural object interaction fields for guided human motion synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 947–957. 142
- Kwak, J.-g., Dong, E., Jin, Y., Ko, H., Mahajan, S., and Yi, K. M. (2024). Vivid-1-to-3: Novel view synthesis with video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6775–6785. 107
- Labs, B. F., Batifol, S., Blattmann, A., Boesel, F., Consul, S., Diagne, C., Dockhorn, T., English, J., English, Z., Esser, P., Kulal, S., Lacey, K., Levi, Y., Li, C., Lorenz, D., Müller, J., Podell, D., Rombach, R., Saini, H., Sauer, A., and Smith, L. (2025). FLUX.1 Kontext: Flow matching for in-context image generation and editing in latent space. 199
- Laine, S., Hellsten, J., Karras, T., Seol, Y., Lehtinen, J., and Aila, T. (2020). Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics (ToG)*, 39(6):1–14. 44, 50
- Lan, L., Kaufman, D. M., Li, M., Jiang, C., and Yang, Y. (2022). Affine body dynamics: Fast, stable & intersection-free simulation of stiff materials. *arXiv preprint arXiv:2201.10022*. 42, 55
- Lee, D.-T. and Schachter, B. J. (1980). Two algorithms for constructing a delaunay triangulation. *International Journal of Computer & Information Sciences*, 9(3):219–242. 112
- Lee, M., Cho, S., Shin, C., Lee, J., Yang, S., and Lee, S. (2024). Video diffusion models are strong video inpainter. *arXiv:2408.11402*. 107
- Levin, D. I., Litven, J., Jones, G. L., Sueda, S., and Pai, D. K. (2011). Eulerian solid simulation with contact. *ACM Transactions on Graphics (ToG)*, 30(4):1–10. 11
- Levin, E. and Fried, O. (2023). Differential diffusion: Giving each pixel its strength. *arXiv:2306.00950*. 120
- Li, C., Li, S., Zhao, Y., Zhu, W., and Lin, Y. (2022a). RT-NeRF: Real-time on-device neural radiance fields towards immersive AR/VR rendering. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, pages 1–9. 85
- Li, H., Yu, H.-X., Li, J., and Wu, J. (2026). ZeroHSI: Zero-shot 4d human-scene interaction by video generation. In *3DV*. 141, 142
- Li, J., Cao, J., Zhang, H., Rempe, D., Kautz, J., Iqbal, U., and Yuan, Y. (2025a). GENMO: A generalist model for human motion. *arXiv preprint arXiv:2505.01425*. 145, 196
- Li, J., Clegg, A., Mottaghi, R., Wu, J., Puig, X., and Liu, C. K. (2024a). Controllable human-object interaction synthesis. In *ECCV*. 142, 153, 199, 202

- Li, J., Tan, H., Zhang, K., Xu, Z., Luan, F., Xu, Y., Hong, Y., Sunkavalli, K., Shakhnarovich, G., and Bi, S. (2023a). Instant3D: Fast text-to-3D with sparse-view generation and large reconstruction model. *arXiv preprint arXiv:2311.06214*. 59
- Li, J., Wu, J., and Liu, C. K. (2023b). Object motion guided human motion synthesis. *ACM Transactions on Graphics (ToG)*, 42(6):1–11. 142, 151, 153
- Li, K., Rolff, T., Bacher, R., and Steinicke, F. (2023c). RealityGit: Cross reality version control of r&d optical workbench. In *2023 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pages 807–808. IEEE. 85
- Li, K., Rolff, T., Schmidt, S., Bacher, R., Frintrop, S., Leemans, W., and Steinicke, F. (2022b). Immersive neural graphics primitives. *arXiv preprint arXiv:2211.13494*. 84
- Li, K., Rolff, T., Schmidt, S., Bacher, R., Leemans, W., and Steinicke, F. (2023d). Interacting with neural radiance fields in immersive virtual reality. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–4. 85
- Li, K., Schmidt, S., Rolff, T., Bacher, R., Leemans, W., and Steinicke, F. (2023e). Magic NeRF lens: Interactive fusion of neural radiance fields for virtual facility inspection. *arXiv preprint arXiv:2307.09860*. 85
- Li, L. and Dai, A. (2024). GenZI: Zero-shot 3D human-scene interaction generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20465–20474. 142, 144
- Li, L., Wang, G., Zhang, Z., Li, Y., Li, X., Dou, Q., Gu, J., Xue, T., and Shan, Y. (2025b). ToonComposer: Streamlining cartoon production with generative post-keyframing. *arXiv preprint arXiv:2508.10881*. 122
- Li, M., Duan, Y., Zhou, J., and Lu, J. (2023f). Diffusion-SDF: Text-to-shape via voxelized diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12642–12651. 43
- Li, M., Ferguson, Z., Schneider, T., Langlois, T. R., Zorin, D., Panozzo, D., Jiang, C., and Kaufman, D. M. (2020). Incremental potential contact: Intersection-and inversion-free, large-deformation dynamics. *ACM Transactions on Graphics (ToG)*, 39(4):49. 9, 13, 14, 17, 18, 19, 31, 55
- Li, M., Jiang, C., and Luo, Z. (2024b). *Physics-Based Simulation*. . <https://phys-sim-book.github.io/>. 110
- Li, M., Wan, B., Moens, M.-F., and Tuytelaars, T. (2024c). Animate your motion: Turning still images into dynamic videos. *arXiv:2403.10179*. 108
- Li, Q., Wang, J., Loy, C. C., and Dai, B. (2023g). Task-oriented human-object interactions generation with implicit neural representations. *arXiv preprint arXiv:2303.13129*. 142

- Li, R., Pan, P., Yang, B., Xu, D., Zhou, S., Zhang, X., Li, Z., Kadambi, A., Wang, Z., Tu, Z., and Fan, Z. (2024d). 4K4DGen: Panoramic 4D generation at 4k resolution. [124](#)
- Li, S. and Pan, Y. (2023). Interactive geometry editing of neural radiance fields. *arXiv preprint arXiv:2303.11537*. [84](#)
- Li, S., Wang, L., Jia, W., Zhao, Y., and Zheng, L. (2022c). An iterative solution for improving the generalization ability of unsupervised skeleton motion retargeting. *Computers & Graphics*, 104:129–139. [126](#)
- Li, W., Chen, R., Chen, X., and Tan, P. (2023h). SweetDreamer: Aligning geometric priors in 2D diffusion for consistent text-to-3D. *arXiv preprint arXiv:2310.02596*. [43](#)
- Li, X., Fang, Y., Li, M., and Jiang, C. (2022d). BFEMP: Interpenetration-free mpm-fem coupling with barrier contact. *Computer Methods in Applied Mechanics and Engineering*, 390:114350. [13](#)
- Li, X., Ma, Q., Lin, T.-Y., Chen, Y., Jiang, C., Liu, M.-Y., and Xiang, D. (2025c). Articulated kinematics distillation from video diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17571–17581. [123](#)
- Li, X., Ma, Q., Lin, T.-Y., Chen, Y., Jiang, C., Liu, M.-Y., and Xiang, D. (2025d). Articulated kinematics distillation from video diffusion models. [125](#)
- Li, X., Qiao, Y.-L., Chen, P. Y., Jatavallabhula, K. M., Lin, M., Jiang, C., and Gan, C. (2022e). PAC-NeRF: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. In *The Eleventh International Conference on Learning Representations*. [44](#)
- Li, X., Qiao, Y.-L., Chen, P. Y., Jatavallabhula, K. M., Lin, M., Jiang, C., and Gan, C. (2023i). PAC-NeRF: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. In *The Eleventh International Conference on Learning Representations*. [64](#), [76](#)
- Li, X., Qiao, Y.-L., Chen, P. Y., Jatavallabhula, K. M., Lin, M., Jiang, C., and Gan, C. (2023j). PAC-NeRF: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. *arXiv preprint arXiv:2303.05512*. [83](#), [95](#), [96](#), [103](#)
- Li, X., Yu, C., Du, W., Jiang, Y., Xie, T., Chen, Y., Yang, Y., and Jiang, C. (2025e). Dress-1-to-3: Single image to simulation-ready 3D outfit with diffusion prior and differentiable physics. *ACM Transactions on Graphics (ToG)*, 44(4):1–16. [132](#), [188](#)
- Li, Y., Chen, H.-y., Larionov, E., Sarafianos, N., Matusik, W., and Stuyck, T. (2023k). DiffAvatar: Simulation-ready garment optimization with differentiable simulation. *arXiv preprint arXiv:2311.12194*. [44](#)
- Li, Y., Du, T., Wu, K., Xu, J., and Matusik, W. (2022f). DiffCloth: Differentiable cloth simulation with dry frictional contact. *ACM Transactions on Graphics (ToG)*, 42(1):1–20. [44](#)

