

Vid2Sim: Generalizable, Video-based Reconstruction of Appearance, Geometry and Physics for Mesh-free Simulation

Chuhao Chen¹ Zhiyang Dou^{1,2} Chen Wang¹ Yiming Huang¹

Anjun Chen^{1,3} Qiao Feng¹ Jiatao Gu¹ Lingjie Liu¹

¹University of Pennsylvania ²The University of Hong Kong ³Zhejiang University

{chuhaoc, zydou, chenw30, ymhuang9, chen3110, fengqiao, jgu32, lingjie.liu}@seas.upenn.edu

<https://czzzzh.github.io/Vid2Sim>

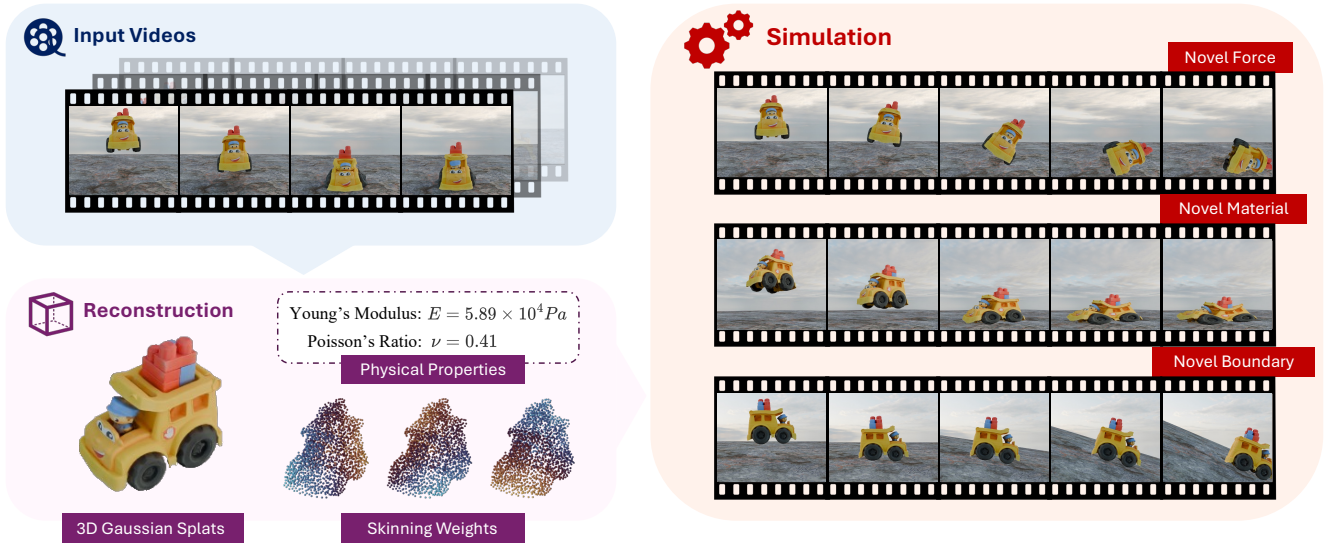


Figure 1. Vid2Sim achieves high-quality reconstruction of appearance, geometry, and physics from multi-view videos effectively. The reconstruction results are simulation-ready, enabling high-fidelity and visually appealing animations via mesh-free simulation. Here, we present our method’s reconstruction and simulation results in the GSO [12] dataset.

Abstract

Faithfully reconstructing textured shapes and physical properties from videos presents an intriguing yet challenging problem. Significant efforts have been dedicated to advancing such a system identification problem in this area. Previous methods often rely on heavy optimization pipelines with a differentiable simulator and renderer to estimate physical parameters. However, these approaches frequently necessitate extensive hyperparameter tuning for each scene and involve a costly optimization process, which limits both their practicality and generalizability. In this work, we propose a novel framework, Vid2Sim, a generalizable video-based approach for recovering geometry and physical properties through a mesh-free reduced simulation based on Linear Blend Skinning (LBS), offering high computational

efficiency and versatile representation capability. Specifically, Vid2Sim first reconstructs the observed configuration of the physical system from video using a feed-forward neural network trained to capture physical world knowledge. A lightweight optimization pipeline then refines the estimated appearance, geometry, and physical properties to closely align with video observations within just a few minutes. Additionally, after the reconstruction, Vid2Sim enables high-quality, mesh-free simulation with high efficiency. Extensive experiments demonstrate that our method achieves superior accuracy and efficiency in reconstructing geometry and physical properties from video data.

1. Introduction

Understanding and reconstructing appearance, geometry, and physical properties from observations with high fidelity,

a.k.a. system identification, is a fundamental yet challenging task in computer vision. Traditional methods [13, 17, 19, 24, 39, 46, 49] often rely on known shape information of given objects, which limits their practicality for broader applications. Recent advancements [5, 27, 32, 64] leverage neural representations, such as NeRF [40] and Gaussian Splatting [28] along with differentiable simulators [25] to create a unified framework that jointly learns 3D geometry, appearance, and physical parameters. That being said, none of the previous efforts have achieved accurate, generalizable, and efficient reconstruction of appearance, geometry, and physical properties from the input video, as they suffer from two main limitations. First, most existing methods [5, 27, 32, 64] employ heavy per-scene optimization to identify physical parameters, making the understanding of various scenes computationally expensive. Second, these approaches struggle to accurately model complex, physics-driven deformations, as they typically use Material Point Methods (MPM) [25] for simulation. This method is limited by its grid-based representation and its typical dependence on symplectic time integration, which constrains expressiveness. Although alternative approaches, such as Spring-Gaus [64], employ more efficient mass-spring models, they are limited to modeling elastic dynamics.

In this paper, we propose a novel framework, named Vid2Sim, for the high-fidelity reconstruction of textured shapes and the estimation of physical properties directly from videos. We first train a feed-forward neural network that integrates general physical knowledge, utilizing a pre-trained video vision transformer [54] to infer a range of physical attributes from the input video sequences. This component is coupled with an advanced 3D reconstruction pipeline [53] that predicts both object geometry and appearance, encoded with 3D Gaussians to facilitate instant system identification. In contrast to prior methods, Vid2Sim incorporates an efficient simulation pipeline leveraging an implicit Euler solver as inspired by [41]. This simulation approach is mesh-free and uses Linear Blend Skinning (LBS) to enable reduced-order, computationally efficient simulations that are highly adaptable to complex deformations and fully end-to-end trainable. Then, we perform a lightweight optimization with a novel Neural Jacobian module to efficiently refine estimates of appearance, geometry, and physical properties, aligning the reconstructed outputs precisely with observed video data. This post-prediction optimization completes in only a few minutes. Upon reconstruction, the system enables high-quality, mesh-free simulations via the implicit Euler solver, supporting accurate dynamic behavior modeling.

We conduct extensive experiments to evaluate our method where Vid2Sim demonstrates remarkable accuracy and efficiency in recovering geometry, appearance, and physical properties from videos compared to existing meth-

ods. In summary, our contributions are three-fold:

- We propose Vid2Sim, a novel framework for generalizable, video-based reconstruction of appearance, geometry, and physical properties for mesh-free, reduced-order simulation.
- We introduce a generalizable feed-forward model with physical world knowledge to estimate the dynamics, followed by an efficient optimization step with Neural Jacobian to improve the reconstruction results further.
- Vid2Sim demonstrates remarkable effectiveness and efficiency, achieving state-of-the-art performance in accuracy and speed compared to existing methods.

2. Related Work

2.1. Physics-aware Dynamic 3D reconstruction

Dynamic 3D reconstruction is one of the critical tasks in computer vision and graphics. Recent advances in 3D representations like NeRF [40] and 3D Gaussian Splatting [28] as well as template-based models [31, 36, 51] make it flexible to reconstruct complex 3D scenes from visual data. These methods are recently extended to a dynamic 3D reconstruction [45, 56, 60] from either monocular videos [16, 47, 52, 55, 57, 58, 61] or multi-view videos [38, 42, 43]. With the introduction of physics-informed learning [6, 8], approaches that incorporate physical priors to enhance the understanding and reconstruction of dynamic scenes have gained popularity. For instance, PAC-NeRF [32] first jointly reconstructed the dynamic scene and a simulatable model using the differentiable Material Point Method [22, 23], and it was subsequently improved regarding the quality [5, 27] and adaptability [64]. While these methods achieve physically complete reconstruction, none of them are generalizable. In contrast to all existing methods, we first propose a generalizable pipeline that achieves simulation-ready geometry and physical property recovery in a feed-forward manner, which is inspired by the recent achievements in large 3D reconstruction model [21, 53, 62] and 4D reconstruction model [48]. A highly efficient optimization step is conducted to further enhance the reconstruction quality.

2.2. Vision-based Physical Simulation

Mesh-free Physical Simulation Traditional physical elasticity simulation, such as the finite element method (FEM) [9], often requires a mesh or tetrahedral representation. This complicates the simulation of scenes reconstructed from visual data, often represented by NeRF or 3D Gaussians, as obtaining high-quality meshes from these models for simulation can be a non-trivial task. Mesh-free models have then been a popular alternative for vision-based physical simulation such as the material point method (MPM) [22, 25] and smoothed-particle hydrodynamics (SPH) [11, 30, 44]. However, neither is a purely point-based

method since SPH needs to update connectivity among neighborhoods and MPM requires maintaining a background grid. More importantly, these approaches bring significant computational burden. The very recent work *Simplicits* [41] thus proposed a mesh-free, geometry-agnostic, and reduced-order elastic simulation method, which offers another feasibility to do a vision-based physical simulation in an efficient and flexible way. Inspired by *Simplicits* [41], we develop a feed-forward model that efficiently delivers a generalizable initial estimate, coupled with a differentiable, reduced-order simulator that employs Linear Blend Skinning for rapid and accurate optimization of appearance, geometry, and physical properties.

Physical reconstruction and simulation from visual data

Apart from physics-aware dynamic 3D reconstruction, there are a lot of other applications in vision-based physical simulation with the help of mesh-free simulation methods. Works such as *PhysGaussian* [59] integrate mesh-free simulators with NeRF [15] or 3D Gaussians [26, 37], making it possible to interact with these representations. Some other works [14, 34, 35, 63] combine the simulation model with the video generation model [2–4, 50] to learn physical properties and generate dynamics. As of yet, all previous methods are limited by their reconstruction accuracy, generalization capability, and runtime cost.

3. Preliminary

We begin by introducing (1) mesh-free simulation [41], which operates without mesh or grid representation using a reduced-order simulator; and (2) 3D Gaussian Splatting [28] for modeling both geometry and appearance.

