NEURAL VOXEL DYNAMICS: LEARNING VOLUMETRIC FEATURE ADVECTION FOR 3D PHYSICS IN V-JEPA LATENT SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

We present *Neural Voxel Dynamics*, a self-supervised framework for learning 3D latent dynamics from monocular video. While generative video models now produce visually compelling motion, their predominantly 2D representations provide limited geometric structure for modeling and controlling physical interactions. Instead, we learn dynamics in a lifted volumetric latent space. Specifically, we unproject semantic Video Joint-Embedding Predictive Architecture (V-JEPA) features into a voxel grid using monocular depth priors, producing a geometrically grounded representation that retains rich video features. We then introduce *Volumetric Feature Advection*, an action-conditioned transition model that predicts the evolution of these latent features directly in 3D. Unlike hybrid approaches that assume access to privileged physics-engine states or explicit simulation during training and/or inference, we learn only using video-derived supervision and, if available, actions, allowing a single latent dynamics model to represent heterogeneous material-dependent phenomena, unifying rigid-body and fluid dynamics. We evaluate across multiple datasets (synthetic and real) and unseen scenarios (e.g., new boundary conditions, OOD generalization), measuring both predictive fidelity and 3D geometric consistency. We report physical plausibility with Physics-IQ using videos decoded from predicted latent states and videos generated from predicted optical flow. Our results show that geometrically lifting pretrained video latent representations provides a scalable route toward dynamic world models that learn structured 3D physical dynamics without access to privileged simulator state.

1 INTRODUCTION

Can we learn implicit physics from videos and use it to generate physically grounded dynamic worlds? Modern video models learn rich representations of appearance and motion, yet typically predict dynamics in 2D image-aligned or token-based latent spaces. Physical dynamics, however, unfold in 3D: objects persist through occlusion, motion occurs in a common spatial frame, and interactions depend on geometry and material properties. While recent large-scale video models achieve impressive visual fidelity, they may struggle to maintain such physical invariants over time, particularly under interaction (Motamed et al., 2026). We humans, in contrast, regularly infer and predict physical behavior from only visual observations without explicitly solving for the parameters of a physical simulator (McCloskey et al., 1983; Battaglia et al., 2013).

Existing approaches loosely trade off *explicit physical structure* and *learned flexibility*. Neural-physics hybrids (Wang et al., 2025a; Li et al., 2025; Xie et al., 2024; Tan et al., 2024; Zhan et al., 2026) couple learned visual representations with classical simulators. While these approaches, by construction, provide strong physical and geometric structure grounding, they require many assumptions such as predicting simulator states and physical parameters from visual observations, and selecting an appropriate material model or simulator a priori. On the other hand, learned video dynamics models (Nguyen-Phuoc et al., 2020; Ehrhardt et al., 2020; Gillman et al., 2025; Gu et al., 2025), being implicit, can model non-trivial visual phenomena without requiring any classical simulator. Although some incorporate 3D-aware structure, most evolve state in 2D or image-aligned latent spaces; hence, 3D motion, occlusion, and interaction continue to be represented implicitly. We investigate a middle ground: *3D geometry grounded, yet materially implicit* dynamics.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

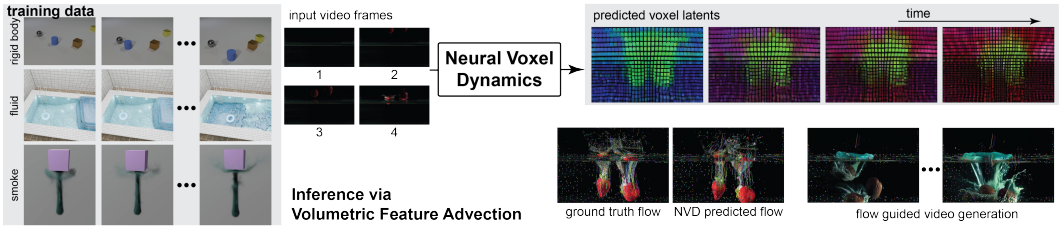


Figure 1: **Unified Implicit 3D Physics.** Trained exclusively on unconstrained 2D videos (left), Neural Voxel Dynamics learns to simulate complex interactions by advecting latent features within a 3D voxel grid (right). Our implicit formulation naturally unifies the prediction of heterogeneous materials like fluids, smoke, and rigid bodies. Here, conditioned on four initial frames, our model predicts, in V-JEPA space, the complex dynamics of fruits dropping into water; this is then decoded to predict 2D flows and drive a trajectory-driven video model. See supplemental for videos.

Our approach leverages the rich semantic representations of Video JEPA (V-JEPA) (Assran et al., 2025; Mur-Labadia et al., 2026), which we unproject into a 3D voxel grid using monocular depth priors (Ren et al., 2025). The lifting places features observed across frames into a single Euclidean coordinate system, allowing the dynamics model to reason directly about spatial motion rather than jointly representing dynamics and camera projection in an image-aligned latent space (i.e., camera motion is factored out). Monocular observations, however, reveal only a partial view of the underlying 3D world, so we additionally track latent state in unobserved voxels over time; note we do not perform object completion. Our persistent volumetric state allows information to survive occlusion to some extent and provides an easier representation for reasoning about object permanence. Our key hypothesis is that such a representation turns future prediction from a sequence-to-sequence problem into a *spatio-temporal latent transport problem in 3D*.

To evolve this latent world state, we introduce *Volumetric Feature Advection*, an action-conditioned predictor that performs grid-to-grid updates in the lifted 3D representation. Here, an *action* can be specified explicitly as a 3D external force vector or implicitly using multiple past frames. Given the current volume and action, our predictor, modeled as a flow matching transformer, estimates a per-voxel 3D flow field and transports V-JEPA features by locally attending and aggregating the volume along it, while specifically adjusting predictions being in observed or occluded regions. We represent material-dependent behavior implicitly by the high-dimensional latent state and the learned transition function. Consequently, the same transition architecture can model different physical behaviors (e.g., rigid-body motion, fluid-like transport, or hybrids) without any material-specific solver. Our predictor is trained end-to-end only using video-derived targets and actions, without supervision from internal physics-engine states or manual physical annotations. Figure 1 visualizes predicted latent states as the model evolves a partially observed scene of fruits falling into water.