- Li, Y., Lin, Z.-H., Forsyth, D., Huang, J.-B., and Wang, S. (2023l). ClimateNeRF: Extreme weather synthesis in neural radiance field. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3227–3238. [63](#)
- Li, Z., Chen, Y., and Liu, P. (2024e). DreamMesh4D: Video-to-4D generation with sparse-controlled Gaussian-mesh hybrid representation. In *Advances in Neural Information Processing Systems (NeurIPS)*. [124](#), [134](#), [135](#), [189](#), [190](#)
- Li, Z., Tucker, R., Snavely, N., and Holynski, A. (2024f). Generative image dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24142–24153. [108](#), [114](#)
- Liang, Y., Yang, X., Lin, J., Li, H., Xu, X., and Chen, Y. (2023). LucidDreamer: Towards high-fidelity text-to-3D generation via interval score matching. *arXiv preprint arXiv:2311.11284*. [98](#)
- Liao, Z., Golyanik, V., Habermann, M., and Theobalt, C. (2024). Vinecs. In *Computer Vision and Pattern Recognition (CVPR)*. [125](#)
- Lin, C.-H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.-Y., and Lin, T.-Y. (2023a). Magic3D: High-resolution text-to-3D content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309. [1](#), [41](#), [43](#), [51](#), [56](#)
- Lin, G., Feng-Lin, L., Shu-Yu, C., Kaiwen, J., Chunpeng, L., Lai, Y., and Hongbo, F. (2023b). SketchFaceNeRF: Sketch-based facial generation and editing in neural radiance fields. *ACM Transactions on Graphics (ToG)*. [84](#)
- Lin, Y., Clark, R., and Torr, P. (2024). DreamPolisher: Towards high-quality text-to-3D generation via geometric diffusion. *arXiv preprint arXiv:2403.17237*. [43](#)
- Lin, Z.-H., Ma, W.-C., Hsu, H.-Y., Wang, Y.-C. F., and Wang, S. (2022). NeurMIPs: Neural mixture of planar experts for view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15702–15712. [63](#)
- Lindell, D. B., Martel, J. N., and Wetzstein, G. (2021). AutoInt: Automatic integration for fast neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14556–14565. [83](#)
- Ling, H., Kim, S. W., Torralba, A., Fidler, S., and Kreis, K. (2024). Align your Gaussians: Text-to-4D with dynamic 3D Gaussians and composed diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8576–8588. [1](#)
- Liu, F., Sun, W., Wang, H., Wang, Y., Sun, H., Ye, J., Zhang, J., and Duan, Y. (2024a). ReconX: Reconstruct any scene from sparse views with video diffusion model. *arXiv:2408.16767*. [107](#)

- Liu, L., Gu, J., Zaw Lin, K., Chua, T.-S., and Theobalt, C. (2020). Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663. 83
- Liu, L. and Hodgins, J. (2017). Learning to schedule control fragments for physics-based characters using deep q-learning. *ACM Transactions on Graphics (ToG)*, 36(3):1–14. 143
- Liu, L. and Hodgins, J. (2018). Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning. *ACM Transactions on Graphics (ToG)*, 37(4):1–14. 143
- Liu, M., Shi, R., Chen, L., Zhang, Z., Xu, C., Wei, X., Chen, H., Zeng, C., Gu, J., and Su, H. (2023a). One-2-3-45++: Fast single image to 3D objects with consistent multi-view generation and 3D diffusion. *arXiv preprint arXiv:2311.07885*. 42
- Liu, M., Xu, C., Jin, H., Chen, L., Varma T, M., Xu, Z., and Su, H. (2024b). One-2-3-45: Any single image to 3D mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems*, 36. 42
- Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., and Vondrick, C. (2023b). Zero-1-to-3: Zero-shot one image to 3D object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9298–9309. 42, 59
- Liu, R., Xiang, J., Zhao, B., Zhang, R., Yu, J., and Zheng, C. (2023c). Neural impostor: Editing neural radiance fields with explicit shape manipulation. *arXiv preprint arXiv:2310.05391*. 64
- Liu, S., Ren, Z., Gupta, S., and Wang, S. (2025). PhysGen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision*, pages 360–378. Springer. 109
- Liu, X., Wang, B., Wang, H., and Yi, L. (2023d). Few-shot physically-aware articulated mesh generation via hierarchical deformation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 854–864. 43
- Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., and Wang, W. (2023e). SyncDreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*. 42
- Lombardi, S., Simon, T., Schwartz, G., Zollhoefer, M., Sheikh, Y., and Saragih, J. (2021). Mixture of volumetric primitives for efficient neural rendering. *ACM Transactions on Graphics (ToG)*, 40(4):1–13. 83
- Long, X., Guo, Y.-C., Lin, C., Liu, Y., Dou, Z., Liu, L., Ma, Y., Zhang, S.-H., Habermann, M., Theobalt, C., et al. (2023). Wonder3D: Single image to 3D using cross-domain diffusion. *arXiv preprint arXiv:2310.15008*. 42
- Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169. 89

- Lou, Y., Wang, Y., Wu, Z., Zhao, R., Wang, W., Shi, M., and Komura, T. (2025). Zero-shot human-object interaction synthesis with multimodal priors. *arXiv preprint arXiv:2503.20118*. 140, 142
- Lu, J., Huang, C.-H. P., Bhattacharya, U., Huang, Q., and Zhou, Y. (2025). HUMOTO: A 4D dataset of mocap human object interactions. *arXiv preprint arXiv:2504.10414*. 140, 142, 143
- Luiten, J., Kopanas, G., Leibe, B., and Ramanan, D. (2023). Dynamic 3D Gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*. 64, 77
- Luo, S. and Hu, W. (2021). Diffusion probabilistic models for 3D point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845. 43
- Luo, Z., Cao, J., Kitani, K., Xu, W., et al. (2023). Perpetual humanoid control for real-time simulated avatars. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10895–10904. 143
- Luo, Z., Wang, J., Liu, K., Zhang, H., Tessler, C., Wang, J., Yuan, Y., Cao, J., Lin, Z., Wang, F., et al. (2024). SMPLOlympics: Sports environments for physically simulated humanoids. *arXiv preprint arXiv:2407.00187*. 143
- Luo, Z., Yuan, Y., Wang, T., Li, C., Chen, S., Castañeda, F., Cao, Z.-A., Li, J., Minor, D., Ben, Q., et al. (2025). SONIC: Supersizing motion tracking for natural humanoid whole-body control. *arXiv preprint arXiv:2511.07820*. 141, 143, 148
- Ma, J., Wei, L.-Y., and Kazi, R. H. (2022). A layered authoring tool for stylized 3D animations. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–14. 107
- Ma, X., Wang, Y., Jia, G., Chen, X., Li, Y.-F., Chen, C., and Qiao, Y. (2024). Cinemo: Consistent and controllable image animation with motion diffusion models. *arXiv:2407.15642*. 105, 116, 117, 182, 183, 184, 185
- Macklin, M. (2022). Warp: A high-performance python framework for GPU simulation and graphics. <https://github.com/nvidia/warp>. NVIDIA GPU Technology Conference (GTC). 47, 134, 166, 189
- Macklin, M. and Müller, M. (2013). Position based fluids. *ACM Transactions on Graphics (ToG)*, 32(4):1–12. 9, 14
- Macklin, M., Müller, M., and Chentanez, N. (2016). XPBD: Position-based simulation of compliant constrained dynamics. In *Proceedings of the 9th International Conference on Motion in Games*, pages 49–54. 82, 90, 145
- Magnenat-Thalmann, N., Laperrière, R., and Thalmann, D. (1989). Joint-dependent local deformations for hand animation and object grasping. In *Proceedings on Graphics interface '88*, pages 26–33. 128

- McKiver, W. J. and Dritschel, D. G. (2003). The motion of a fluid ellipsoid in a general linear background flow. *Journal of Fluid Mechanics*, 474:147–173. 70
- Meng, F., Liao, J., Tan, X., Shao, W., Lu, Q., Zhang, K., Cheng, Y., Li, D., Qiao, Y., and Luo, P. (2024). Towards world simulator: Crafting physical commonsense-based benchmark for video generation. *arXiv:2410.05363*. 109
- Meng, X., Chen, W., and Yang, B. (2023). NeAT: Learning neural implicit surfaces with arbitrary topologies from multi-view images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 248–258. 83
- Mercier, A., Nakhli, R., Reddy, M., Yasarla, R., Cai, H., Porikli, F., and Berger, G. (2024). HexaGen3D: Stablediffusion is just one step away from fast and diverse text-to-3D generation. *arXiv:2401.07727*. 43
- Meshy (2025). Meshy AI 3D model generator. <https://www.meshy.ai/>. Accessed: 2025-11-1. 133
- Metzer, G., Richardson, E., Patashnik, O., Giryes, R., and Cohen-Or, D. (2023). Latent-nerf for shape-guided generation of 3D shapes and textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12663–12673. 41
- Mezghanni, M., Bodrito, T., Boulkenafed, M., and Ovsjanikov, M. (2022). Physical simulation layer for accurate 3D modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13514–13523. 44
- Mezghanni, M., Boulkenafed, M., Lieutier, A., and Ovsjanikov, M. (2021). Physically-aware generative network for 3D shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9330–9341. 40, 43
- Michel, O., Bar-On, R., Liu, R., Benaim, S., and Hanocka, R. (2022). Text2Mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502. 59
- Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2021). NeRF: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106. 44, 60, 63, 65, 83, 123, 177
- Mo, S., Xie, E., Chu, R., Hong, L., Niessner, M., and Li, Z. (2024). Dit-3D: Exploring plain diffusion transformers for 3D shape generation. *Advances in Neural Information Processing Systems*, 36. 43
- Monaghan, J. J. (1992). Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 30:543–574. 1, 15
- Monaghan, J. J. (1994). Simulating free surface flows with SPH. *Journal of Computational Physics*, 110(2):399–406. 11