Mesh-Free, Reduced-Order Simulation Given a set of points $\{\mathbf{X}_i \in \mathbb{R}^3 \mid i = 1, 2, \dots, n\}$ at the rest position, following [41], we simulate the dynamics of the points with a set of handles (full affine transformations) $\{\mathbf{Z}_j \in \mathbb{R}^{3 \times 4} \mid j = 1, 2, \dots, m\}$ (or $\mathbf{z}_j \in \mathbb{R}^{12}$ in an equivalent vector form) with a reduced $m \ll n$. The deformation of the point $\mathbf{X}_i$ is then defined as

$$\mathbf{x}_i = \phi_i(\mathbf{X}_i, \mathbf{Z}) = \mathbf{X}_i + \sum_{j=1}^m W_{\theta;j}(\mathbf{X}_i) \mathbf{Z}_j [\mathbf{X}_i, 1]^\top, \quad (1)$$

where $\mathbf{x}_i$ represents the deformed position, and $W_{\theta;j}(\mathbf{X}_i)$ is a scalar weight for Linear Blending Skinning (LBS), predicted by a small Multilayer Perception (MLP) that models the transformation of each point based on the combined influence of the handles.

The handles $\mathbf{z}_i$ are initialized to zero to make sure the points are at the rest position at $t = 0$. Then, at each discrete

time step, the handles vary according to the implicit time integration with the following incremental potential equation containing an inertia term and a potential energy term:

$$\mathbf{z}_{t+1} = \underset{\mathbf{z}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{z} - \tilde{\mathbf{z}}_t\|_M + \Delta t^2 E_{\text{potential}}(\mathbf{z}_t) \quad (2)$$

where Δt is the simulation time step, $\tilde{\mathbf{z}}_t = \mathbf{z}_t + \Delta t \dot{\mathbf{z}}_t$ is the first order prediction of $\mathbf{z}_t$ and $E_{\text{potential}}(\mathbf{z}_t)$ is the potential energy from both internal and external forces. Following [41], when evolving $\mathbf{z}_t$ at each timestep, we usually sample a small set of key control points $\{\mathbf{X}_i^c \in \mathbb{R}^3 \mid i = 1, 2, \dots, k\}$, $k \ll n$, which is also called *cubature points*, to save the computational time and memory.

3D Gaussian Splatting 3D Gaussian Splatting [28] represents 3D scenes as Gaussian primitives. Each primitive is defined by the Gaussian function:

$$\mathcal{G}(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{p})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mathbf{p})} \quad (3)$$

where $\mathbf{p}$ is the center and $\boldsymbol{\Sigma} = \mathbf{R}\mathbf{S}\mathbf{S}^\top \mathbf{R}^\top$ is the covariance matrix, factorized into rotation matrix $\mathbf{R}$ and scaling matrix $\mathbf{S}$. For rendering, learnable parameters $\mathbf{p}$ and $\boldsymbol{\Sigma}$ are projected into camera coordinates as $\mathbf{p}' = \mathbf{K}\mathbf{W}[\mathbf{p}, 1]^\top$, $\boldsymbol{\Sigma}' = \mathbf{J}\mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^\top \mathbf{J}^\top$, where $\mathbf{K}$ is the camera’s intrinsic matrix, $\mathbf{W}$ the extrinsic matrix, and $\mathbf{J}$ the Jacobian matrix of the affine perspective projection. The Gaussian in image space is then: $\mathcal{G}'(\mathbf{x}') = e^{-\frac{1}{2}(\mathbf{x}'-\mathbf{p}')^\top \boldsymbol{\Sigma}'^{-1}(\mathbf{x}'-\mathbf{p}')}$, where $\mathbf{x}'$ is the pixel position transformed similarly to $\mathbf{p} \mapsto \mathbf{p}'$. Each 3D Gaussian primitive uses $\mathbf{c}$ and α to model appearance, with $\mathbf{c}$ representing view-dependent color (parameterized by spherical harmonics) and α the opacity. The pixel color $\mathbf{C}$ at $\mathbf{x}'$ is computed via volumetric alpha blending:

$$\mathbf{C}(\mathbf{x}') = \sum_{i=1}^N T_i \alpha_i \mathcal{G}'_i(\mathbf{x}') \mathbf{c}_i \quad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j \mathcal{G}'_j(\mathbf{x}')) \quad (4)$$

where $\mathcal{G}'(\mathbf{x}')$ is the Gaussian with transformed $\mathbf{p}'$ and $\boldsymbol{\Sigma}'$, and T_i is the transmittance along the ray.

To apply deformation to each Gaussian primitive, we apply $\phi(\mathbf{X}, \mathbf{Z})$ to $\mathbf{p}$ and construct $\boldsymbol{\Sigma} = \mathbf{L}'\mathbf{L}'^\top$ with $\mathbf{L}' = \mathbf{F}(\mathbf{R}\mathbf{S})$. Here, $\mathbf{F} = \frac{\partial \phi(\mathbf{p}, \mathbf{Z})}{\partial \mathbf{p}}$ is the deformation gradient, reflecting local deformation in continuum mechanics.

4. Method

We aim to jointly reconstruct the appearance, geometry, and physical properties of the given target from posed multiview videos that describe the dynamics. We focus on elastic material modeled by the Neo-Hookean constitutive model to reduce the state space that our feed-forward predictor needs to learn, where we only predict Young’s modulus E , Poisson’s ratio ν and estimated scalar LBS weight $W_{\theta;j}(\mathbf{X}_i)$.

Notably, our framework is not restricted to elastic materials and can be readily extended to various physical phenomena, which we demonstrate in the supplementary materials that our method generalizes across different material types. Our two-stage pipeline, illustrated in Fig. 2, is detailed below.

4.1. Feed-forward Physical System Identification

In the first stage, we develop several neural networks that learn physical world knowledge, enabling feed-forward reconstruction of the observed appearance, geometry, and physical configuration of the physical system from the video.

We leverage the prior knowledge of physical dynamics by utilizing VideoMAE [54] as the network backbone of our feed-forward predictor, which is a large video vision transformer pre-trained on a vast dataset of videos. The visual features extracted from the backbone are then decoded by several small MLPs, which function as the regression head to estimate physical properties. The whole network takes a single front-view video as input and regresses it to two physical parameters, $\{E, \nu\}$, relevant to elastic materials. Additionally, to enable mesh-free, reduced-order simulation, the network should also regress the LBS values $W_{\theta;j}(\mathbf{X}_i)$ used to deform positions for dynamics, as specified in Eq. (1). However, as the LBS values are implicitly modeled using an MLP in [41], it becomes challenging to estimate them directly in a feed-forward manner.

To address this problem, we introduce a HyperNetwork [18] approach for predicting the weights of MLP $\hat{\theta}_{lbs}$ for LBS estimation. This HyperNetwork is also implemented in a small MLP as a regression head, similar to the ones to predict E and ν . Additionally, it is tasked with regressing only the weights and biases of the final linear layer, keeping the other layers fixed at their default initialization. This design enhances the generalizability and robustness of LBS prediction during feed-forward inference. We demonstrate more details in our supplementary material.

To recover geometry and appearance, we process the first multiview frames of the input videos by applying the pre-trained Large Multi-view Gaussian Model [53], which leverages the generalizable knowledge of the textured shape recovery trained with large-scale 3D datasets, and efficiently reconstruct them into 3D Gaussians as the shape representation, which is then normalized into a canonical space.

Together, we recover the geometry, appearance, and physical properties through the two branches, as shown in Sec. 4.1 Stage I, with a short inference time. This produces a simulation-ready prediction that meets all the requirements to be simulated with our simulation method. The feed-forward prediction is considered as a *general* estimation, which is then further refined to more closely match the reference videos, resulting in a *specific* estimation. More

implementation details can be found in Sec. 5 and our supplementary material.

4.2. Scene-specific Refinement

We conduct joint optimization of geometry, appearance, LBS, and physical parameters to better fit the reconstruction with the input multiview videos. Our lightweight optimization is significantly more efficient, completing in approximately 15 minutes, compared to existing methods that typically require around 1.5 hours. Detailed statistics are provided in Tab. 5.

To improve the reconstruction quality of the shape and appearance, we first refine the 3D Gaussians via standard 3DGS training [28]. Next, we refine the LBS estimation model to capture physical dynamics, enhancing its alignment with the specific dynamics of the given object. Usually, optimizing the LBS, as in Simplicits [41], requires precomputing the Jacobian of the deformation gradient with respect to transformations, $\mathbf{J}(\mathbf{X}) = \frac{\partial \mathbf{F}(\mathbf{X}, \mathbf{z})}{\partial \mathbf{z}}$, where $\mathbf{z}$ is the vector form of transformation $\mathbf{Z}$. Since $\mathbf{F} = \frac{\partial \phi(\mathbf{X}, \mathbf{z})}{\partial \mathbf{X}}$ includes only linear terms of $\mathbf{z}$, $\mathbf{J}$ depends solely on $\mathbf{X}$. For cubature points $C \subseteq \{\mathbf{X}_i \in \mathbb{R}^3 \mid i = 1, 2, \dots, n\}$, the Jacobian $\mathbf{J} \in \mathbb{R}^{9N_c \times m \times m}$ grows large with increasing cubature points N_c and handles m , necessitating computation through auto-differentiation. Precomputing this Jacobian is manageable if done once for fixed neural LBS, but further LBS optimization makes this cost-prohibitive.

In our method, we accelerate the refinement (and simulation) by introducing a Neural Jacobian module.

Neural Jacobian. We employ a neural network trained to predict $\mathbf{J}_\theta(\mathbf{X})$ instead of computing it explicitly. The Neural Jacobian is trained following the LBS training using the loss function below

$$\mathcal{L}_J = \|\mathbf{J}_\theta(\mathbf{X})\mathbf{z} + \mathbf{I} - \mathbf{F}(\mathbf{X}, \mathbf{z})\|_1, \quad (5)$$

where $\mathbf{J}_\theta(\mathbf{X})\mathbf{z} + \mathbf{I}$ is an estimation of the deformation gradient $\mathbf{F}(\mathbf{X}, \mathbf{z})$ and its ground truth is much cheaper to get via finite differences. The training samples for $\mathbf{X}$ and $\mathbf{z}$ are generated in a data-free manner the same as [41]. We validate the effectiveness of the Neural Jacobian in Sec. 6.3 and our supplementary material. The speed-up is shown in Tab. 5.