We evaluate Neural Voxel Dynamics across heterogeneous physical domains using CLEVRER (Yi et al., 2019), PhysInOne (Zhou et al., 2026), PhysGaia (Kim et al., 2026), and Kubric (Greff et al., 2022). We measure predictive fidelity (feature-space error and reconstructed-frame PSNR), geometric consistency (occupancy loss), and generalization (e.g., changing object counts, shapes, or material configurations unseen during training). We further assess the physical plausibility of generated videos using Physics-IQ (Motamed et al., 2026). Against representative neural-physics hybrids (Wang et al., 2025a; Xie et al., 2024; Liu et al., 2025), our NVD reduces feature-space prediction error by 6% - 19% without access to simulator state; against a 2D latent diffusion video model (Yang et al., 2024), it improves dynamic consistency for synthetic videos (evaluated from Physics IQ benchmark) by up to 28%. It also improves the same video models on real videos by a score of 0.2 by injecting predicted trajectory controls. Together, these results suggest that explicit 3D structure with spatially anchored latent features is a useful inductive bias for learning heterogeneous physical dynamics.

Our contributions are: (i) a persistent, geometrically grounded latent world state that lifts pretrained video features into 3D, maintaining partially observed state through time; (ii) Volumetric Feature Advection, an action-conditioned transition operator that generatively evolves this state without material-specific solvers or privileged information; and (iii) empirical evaluation showing that a single shared 3D latent transition model captures diverse dynamics outperforming hybrid and 2D-latent baselines on flow and video prediction.

2 Related Works

Video Generation and Controllability. Diffusion models are the dominant paradigm for video synthesis, extending foundational image-based frameworks such as DDPM (Ho et al., 2020) and DDIM (Song et al., 2022) to the temporal domain through spatio-temporal attention and latent diffusion architectures (Ho et al., 2022; Rombach et al., 2022). While existing video models and world models demonstrate remarkable scalability, they often lack explicit mechanisms for physically-grounded generation or control. To bridge this gap, recent research has explored trajectory-based conditioning. Multiple methods (He et al., 2024; Fu et al., 2024) enable explicit camera or object trajectory transfer, while point-based motion guidance techniques (Geng et al., 2025; Gillman et al., 2025; Chu et al., 2025) allow for user-specified 2D motion cues. However, these approaches primarily operate in image space, where control signals remain heuristic. More 3D-aware representations (Gu et al., 2025; Sheng et al., 2026; Bhat et al., 2026) do not explicitly model the underlying physical forces or 3D interactions, often resulting in motions that lack causal and/or dynamical consistency.

Explicit Physics-Based Generative Methods. A parallel line of work integrates classical physics simulation directly into generative pipelines. Early approaches utilized image-space modal bases (Davis et al., 2015) or 2D physics proxies like PhysGen (Liu et al., 2025). More recent approaches utilize 3D Gaussian Splatting as a substrate for simulation; e.g., SpringGauss (Zhong et al., 2024) and PhysGaussian (Xie et al., 2024) respectively incorporate spring-mass systems or Material Point Method (MPM) (Stomakhin et al., 2013; Jiang et al., 2016) to achieve physically plausible dynamics. While producing high-fidelity output, these methods require scene-specific reconstructions and explicit parameter estimation.

Hybrid methods (Chen et al., 2025a; Tan et al., 2024; Lin et al., 2025) combine explicit simulation with generative priors, often relying on external engines like PyBullet (Coumans & Bai, 2016), MuJoCo (Todorov et al., 2012), or Warp (Zong et al., 2023). However, a fundamental bottleneck remains: these systems assume access to precise material properties and system states as input, which are often manually specified or crudely estimated. To mitigate this, PhysDreamer (Zhang et al., 2025), Physics3D (Liu et al., 2024), and PSIVG (Foo et al., 2026) refine physical parameters through generative feedback. Despite such advances, the methods are restricted to preprocessing, segmentation, and isolated solid objects, and struggle to model complex multi-object interactions or diverse materials. Approaches like PhysCtrl (Wang et al., 2025a) use diffusion models as neural proxies for simulators to bypass explicit tuning, yet still face challenges in robust material estimation and handling different physical interactions (e.g., fluids versus rigid body dynamics).

Implicit Representations and Predictive World Models. Alternately, implicit generative models learn structured scene representations via latent decomposition. Early works like BlockGAN (Nguyen-Phuoc et al., 2020) and BlobGAN (Epstein et al., 2022) demonstrate the ability to disentangle scene components into latent variables, though they generally lack temporal dynamics or force-based reasoning. Joint-Embedding Predictive Architectures (JEPA) (Assran et al., 2023) have shown promise in learning predictive world models. Recently, V-JEPA (Drozdov et al., 2024; Assran et al., 2025), 3D-JEPA (Hu et al., 2024) and LeWM (Maes et al., 2026) learn high-level representations for future state prediction. Concurrent work Phantom (Shen et al., 2026) also looks at modeling the visual and latent physics jointly. While these models capture the *gist* of physical movement, they lack any fine-grained, user-addressable force control required for 4D animation. Our work bridges this divide by providing a factorized 4D representation that can integrate geometric strength of reconstruction priors (e.g., particles, splats (Tang et al., 2024), deformation fields (Sabathier et al., 2026)) with the generality of a disentangled motion (V-JEPA) latents.