- Monaghan, J. J. (2005). Smoothed particle hydrodynamics. *Reports on Progress in Physics*, 68(8):1703. [1](#), [16](#)
- Moreau, J. J. (2011). On unilateral constraints, friction and plasticity. In *New Variational Techniques in Mathematical Physics*, pages 171–322. Springer. [14](#)
- Mou, L., Chen, J.-K., and Wang, Y.-X. (2024). Instruct 4D-to-4D: Editing 4D scenes as pseudo-3D scenes using 2D diffusion. [124](#)
- Müller, M., Charypar, D., and Gross, M. H. (2003). Particle-based fluid simulation for interactive applications. In *Symposium on Computer Animation*, volume 2. [16](#)
- Müller, M., Chentanez, N., and Macklin, M. (2016). Simulating visual geometry. In *Proceedings of the 9th International Conference on Motion in Games*, pages 31–38. [61](#)
- Müller, M., Chentanez, N., and Macklin, M. (2016). Simulating visual geometry. In *Proceedings of the 9th International Conference on Motion in Games*, MIG ’16, page 31–38, New York, NY, USA. Association for Computing Machinery. [86](#)
- Müller, M., Heidelberger, B., Hennix, M., and Ratcliff, J. (2007). Position based dynamics. *J. Vis. Comun. Image Represent.*, 18(2):109–118. [126](#)
- Müller, N., Siddiqui, Y., Porzi, L., Buló, S. R., Kontschieder, P., and Nießner, M. (2023). DiffRF: Rendering-guided 3D radiance field diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4328–4338. [43](#)
- Müller, T., Evans, A., Schied, C., and Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15. [63](#), [72](#), [83](#), [97](#)
- Murthy, J. K., Macklin, M., Golemo, F., Voleti, V., Petrini, L., Weiss, M., Considine, B., Parent-Lévesque, J., Xie, K., Erleben, K., et al. (2020). gradSim: Differentiable simulation for system identification and visuomotor control. In *International Conference on Learning Representations*. [44](#)
- Museth, K. (2013). VDB: High-resolution sparse volumes with dynamic topology. *ACM Transactions on Graphics (ToG)*, 32(3). [89](#)
- Nag, S., Cohen-Or, D., Zhang, H., and Mahdavi-Amiri, A. (2025). In-2-4D: Inbetweening from two single-view images to 4D generation. [124](#)
- Nai, R., Zheng, B., Zhao, J., Zhu, H., Dai, S., Chen, Z., Hu, Y., Hu, Y., Zhang, T., Wen, C., and Gao, Y. (2026). HuMI: Humanoid whole-body manipulation from robot-free demonstrations. [143](#)
- Nasiriany, S., Maddukuri, A., Zhang, L., Parikh, A., Lo, A., Joshi, A., Mandlekar, A., and Zhu, Y. (2024). RoboCasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*. [151](#)

- Ni, J., Chen, Y., Jing, B., Jiang, N., Wang, B., Dai, B., Zhu, Y., Zhu, S.-C., and Huang, S. (2024). PhyRecon: Physically plausible neural scene reconstruction. *arXiv preprint arXiv:2404.16666*. 41, 44
- Ni, J., Wang, Z., Lin, W., Bar, A., LeCun, Y., Darrell, T., Malik, J., and Herzig, R. (2025). From generated human videos to physically plausible robot trajectories. *arXiv preprint arXiv:2512.05094*. 140
- Niemeyer, M. and Geiger, A. (2021). CAMPARI: Camera-aware decomposed generative neural radiance fields. In *2021 International Conference on 3D Vision (3DV)*, pages 951–961. IEEE. 83
- Niklaus, S. and Liu, F. (2020). Softmax splatting for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5437–5446. 114
- Niu, M., Cun, X., Wang, X., Zhang, Y., Shan, Y., and Zheng, Y. (2024). MOFA-Video: Controllable image animation via generative motion field adaptations in frozen image-to-video diffusion model. *arXiv:2405.20222*. 105
- Ntavelis, E., Siarohin, A., Olszewski, K., Wang, C., Gool, L. V., and Tulyakov, S. (2023). Autodecoding latent 3D diffusion models. *Advances in Neural Information Processing Systems*, 36:67021–67047. 43
- Oechsle, M., Peng, S., and Geiger, A. (2021). UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599. 83
- OpenAI (2024). Chatgpt. <https://chat.openai.com/>. Large language model used for text generation and editing. 145, 147, 153, 194, 195
- Pan, Z., Zhu, Y., and Mu, Y. (2024). Sakuga-42M dataset: Scaling up cartoon research. *arXiv:2405.07425*. 116
- Park, K., Sinha, U., Barron, J. T., Bouaziz, S., Goldman, D. B., Seitz, S. M., and Martin-Brualla, R. (2021). Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874. 1, 63, 82, 84
- Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in PyTorch. *OpenReview.net*. 166
- Patel, S., Mohan, S., Mai, H., Jain, U., Lazebnik, S., and Li, Y. (2025). Robotic manipulation by imitating generated videos without physical demonstrations. 143
- Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A. A., Tzionas, D., and Black, M. J. (2019). Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10975–10985. 145

- Peer, A., Gissler, C., Band, S., and Teschner, M. (2018). An implicit SPH formulation for incompressible linearly elastic solids. *Computer Graphics Forum*, 37(6):135–148. 12, 32, 37
- Peer, A., Ihmsen, M., Cornelis, J., and Teschner, M. (2015). An implicit viscosity formulation for SPH fluids. *ACM Transactions on Graphics (ToG)*, 34(4):1–10. 16
- Peer, A. and Teschner, M. (2016). Prescribed velocity gradients for highly viscous SPH fluids with vorticity diffusion. *IEEE Transactions on Visualization and Computer Graphics*, 23(12):2656–2662. 16
- Peng, M., Wei, L.-y., Kazi, R. H., and Kim, V. G. (2020). Autocomplete animated sculpting. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, pages 760–777. 107
- Peng, S., Dong, J., Wang, Q., Zhang, S., Shuai, Q., Zhou, X., and Bao, H. (2021a). Animatable neural radiance fields for modeling dynamic human bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14314–14323. 83
- Peng, X., Xie, Y., Wu, Z., Jampani, V., Sun, D., and Jiang, H. (2025). HOI-Diff: Text-driven synthesis of 3D human-object interactions using diffusion models. In *CVPR 2025 Workshop of HuMoGen*. 142, 153, 199, 202
- Peng, X. B., Abbeel, P., Levine, S., and Van de Panne, M. (2018). DeepMimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (ToG)*, 37(4):1–14. 143
- Peng, X. B., Guo, Y., Halper, L., Levine, S., and Fidler, S. (2022a). ASE: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions On Graphics (ToG)*, 41(4):1–17. 143
- Peng, X. B., Ma, Z., Abbeel, P., Levine, S., and Kanazawa, A. (2021b). AMP: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (ToG)*, 40(4):1–20. 143
- Peng, Y., Yan, Y., Liu, S., Cheng, Y., Guan, S., Pan, B., Zhai, G., and Yang, X. (2022b). CageNeRF: Cage-based neural radiance field for generalized 3D deformation and animation. *Advances in Neural Information Processing Systems*, 35:31402–31415. 61, 64
- Peskin, C. S. (2002). The immersed boundary method. *Acta Numerica*, 11:479–517. 11
- Petrov, I. A., Marin, R., Chibane, J., and Pons-Moll, G. (2023). Object pop-up: Can we infer 3D objects and their poses from human interactions alone? In *CVPR*. 142
- Pilkington, N. (2022). Dronedeploy nerf dataset. 72
- Po, R. and Wetzstein, G. (2023). Compositional 3D scene generation using locally conditioned diffusion. *arXiv preprint arXiv:2303.12218*. 43

- Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. (2022). DreamFusion: Text-to-3D using 2D diffusion. *arXiv preprint arXiv:2209.14988*. 41, 43, 56, 123
- Potamias, R. A., Zhang, J., Deng, J., and Zafeiriou, S. (2025). WiLoR: End-to-end 3D hand localization and reconstruction in-the-wild. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12242–12254. 145, 147, 195
- Prévost, R., Whiting, E., Lefebvre, S., and Sorkine-Hornung, O. (2013). Make it stand: balancing shapes for 3D fabrication. *ACM Transactions on Graphics (ToG)*, 32(4):1–10. 41
- Pumarola, A., Corona, E., Pons-Moll, G., and Moreno-Noguer, F. (2021). D-NeRF: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327. 63, 81, 82, 84
- Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660. 83
- Qian, G., Mai, J., Hamdi, A., Ren, J., Siarohin, A., Li, B., Lee, H.-Y., Skorokhodov, I., Wonka, P., Tulyakov, S., and Ghanem, B. (2024). Magic123: One image to high-quality 3D object generation using both 2D and 3D diffusion priors. In *The Twelfth International Conference on Learning Representations (ICLR)*. 43, 59
- Qiao, Y.-L., Gao, A., and Lin, M. (2022). NeuPhysics: Editable neural geometry and physics from monocular videos. *Advances in Neural Information Processing Systems*, 35:12841–12854. 64
- Qiao, Y.-L., Gao, A., Xu, Y., Feng, Y., Huang, J.-B., and Lin, M. C. (2023). Dynamic mesh-aware radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 385–396. 64, 83
- Qiao, Y.-L., Liang, J., Koltun, V., and Lin, M. C. (2021). Efficient differentiable simulation of articulated bodies. In *International Conference on Machine Learning*, pages 8661–8671. PMLR. 44
- Qiu, R.-Z., Yang, G., Zeng, W., and Wang, X. (2024). Feature splatting: Language-driven physics-based scene synthesis and editing. *arXiv preprint arXiv:2404.01223*. 43
- Qiu, Y., Reeve, S. T., Li, M., Yang, Y., Slattery, S. R., and Jiang, C. (2023). A sparse distributed gigascale resolution material point method. *ACM Transactions on Graphics (ToG)*, 42(2):1–21. 64
- Raafat, M. (2023). BlenderNeRF. <https://github.com/maximeraafat/BlenderNeRF>. 72, 74, 177

- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR. 56
- Raistrick, A., Lipson, L., Ma, Z., Mei, L., Wang, M., Zuo, Y., Kayan, K., Wen, H., Han, B., Wang, Y., et al. (2023). Infinite photorealistic worlds using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12630–12641. 145, 194
- Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., et al. (2024). SAM 2: Segment anything in images and videos. *arXiv:2408.00714*. 112, 129, 146, 147, 193, 198
- Ravi, N., Reizenstein, J., Novotny, D., Gordon, T., Lo, W.-Y., Johnson, J., and Gkioxari, G. (2020). Accelerating 3D deep learning with pytorch3D. *arXiv preprint arXiv:2007.08501*. 129
- Rebain, D., Jiang, W., Yazdani, S., Li, K., Yi, K. M., and Tagliasacchi, A. (2021). DeRF: Decomposed radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14153–14161. 83
- Ren, X., Shen, T., Huang, J., Ling, H., Lu, Y., Nimier-David, M., Müller, T., Keller, A., Fidler, S., and Gao, J. (2025). GEN3C: 3D-informed world-consistent video generation with precise camera control. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6121–6132. 122
- Ren, Z., Pan, Z., Zhou, X., and Kang, L. (2023). Diffusion motion: Generate text-guided 3D human motion by diffusion model. 126
- Robinson-Mosher, A., Schroeder, C., and Fedkiw, R. (2011). A symmetric positive definite formulation for monolithic fluid structure interaction. *Journal of Computational Physics*, 230(4):1547–1566. 10
- Robinson-Mosher, A., Shinar, T., Gretarsson, J., Su, J., and Fedkiw, R. (2008). Two-way coupling of fluids to rigid and deformable solids and shells. *ACM Transactions on Graphics (ToG)*, 27(3):1–9. 10
- Roble, D., Zafar, N. b., and Falt, H. (2005). Cartesian grid fluid simulation with irregular boundary voxels. In *ACM SIGGRAPH 2005 Sketches*, pages 138–es. 10
- Rolff, T., Schmidt, S., Li, K., Steinicke, F., and Frintrop, S. (2023). VRS-NeRF: Accelerating neural radiance field rendering with variable rate shading. In *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 243–252. IEEE. 85
- Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2021). High-resolution image synthesis with latent diffusion models. 42

- Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695. 44, 50, 106, 110
- Romero, J., Tzionas, D., and Black, M. J. (2017). Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics (ToG)*, 36(6):1–17. 145, 195
- Sang, L., Canfes, Z., Cao, D., Marin, R., Bernard, F., and Cremers, D. (2025). TwoSquared: 4D generation from 2D image pairs. 124
- Savitzky, A. and Golay, M. J. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639. 195
- Schönberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. 72
- Schönberger, J. L., Zheng, E., Pollefeys, M., and Frahm, J.-M. (2016). Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*. 72
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*. 150, 197
- Seo, J., Jang, W., Kwak, M.-S., Kim, H., Ko, J., Kim, J., Kim, J.-H., Lee, J., and Kim, S. (2023). Let 2D diffusion model know 3D-consistency for robust text-to-3D generation. *arXiv preprint arXiv:2303.07937*. 43
- Shao, J., Yang, Y., Zhou, H., Zhang, Y., Shen, Y., Poggi, M., and Liao, Y. (2024a). Learning temporally consistent video depth from video diffusion priors. *arXiv:2406.01493*. 107
- Shao, R., Pang, Y., Zheng, Z., Sun, J., and Liu, Y. (2024b). Human4DiT: 360-degree human video generation with 4D diffusion transformer. *ACM Transactions on Graphics (ToG)*, 43(6). 124
- Shen, T., Gao, J., Yin, K., Liu, M.-Y., and Fidler, S. (2021). Deep marching tetrahedra: a hybrid representation for high-resolution 3D shape synthesis. *Advances in Neural Information Processing Systems*, 34:6087–6101. 44, 50
- Shi, M., Peng, S., Chen, J., Jiang, H., Li, Y., Huang, D., Luo, P., Li, H., and Chen, L. (2026). EgoHumanoid: Unlocking in-the-wild loco-manipulation with robot-free egocentric demonstration. 143
- Shi, R., Chen, H., Zhang, Z., Liu, M., Xu, C., Wei, X., Chen, L., Zeng, C., and Su, H. (2023a). Zero123++: a single image to consistent multi-view diffusion base model. 42

- Shi, X., Huang, Z., Wang, F.-Y., Bian, W., Li, D., Zhang, Y., Zhang, M., Cheung, K. C., See, S., Qin, H., et al. (2024). Motion-I2V: Consistent and controllable image-to-video generation with explicit motion modeling. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11. 105, 107, 108, 111, 116, 117, 182, 183, 184, 185
- Shi, Y., Wang, P., Ye, J., Long, M., Li, K., and Yang, X. (2023b). MVDream: Multi-view diffusion for 3D generation. *arXiv preprint arXiv:2308.16512*. 1, 41, 43, 52
- Shim, J., Kang, C., and Joo, K. (2023). Diffusion-based signed distance fields for 3D shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20887–20897. 43
- Shu, D., Cunningham, J., Stump, G., Miller, S. W., Yukish, M. A., Simpson, T. W., and Tucker, C. S. (2020). 3D design using generative adversarial networks and physics-based validation. *Journal of Mechanical Design*, 142(7):071701. 43
- Shue, J. R., Chan, E. R., Po, R., Ankner, Z., Wu, J., and Wetzstein, G. (2023). 3D neural field generation using triplane diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20875–20886. 43
- Si, H. (2015). TetGen, a delaunay-based quality tetrahedral mesh generator. *ACM Transactions on Mathematical Software*, 41(2). 89
- Sifakis, E. and Barbic, J. (2012). FEM simulation of 3D deformable solids: A practitioner’s guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 courses*, pages 1–50. 1, 14
- Simo, J. C. and Hughes, T. J. (2006). *Computational inelasticity*, volume 7. Springer Science & Business Media. 67
- Singer, U., Polyak, A., Hayes, T., Yin, X., An, J., Zhang, S., Hu, Q., Yang, H., Ashual, O., Gafni, O., et al. (2022). Make-a-Video: Text-to-video generation without text-video data. *arXiv:2209.14792*. 107
- Singer, U., Sheynin, S., Polyak, A., Ashual, O., Makarov, I., Kokkinos, F., Goyal, N., Vedaldi, A., Parikh, D., Johnson, J., et al. (2023). Text-to-4D dynamic scene generation. *arXiv preprint arXiv:2301.11280*. 1
- Solenthaler, B. and Pajarola, R. (2009). Predictive-corrective incompressible SPH. In *ACM SIGGRAPH 2009 papers*, pages 1–6. 11
- Solenthaler, B., Schläfli, J., and Pajarola, R. (2007). A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds*, 18(1):69–82. 12
- Song, C., Li, X., Yang, F., Xu, Z., Wei, J., Liu, F., Feng, J., Lin, G., and Zhang, J. (2025). Puppeteer: Rig and animate your 3D models. *Advances in Neural Information Processing Systems*. 125, 134, 135, 189, 190