Then, we optimize the physical parameters, with fine-tuning the LBS and the corresponding Neural Jacobian at the same time, to match the input videos. We use rendering loss to supervise the optimization. This process can be formulated as:

$$\begin{aligned} \theta_{lbs}^*, \theta_{jac}^*, E^*, \nu^* &= \underset{\theta_{lbs}, \theta_{jac}, E, \nu}{\operatorname{argmin}} \mathcal{L}_{rendering} \\ \mathcal{L}_{rendering} &= \frac{1}{N\Delta s} \sum_{i=1}^N \sum_{t=s}^{s+\Delta s} \|\mathbf{C}_{pred}(i, t) - \mathbf{C}_{gt}(i, t)\|_2^2. \end{aligned} \quad (6)$$

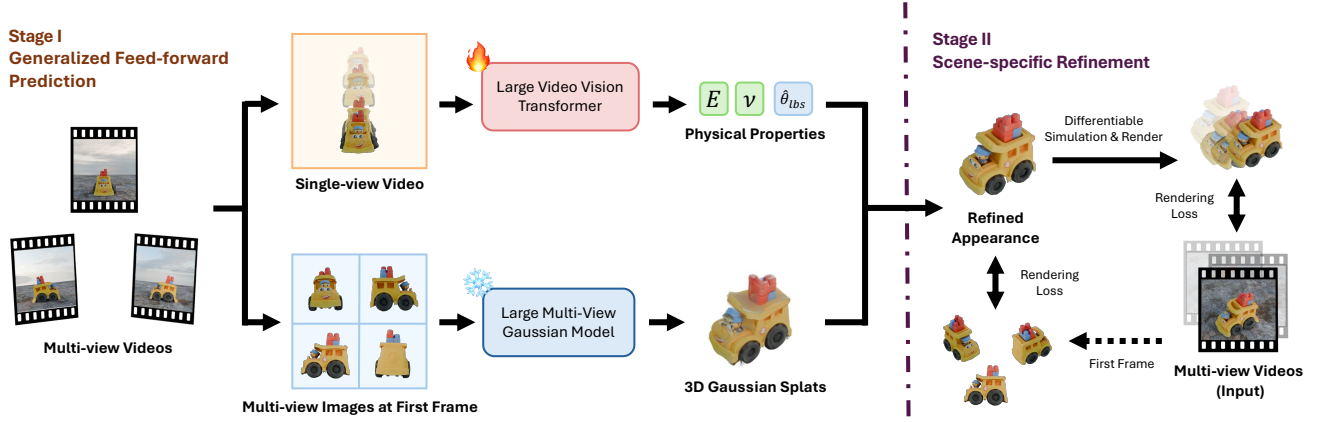


Figure 2. **An overview of Vid2Sim, comprising two stages.** In Stage I, a generalizable feed-forward model reconstructs appearance, geometry, and physical properties, generating simulation-ready outputs. In Stage II, a lightweight optimization pipeline refines these estimated attributes to closely match the input video. We introduce a mesh-free reduced simulation based on Linear Blend Skinning (LBS), which provides high computational efficiency and versatile representational capability for high-fidelity dynamic reconstruction.

Here, C_{pred} represents the rendering sequence from the simulation steps $\{z_s, z_{s+1}, \dots, z_{s+\Delta s}\}$, C_{gt} is the reference rendering sequence, and N denotes the number of views. For efficiency, we set $\Delta s = 4$ and randomly sample s from s' to $T - \Delta s$ in each iteration, in which s' is the first frame where C_{pred} is different from C_{gt} . This allows the process to cover the entire valid observation.

5. Implementation Details

5.1. Feed-forward Physical System Identification

Dataset. We choose 50k high-quality 3D objects from Objaverse [10] to construct our dataset (49k for training and 1k for validation). For each object, we generate an animation with the motion of falling to the ground at 448×448 resolution simulated by our reduced simulator, with randomly sampled $E \in [10^4, 10^6]$, $\nu \in [0.2, 0.5]$.

Implementation. We use two identical 4-layer MLPs to predict the scalar E and ν and a 4-layer MLP as the hyper-network to predict the final linear layer of the LBS network. We trained the whole network on one NVIDIA-L40 GPU for 1 day with the Adam [29] and a learning rate of 10^{-5} , where the backbone’s weights are fine-tuned from pretraining and the regression heads are trained from scratch.

5.2. Physical System Refinement

Dataset. To evaluate the performance of our full pipeline, we use both a synthetic dataset and a real-world dataset.

The synthetic dataset is a mesh dataset that contains 12 delicate objects collected from Google Scanned Objects (GSO) [12] with complex geometry and detailed texture. We use FEM to simulate animations in the most accurate physic as references. We rendered each animation from 12 different viewpoints at 448×448 resolution for 24 frames.

The first 16 frames are treated as observation, and the 8 frames remaining are references for future state prediction.

For the real-world dataset, we captured 3 different animations (See Fig. 4) *orange*, *bird* and *cup* with four posed cameras at surrounding views. We use BackgroundMattingV2 [33] with post-processing to obtain the mask of the object.

Implementation. We first refine the 3D Gaussians following the original 3DGS [28] and use the data-free method from [41] to train the full LBS layers and the corresponding Neural Jacobian. Afterwards, we jointly optimize $\{\theta_{lbs}, \theta_{jac}, E, \nu\}$ for 400 iterations. We also use the Adam optimizer and the learning rates are set to $\{5 \times 10^{-7}, 5 \times 10^{-7}, 5 \times 10^{-3}, 1 \times 10^{-3}\}$. We use 10 control handles and 500 cubature points for simulation. We use Farthest Point Sampling (FPS) to sample cubature points.

6. Experiments

6.1. Baselines and Metrics

We compare our method with the state-of-the-art methods: GIC [5], Spring-GS [64], and PAC-NeRF [32] on the dynamic reconstruction task and the future state prediction task at both synthetic and real-world datasets. We use the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Metric (SSIM), and video perceptual loss (FoVDP) as the metrics for evaluation. We additionally report the running time of each method to assess runtime efficiency in Tab. 5.

6.2. Evaluation on the synthetic dataset

Following previous methods [5, 27, 32, 64], we evaluate our method and baselines for dynamic reconstruction on the 12 diverse synthetic test cases. Both qualitative results (Fig. 3) and quantitative results (Tab. 1) show that our

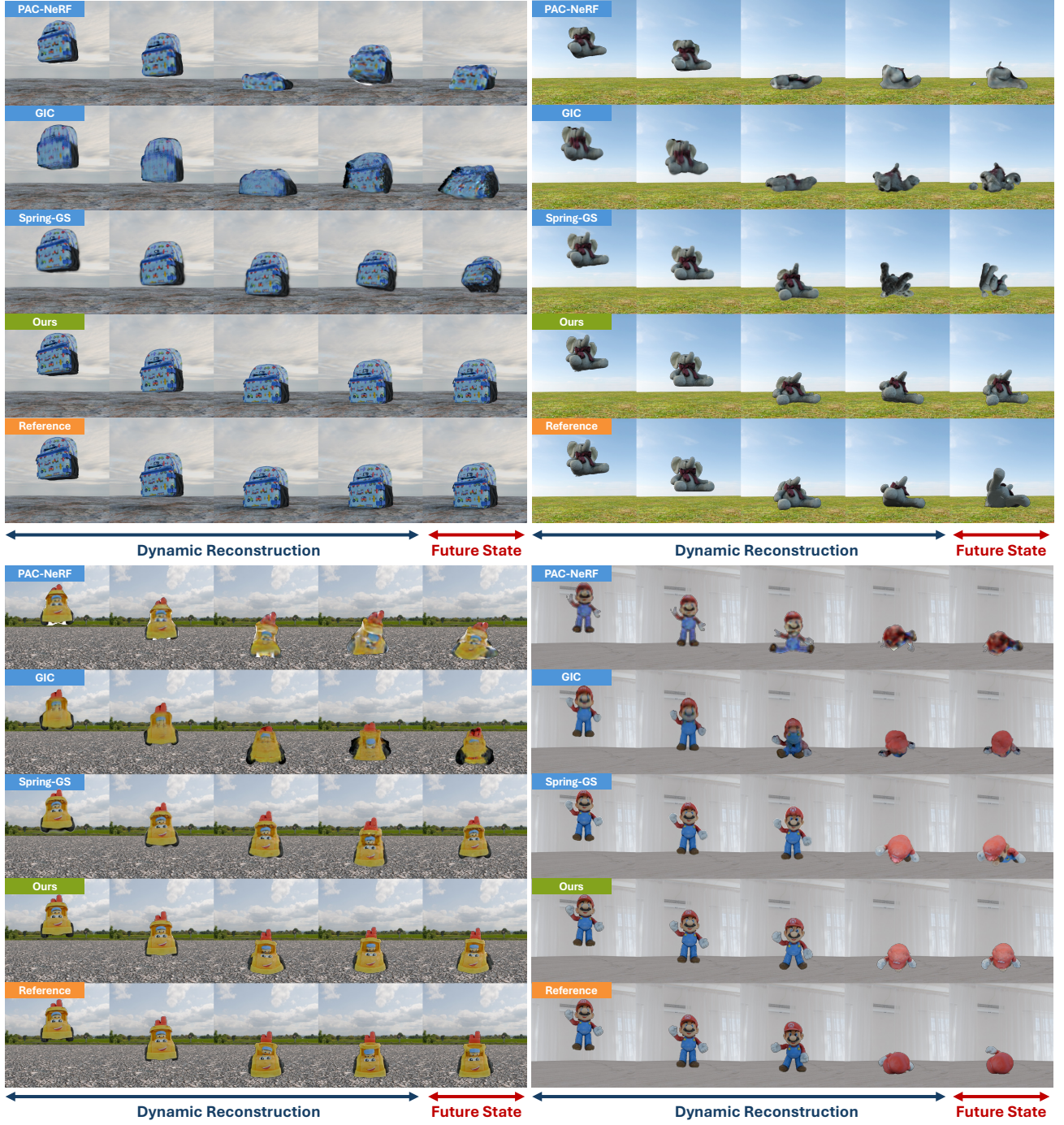


Figure 3. Comparison with the SOTA methods [5, 32, 64] on physics-aware dynamic reconstruction from multi-view videos (reference). Our method achieves the best quality in terms of textured shape and physical dynamics.

method Vid2Sim achieves a much higher quality of reconstruction for appearance and physics compared with all the SOTA methods across different objects. To be more specific, previous methods rely on optimizing dynamic NeRF or 3D Gaussians to model appearance, a process that is challenging in high-dimensional spaces and often results in blurred textures as shown in Fig. 3. In contrast, our pipeline enables explicit deformation guided by a deformation field

based on 3D Gaussians, preserving high-quality details optimized in the canonical space. Furthermore, baseline models are constrained to a differentiable simulator with a symplectic solver, which introduces oscillations and instability, compromising the realism of the simulations. Unlike these models, our implicit solver within the differentiable simulator provides a more accurate and efficient simulation.