3 Method

Learning physics from raw video is challenging as pixel observations entangle distinct factors: camera perspective, scene geometry, and material-dependent dynamics. In contrast to 2D generative models or explicit 3D simulators, we propose *Neural Voxel Dynamics* as a latent middle ground. We factorize the problem (Figure 2): (i) *Geometric Lifting* (Section 3.1), which unprojects monocular semantic features into a partially observed 3D latent voxel grid in the world frame; and (ii) *Volumetric Feature Advection* (Section 3.2), an action-conditioned generative operator that evolves this spatial latent state forward in time. Our design is: (i) *materially implicit:* by evolving dynamics in a high-dimensional

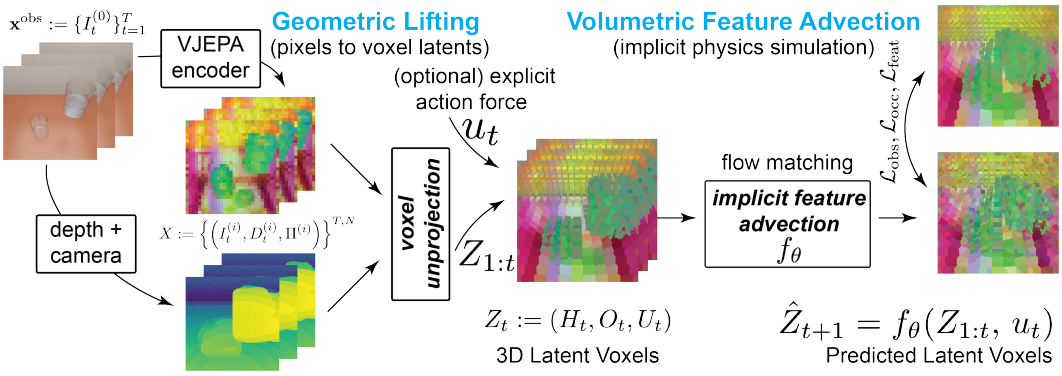


Figure 2: **Neural Voxel Dynamics.** By lifting 2D semantic features into a 3D latent voxel grid, our generative feature advector f_θ learns to implicitly simulate action-conditioned physics directly in the latent space via flow matching; the model (comprising multiple stages of *local attention* and *temporal attention* layers, see section A.4) is supervised with video-derived signals and optional explicit action.

V-JEPA latent space, a single operator handles heterogeneous phenomena without any material label; (ii) *geometrically grounded:* the voxel structure makes spatial occupancy and occlusion explicit, rather than something the model must infer from entangled 2D latents; (iii) *perspective-independent:* being anchored in a world frame, it supports multiview input and a predicted 4D trajectory can be rendered from novel viewpoints *without* re-running the dynamics operator.

Problem setting. We are given T frames $\mathbf{x}^{\text{obs}} := \{I_t\}_{t=1}^{T}$ from a single fixed viewpoint. Geometric Lifting maps these observations to a sequence of 3D latent voxel states $Z_{1:t}$, each carrying monocular depth estimates and semantic V-JEPA features, and marks every voxel as `empty`, `occupied`, or `occluded`. Given a history $Z_{1:t}$ and an (optional) external action u_t applied at step t, the advector predicts the next state $\hat{Z}_{t+1}$; rolling this forward autoregressively yields a full latent trajectory. Finally, a projection head projects any predicted state back to patch-aligned latents for a target camera. For notational simplicity, we use a single index t for both observed and predicted steps; note that the V-JEPA encoder has a tubelet size of two, so one latent step spans two source frames.

3.1 Geometric Lifting to Voxel Latents

Instead of training a 4D JEPA video encoder from scratch, we adopt a reconstruction-based alternative that reuses a strong pretrained backbone. We describe the monocular setting; the formulation extends directly when real multi-view data, or a separate view-synthesis or reconstruction module, provides additional views. We estimate per-frame depth and camera parameters with MoGe (Wang et al., 2025c). Given $\mathbf{x}^{\text{obs}} \in \mathbb{R}^{T\times H\times W\times 3}$, and using $N = 1$ in the monocular case,

$$X := \left\{\left(I_t^{(i)}, D_t^{(i)}, \Pi^{(i)}\right)\right\}^{T,N} \text{ with } \{I, D\} \in \mathbb{R}^{N\times T\times H\times W\times(3+1)},$$

where $I^{(i)}$ and $D^{(i)}$ are the RGB and depth observations of each view and $\Pi^{(i)}$ its intrinsics and extrinsics; we take the first camera frame as the world frame. In parallel, the frozen V-JEPA 2.1 encoder (Mur-Labadia et al., 2026) produces patch-level features,

$$\left\{z_t^{(i)}\right\}^{T'} := \text{VJEPA}\left(\left\{I_t^{(i)}\right\}^{T}\right) \in \mathbb{R}^{T'\times H'\times W'\times d},$$

with temporal ratio $T/T' = 2$ and $d = 1408$. We use V-JEPA as it has temporally consistent dense features and has been shown to encode a degree of physical understanding (Garrido et al., 2025), both of which we exploit when evolving state directly in feature space.

3.1.1 Voxel unprojection

For each view, the frozen encoder produces a dense 2D grid of patch-level features, and we pool the depth map to the patch resolution. Using the intrinsics, extrinsics, and pooled depth, we unproject

each patch feature z_t from image space to a 3D point in the world frame, and aggregate its contribution into a shared voxel lattice by trilinear interpolation, yielding the feature channel H_t (see section A.1). Alongside the features we maintain two scalar channels: an `occupancy` channel O_t, marking voxels that contain observed scene content, and an `occlusion` channel U_t, marking cells that lie behind an observed surface and are therefore currently unseen (i.e., a subset of the complement of O_t, at depths beyond the observed surface). The full latent state is $Z_t := (H_t, O_t, U_t)$. Because only `occupied` and `occluded` cells carry information, isolated `empty` voxels are never materialized, saving space.

3.2 Volumetric Feature Advection

Given a history $Z_{1:t}$ and action u_t, the advector f_θ predicts the next state,

$$\hat{Z}_{t+1} = f_\theta(Z_{1:t},\, u_t)\,.$$

Rather than deterministically regressing the next state, we adopt a flow-matching formulation (Lipman et al., 2022) (section A.2): f_θ learns a time-dependent velocity field that transports a noisy source distribution toward the target latent manifold. Our model has a flexible generative prior over physically plausible transitions and lets it represent the conditional uncertainty arising from partial observation and unresolved dynamics, without the rigid constraints of a predefined analytical solver.