- Song, L., Chen, A., Li, Z., Chen, Z., Chen, L., Yuan, J., Xu, Y., and Geiger, A. (2023a). NeRFPlayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2732–2742. 84
- Song, W., Zhang, X., Guo, Y., Li, S., Hao, A., and Qin, H. (2023b). Automatic generation of 3D scene animation based on dynamic knowledge graphs and contextual encoding. *International Journal of Computer Vision*, 131(11):2816–2844. 124
- Sorkine, O. and Alexa, M. (2007). As-rigid-as-possible surface modeling. In *Symposium on Geometry Processing*, volume 4, pages 109–116. Citeseer. 78
- Sorkine, O. and Cohen-Or, D. (2004). Least-squares meshes. In *Proceedings Shape Modeling Applications, 2004.*, pages 191–199. IEEE. 83
- Stanney, K. M., Mollaghasemi, M., Reeves, L., Breaux, R., and Graeber, D. A. (2003). Usability engineering of virtual environments (VEs): identifying multiple criteria that drive effective ve system design. *International Journal of Human-Computer Studies*, 58(4):447–481. 87
- Starke, S., Mason, I., and Komura, T. (2022). DeepPhase: Periodic autoencoders for learning motion phase manifolds. *ACM Transactions on Graphics (ToG)*, 41(4):1–13. 143
- Starke, S., Zhao, Y., Komura, T., and Zaman, K. (2020). Local motion phases for learning multi-contact character movements. *ACM Transactions on Graphics (ToG)*, 39(4). 143
- Stomakhin, A., Howes, R., Schroeder, C. A., and Teran, J. M. (2012). Energetically consistent invertible elasticity. In *Symposium on Computer Animation*, pages 25–32. 23, 113, 132, 179, 187
- Stomakhin, A., Schroeder, C., Chai, L., Teran, J., and Selle, A. (2013). A material point method for snow simulation. *ACM Transactions on Graphics (ToG)*, 32(4):1–10. 12, 64, 66, 67
- Stomakhin, A., Schroeder, C., Jiang, C., Chai, L., Teran, J., and Selle, A. (2014). Augmented MPM for phase-change and varied materials. *ACM Transactions on Graphics (ToG)*, 33(4):1–11. 12
- Strecke, M. and Stueckler, J. (2021). DiffSDFSIm: Differentiable rigid-body dynamics with implicit shapes. In *2021 International Conference on 3D Vision (3DV)*, pages 96–105. IEEE. 44
- Su, Q., Bai, X., Fu, H., Tai, C.-L., and Wang, J. (2018). Live Sketch: Video-driven dynamic deformation of static drawings. In *Proceedings of the 2018 Chi Conference on Human Factors in Computing Systems*, pages 1–12. 105

- Sulsky, D., Zhou, S.-J., and Schreyer, H. L. (1995). Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications*, 87(1-2):236–252. [12](#)
- Sun, C., Sun, M., and Chen, H.-T. (2022). Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469. [63](#)
- Sun, J.-M., Wu, T., Yang, Y.-L., Lai, Y.-K., and Gao, L. (2023). SOL-NeRF: Sunlight modeling for outdoor scene decomposition and relighting. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–11. [83](#)
- Sun, Q., Guo, Z., Wan, Z., Yan, J. N., Yin, S., Zhou, W., Liao, J., and Li, H. (2025). EG4D: Explicit generation of 4D object without score distillation. In *ICLR*. [124](#)
- Sundaresan, P., Antonova, R., and Bohgl, J. (2022). DiffCloud: Real-to-sim from point clouds with differentiable simulation and rendering of deformable objects. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10828–10835. IEEE. [44](#)
- Sutcliffe, A. G., Poullis, C., Gregoriades, A., Katsouri, I., Tzanavari, A., and Herakleous, K. (2019). Reflecting on the design process for virtual reality applications. *International Journal of Human-Computer Interaction*, 35(2):168–179. [86](#)
- Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Silvestrov, A., Kong, N., Goka, H., Park, K., and Lempitsky, V. (2022). Resolution-robust large mask inpainting with fourier convolutions. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2149–2159. [89](#), [92](#), [93](#), [100](#)
- Szymanowicz, S., Rupprecht, C., and Vedaldi, A. (2023). Viewset diffusion:(0-) image-conditioned 3D generative models from 2D data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8863–8873. [43](#)
- Taheri, O., Ghorbani, N., Black, M. J., and Tzionas, D. (2020). GRAB: A dataset of whole-body human grasping of objects. In *ECCV*. [140](#), [142](#), [143](#)
- Takahashi, T. and Batty, C. (2020). Monolith: A monolithic pressure-viscosity-contact solver for strong two-way rigid-rigid rigid-fluid coupling. *ACM Transactions on Graphics (ToG)*, 39(6):1–16. [11](#)
- Takahashi, T. and Batty, C. (2021). FrictionalMonolith: A monolithic optimization-based approach for granular flow with contact-aware rigid-body coupling. *ACM Transactions on Graphics (ToG)*, 40(6):1–20. [11](#)
- Takahashi, T. and Batty, C. (2022). ElastoMonolith: A monolithic optimization-based liquid solver for contact-aware elastic-solid coupling. *ACM Transactions on Graphics (ToG)*, 41(6):1–19. [26](#), [31](#), [32](#), [33](#), [37](#)
- Takahashi, T., Dobashi, Y., Fujishiro, I., Nishita, T., and Lin, M. C. (2015). Implicit formulation for SPH-based viscous fluids. *Computer Graphics Forum*, 34(2):493–502. [16](#)

- Takahashi, T., Liang, J., Qiao, Y.-L., and Lin, M. C. (2021). Differentiable fluids with solid coupling for learning and control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6138–6146. 44
- Tampubolon, A. P., Gast, T., Klár, G., Fu, C., Teran, J., Jiang, C., and Museth, K. (2017). Multi-species simulation of porous sand and water mixtures. *ACM Transactions on Graphics (ToG)*, 36(4):1–11. 12
- Tancik, M., Weber, E., Ng, E., Li, R., Yi, B., Wang, T., Kristoffersen, A., Austin, J., Salahi, K., Ahuja, A., et al. (2023). Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–12. 72
- Tang, B., Wang, J., Wu, Z., and Zhang, L. (2023a). Stable score distillation for high-quality 3D generation. *arXiv preprint arXiv:2312.09305*. 43
- Tang, J., Chen, Z., Chen, X., Wang, T., Zeng, G., and Liu, Z. (2025). LGM: Large multi-view Gaussian model for high-resolution 3D content creation. In *European Conference on Computer Vision*, pages 1–18. Springer. 1, 111
- Tang, J., Ren, J., Zhou, H., Liu, Z., and Zeng, G. (2023b). DreamGaussian: Generative Gaussian splatting for efficient 3D content creation. *arXiv preprint arXiv:2309.16653*. 43, 70
- Tang, J., Wang, T., Zhang, B., Zhang, T., Yi, R., Ma, L., and Chen, D. (2023c). Make-it-3D: High-fidelity 3D creation from a single image with diffusion prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22819–22829. 43
- Tang, L., Jia, M., Wang, Q., Phoo, C. P., and Hariharan, B. (2023d). Emergent correspondence from image diffusion. *Advances in Neural Information Processing Systems*, 36:1363–1389. 42
- Team, T. H. (2025). Hunyuan3D 2.1: From images to high-fidelity 3D assets with production-ready pbr material. 151
- Technology, K. (2025). Kling AI video generator (image-to-video model). Version 2.1, image-to-video generative model. 133, 145, 195
- Teng, Y., Levin, D. I., and Kim, T. (2016). Eulerian solid-fluid coupling. *ACM Transactions on Graphics (ToG)*, 35(6):1–8. 8, 11
- Tessler, C., Kasten, Y., Guo, Y., Mannor, S., Chechik, G., and Peng, X. B. (2023). CALM: Conditional adversarial latent models for directable virtual characters. In *SIGGRAPH*. 143
- Tevet, G., Raab, S., Cohan, S., Reda, D., Luo, Z., Peng, X. B., Bermano, A. H., and van de Panne, M. (2025). CLoSD: Closing the loop between simulation and diffusion for multi-task character control. In *ICLR*. 143

- Tevet, G., Raab, S., Gordon, B., Shafir, Y., Cohen-Or, D., and Bermano, A. H. (2022). Human motion diffusion model. 126
- Tripo3D (2025). Tripo studio 3D model generator. <https://studio.tripo3d.ai/>. 133
- Turki, H., Agrawal, V., Bulò, S. R., Porzi, L., Kotschieder, P., Ramanan, D., Zollhöfer, M., and Richardt, C. (2023). HybridNeRF: Efficient neural rendering via adaptive volumetric surfaces. *arXiv preprint arXiv:2312.03160*. 85
- Vahdat, A., Williams, F., Gojcic, Z., Litany, O., Fidler, S., Kreis, K., et al. (2022). Lion: Latent point diffusion models for 3D shape generation. *Advances in Neural Information Processing Systems*, 35:10021–10039. 43
- Valkov, B., Rycroft, C. H., and Kamrin, K. (2015). Eulerian method for multiphase interactions of soft solid bodies in fluids. *Journal of Applied Mechanics*, 82(4):041011. 8, 11
- Vilesov, A., Chari, P., and Kadambi, A. (2023). CG3D: Compositional generation for text-to-3D via Gaussian splatting. *arXiv preprint arXiv:2311.17907*. 43
- Voleti, V., Yao, C.-H., Boss, M., Letts, A., Pankratz, D., Tochilkin, D., Laforte, C., Rombach, R., and Jampani, V. (2024). Sv3D: Novel multi-view synthesis and 3D generation from a single image using latent video diffusion. *arXiv preprint arXiv:2403.12008*. 42
- Voleti, V., Yao, C.-H., Boss, M., Letts, A., Pankratz, D., Tochilkin, D., Laforte, C., Rombach, R., and Jampani, V. (2025). SV3D: Novel multi-view synthesis and 3D generation from a single image using latent video diffusion. In *European Conference on Computer Vision*, pages 439–457. Springer. 107
- Wang, C., Jiang, R., Chai, M., He, M., Chen, D., and Liao, J. (2023a). NeRF-Art: Text-driven neural radiance fields stylization. *IEEE Transactions on Visualization and Computer Graphics*. 84
- Wang, H., Du, X., Li, J., Yeh, R. A., and Shakhnarovich, G. (2023b). Score Jacobian chaining: Lifting pretrained 2D diffusion models for 3D generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12619–12629. 43
- Wang, H., Jin, Y., Luo, A., Yang, X., and Zhu, B. (2020a). Codimensional surface tension flow using moving-least-squares particles. *ACM Transactions on Graphics (ToG)*, 39(4):42–1. 11
- Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupperecht, C., and Novotny, D. (2025a). VGGT: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306. 130