Table 1. Quantitative Comparison with Previous Methods in Dynamic Reconstruction.

		backpack	bell	blocks	bus	cream	elephant	grandpa	leather	lion	mario	sofa	turtle	Mean
PSNR $\uparrow$	PAC-NeRF	19.37	25.00	23.36	20.72	23.24	22.27	21.63	20.85	22.66	21.01	22.49	22.19	22.06
	Spring-Gaus	17.42	20.49	22.78	20.06	23.58	21.30	21.64	18.29	21.95	20.23	21.89	21.23	20.91
	GIC	18.94	19.55	18.78	20.84	23.81	21.86	20.50	21.00	19.33	21.28	24.16	22.09	21.01
	Ours (full)	28.30	30.14	33.49	27.76	35.75	29.29	27.52	32.55	28.27	27.35	30.44	31.13	30.17
SSIM $\uparrow$	PAC-NeRF	0.887	0.956	0.940	0.908	0.893	0.922	0.939	0.932	0.936	0.921	0.926	0.923	0.924
	Spring-Gaus	0.867	0.941	0.941	0.903	0.912	0.919	0.948	0.917	0.937	0.920	0.921	0.920	0.920
	GIC	0.903	0.945	0.930	0.925	0.922	0.936	0.948	0.949	0.934	0.938	0.942	0.936	0.934
	Ours (full)	0.944	0.972	0.978	0.944	0.966	0.955	0.962	0.977	0.957	0.949	0.954	0.969	0.961
FoVDP $\uparrow$	PAC-NeRF	6.043	7.473	7.001	6.540	5.991	6.791	6.626	6.485	7.006	6.876	6.543	6.711	6.674
	Spring-Gaus	5.455	6.862	6.890	6.377	5.899	6.524	6.998	5.988	6.902	6.153	6.300	6.569	6.410
	GIC	6.130	6.230	6.062	6.552	5.889	6.907	6.855	6.737	6.331	6.985	7.069	6.782	6.544
	Ours (full)	8.341	8.288	8.943	7.948	9.181	8.307	7.830	9.007	7.866	7.771	8.049	8.820	8.363

6.3. Ablation Study

We conduct extensive ablation studies on our key designs. Tab. 2 summarize the quantitative results. Since only 4 views are used in LGM [53] in our Stage I, it is difficult to reconstruct the accurate appearance and geometry at inference time, resulting in compromised quantitative results (*Ours (Stage I only)*). Nevertheless, the predicted physical properties from Stage I are effective enough to produce high-quality simulations. This is validated by *Ours (Stage I+refine GS)*, where we solely refine the 3D Gaussians from LGM initialization without changing any physical properties. This demonstrates that appearance and geometry are critical for the overall dynamic reconstruction. *Ours (Stage I+fit GS)* is a similar ablation where the 3D Gaussians are trained from scratch, demonstrating a worse result than using LGM prediction as initialization. *Ours (full w/o fine-tune LBS)* shows a further improvement when adding the optimization of the E and ν , and our full model that unlocks the LBS reaches the best. Additionally, *Ours (full w/o Stage I Phys.)* shows purely optimization results with random physics initialization, for which we ran the experiments 3 times with random samples of $E \in [10^4, 10^6]$, $\nu \in [0.2, 0.5]$, same as the prediction range of our feed-forward predictor. This result suggests that a reliable initialization is crucial for achieving final convergence.

6.4. Future State Prediction

Like Spring-Gaus [64] and GIC [5], we also perform a test of future state prediction to evaluate how our model’s simulation aligns the observed videos in future frames. We report an average result across all the test cases on our synthetic dataset in Tab. 4, for which both our method and baselines predict 8 frames after reconstructing from 16 frames. The results show that our method keeps better accuracy than all the baselines.

6.5. Evaluation on the real-world dataset

We next evaluate our model on the real-world dataset. Obtaining accurate 3D Gaussian representations from sparse

viewpoints in our real-world dataset poses a significant challenge. To address this issue, we employ the registration network introduced by Spring-Gaus [64] to align the poses of the 3D Gaussians estimated by LGM [53] in Stage I with the real-world camera poses. Our approach then leverages these registered static 3D Gaussians, in the manner of Spring-Gaus, to facilitate reconstruction and simulation. We compare our method with Spring-Gaus for both dynamic reconstruction and future state prediction, as shown in Fig. 4 and Tab. 3. Our approach demonstrates enhanced capability in modeling real-world objects, particularly in future state prediction.

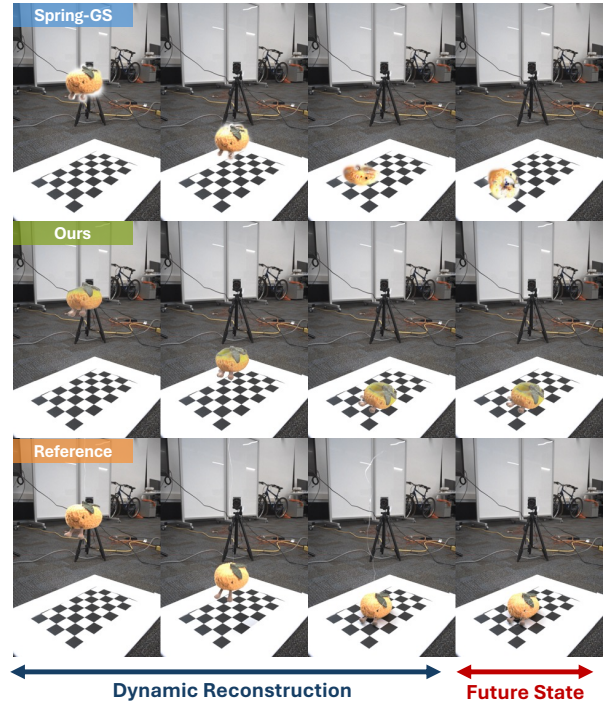


Figure 4. Visualization of dynamic reconstruction results of Vid2Sim on the real-world object.

Table 2. Ablation of dynamic reconstruction.

		backpack	bell	blocks	bus	cream	elephant	grandpa	leather	lion	mario	sofa	turtle	Mean
PSNR $\uparrow$	Ours (Stage I only)	19.54	21.34	20.50	19.62	18.08	19.89	19.46	15.53	21.36	19.07	20.48	22.36	19.77
	Ours (Stage I+fit GS)	26.87	26.59	32.57	26.53	34.82	26.99	24.45	31.21	27.62	24.83	30.01	30.33	28.57
	Ours (Stage I+refine GS)	26.52	27.61	32.49	26.63	34.54	27.05	24.42	31.32	27.54	25.73	30.11	30.37	28.69
	Ours (full w/o fine-tune LBS)	27.37	27.97	32.98	27.01	35.31	27.93	27.34	31.06	28.27	26.60	29.90	31.00	29.40
	Ours (full w/o Stage I Phys.)	27.63	30.79	33.02	28.35	35.75	28.59	26.72	32.71	27.06	26.58	30.09	31.16	29.87
	Ours (full)	28.30	30.14	33.49	27.76	35.75	29.29	27.52	32.55	28.27	27.35	30.44	31.13	30.17
SSIM $\uparrow$	Ours (Stage I only)	0.883	0.950	0.924	0.897	0.909	0.904	0.938	0.895	0.931	0.917	0.913	0.921	0.915
	Ours (Stage I+fit GS)	0.931	0.959	0.973	0.937	0.955	0.942	0.944	0.973	0.954	0.936	0.952	0.963	0.952
	Ours (Stage I+refine GS)	0.929	0.960	0.974	0.937	0.953	0.943	0.945	0.974	0.954	0.942	0.952	0.962	0.952
	Ours (full w/o fine-tune LBS)	0.934	0.965	0.975	0.942	0.960	0.947	0.961	0.974	0.957	0.946	0.951	0.968	0.957
	Ours (full w/o Stage I Phys.)	0.938	0.973	0.975	0.948	0.962	0.952	0.961	0.977	0.952	0.951	0.953	0.969	0.959
	Ours (full)	0.944	0.972	0.978	0.944	0.966	0.955	0.962	0.977	0.957	0.949	0.954	0.969	0.961
FoVDP $\uparrow$	Ours (Stage I only)	6.616	6.175	6.481	6.545	4.450	6.188	6.064	5.661	6.834	5.912	6.062	7.330	6.193
	Ours (Stage I+fit GS)	7.778	6.884	8.792	7.622	9.117	7.468	6.246	8.949	7.681	6.325	8.034	8.628	7.794
	Ours (Stage I+refine GS)	7.664	7.000	8.802	7.657	9.105	7.473	6.249	8.968	7.675	6.825	8.060	8.632	7.843
	Ours (full w/o fine-tune LBS)	8.080	7.680	8.873	7.770	9.199	7.916	7.676	8.929	7.866	7.312	7.994	8.729	8.169
	Ours (full w/o Stage I Phys.)	7.996	8.439	8.861	8.107	9.158	8.064	7.433	9.011	7.438	7.218	7.942	8.782	8.204
	Ours (full)	8.341	8.288	8.943	7.948	9.181	8.307	7.830	9.007	7.866	7.771	8.049	8.820	8.363

Table 3. Evaluation on the real-world object.

		orange	bird	cup	Mean
PSNR $\uparrow$	Spring-Gaus	28.69	25.08	24.39	26.05
	Ours (full)	30.11	26.02	25.24	27.12
SSIM $\uparrow$	Spring-Gaus	0.987	0.980	0.979	0.982
	Ours (full)	0.987	0.981	0.980	0.983
FoVDP $\uparrow$	Spring-Gaus	8.379	7.494	7.447	7.773
	Ours (full)	8.623	7.554	7.447	7.875

Table 4. Comparison on future state prediction.

	PSNR $\uparrow$	SSIM $\uparrow$	FoVDP $\uparrow$
PAC-NeRF	20.11	0.913	5.948
Spring-Gaus	18.32	0.905	5.443
GIC	19.20	0.916	5.702
Ours (full)	25.07	0.945	7.770

6.6. Comparison of Efficiency

Though using an implicit Euler solver with Newton’s method and line search, our method is still much more efficient regarding differentiable simulation. This is because of four reasons: **(1)** The implicit Euler solver requires fewer time steps; **(2)** The simulation and optimization is operated on a reduced dimension; **(3)** We design a neural Jacobian for faster precomputation and **(4)** Our strategy of using partial frames.

We compare the computation time among our method and baselines for one optimization iteration that contains one forward and backward pass (consider using all 12 views on *backpack* case). We also report the whole training time for all the methods with the default settings. Our results in Tab. 5 show that our method is even faster than the efficient Spring-Gaus method, and our proposed neural Jacobian saves more time when using more cubature points and handles in simulation. All the performances are tested on one NVIDIA-RTX-4090 GPU.

Table 5. Comparison with existing methods on runtime performance. The results in $(\cdot)$ is the case that uses 40 handles and 2000 cubature points for more accurate simulation.