Explicit control. Our model also takes explicit (but optional) force control input u_t in addition to implicit temporal frames. We augment each voxel token with 3D spatial and temporal positional embeddings. The control is a time-varying action descriptor $u_t \in \mathbb{R}^m$: a 12-dimensional ($m = 12$) encoding of the contact point, applied direction, magnitude, temporal duration, active-frame indicator, and local/global flags, expressed and normalized in world coordinates to reduce ambiguity. We inject it through two complementary paths: (i) *local integration*, in which the force is projected to the token dimension and added element-wise to each voxel token, giving a spatially uniform contextual field; and (ii) *global modulation*, in which the force is encoded into a set of conditioning tokens concatenated to the sequence, letting self-attention modulate the global scene transition (section A). Note that u_t is not a privileged simulator state; it does not specify the material parameters or internal solver variables that hybrid methods require. Note that the advector never receives material parameters (e.g., stiffness, viscosity), engine states, or per-particle states; material-dependent behavior emerges purely from the latent state and learned transition. In our experiments, actions are drawn only for the CLEVRER/MuJoCo dataset; on data without such annotations, the action reduces to the zero vector, so the model still predicts implicit dynamics. At inference time, users can optionally provide explicit force input (see supplemental for examples).

Sparse tokenization and efficiency. To avoid the cubic cost of dense grids, f_θ operates on a *sparse set of active voxels* and their neighbors. We define the set as the union of `occupied` and `occluded` voxels across the context window, dilated morphologically to leave room for predicted motion, with size M. This concentrates capacity on scene-relevant geometry, and scales more with object complexity than with grid resolution. We operate at a single grid spacing.

Dynamic local spatiotemporal transformer. Our feature advector is a Diffusion Transformer (DiT) variant whose layers alternate between two attention modes. (i) *Local spatial attention*, inspired by the locality of interactions in classical simulators (e.g., MPM (Jiang et al., 2016; Welschinger et al., 2025)), restricts each voxel to a $k \times k \times k$ sparse neighborhood; this center-query mechanism mimics step-wise force propagation in the physics engine and gives cost that scales linearly with M, $O(Mk^3)$, rather than quadratically in the number of active tokens. We use 5 layers and $k = 5$, with M set by object complexity (i.e., number of active voxels). (ii) *Temporal attention* regroups tokens by spatial location so that each voxel attends causally along its own trajectory, capturing temporal changes in local features and occupancy. Because a single step propagates information only within a local neighborhood, longer-range transport is composed over stacked layers and successive prediction steps; we discuss the resulting displacement range in section A.4. The transformer is modulated by the flow-matching time τ via adaptive layer normalization (AdaLN) (Wang et al., 2025a; Yang et al., 2024), outputting feature velocities that are scattered back into the lattice; we integrate the predicted velocity field v_θ with Euler steps to solve the probability-flow ODE.

3.2.1 TRAINING OBJECTIVES

We optimize a multi-task objective that decouples feature dynamics from geometric occupancy across the three channels of $Z_t := (H_t, O_t, U_t)$, where O_t is the `occupancy` probability and U_t the `occlusion` probability.

Feature velocity loss. Under the flow-matching formulation, the feature branch regresses a velocity field v_θ toward the target manifold. To prevent empty space from dominating, we weight the loss by occupancy as:

$$\mathcal{L}_{\text{feat},r} := \frac{\sum_j O_r(j)\, \|v_{\theta,r}(j) - v^\star(j)\|_2^2}{\sum_j O_r(j) + \xi}, \tag{1}$$

where the target velocity $v^\star = \epsilon - Z_{t+1}$ is the derivative of the linear interpolant $(1-\tau)Z_{t+1} + \tau\epsilon$ with ϵ Gaussian noise (section A.2). The occupancy $\mathcal{L}_{O,r}$ and occlusion $\mathcal{L}_{U,r}$ channels use the same flow-matching objective, which we find outperforms a binary target (Table 15), likely because it lets geometry and features share one denoising dynamics.

Diffusion-forcing objective. Following diffusion forcing, we corrupt history frames at varying noise levels and explicitly condition the predictor on the noise level of each frame, limiting error accumulation over long rollouts (Chen et al., 2024). For stable autoregressive rollout, we sample an independent noise level $\tau_r \sim p_\tau$ per latent frame (Chen et al., 2024), shared across channels within a frame, and averaged over the clip. Each training window contains C context states and S predicted states, giving $K = C + S$ states in total. The frame index $r = C+1, \ldots, K$ covers only the predicted states.

$$\mathcal{L}_{\text{DF}} = \mathbb{E}\left[\frac{1}{S}\sum_{r=C+1}^{K}\left(\mathcal{L}_{\text{feat},r} + \lambda_O \mathcal{L}_{O,r} + \lambda_U \mathcal{L}_{U,r}\right)\right]. \tag{2}$$

where the expectation is over training clips, sampled noise levels, and Gaussian noise, and $\lambda_O, \lambda_U > 0$ weight the geometry channels.

Projection loss (2D). We encourage the projected features to match the encoded future observations in V-JEPA space. Using differentiable front-to-back compositing, we supervise the projected features and accumulated opacity of the estimated clean endpoint. For each 2D latent patch at u, v:

$$\mathcal{L}_{\text{featproj}} = \frac{\sum_{u,v} M_{u,v}\|\hat{z}_{u,v} - z_{u,v}\|_2}{D(\sum_{u,v} M_{u,v} + \delta)}, \qquad \mathcal{L}_{\text{occproj}} = \frac{1}{|\mathcal{P}|}\sum_{u,v}(\hat{M}_{u,v} - M_{u,v})^2. \tag{3}$$

where $\hat{z}_{u,v}$ is the projected 2D features, $z_{u,v}$ the encoder's target patch features, $M_{u,v} \in [0,1]$ is the reference depth-validity mask, restricted to observations inside the voxel volume, and $\hat{M}_{u,v}$ is the accumulated ray opacity; details in Section A.3.

Total objective. The full loss combines the volumetric and projection terms,

$$\mathcal{L} = \mathcal{L}_{\text{DF}} + \mathbb{E}\left[\frac{1}{S}\sum_{r=C+1}^{K}\left(\lambda_{\text{fp}}\mathcal{L}_{\text{featproj},r} + \lambda_{\text{op}}\mathcal{L}_{\text{occproj},r}\right)\right]. \tag{4}$$

with weights $\lambda_{\text{fp}}, \lambda_{\text{op}}$ (set to 1 in our tests). We train the voxel predictor using video-derived feature and geometry targets, conditioned on supplied force inputs, with no supervision from physics-engine states or manual annotations.