- Wang, J. and He, Y. (2019). Physics-aware 3D mesh synthesis. In *2019 International Conference on 3D Vision (3DV)*, pages 502–512. IEEE. 40, 43
- Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., and Wang, W. (2021). NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*. 44, 76, 83
- Wang, P. and Shi, Y. (2023). ImageDream: Image-prompt multi-view diffusion for 3D generation. *arXiv preprint arXiv:2312.02201*. 43
- Wang, Q., Eldesokey, A., Mendiratta, M., Zhan, F., Kortylewski, A., Theobalt, C., and Wonka, P. (2024a). Zero-shot video semantic segmentation based on pre-trained diffusion models. *arXiv:2405.16947*. 107
- Wang, R., Xu, S., Dong, Y., Deng, Y., Xiang, J., Lv, Z., Sun, G., Tong, X., and Yang, J. (2025b). MoGe-2: Accurate monocular geometry with metric scale and sharp details. *arXiv preprint arXiv:2507.02546*. 147, 193
- Wang, X., Li, G., Kuo, Y.-L., Kocabas, M., Aksan, E., and Hilliges, O. (2022). Reconstructing action-conditioned human-object interactions using commonsense knowledge priors. In *3DV*. 142
- Wang, X., Qiu, Y., Slattery, S. R., Fang, Y., Li, M., Zhu, S.-C., Zhu, Y., Tang, M., Manocha, D., and Jiang, C. (2020b). A massively parallel and scalable multi-GPU material point method. *ACM Transactions on Graphics (ToG)*, 39(4):30–1. 64
- Wang, Y., Wang, X., Chen, Z., Wang, Z., Sun, F., and Zhu, J. (2024b). Vidu4D: Single generated video to high-fidelity 4D reconstruction with dynamic Gaussian surfels. In *Advances in Neural Information Processing Systems (NeurIPS)*. 124
- Wang, Y., Zhao, Q., Yu, R., Zeng, A., Lin, J., Luo, Z., Tsui, H. W., Yu, J., Li, X., Chen, Q., et al. (2024c). SkillMimic: Learning reusable basketball skills from demonstrations. *arXiv preprint arXiv:2408.15270*. 143
- Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612. 134
- Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., and Zhu, J. (2024d). ProlificDreamer: High-fidelity and diverse text-to-3D generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36. 41, 43
- Wang, Z., Yuan, Z., Wang, X., Li, Y., Chen, T., Xia, M., Luo, P., and Shan, Y. (2024e). MotionCtrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11. 1, 108
- Wanger, L., Ferwerda, J., and Greenberg, D. (1992). Perceiving spatial relationships in computer-generated images. *IEEE Computer Graphics and Applications*, 12(3):44–58. 91

- Wei, Y., Zhang, S., Qing, Z., Yuan, H., Liu, Z., Liu, Y., Zhang, Y., Zhou, J., and Shan, H. (2024). DreamVideo: Composing your dream videos with customized subject and motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6537–6549. 108
- Weiler, M., Koschier, D., Brand, M., and Bender, J. (2018). A physically consistent implicit viscosity solver for SPH fluids. *Computer Graphics Forum*, 37(2):145–155. 16
- Wen, B., Yang, W., Kautz, J., and Birchfield, S. (2024). FoundationPose: Unified 6D pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17868–17879. 146, 147, 195
- Weng, H., Yang, T., Wang, J., Li, Y., Zhang, T., Chen, C. L. P., and Zhang, L. (2023). Consistent123: Improve consistency for one image to 3D object synthesis. 42
- Willett, N. S., Kazi, R. H., Chen, M., Fitzmaurice, G., Finkelstein, A., and Grossman, T. (2018). A mixed-initiative interface for animating static pictures. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, pages 649–661. 107
- Willett, N. S., Li, W., Popovic, J., Berthouzoz, F., and Finkelstein, A. (2017). Secondary motion for performed 2D animation. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 97–108. 108
- Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., and Wang, X. (2023a). 4D Gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*. 64, 65, 69, 77
- Wu, R., Su, W., Ma, K., and Liao, J. (2024a). AniClipart: Clipart animation with text-to-video priors. *arXiv:2404.12347*. 107
- Wu, T., Sun, J.-M., Lai, Y.-K., and Gao, L. (2023b). DE-NeRF: Decoupled neural radiance fields for view-consistent appearance editing and high-frequency environmental relighting. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11. 83, 84
- Wu, T., Yang, G., Li, Z., Zhang, K., Liu, Z., Guibas, L., Lin, D., and Wetzstein, G. (2024b). Gpt-4v (ision) is a human-aligned evaluator for text-to-3D generation. *arXiv preprint arXiv:2401.04092*. 57, 167
- Wu, W., Li, Z., Gu, Y., Zhao, R., He, Y., Zhang, D. J., Shou, M. Z., Li, Y., Gao, T., and Zhang, D. (2025a). DragAnything: Motion control for anything using entity representation. In *European Conference on Computer Vision*, pages 331–348. Springer. 105, 108, 111, 116, 117, 182, 183, 184, 185
- Wu, Z., Li, J., Xu, P., and Liu, C. K. (2025b). Human-object interaction from human-level instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11176–11186. 142

- Wu, Z., Yu, C., Jiang, Y., Cao, C., Wang, F., and Bai, X. (2024c). SC4D: Sparse-controlled video-to-4D generation and motion transfer. In *European Conference on Computer Vision*, pages 361–379. Springer. 123, 134, 135, 189, 190
- Wu, Z., Yu, C., Wang, F., and Bai, X. (2025c). AnimateAnyMesh: A feed-forward 4D foundation model for text-driven universal mesh animation. *arXiv preprint arxiv:2506.09982*. 124
- Xia, H., Lin, Z.-H., Ma, W.-C., and Wang, S. (2024). Video2Game: Real-time, interactive, realistic and browser-compatible environment from a single video. In *CVPR*. 83
- Xie, T., Chen, Y., Guo, Y., Yang, Y., Zhou, B., Terzopoulos, D., Jiang, Y., and Jiang, C. (2025a). AnimaMimic: Imitating 3D animation from video priors. *arXiv preprint arXiv:2512.14133*. 6
- Xie, T., Li, M., Yang, Y., and Jiang, C. (2023a). A contact proxy splitting method for Lagrangian solid-fluid coupling. *ACM Transactions on Graphics (ToG)*, 42(4):1–14. 2
- Xie, T., Zhao, Y., Jiang, Y., and Jiang, C. (2025b). PhysAnimator: Physics-guided generative cartoon animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10793–10804. 4
- Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., and Jiang, C. (2023b). Phys-Gaussian: Physics-integrated 3D Gaussians for generative dynamics. *arXiv preprint arXiv:2311.12198*. 4, 43, 82, 84, 86, 89, 90, 91, 95, 96
- Xie, X., Bhatnagar, B. L., Lenssen, J. E., and Pons-Moll, G. (2024a). Template free reconstruction of human-object interaction with procedural interaction generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10003–10015. 140, 142
- Xie, X., Bhatnagar, B. L., and Pons-Moll, G. (2022a). CHORE: Contact, human and object reconstruction from a single RGB image. In *ECCV*. 142
- Xie, X., Lenssen, J. E., and Pons-Moll, G. (2025c). InterTrack: Tracking human object interaction without object templates. In *2025 International Conference on 3D Vision (3DV)*, pages 1427–1439. IEEE. 140, 142
- Xie, X., Wen, B., Chang, Y., Rabeti, H., Li, J., Yuan, Y., Pons-Moll, G., and Birchfield, S. (2026). CARI4D: Category agnostic 4D reconstruction of human-object interaction. In *CVPR*. 142
- Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., and Sridhar, S. (2022b). Neural fields in visual computing and beyond. *Computer Graphics Forum*, 41(2):641–676. 83
- Xie, Y., Yao, C.-H., Voleti, V., Jiang, H., and Jampani, V. (2024b). SV4D: Dynamic 3D content generation with multi-frame and multi-view consistency. *arXiv preprint arXiv:2407.17470*. 123