	Per Iteration Time	Total Training Time
GIC	37.33s	120min
PAC-NeRF	29.04s	84min
Spring-GS	8.08s	54min
Ours (w/o J_θ)	3.22s (13.11s)	26 min
Ours (full)	1.44s (2.11s)	15 min

7. Conclusion

In this paper, we present Vid2Sim, a novel and robust framework for high-fidelity and generalizable reconstruction of textured shapes and physical properties directly from video data. Our approach overcomes key limitations in existing methods by incorporating a feed-forward model that efficiently provides generalizable initial estimation, alongside a differentiable, reduced-order simulator utilizing Linear Blend Skinning for fast and precise optimization of appearance, geometry, and physical properties. After the reconstruction, Vid2Sim enables high-quality, mesh-free simulation with high efficiency. Comprehensive experiments demonstrate that Vid2Sim achieves state-of-the-art performance in both accuracy and efficiency, representing a significant advancement in video-based system identification.

8. Limitation and Future Work

Our approach is limited in reconstructing and simulating complex materials, e.g. fluid, since we use a reduced-order simulation method. Future works include further enhancing the ability to express more complex material and motions. Another direction is to merge the two branches of our Stage I and train a unified feed-forward network to predict 3D Gaussians together with point-wise physical properties.

References

- [1] Noam Aigerman, Kunal Gupta, Vladimir G Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. Neural jacobian fields: Learning intrinsic mappings of arbitrary meshes. *arXiv preprint arXiv:2205.02904*, 2022. 1
- [2] Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, et al. Lumiere: A space-time diffusion model for video generation. *arXiv preprint arXiv:2401.12945*, 2024. 3
- [3] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.
- [4] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. 2024. URL <https://openai.com/research/video-generation-models-as-world-simulators>, 3, 2024. 3
- [5] Junhao Cai, Yuji Yang, Weihao Yuan, Yisheng He, Zilong Dong, Liefeng Bo, Hui Cheng, and Qifeng Chen. Gaussian-informed continuum for physical property identification and simulation. *arXiv preprint arXiv:2406.14927*, 2024. 2, 5, 6, 7, 3
- [6] Pao-Hsiung Chiu, Jian Cheng Wong, Chinchun Ooi, My Ha Dao, and Yew-Soon Ong. Can-pinn: A fast physics-informed neural network based on coupled-automatic-numerical differentiation method. *Computer Methods in Applied Mechanics and Engineering*, 395:114909, 2022. 2
- [7] Djork-Arné Clevert. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015. 1
- [8] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics-informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022. 2
- [9] Barbara Cutler, Julie Dorsey, Leonard McMillan, Matthias Müller, and Robert Jagnow. A procedural approach to authoring solid models. *ACM Transactions on Graphics (TOG)*, 21(3):302–311, 2002. 2
- [10] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023. 5
- [11] M Desbrun. Smoothed particles: A new paradigm for animating highly deformable bodies. *Computer Animation and Simulation/Springer Vienna*, 1996. 2
- [12] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022. 1, 5
- [13] Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. Diffpd: Differentiable projective dynamics. *ACM Transactions on Graphics (TOG)*, 41(2):1–21, 2021. 2
- [14] Yutao Feng, Yintong Shang, Xiang Feng, Lei Lan, Shandian Zhe, Tianjia Shao, Hongzhi Wu, Kun Zhou, Hao Su, Chenfanfu Jiang, et al. Elastogen: 4d generative elastodynamics. *arXiv preprint arXiv:2405.15056*, 2024. 3
- [15] Yutao Feng, Yintong Shang, Xuan Li, Tianjia Shao, Chenfanfu Jiang, and Yin Yang. Pie-nerf: Physics-based interactive elastodynamics with nerf. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4450–4461, 2024. 3
- [16] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 2
- [17] Moritz Geilinger, David Hahn, Jonas Zehnder, Moritz Bächer, Bernhard Thomaszewski, and Stelian Coros. Add: Analytically differentiable dynamics for multi-body systems with frictional contact. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020. 2
- [18] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016. 4
- [19] Eric Heiden, Miles Macklin, Yashraj Narang, Dieter Fox, Animesh Garg, and Fabio Ramos. Disect: A differentiable simulation engine for autonomous robotic cutting. *arXiv preprint arXiv:2105.12244*, 2021. 2
- [20] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 1
- [21] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023. 2
- [22] Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. 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, 2018. 2
- [23] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. DiffTaichi: Differentiable programming for physical simulation. *arXiv preprint arXiv:1910.00935*, 2019. 2
- [24] Krishna Murthy Jatavallabhula, Miles Macklin, Florian Golemo, Vikram Voleti, Linda Petrini, Martin Weiss, Breandan Considine, Jérôme Parent-Lévesque, Kevin Xie, Kenny Erleben, et al. gradsim: Differentiable simulation for system identification and visuomotor control. *arXiv preprint arXiv:2104.02646*, 2021. 2
- [25] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *Acm siggraph 2016 courses*, pages 1–52. 2016. 2
- [26] Ying Jiang, Chang Yu, Tianyi Xie, Xuan Li, Yutao Feng, Huamin Wang, Minchen Li, Henry Lau, Feng Gao, Yin Yang, et al. Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–1, 2024. 3

- [27] Takuhiro Kaneko. Improving physics-augmented continuum neural radiance field-based geometry-agnostic system identification with lagrangian particle optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5480, 2024. 2, 5
- [28] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2, 3, 4, 5
- [29] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015. 5
- [30] Tassilo Kugelstadt, Jan Bender, José Antonio Fernández-Fernández, Stefan Rhys Jeske, Fabian Löschner, and Andreas Longva. Fast corotated elastic sph solids with implicit zero-energy mode control. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 4(3):1–21, 2021. 2
- [31] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph.*, 36(6):194–1, 2017. 2
- [32] Xuan Li, Yi-Ling Qiao, Peter Yichen Chen, Krishna Murthy Jatavallabhula, Ming Lin, Chenfanfu Jiang, and Chuang Gan. Pac-nerf: Physics augmented continuum neural radiance fields for geometry-agnostic system identification. *arXiv preprint arXiv:2303.05512*, 2023. 2, 5, 6, 3
- [33] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian Curless, Steve Seitz, and Ira Kemelmacher-Shlizerman. Real-time high-resolution background matting. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 5
- [34] Fangfu Liu, Hanyang Wang, Shunyu Yao, Shengjun Zhang, Jie Zhou, and Yueqi Duan. Physics3d: Learning physical properties of 3d gaussians via video diffusion. *arXiv preprint arXiv:2406.04338*, 2024. 3
- [35] Shaowei Liu, Zhongzheng Ren, Saurabh Gupta, and Shenglong Wang. Physgen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision*, pages 360–378. Springer, 2025. 3
- [36] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 851–866. 2023. 2
- [37] Guanxing Lu, Shiyi Zhang, Ziwei Wang, Changliu Liu, Jiwen Lu, and Yansong Tang. Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation. In *European Conference on Computer Vision*, pages 349–366. Springer, 2025. 3
- [38] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023. 2
- [39] Pingchuan Ma, Tao Du, Joshua B Tenenbaum, Wojciech Matusik, and Chuang Gan. Risp: Rendering-invariant state predictor with differentiable simulation and rendering for cross-domain parameter estimation. *arXiv preprint arXiv:2205.05678*, 2022. 2
- [40] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2
- [41] Vismay Modi, Nicholas Sharp, Or Perel, Shinjiro Sueda, and David IW Levin. Simplicits: Mesh-free, geometry-agnostic elastic simulation. *ACM Transactions on Graphics (TOG)*, 43(4):1–11, 2024. 2, 3, 4, 5, 1
- [42] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2
- [43] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. 2
- [44] Andreas Peer, Christoph Gissler, Stefan Band, and Matthias Teschner. An implicit sph formulation for incompressible linearly elastic solids. In *Computer Graphics Forum*, pages 135–148. Wiley Online Library, 2018. 2
- [45] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2
- [46] Yiling Qiao, Junbang Liang, Vladlen Koltun, and Ming Lin. Differentiable simulation of soft multi-body systems. *Advances in Neural Information Processing Systems*, 34: 17123–17135, 2021. 2
- [47] Yi-Ling Qiao, Alexander Gao, and Ming Lin. Neu-physics: Editable neural geometry and physics from monocular videos. *Advances in Neural Information Processing Systems*, 35:12841–12854, 2022. 2
- [48] Jiawei Ren, Kevin Xie, Ashkan Mirzaei, Hanxue Liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung Wook Kim, et al. L4gm: Large 4d gaussian reconstruction model. *arXiv preprint arXiv:2406.10324*, 2024. 2
- [49] Junior Rojas, Eftychios Sifakis, and Ladislav Kavan. Differentiable implicit soft-body physics. *arXiv preprint arXiv:2102.05791*, 2021. 2
- [50] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 3
- [51] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *arXiv preprint arXiv:2201.02610*, 2022. 2
- [52] Soshi Shimada, Vladislav Golyanik, Patrick Pérez, and Christian Theobalt. Decaf: Monocular deformation capture for face and hand interactions. *ACM Transactions on Graphics (ToG)*, 42(6):1–16, 2023. 2
- [53] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European*

Conference on Computer Vision, pages 1–18. Springer, 2025. [2](#), [4](#), [7](#)

- [54] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022. [2](#), [4](#), [1](#)
- [55] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021. [2](#)
- [56] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024. [2](#)
- [57] Qingxuan Wu, Zhiyang Dou#, Sirui Xu, Soshi Shimada, Chen Wang, Zhengming Yu, Yuan Liu, Cheng Lin, Zeyu Cao, Taku Komura, et al. Dice: End-to-end deformation capture of hand-face interactions from a single image. *arXiv preprint arXiv:2406.17988*, 2024. [2](#)
- [58] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9421–9431, 2021. [2](#)
- [59] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4389–4398, 2024. [3](#)
- [60] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642*, 2023. [2](#)
- [61] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20331–20341, 2024. [2](#)
- [62] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *European Conference on Computer Vision*, pages 1–19. Springer, 2025. [2](#)
- [63] Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T Freeman. Physics-based interaction with 3d objects via video generation. [3](#)
- [64] Licheng Zhong, Hong-Xing Yu, Jiajun Wu, and Yunzhu Li. Reconstruction and simulation of elastic objects with spring-mass 3d gaussians. In *European Conference on Computer Vision*, pages 407–423. Springer, 2025. [2](#), [5](#), [6](#), [7](#), [3](#)

Vid2Sim: Generalizable, Video-based Reconstruction of Appearance, Geometry and Physics for Mesh-free Simulation

Supplementary Material

This supplementary material covers the following sections: More Implementation Details (Sec. 9); More Results on Dynamic Reconstruction (Sec. 10); Generalization Capability (Sec. 11). Please refer to our supplementary video for a more comprehensive overview,

9. More Implementation Details

9.1. Large Video Vision Transformer

The pipeline of our Large Video Vision Transformer is shown in Fig. 5. In our framework, we fine-tune the backbone network, VideoMAE [54], which is pre-trained on 16-frame videos at a resolution of 224×224 . To adapt it to a higher resolution (448×448 in our setting), we interpolate the pre-trained positional embeddings to align with the updated number of input tokens. The output tokens are averaged across all the patches before being sent into the regression MLPs. The regression MLPs for predicting E and ν are identical and with widths of [768, 512, 256, 128, 1]. The regression MLP for predicting $\hat{\theta}_{lbs}$ has widths of [768, 650, 650, 650, 650] where the width of the last layer is equal to the number of trainable parameters for a linear layer. We demonstrate in Tab. 6 that it is better to predict only the last layer of $\hat{\theta}_{lbs}$ and keep the first 7 layers fixed for consistency with the optimization stage (Stage II) than to predict full layers in our task. This is because Hypernetwork predicts $\sim 30k$ network parameters for full-layer LBS, making training much more difficult than our one-layer prediction design. We use GELU [20] as the activation function for all regression MLPs.