Inference. At test time, we maintain a rolling history of 2 latent states (i.e., 4 image frames). Given $Z_{1:2}$ and an (optional) action u_2, we integrate the learned velocity field from Gaussian noise to a clean prediction by solving the probability-flow ODE with 20 Euler steps, scatter the result into the sparse lattice to form $\hat{Z}_{3:10}$ (i.e., 16 RGB frames), append it to the history, and repeat. Diffusion forcing supports history-guided sampling (Song et al., 2025). At each denoising step, we subtract the velocity predicted with fully noised history from that predicted with clean history, then add a scaled version of this difference to the history-conditioned velocity as a correction.

4 EXPERIMENTS

Datasets. We evaluate on four sources with complementary dynamics. Our synthetic CLEVRER-style data (Yi et al., 2019) focuses on rigid-body interactions simulated with MuJoCo (Todorov et al., 2012), augmented with a labeled force vector not present in the original CLEVRER. PhysInOne (Zhou et al., 2026) adds fluids and rigid bodies, and PhysGaia (Kim et al., 2026) contributes more complex scenes with cloth, smoke, and soft bodies. Although the pipeline supports multi-camera inputs, to have a reasonable comparison against monocular baselines, training uses single-camera inputs. See Section A.5. To test *zero-shot generalization*, we additionally evaluate the *frozen* NVD predictor on 30 samples from the unseen Kubric MOVI-C dataset (Greff et al., 2022). We also take the Physics-IQ (Motamed et al., 2026) benchmark dataset for video physics evaluation.

Model variants. *Ours-GT* uses ground-truth depth (available only for the simulated CLEVRER data and PhysInOne) and serves as a reference for ablation; *Ours-Estimate* uses MoGe-estimated depth (Wang et al., 2025c) on all datasets and is our deployable model for real scenes and generalization. To isolate the contribution of the 3D lift, we also train a *2D baseline* of comparable capacity that removes the 3D projection and applies the same local/temporal attention and autoregressive flow-matching objective directly on the 2D V-JEPA latents. As this baseline is optimized with identical latent objective, it also bounds how much the V-JEPA-space metric could favor our method.

Baselines. We compare against simulator-based methods PhysGen(2D) (Liu et al., 2025) and PhysGaussian(3D) (Xie et al., 2024), the simulator-proxy PhysCtrl (Wang et al., 2025a), and the I2V model CogVideoX (Yang et al., 2024). All video baselines receive the same monocular initial frames and predict the subsequent 16 frames, while PhysGaussian consumes all camera angles. PhysCtrl and PhysGaussian consume the 3D force directly; for PhysGen we project the force into 2D camera space, and for CogVideoX we supply it via text conditioning. See supplemental webpage.

4.1 LATENT-STATE PREDICTION ACROSS SIMULATION REGIMES

We first test prediction quality directly in latent space. For each method, we compare predicted future latents to ground-truth future latents, reporting L2 error in both the 2D latent space and the 3D (unprojected from estimated depth) latent space; baseline outputs are re-encoded into V-JEPA space for a common comparison. This measures whether a model's transition stays on the V-JEPA manifold and follows the correct motion, rather than merely producing smooth-looking frames. In the multi-camera setting, all models receive only a single camera at inference but are scored across all views; in contrast, PhysGaussian is given multiple views to enable its 3DGS reconstruction. Results grouped by material appear in Table 1; the full per-dataset breakdown is in Section B.1, Table 4.

Table 1: Latent prediction on different dynamics categories (synthetic). Each entry reports 2D / 3D latent L2 loss (↓). Ours, being material implicit, has a single model across settings.

Method	Rigid body	Fluid	Smoke
CogVideoX (Yang et al., 2024)	3.39 ± 0.64 / 0.91 ± 0.33	4.21 ± 0.31 / 1.15 ± 0.13	3.07 ± 0.68 / 1.56 ± 0.48
PhysGen (Liu et al., 2025)	2.54 ± 0.35 / 0.59 ± 0.37	5.22 ± 1.10 / 1.22 ± 0.39	4.37 ± 0.45 / 1.80 ± 0.47
PhysGaussian (Xie et al., 2024)	3.32 ± 0.15 / 0.90 ± 0.31	4.80 ± 0.20 / 1.18 ± 0.19	2.67 ± 1.06 / 1.70 ± 0.47
PhysCtrl (Wang et al., 2025a)	3.58 ± 0.37 / 0.83 ± 0.33	5.03 ± 0.34 / 1.19 ± 0.22	3.72 ± 0.96 / 1.73 ± 0.48
2D baseline	2.09 ± 0.13 / 0.31 ± 0.21	$\underline{2.99 \pm 0.27}$ / 1.20 ± 0.33	2.51 ± 0.53 / 1.62 ± 0.25
Ours-GT	$\mathbf{1.94 \pm 0.14}$ / $\mathbf{0.19 \pm 0.11}$	$\mathbf{2.12 \pm 0.11}$ / 0.82 ± 0.09	2.22 ± 0.33 / 1.15 ± 0.20
Ours-estimate	$\underline{1.97 \pm 0.16}$ / $\underline{0.26 \pm 0.12}$	$\mathbf{2.12 \pm 0.21}$ / $\mathbf{0.80 \pm 0.13}$	$\mathbf{2.03 \pm 0.41}$ / $\mathbf{1.00 \pm 0.29}$

An accurate and material-robust model. Both our variants achieve the best or second-best 3D latent error in every material category, improving over the strongest simulator-based baseline by 7% to up to 29% across material types. More importantly for our central claim, our error is the *most stable across materials*: the 2D latent error varies by only ± 0.12 across rigid/fluid/smoke for Ours-Estimate, whereas the specialized baselines swing sharply. PhysGen is competitive on rigid bodies but degrades markedly on fluid and smoke, reflecting their reliance on 2D structure and per-material solver assumptions. While CogVideoX catches up in the fluid and smoke scenarios, it fails in rigid-body scenarios because it lacks fine-grained control input and object permanence. PhysGaussian is primarily designed for soft-body dynamics and relies on multiview input for accurate

scene reconstruction. Like PhysCtrl, it also requires segmentation of the main object and is best suited to single-object scenarios, which may partly explain its relatively weaker performance.