- Xing, J., Kazi, R. H., Grossman, T., Wei, L.-Y., Stam, J., and Fitzmaurice, G. (2016). Energy-brushes: Interactive tools for illustrating stylized elemental dynamics. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 755–766. 108, 112, 113
- Xing, J., Liu, H., Xia, M., Zhang, Y., Wang, X., Shan, Y., and Wong, T.-T. (2024). ToonCrafter: Generative cartoon interpolation. *arXiv:2405.17933*. 107, 109, 115, 120, 122
- Xing, J., Wei, L.-Y., Shiratori, T., and Yatani, K. (2015). Autocomplete hand-drawn animations. *ACM Transactions on Graphics (ToG)*, 34(6):1–11. 107
- Xing, J., Xia, M., Zhang, Y., Chen, H., Yu, W., Liu, H., Liu, G., Wang, X., Shan, Y., and Wong, T.-T. (2025). DynamiCrafter: Animating open-domain images with video diffusion priors. In *European Conference on Computer Vision*, pages 399–417. Springer. 1, 105, 107, 108, 116, 117, 182, 183, 184, 185
- Xing, Z., Feng, Q., Chen, H., Dai, Q., Hu, H., Xu, H., Wu, Z., and Jiang, Y.-G. (2023). A survey on video diffusion models. *ACM Computing Surveys*. 107
- Xu, D., Liang, H., Bhatt, N. P., Hu, H., Liang, H., Plataniotis, K. N., and Wang, Z. (2024a). Comp4D: Llm-guided compositional 4D scene generation. *arXiv preprint arXiv:2403.16993*. 124
- Xu, L., Agrawal, V., Laney, W., Garcia, T., Bansal, A., Kim, C., Rota Bulò, S., Porzi, L., Kotschieder, P., Božič, A., et al. (2023a). VR-NeRF: High-fidelity virtualized walkable spaces. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–12. 85
- Xu, Q., Liu, J., Wong, M., Chen, C., and Ong, Y.-S. (2024b). Precise-physics driven text-to-3D generation. *arXiv preprint arXiv:2403.12438*. 44
- Xu, Q., Xu, Z., Philip, J., Bi, S., Shu, Z., Sunkavalli, K., and Neumann, U. (2022a). Point-NeRF: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448. 63, 64
- Xu, S., Li, Z., Wang, Y.-X., and Gui, L.-Y. (2023b). InterDiff: Generating 3D human-object interactions with physics-informed diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14928–14940. 142
- Xu, S., Ling, H. Y., Wang, Y.-X., and Gui, L.-Y. (2025). InterMimic: Towards universal whole-body control for physics-based human-object interactions. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12266–12277. 143, 154, 155, 199
- Xu, S., Wang, Z., Wang, Y.-X., and Gui, L.-Y. (2024c). InterDreamer: Zero-shot text to 3D dynamic human-object interaction. In *NeurIPS*. 142
- Xu, T. and Harada, T. (2022). Deforming radiance fields with cages. In *European Conference on Computer Vision*, pages 159–175. Springer. 61, 76

- Xu, Y., Chen, B., Li, Z., Zhang, H., Wang, L., Zheng, Z., and Liu, Y. (2023c). Gaussian head avatar: Ultra high-fidelity head avatar via dynamic Gaussians. *arXiv preprint arXiv:2312.03029*. 82
- Xu, Y., Tan, H., Luan, F., Bi, S., Wang, P., Li, J., Shi, Z., Sunkavalli, K., Wetzstein, G., Xu, Z., et al. (2023d). DMV3D: Denoising multi-view diffusion using 3D large reconstruction model. *arXiv preprint arXiv:2311.09217*. 43
- Xu, Y., Zhang, J., Zhang, Q., and Tao, D. (2022b). ViTPose: Simple vision transformer baselines for human pose estimation. *Advances in Neural Information Processing Systems*, 35:38571–38584. 147
- Xu, Z., Zhou, Y., Kalogerakis, E., Landreth, C., and Singh, K. (2020). RigNet: Neural rigging for articulated characters. *ACM Transactions on Graphics (ToG)*, 39. 125, 133
- Yan, X., Li, C.-F., Chen, X.-S., and Hu, S.-M. (2018). MPM simulation of interacting fluids and solids. *Computer Graphics Forum*, 37(8):183–193. 12
- Yang, F., Zhang, J., Shi, Y., Chen, B., Zhang, C., Zhang, H., Yang, X., Feng, J., and Lin, G. (2024a). Magic-Boost: Boost 3D generation with mutli-view conditioned diffusion. *arXiv preprint arXiv:2404.06429*. 43
- Yang, J., Cheng, Z., Duan, Y., Ji, P., and Li, H. (2023a). ConsistNet: Enforcing 3D consistency for multi-view images diffusion. *arXiv*. 42
- Yang, S., Hou, L., Huang, H., Ma, C., Wan, P., Zhang, D., Chen, X., and Liao, J. (2024b). Direct-a-Video: Customized video generation with user-directed camera movement and object motion. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–12. 108
- Yang, X., Chen, Y., Chen, C., Zhang, C., Xu, Y., Yang, X., Liu, F., and Lin, G. (2023b). Learn to optimize denoising scores for 3D generation: A unified and improved diffusion prior on nerf and 3D gaussian splatting. *arXiv preprint arXiv:2312.04820*. 43
- Yang, Y., Jia, B., Zhi, P., and Huang, S. (2024c). PhyScene: Physically interactable 3D scene synthesis for embodied AI. *arXiv preprint arXiv:2404.09465*. 40, 43
- Yang, Y., Sun, F.-Y., Weihs, L., VanderBilt, E., Herrasti, A., Han, W., Wu, J., Haber, N., Krishna, R., Liu, L., et al. (2024d). Holodeck: Language guided generation of 3D embodied AI environments. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2024)*, volume 30, pages 20–25. IEEE/CVF. 40
- Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., and Jin, X. (2023c). Deformable 3D Gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*. 64
- Yang, Z., Pan, Z., Gu, C., and Zhang, L. (2025). Diffusion²: Dynamic 3D content generation via score composition of video and multi-view diffusion models. In *International Conference on Learning Representations (ICLR)*. 124

- Yang, Z., Yang, H., Pan, Z., Zhu, X., and Zhang, L. (2023d). Real-time photorealistic dynamic scene representation and rendering with 4D Gaussian splatting. *arXiv preprint arXiv:2310.10642*. 64, 82, 84
- Yariv, L., Gu, J., Kasten, Y., and Lipman, Y. (2021). Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34:4805–4815. 83
- Ye, J., Wang, P., Li, K., Shi, Y., and Wang, H. (2023a). Consistent-1-to-3: Consistent image to 3D view synthesis via geometry-aware diffusion models. *arXiv preprint arXiv:2310.03020*. 42
- Ye, M., Danelljan, M., Yu, F., and Ke, L. (2023b). Gaussian grouping: Segment and edit anything in 3D scenes. *arXiv preprint arXiv:2312.00732*. 80, 84
- Ye, Y., Li, X., Gupta, A., De Mello, S., Birchfield, S., Song, J., Tulsiani, S., and Liu, S. (2023c). Affordance diffusion: Synthesizing hand-object interactions. In *CVPR*. 142
- Yi, X., Wu, Z., Xu, Q., Zhou, P., Lim, J.-H., and Zhang, H. (2024). Diffusion time-step curriculum for one image to 3D generation. *arXiv preprint arXiv:2404.04562*. 43
- Yin, S., Wu, C., Liang, J., Shi, J., Li, H., Ming, G., and Duan, N. (2023). DragNUWA: Fine-grained control in video generation by integrating text, image, and trajectory. *arXiv:2308.08089*. 107
- Yu, J., Fu, L., Huang, H., El-Refai, K., Ambrus, R. A., Cheng, R., Irshad, M. Z., and Goldberg, K. (2025). Real2Render2Real: Scaling robot data without dynamics simulation or robot hardware. In *Conference on Robot Learning (CoRL)*. 143
- YU, M., Hu, W., Xing, J., and Shan, Y. (2025). TrajectoryCrafter: Redirecting camera trajectory for monocular videos via diffusion models. *arXiv preprint arXiv:2503.05638*. 122
- Yu, X., Guo, Y.-C., Li, Y., Liang, D., Zhang, S.-H., and QI, X. (2024). Text-to-3D with classifier score distillation. In *The Twelfth International Conference on Learning Representations*. 43
- Yuan, Y., Song, J., Iqbal, U., Vahdat, A., and Kautz, J. (2023). PhysDiff: Physics-guided human motion diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16010–16021. 43, 126
- Yuan, Y.-J., Sun, Y.-T., Lai, Y.-K., Ma, Y., Jia, R., and Gao, L. (2022). NeRF-Editing: Geometry editing of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18353–18364. 61, 64, 76, 78, 79
- Yue, Y., Smith, B., Batty, C., Zheng, C., and Grinspun, E. (2015). Continuum foam: A material point method for shear-dependent flows. *ACM Transactions on Graphics (ToG)*, 34(5):1–20. 74, 177