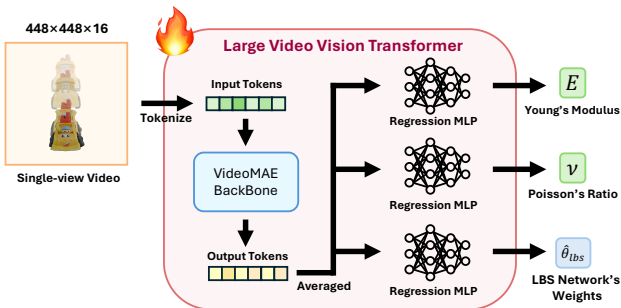


Figure 5. Detailed pipeline of the large video vision transformer.

9.2. LBS and Jacobian Network

The implementation of the LBS and Jacobian network is visualized in Fig. 6. Specifically, the LBS network com-

	PSNR $\uparrow$	SSIM $\uparrow$	FoVDP $\uparrow$
One-layer prediction (Ours)	28.83$\pm$3.06	0.954$\pm$0.014	7.907$\pm$0.859
Full-layer prediction	28.53 $\pm$ 3.21	0.953 $\pm$ 0.015	7.782 $\pm$ 0.886
GT (data-free train)	29.40 $\pm$ 2.59	0.957 $\pm$ 0.012	8.169 $\pm$ 0.579

Table 6. Quantitative results in Dynamic Reconstruction across different LBS prediction settings using the same optimized geometry and physical parameters for fairness.

	Forward Time	Backward Time	$\frac{1}{N} \sum_{i=1}^N \mathbf{J}_{\theta}^i - \mathbf{J}_{gt}^i _2^2$
4 blocks w/o PE	0.00081s	0.00158s	9.22×10^{-7}
2 blocks	0.00052s	0.00099s	5.12×10^{-7}
4 blocks (Ours)	0.00086s	0.00166s	4.04×10^{-8}
GT Jacobian	6.51398s	0.00014s	-

Table 7. Speed (time per iteration) and average accuracy across different Neural Jacobian models. All the values are tested under the setting of 2000 points & 10 handles on one NVIDIA-RTX-4090 GPU.

prises 8 linear layers with a constant layer width of 64 and ELU [7] activation function. We observe that the neural Jacobian predominantly focuses on learning to predict high-frequency features, rather than the low-frequency signals typically modeled by the LBS prediction network. This insight motivates us to adopt a design for predicting the Jacobian that differs from the standard MLP architecture used in the LBS network, where we incorporate positional encoding into the input to capture the high-frequency features effectively. The input positions are embedded into a 512-dimensional space using positional encoding. The model comprises four residual blocks, each containing two linear layers. The first two residual blocks have a layer width of 512, while the last two have a layer width of 1024. The output is projected with a linear layer from the features. We use the GELU [20] activation function in the Jacobian network. We found that 4 blocks with positional encoding are sufficient to predict the Jacobian that is accurate enough for simulation, so we didn't scale up it further to save data-free training time. We report the speed-accuracy trade-off for different Neural Jacobian models in Tab. 7. Note that the time cost of Neural Jacobian is only meaningful in its data-free training, which contains 10k iterations. It can be ignored in the joint optimization of Stage II.

The LBS and Jacobian networks are first trained in a data-free manner, supervised by randomly sampled $\mathbf{X}$ and $\mathbf{z}$, inspired by [1, 41]. The LBS network is optimized by minimizing an elastic loss and orthogonal regulariza-

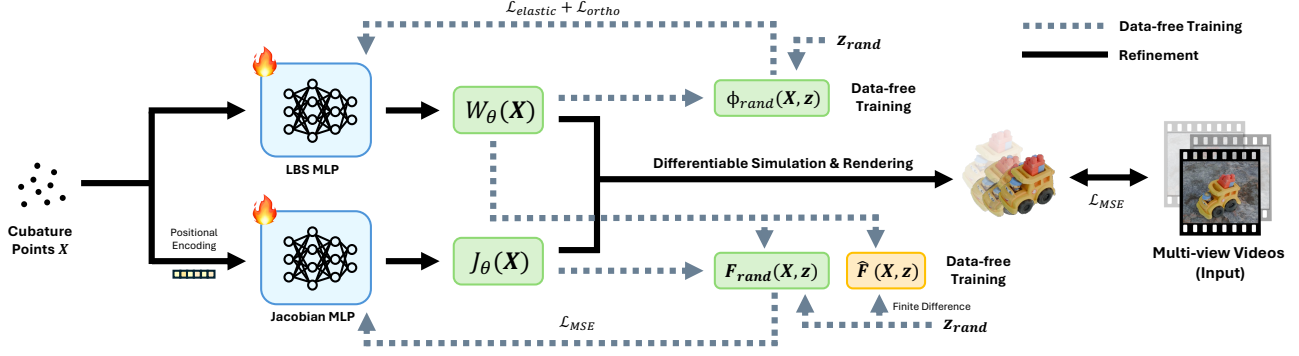


Figure 6. Network structure of LBS network and Jacobian network.

		backpack	bell	blocks	bus	cream	elephant	grandpa	leather	lion	mario	sofa	turtle	Mean
PSNR $\uparrow$	PAC-NeRF	18.03	21.74	21.67	19.05	19.81	20.68	20.20	19.48	20.67	17.06	19.60	22.09	20.01
	Spring-Gaus	17.35	21.04	22.93	19.77	24.80	20.97	21.76	19.28	21.32	20.51	21.55	22.42	21.14
	GIC	18.22	19.33	18.26	20.23	23.94	21.50	20.45	20.76	18.53	21.17	23.38	22.60	20.70
	Ours (full)	26.59	27.26	31.29	25.64	33.85	27.95	24.09	31.11	25.89	26.54	27.82	30.81	28.24
SSIM $\uparrow$	PAC-NeRF	0.882	0.956	0.935	0.900	0.900	0.924	0.941	0.929	0.931	0.932	0.918	0.933	0.924
	Spring-Gaus	0.866	0.945	0.936	0.899	0.918	0.921	0.950	0.924	0.931	0.922	0.913	0.931	0.921
	GIC	0.879	0.938	0.918	0.903	0.901	0.924	0.943	0.937	0.917	0.930	0.920	0.934	0.920
	Ours (full)	0.940	0.964	0.971	0.935	0.951	0.955	0.953	0.975	0.949	0.951	0.942	0.973	0.955
FoV/DP $\uparrow$	PAC-NeRF	5.873	6.514	6.803	6.330	4.695	6.681	6.516	6.437	6.581	3.989	5.779	6.943	6.095
	Spring-Gaus	5.535	6.945	6.920	6.344	5.970	6.482	7.045	6.260	6.766	6.201	6.296	7.011	6.481
	GIC	5.959	6.129	5.947	6.394	5.805	6.921	6.930	6.696	6.023	7.029	6.905	6.939	6.473
	Ours (full)	8.331	7.640	8.921	7.898	8.900	8.398	7.020	9.053	7.659	8.015	7.912	9.000	8.229

Table 8. Quantitative comparison with previous methods on dynamic reconstruction (novel views).

		backpack	bell	blocks	bus	cream	elephant	grandpa	leather	lion	mario	sofa	turtle	Mean
$\log(E)$	PAC-NeRF	3.28	1.08	4.02	3.30	3.22	3.05	2.99	1.20	2.34	3.37	0.20	1.94	2.50
	GIC	1.16	2.87	2.12	1.93	2.13	1.53	0.42	3.45	2.85	1.82	0.65	3.18	2.01
	Ours (full)	0.69	0.15	0.54	0.26	0.95	0.18	1.07	0.73	0.48	0.50	0.18	0.44	0.51
ν	PAC-NeRF	0.21	0.23	0.33	0.16	0.12	0.06	0.36	0.26	0.14	0.33	0.30	0.01	0.21
	GIC	0.11	0.24	0.29	0.18	0.08	0.24	0.26	0.02	0.14	0.22	0.01	0.08	0.16
	Ours (full)	0.10	0.10	0.07	0.06	0.11	0.05	0.02	0.06	0.05	0.06	0.08	0.02	0.06

Table 9. Mean Absolute Error (MAE) among baselines and our method on physical property predictions.

tion loss. The Jacobian network is optimized by minimizing the L2 loss between the predicted deformation gradient $\mathbf{F}(\mathbf{X}, \mathbf{z})$ and the estimated $\hat{\mathbf{F}}(\mathbf{X}, \mathbf{z})$ from the finite difference.

The two networks are then jointly trained along with physical parameters according to the observed multi-view videos, where we only minimize the L2 loss between simulated animations and the observed multi-view videos, as described in Sec. 4.3 in the main paper.

9.3. Boundary Condition Implementation

We follow [41] to implement boundary conditions with incremental potential contact for handling collision, the con-

straints are formulated with barrier functions that provide extra potential energy. For example, our floor barrier in the dynamic reconstruction and the future state prediction task uses $E_f = 10^5 \times \sum_{i=1}^N [\max(0, h_f - h_i)]^2$ as potential energy, where E_f is part of the external energy (See Eq. 2 in the main paper). Barrier functions can be very flexible in our method, and we provide more examples in Sec. 11.2.

10. More Results on Dynamic Reconstruction

In Sec. 10.1, we provide a comprehensive investigation by showcasing additional qualitative results of dynamic reconstruction and future state prediction across baselines, our

Stage I model, and our full model. In Sec. 10.2, we evaluate the performance of dynamic reconstruction on novel views. Additionally, we evaluate the prediction of physical properties E and ν in Sec. 10.3.

10.1. More Qualitative Results on Dynamic Reconstruction and Future States Prediction

As illustrated in Fig. 7 and Fig. 8, our model demonstrates remarkable physics-aware dynamic reconstruction quality compared to existing methods [5, 32, 64] that suffer from reconstructing blurry textures and incorrect dynamics due to the use of dynamic representations and symplectic solver. This is further evidenced by real-world test cases presented in Fig. 9, where the reconstruction results of the SOTA method Spring-Gaus [64] collapses when hitting the ground plane, while ours successfully capture the physical dynamics and produce higher realistic results.