Using a similar (latent) error metric as our objective did give us an advantage, as shown in the 2D baseline result being better than most other baselines. However, Ours-Estimate reduces the 3D latent error by around 29% relative to the 2D baseline and also performs better in the additional evaluations later on. This suggests that the improvement is not solely due to the choice of metric, but also benefits from lifting into 3D. Finally, the Ours-GT ablation indicates that our method has a lighter assumption on privileged geometry.

4.2 Flow Estimation From Latent Dynamics

We next test whether the learned latents encode transferable motion prediction. Keeping V-JEPA frozen, we train a small convolutional head that maps two consecutive latent maps to an optical-flow field (Section C.2); predicted displacements are scaled to pixels and integrated over time. At inference, the head needs only the latent sequence; baseline flows are computed from their generated videos. Table 2 reports optical-flow displacement error, normalized by the ground-truth flow magnitude, and optical flow vector angle difference.

Table 2: Optical-flow error (↓). The table reports magnitude/angle. See supplemental for videos.

Method	CLEVRER ↓	PhysGaia ↓	PhysInOne ↓	Kubric MOVI-C (zero-shot) ↓
CogVideoX	1.551 ± 0.915 / $\underline{17.7 \pm 5.3}$	$\underline{0.996 \pm 0.037}$ / $\mathbf{21.8 \pm 6.3}$	1.094 ± 0.113 / $\mathbf{24.0 \pm 6.0}$	1.086 ± 0.228 / $\underline{23.4 \pm 2.7}$
PhysGen	0.993 ± 0.057 / 19.2 ± 3.6	1.001 ± 0.003 / 50.2 ± 7.5	1.072 ± 0.151 / 52.3 ± 12.9	1.000 ± 0.061 / 24.1 ± 20.2
PhysGaussian	0.804 ± 0.338 / 26.4 ± 4.1	1.000 ± 0.002 / 37.8 ± 4.0	1.068 ± 0.201 / 60.9 ∓ 13.1	0.982 ± 0.076 / 29.7 ± 13.8
PhysCtrl	10.93 ± 20.97 / 64.5 ± 16.9	1.085 ± 0.173 / 41.0 ± 8.5	1.295 ± 0.455 / 47.7 ± 13.7	1.047 ± 0.121 / 42.3 ± 14.1
2D baseline	$\underline{0.583 \pm 0.032}$ / 21.24 ± 5.2	1.055 ± 0.124 / 30.35 ± 9.5	$\underline{1.005 \pm 0.314}$ / 43.33 ± 13.2	$\underline{0.981 \pm 0.030}$ / 29.24 ± 19.9
Ours-Estimate	$\mathbf{0.545 \pm 0.186}$ / $\mathbf{13.5 \pm 2.5}$	$\mathbf{0.977 \pm 0.165}$ / $\underline{29.0 \pm 6.0}$	$\mathbf{0.980 \pm 0.035}$ / $\underline{25.9 \pm 8.6}$	$\mathbf{0.917 \pm 0.059}$ / $\mathbf{23.1 \pm 9.7}$

Generalizable flow across multiple datasets. Ours-Estimate attains the lowest flow error on all four datasets. The result holds across materials and, most notably, on unseen Kubric MOVI-C, where we improve over the best baseline by 6.6% in magnitude and 1.2% in direction despite never training on it, which is evidence that the 3D-grounded dynamics generalize beyond the training distribution rather than memorizing per-dataset motion. A complementary voxel-space latent flow-consistency evaluation is deferred to Section B.2, where Ours-Estimate also achieves the lowest error.

4.3 Physical Plausibility (Physics-IQ)

Using our generated trajectory, we generate motion-guided video with Wan-move (Chu et al., 2025), given the first frame and the predicted flow trajectories (see above and Section C.2). We assess whether our generated dynamics are physically plausible using Physics-IQ metrics (Motamed et al., 2026). Table 3 reports the overall Physics-IQ verified score on synthetic videos across our four domains, in addition to a subset (10%) of the Physics-IQ test set. And Figure 3 shows a qualitative result (more in the supplementary).

Table 3: Physics-IQ benchmark on different video datasets (overall score ↑).

Method	CLEVRER	Kubric	PhysGaia	PhysInOne	Physics IQ	Total
CogVideoX	0.008	0.308	<u>0.296</u>	0.161	**0.405**	0.237
PhysGen	0.073	0.126	0.031	0.194	0.191	0.163
PhysGaussian (all cameras)	<u>0.368</u>	0.292	0.023	0.225	0.253	0.272
PhysCtrl	0.047	0.268	0.270	<u>0.229</u>	0.289	<u>0.281</u>
Wan-move (no tracks)	0.114	<u>0.335</u>	**0.303**	0.226	0.389	0.259
Ours-Estimate	**0.478**	**0.420**	0.271	**0.293**	<u>0.391</u>	**0.384**

Results. Ours-Estimate obtains the highest score overall and improves the score of the Wan-move baseline with no trajectories. However, performance degrades beyond 3 seconds due to latent noise, while Physics-IQ evaluates 5-second videos. Consequently, CogVideoX performs slightly better on some metrics. Full per-metric breakdowns appear in Section B.4.1.