- Zarifi, O. and Batty, C. (2017). A positive-definite cut-cell method for strong two-way coupling between fluids and deformable bodies. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 1–11. 8, 11, 35, 37
- Ze, Y., Araújo, J. P., Wu, J., and Liu, C. K. (2025a). GMR: General motion retargeting. <https://github.com/YanjieZe/GMR>. 148
- Ze, Y., Chen, Z., Araújo, J. P., Cao, Z.-a., Peng, X. B., Wu, J., and Liu, C. K. (2025b). TWIST: Teleoperated whole-body imitation system. In *Conference on Robot Learning (CoRL)*. 140, 143
- Ze, Y., Zhao, S., Wang, W., Kanazawa, A., Duan, R., Abbeel, P., Shi, G., Wu, J., and Liu, C. K. (2025c). TWIST2: Scalable, portable, and holistic humanoid data collection system. *arXiv preprint arXiv:2511.02832*. 140, 143
- Zhan, X., Fu, R., and Ritchie, D. (2024). CharacterMixer: Rig-aware interpolation of 3D characters. 126
- Zhang, F. (2006). *The Schur Complement and its Applications*, volume 4. Springer Science & Business Media. 24
- Zhang, H., Chen, Z., Xu, H., Hao, L., Wu, X., Xu, S., Zhang, Z., Wang, Y., and Xiong, R. (2024a). Semantics-aware motion retargeting with vision-language models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 126
- Zhang, H., Christen, S., Fan, Z., Zheng, L., Hwangbo, J., Song, J., and Hilliges, O. (2023a). ArtiGrasp: Physically plausible synthesis of bi-manual dexterous grasping and articulation. *arXiv preprint arXiv:2309.03891*. 142
- Zhang, H., Yuan, Y., Makoviychuk, V., Guo, Y., Fidler, S., Peng, X., and Fatahalian, K. (2023b). Learning physically simulated tennis skills from broadcast videos. *ACM Transactions on Graphics (ToG)*, 42(4). 143
- Zhang, J., Chen, Y., Wang, Z., Yang, J., Wang, Y., and Huang, S. (2025a). InteractAnything: Zero-shot human object interaction synthesis via llm feedback and object affordance parsing. In *CVPR*. 142
- Zhang, J., Luo, H., Yang, H., Xu, X., Wu, Q., Shi, Y., Yu, J., Xu, L., and Wang, J. (2023c). NeuralDome: A neural modeling pipeline on multi-view human-object interactions. In *CVPR*. 142
- Zhang, J. E., Bang, S., Levin, D. I., and Jacobson, A. (2020a). Complementary dynamics. *arXiv:2009.02462*. 108
- Zhang, J. E., Bang, S., Levin, D. I., and Jacobson, A. (2020b). Complementary dynamics. *ACM Transactions on Graphics (ToG)*. 126

- Zhang, J.-P., Pu, C.-F., Guo, M.-H., Cao, Y.-P., and Hu, S.-M. (2025b). One model to rig them all: Diverse skeleton rigging with unirig. **125**
- Zhang, J.-P., Pu, C.-F., Guo, M.-H., Cao, Y.-P., and Hu, S.-M. (2025c). One model to rig them all: Diverse skeleton rigging with unirig. *ACM Transactions on Graphics (ToG)*, 44(4):1–18. **128**
- Zhang, J. Y., Pepose, S., Joo, H., Ramanan, D., Malik, J., and Kanazawa, A. (2020c). Perceiving 3D human-object spatial arrangements from a single image in the wild. In *European Conference on Computer Vision (ECCV)*. **142**
- Zhang, L., Rao, A., and Agrawala, M. (2023d). Adding conditional control to text-to-image diffusion models. **108, 109, 114**
- Zhang, M., Cai, Z., Pan, L., Hong, F., Guo, X., Yang, L., and Liu, Z. (2022). Motion-Diffuse: Text-driven human motion generation with diffusion model. *arXiv preprint arXiv:2208.15001*. **126**
- Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595. **134**
- Zhang, S., Qiao, Y.-L., Zhu, G., Heiden, E., Turpin, D., Liu, J., Lin, M., Macklin, M., and Garg, A. (2023e). HandyPriors: Physically consistent perception of hand-object interactions with differentiable priors. *arXiv preprint arXiv:2311.16552*. **44**
- Zhang, T., Yu, H.-X., Wu, R., Feng, B. Y., Zheng, C., Snavely, N., Wu, J., and Freeman, W. T. (2024b). PhysDreamer: Physics-based interaction with 3D objects via video generation. *arXiv preprint arXiv:2404.13026*. **44**
- Zhang, Y., Gu, J., Wang, L.-W., Wang, H., Cheng, J., Zhu, Y., and Zou, F. (2024c). MimicMotion: High-quality human motion video generation with confidence-aware pose guidance. *arXiv:2406.19680*. **107**
- Zhang, Z., Liao, J., Li, M., Qin, L., and Wang, W. (2024d). Tora: Trajectory-oriented diffusion transformer for video generation. *arXiv:2407.21705*. **108**
- Zhang, Z., Long, F., Pan, Y., Qiu, Z., Yao, T., Cao, Y., and Mei, T. (2024e). TRIP: Temporal residual learning with image noise prior for image-to-video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8671–8681. **107**
- Zhao, C., Zhang, J., Du, J., Shan, Z., Wang, J., Yu, J., Wang, J., and Xu, L. (2024). I²M HOI: Inertia-aware monocular capture of 3D human-object interactions. In *CVPR*. **142**
- Zhao, M., Zhao, C., Liang, X., Li, L., Zhao, Z., Hu, Z., Fan, C., and Yu, X. (2023a). EfficientDreamer: High-fidelity and robust 3D creation via orthogonal-view diffusion prior. *arXiv preprint arXiv:2308.13223*. **43**

- Zhao, R., Gu, Y., Wu, J. Z., Zhang, D. J., Liu, J., Wu, W., Keppo, J., and Shou, M. Z. (2023b). MotionDirector: Motion customization of text-to-video diffusion models. *arXiv:2310.08465*. 108
- Zhao, Y., Lin, C.-C., Lin, K., Yan, Z., Li, L., Yang, Z., Wang, J., Lee, G. H., and Wang, L. (2025). GenXD: Generating any 3D and 4D scenes. In *ICLR*. 124
- Zheng, C. and Vedaldi, A. (2023). Free3D: Consistent novel view synthesis without 3D representation. *arXiv*. 42
- Zheng, J., Zheng, Q., Fang, L., Liu, Y., and Yi, L. (2023a). CAMS: Canonicalized manipulation spaces for category-level functional hand-object manipulation synthesis. In *CVPR*. 142
- Zheng, X.-Y., Liu, Y., Wang, P.-S., and Tong, X. (2022). SDF-StyleGAN: Implicit sdf-based stylegan for 3D shape generation. 126
- Zheng, X.-Y., Pan, H., Wang, P.-S., Tong, X., Liu, Y., and Shum, H.-Y. (2023b). Locally attentional sdf diffusion for controllable 3D shape generation. *ACM Transactions on Graphics (SIGGRAPH)*, 42(4). 43
- Zhou, H., Yu, W., Guan, J., Cheng, X., Tian, Y., and Yuan, L. (2025). HoloTime: Taming video diffusion models for panoramic 4D scene generation. 124
- Zhou, K., Bhatnagar, B. L., Lenssen, J. E., and Pons-Moll, G. (2022). TOCH: Spatio-temporal object-to-hand correspondence for motion refinement. In *ECCV*. 142
- Zhou, Y., Barnes, C., Lu, J., Yang, J., and Li, H. (2019). On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753. 128, 195
- Zhou, Z. and Tulsiani, S. (2023). SparseFusion: Distilling view-conditioned diffusion for 3D reconstruction. In *CVPR*. 43
- Zhu, H., He, T., Yu, X., Guo, J., Chen, Z., and Bian, J. (2025). AR4D: Autoregressive 4D generation from monocular videos. 124
- Zhu, J. and Zhuang, P. (2023). Hifa: High-fidelity text-to-3D with advanced diffusion guidance. *arXiv preprint arXiv:2305.18766*. 41, 43
- Zhu, S., Chen, J. L., Dai, Z., Dong, Z., Xu, Y., Cao, X., Yao, Y., Zhu, H., and Zhu, S. (2024). Champ: Controllable and consistent human image animation with 3D parametric guidance. In *European Conference on Computer Vision*, pages 145–162. Springer. 122
- Zhu, Y. and Bridson, R. (2005). Animating sand as a fluid. *ACM Transactions on Graphics (ToG)*, 24(3):965–972. 83
- Zielonka, W., Bagautdinov, T., Saito, S., Zollhöfer, M., Thies, J., and Romero, J. (2023). Drivable 3D Gaussian avatars. *arXiv preprint arXiv:2311.08581*. 82

Zong, Z., Li, X., Li, M., Chiaramonte, M. M., Matusik, W., Grinspun, E., Carlberg, K., Jiang, C., and Chen, P. Y. (2023). Neural stress fields for reduced-order elastoplasticity and fracture. *arXiv preprint arXiv:2310.17790*. 72, 174