10.2. Evaluation on Novel View Synthesis

We further evaluate the performance of our method on novel view synthesis by randomly sampling 6 novel views for each synthetic test case and evaluate the dynamic reconstruction performance among our method and baselines. We show qualitative results in Fig. 10 and quantitative results in Tab. 8, where our method consistently outperforms other models.

10.3. Evaluation on Physical Parameters Estimation

Next, we evaluate the Mean Absolute Error (MAE) on the estimated $\log(E)$ and ν in the Neo-Hookean elastic model used by PAC-NeRF [32], GIC [5] and our method. As shown in Tab. 9, our method outperforms all the other approaches in most cases while showing its competitive performance on the remaining samples, which validates the effectiveness of our model on physical property estimation.

11. Generalization Capability

We provide more simulation results on changed materials in Sec. 11.1 and provide additional simulation results on different boundary conditions in Sec. 11.2.

11.1. Generalized to Different Materials

Although our method mainly focuses on reconstructing elastic objects in this paper, our framework can be generalized to materials characterized by various constitutive models. Here, we show simulation results regarding three different materials: Elasticity, Plasticine, and Sand following [5, 32]. The qualitative results are shown in Fig. 11, where different materials are simulated precisely as our method is combined with different constitutive models effectively.

In order to compute the potential energy $E_{potential}$ for simulation, we derive the corresponding energy density function $\Psi(\mathbf{F})$ for each constitutive model below.

Elasticity. The energy density function can be formulated as

$$\Psi(\mathbf{F}) = \frac{\mu}{2} [tr(\mathbf{F}^\top \mathbf{F}) - d] - \mu \ln(J) + \frac{\lambda}{2} \ln^2(J) \quad (7)$$

where $d = 3$ is the space dimension, $\mathbf{F}$ is the deformation gradient and J is the determinant of $\mathbf{F}$, μ and λ are Lamé parameters related to Young’s modulus E and Poisson’s ratio ν :

$$\mu = \frac{E}{2(1+\nu)} \quad \lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \quad (8)$$

Plasticine Plasticine material is modeled with a combination of Saint Venant-Kirchhoff Model (StVK) and von Mises return mapping function. The energy density function of StVK can be formulated as

$$\Psi(\mathbf{F}) = \mu [tr(\mathbf{G}^2)] + \frac{\lambda}{2} [tr^2(\mathbf{G})] \quad (9)$$

where $\mathbf{G} = \frac{1}{2}(\mathbf{F}^\top \mathbf{F} - d)$ is the Green strain. The von-Mises return mapping function projects the deformation gradient back onto the boundary of the elastic region according to the von-Mises yielding condition. The mapping function can be formulated as

$$\mathcal{Z}(\mathbf{F}) = \begin{cases} \mathbf{F} & \delta\gamma \leq 0 \\ \mathbf{U} \exp(\epsilon - \delta\gamma \frac{\hat{\epsilon}}{\|\hat{\epsilon}\|}) \mathbf{V}^\top & \text{otherwise} \end{cases} \quad (10)$$

where $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^\top$ is the singular value decomposition (SVD) of $\mathbf{F}$, $\epsilon = \log(\Sigma)$ is the Hencky strain, $\hat{\epsilon} = \epsilon - \bar{\epsilon}$ is the normalized Hencky strain and $\delta\gamma = \|\hat{\epsilon}\| - \frac{\tau_Y}{2\mu}$ is von-Mises yielding condition with the yield stress τ_Y as a physical parameter.

Sand Similar to the Plasticine material, we also use StVK as the constitutive model and apply its energy density function to the Sand material. The difference is that we use Drucker-Prager yield criteria instead of von-Mises yield criteria. The mapping function can be formulated as

$$\mathcal{Z}(\mathbf{F}) = \begin{cases} \mathbf{U}\mathbf{V}^\top & tr(\epsilon) > 0 \\ \mathbf{F} & \delta\gamma \leq 0, tr(\epsilon) \leq 0 \\ \mathbf{U} \exp(\epsilon - \delta\gamma \frac{\hat{\epsilon}}{\|\hat{\epsilon}\|}) \mathbf{V}^\top & \text{otherwise} \end{cases} \quad (11)$$

where $\delta\gamma = \|\hat{\epsilon}\|_F + \alpha \frac{(d\lambda+2\mu)tr(\epsilon)}{2\mu}$ is the yield stress, $\alpha = \sqrt{\frac{2}{3} \frac{2\sin\theta_f}{3-\sin\theta_f}}$ and θ_f is the friction angle.

11.2. Generalized to Complex Boundary Conditions

In this section, we demonstrate that the reconstruction results of our method, Vid2Sim, integrate seamlessly into the simulation of various animations under complex boundary conditions. Two examples are presented in Fig. 12, highlighting Vid2Sim’s ability to generate high-quality animations across diverse boundary scenarios.

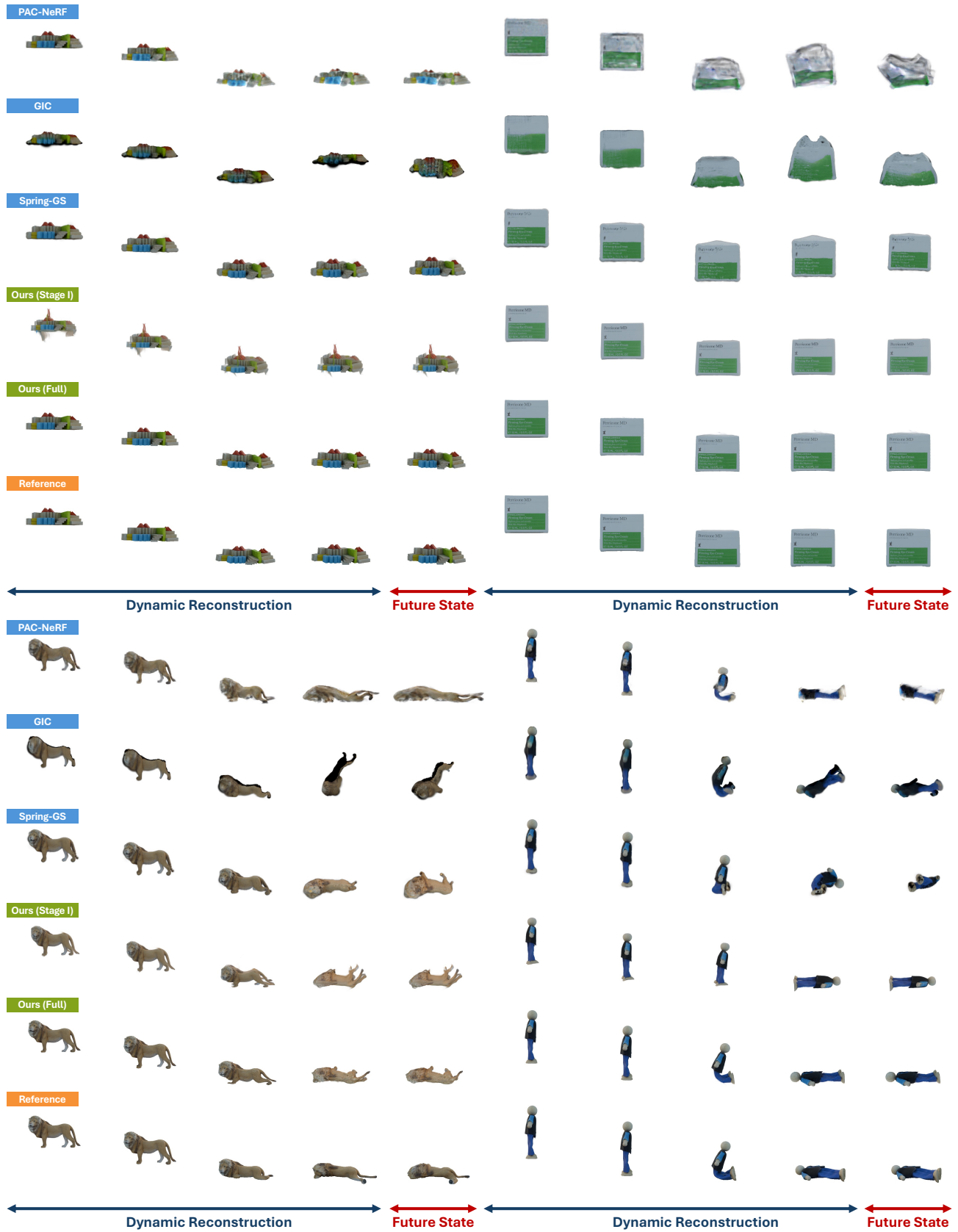


Figure 7. More dynamic reconstruction results from the input videos.