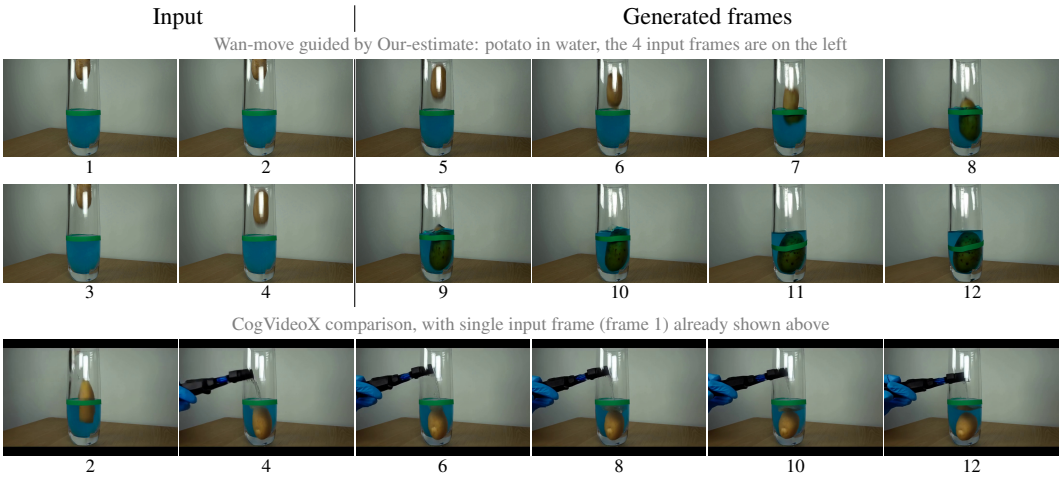


Figure 3: Input frames and the corresponding generated sequences. Note, for ours how the water level rises as the potato sinks demonstrating materially implicit dynamics.

4.4 Other evaluations

Occupancy as a proxy for mass conservation. Because our transition operator predicts an explicit 3D occupancy field, we can measure how the occupancy matches the ground truth as a scene "mass" conservation. We report occupancy IoU against ground truth and the drift in total occupied volume. Full results are in Section B.5. Our model maintains high occupancy IoU, consistent with the behavior seen in the flow and Physics-IQ evaluations.

Controllable Generalization. We probe controllability in a simple, interpretable setting: a single ball on a floor, actuated by an impulse force applied at chosen contact points. The results (see supplementary for videos) show that varying the force produces correspondingly different, physically consistent trajectories, and that the model respects contact geometry (e.g., object permanence across temporary occlusion, plausible collision with different objects). This demonstrates that control transfers to configurations outside the training clips.

4.5 Ablations

We highlight the findings from our ablations here; further details are in Section B.5.

(i) 2D vs. 3D latent dynamics. Across both latent prediction (Table 1) and flow (Table 2), the 2D baseline is consistently beaten by our 3D model under the same objective and comparable capacity. The gap widens on non-rigid and occluded content, where reasoning in a shared Euclidean frame matters most. This suggests that the improvement mainly comes from the 3D lifting rather than differences in the training setup.

(ii) Regression vs. generative transition. Section B.5 compares a deterministic-regression variant against the generative flow-matching formulation (with diffusion forcing and history guidance) over 24-frame rollouts. Regression is clearly worse on occupancy IoU and both flow metrics (0.208 vs. 0.709 IoU), confirming that a generative prior is needed to model the conditional uncertainty of partially observed dynamics. Among generative variants, vanilla history guidance (DF+HG) is best on all three metrics and is used for Ours-Estimate.

5 Conclusion

Neural Voxel Dynamics bridges 2D video generation and 3D physics simulation. By lifting monocular video into a 3D latent voxels and learning an action-conditioned feature advection model, we achieve materially implicit, geometrically grounded physical transitions from monocular video, disentangling camera from material dynamics and forecasting heterogeneous phenomena without segmentation, material estimation, or physics-engine states.

Limitations and Future Work. Our geometric fidelity is bounded by the V-JEPA encoder resolution and monocular depth accuracy. Sparse tokenization scales poorly to unbounded environments, moving cameras, or high-frequency deformations, and long rollouts accumulate noise; possibly mitigable via larger training chunks or feeding refined depth and V-JEPA latents back to the model. Our model also is not designed to generate unseen voxels. Future work includes constructing/generating full voxel volumes from image input and a universal V-JEPA decoder for photorealistic video.

AI USE STATEMENT

(This section is **required** and does not count toward the page limit.)

In this work, we used generative AI tools for polishing writing, finding related work, and implementing standard methods like generating, cleaning, and reformatting datasets, projection and soft projection, and evaluation code. We have not used generative AI tools for design or to provide feedback on research methodology or experiments, help develop theoretical models or conceptual frameworks. We have reviewed all AI-assisted work. We checked LLM-generated related works for potential hallucination; LLM-generated code was verified and tested for correctness. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

See the ICLR 2027 AI Policy for Authors for more details. This statement should not be more than 1 page.

REPRODUCIBILITY STATEMENT

(This section is **recommended** and does not count toward the page limit.)

We have shown the dataset and methods in the evaluation section. We have also shown in the supplementary the necessary hyperparameter values and data preprocessing used to recreate the results.

We will release the code subject to acceptance.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture. *arXiv preprint arXiv:2301.08243*, April 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-Supervised Video Models Enable Understanding, Prediction and Planning. *arXiv preprint arXiv:2506.09985*, June 2025.

P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013. doi: 10.1073/pnas.1306572110.

Shariq Farooq Bhat, Niloy J. Mitra, and Kalyan Sunkavalli. Loosecontrolvideo: Directorial video control using spatial blocking. In *ECCV*, 2026.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion Forcing: Next-token Prediction Meets Full-Sequence Diffusion. In *NeurIPS*, volume 37, 2024. doi: 10.52202/079017-0759.

Boyuan Chen, Hanxiao Jiang, Shaowei Liu, Saurabh Gupta, Yunzhu Li, Hao Zhao, and Shenlong Wang. PhysGen3D: Crafting a Miniature Interactive World from a Single Image. In *Proc. CVPR*, pp. 6178–6189. IEEE, June 2025a.

Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. *arXiv:2501.12375*, 2025b.

Ruihang Chu, Yefei He, Zhekai Chen, Shiwei Zhang, Xiaogang Xu, Bin Xia, Dingdong Wang, Hongwei Yi, Xihui Liu, Hengshuang Zhao, Yu Liu, Yingya Zhang, and Yujiu Yang. Wan-move: Motion-controllable video generation via latent trajectory guidance. *arXiv preprint*, 2025.

Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016. Accessed: 2026-04-08.

Abe Davis, Justin G. Chen, and Frédo Durand. Image-space modal bases for plausible manipulation of objects in video. *ACM TOG*, 34(6):1–7, November 2015.