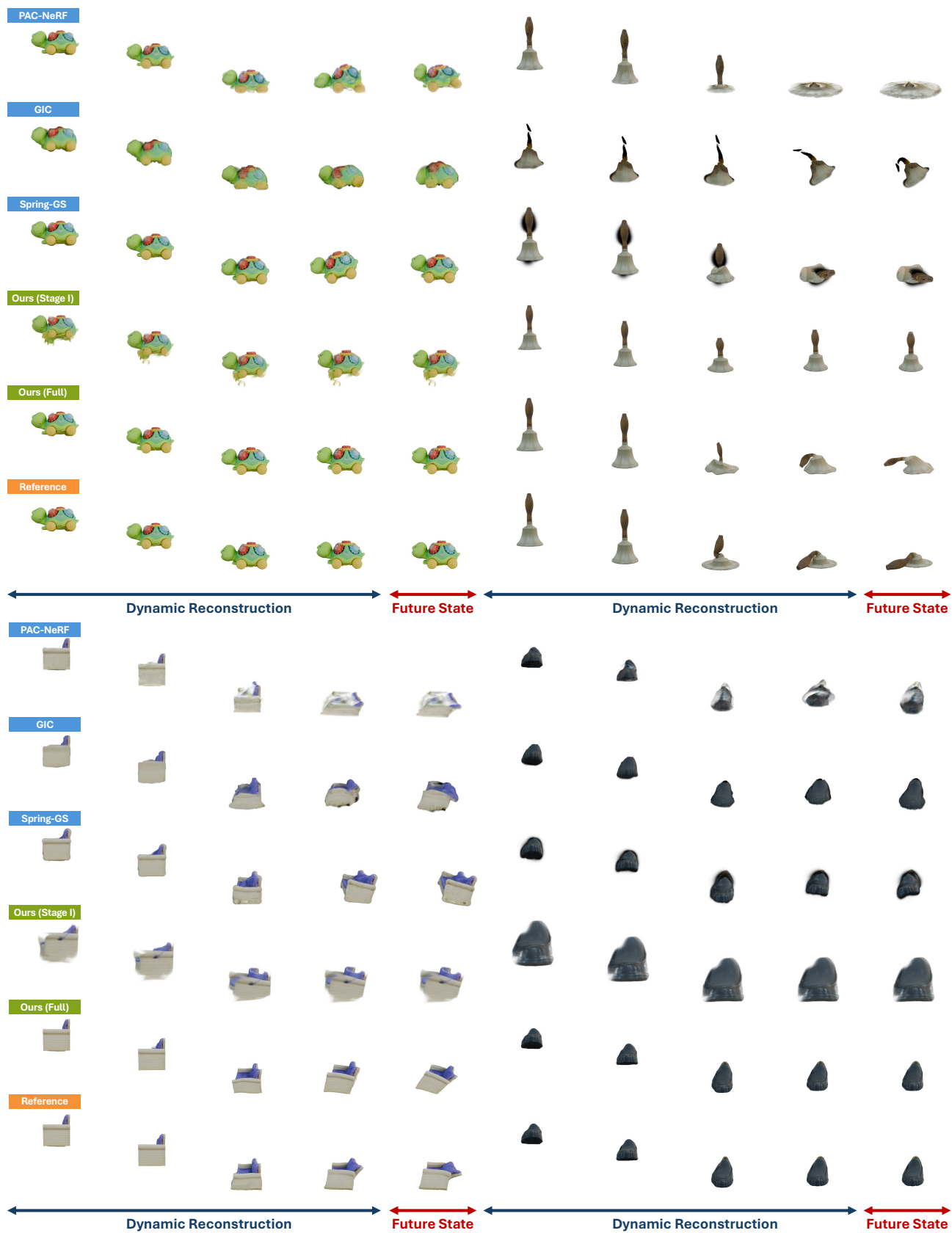


Figure 8. More dynamic reconstruction results from the input videos.

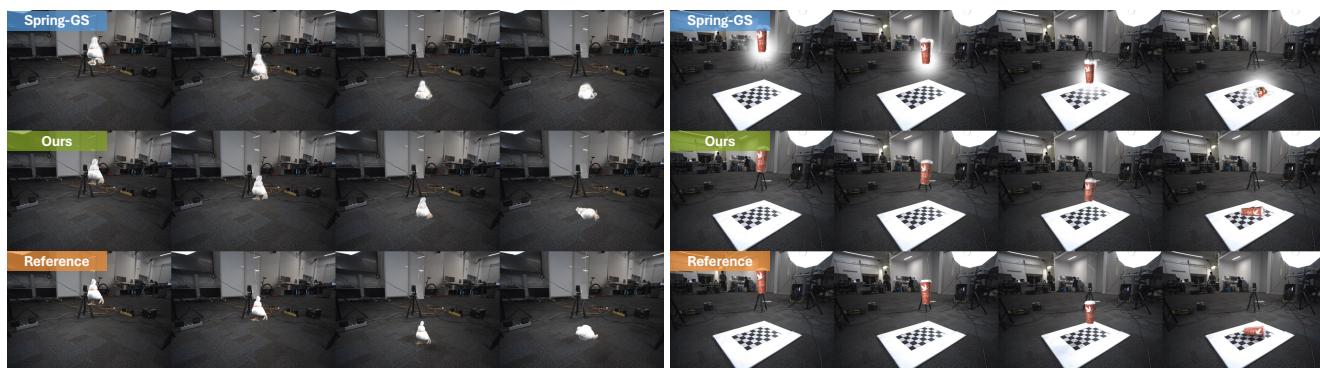


Figure 9. More dynamic reconstruction results from real-world input videos.



Figure 10. Novel view synthesis of the dynamic reconstruction results.

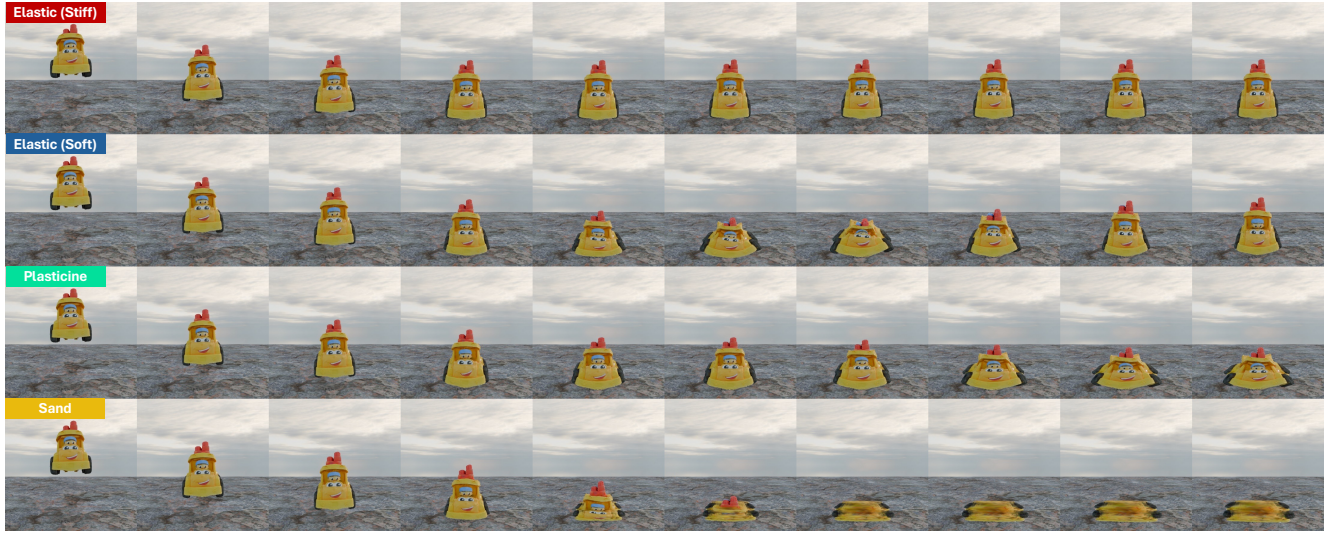
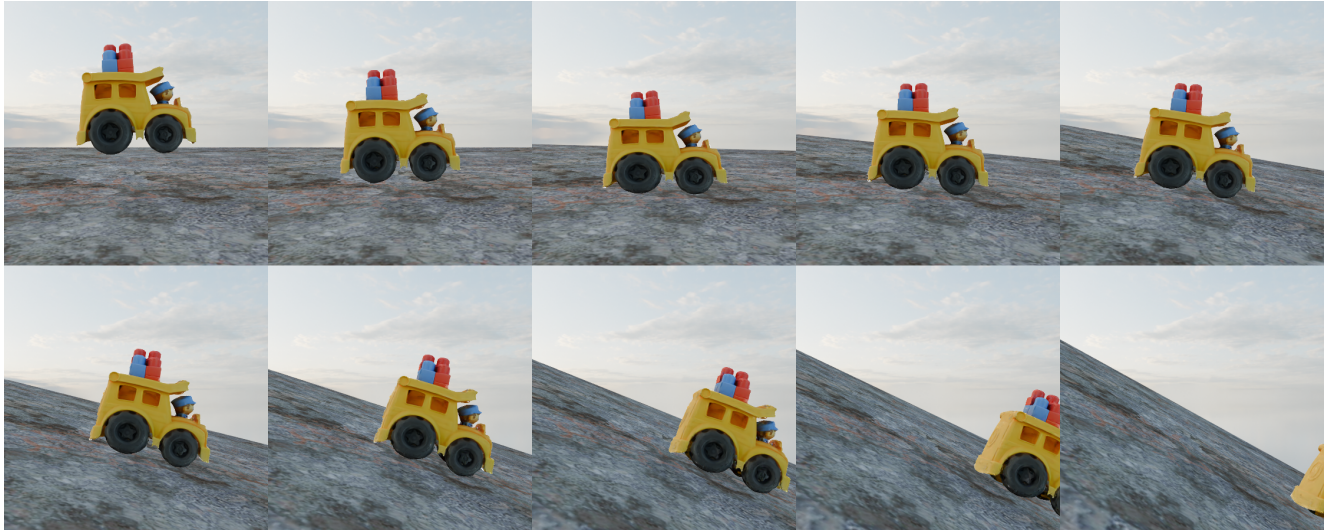
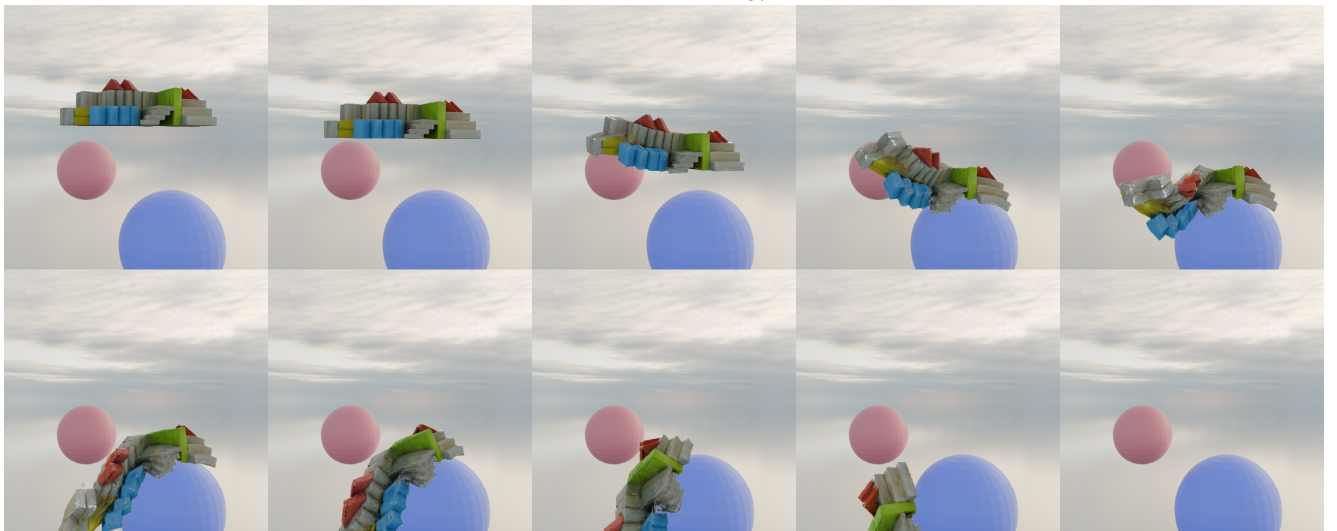


Figure 11. Simulation with different materials. We use $E = 10^7$, $\nu = 0.49$ for the stiff elastic and $E = 8000$, $\nu = 0.4$ for the soft elastic. In Plasticine material τ_γ is set to 500 and in Sand material θ_f is set to 10° .



(a) A bus slides at a *moving* floor.



(b) Blocks drop on the *balls*.

Figure 12. Simulation results based on different boundary conditions.