Katrina Drozdov, Ravid Shwartz-Ziv, and Yann LeCun. Video Representation Learning with Joint-Embedding Predictive Architectures. *arXiv preprint arXiv:2412.10925*, December 2024.

Sébastien Ehrhardt, Oliver Groth, Aron Monszpart, Martin Engelcke, Ingmar Posner, Niloy Mitra, and Andrea Vedaldi. Relate: Physically plausible multi-object scene synthesis using structured latent spaces. *Proc. NeurIPS*, 33:11202–11213, 2020.

Dave Epstein, Taesung Park, Richard Zhang, Eli Shechtman, and Alexei A. Efros. BlobGAN: Spatially Disentangled Scene Representations. In *ECCV*, volume 13675, pp. 616–635. Springer Nature Switzerland, 2022.

Lin Geng Foo, Mark He Huang, Alexandros Lattas, Stylianos Moschoglou, Thabo Beeler, and Christian Theobalt. Physical Simulator In-the-Loop Video Generation. *arXiv preprint arXiv:2603.06408*, March 2026.

Xiao Fu, Xian Liu, Xintao Wang, Sida Peng, Menghan Xia, Xiaoyu Shi, Ziyang Yuan, Pengfei Wan, Di Zhang, and Dahua Lin. 3Dtrajmaster: Mastering 3D trajectory for multi-entity motion in video generation. *arXiv preprint arXiv:2412.07759*, 2024.

Quentin Garrido, Nicolas Ballas, Mahmoud Assran, Adrien Bardes, Laurent Najman, Michael Rabbat, Emmanuel Dupoux, and Yann LeCun. Intuitive physics understanding emerges from self-supervised pretraining on natural videos. *arXiv preprint arXiv:2502.11831*, February 2025.

Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, Oliver Wang, Andrew Owens, and Deqing Sun. Motion Prompting: Controlling Video Generation with Motion Trajectories. In *Proc. CVPR*, pp. 1–12. IEEE, June 2025.

Nate Gillman, Charles Herrmann, Michael Freeman, Daksh Aggarwal, Evan Luo, Deqing Sun, and Chen Sun. Force Prompting: Video Generation Models Can Learn and Generalize Physics-based Control Signals. *arXiv preprint arXiv:2505.19386*, May 2025.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3749–3761, 2022.

Zekai Gu, Rui Yan, Jiahao Lu, Peng Li, Zhiyang Dou, Chenyang Si, Zhen Dong, Qifeng Liu, Cheng Lin, Ziwei Liu, Wenping Wang, and Yuan Liu. Diffusion as Shader: 3D-aware Video Diffusion for Versatile Video Generation Control. In *ACM SIGGRAPH*, pp. 1–12. ACM, August 2025.

Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation. *arXiv preprint arXiv:2404.02101*, 2024.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. *arXiv preprint arXiv:2006.11239*, December 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video Diffusion Models. *Proc. NeurIPS*, 35, June 2022.

Naiwen Hu, Haozhe Cheng, Yifan Xie, Shiqi Li, and Jihua Zhu. 3D-JEPA: A Joint Embedding Predictive Architecture for 3D Self-Supervised Representation Learning. *arXiv preprint arXiv:2409.15803*, September 2024.

Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *ACM SIGGRAPH Courses*, pp. 1–52, Anaheim California, July 2016. ACM.

Mijeong Kim, Gunhee Kim, Jungyoon Choi, Wonjae Roh, and Bohyung Han. Physgaia:physics-aware benchmark with multi-body interactions for dynamic novel view synthesis. In *Proc. CVPR*, 2026.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *ECCV*, pp. 71–91. Springer, 2024.

Zizhang Li, Hong-Xing Yu, Wei Liu, Yin Yang, Charles Herrmann, Gordon Wetzstein, and Jiajun Wu. WonderPlay: Dynamic 3D Scene Generation from a Single Image and Actions. In *ICCV*, pp. 9080–9090. arXiv, May 2025.

Jiajing Lin, Zhenzhong Wang, Dejun Xu, Shu Jiang, Yunpeng Gong, and Min Jiang. Phys4dgen: Physics-compliant 4d generation with multi-material composition perception. In *ACM MM*, pp. 10398–10407, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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*, June 2024.

Shaowei Liu, Zhongzheng Ren, Saurabh Gupta, and Shenlong Wang. PhysGen: Rigid-Body Physics-Grounded Image-to-Video Generation. In *ECCV*, volume 15140, pp. 360–378. Springer Nature Switzerland, 2025.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorld-Model: Stable End-to-End Joint-Embedding Predictive Architecture from Pixels. *arXiv preprint arXiv:2603.19312*, March 2026.

M. McCloskey, A. Washburn, and L. Felch. Intuitive physics: the straight-down belief and its origin. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 9(4):636–649, Oct 1983. doi: 10.1037//0278-7393.9.4.636.

Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles? In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 948–958, 2026.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mido Assran, Koustuv Sinha, Mike Rabbat, Yann LeCun, Nicolas Ballas, and Adrien Bardes. V-JEPA 2.1: Unlocking Dense Features in Video Self-Supervised Learning. *arXiv preprint arXiv:2603.14482*, March 2026.

Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *Proc. NeurIPS*, 33: 6767–6778, 2020.

Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. Gen3C: 3D-Informed World-Consistent Video Generation with Precise Camera Control. In *Proc. CVPR*, pp. 6121–6132. IEEE, June 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *Proc. CVPR*, pp. 10674–10685. IEEE, June 2022.

Remy Sabathier, David Novotny, Niloy J. Mitra, and Tom Monnier. ActionMesh: Animated 3D Mesh Generation with Temporal 3D Diffusion. *arXiv preprint arXiv:2601.16148*, April 2026.

Ying Shen, Jerry Xiong, Tianjiao Yu, and Ismini Lourentzou. Phantom: Physics-Infused Video Generation via Joint Modeling of Visual and Latent Physical Dynamics. *arXiv preprint arXiv:2604.08503*, April 2026.

Mingzhi Sheng, Zekai Gu, Peng Li, Cheng Lin, Hao-Xiang Guo, Ying-Cong Chen, and Yuan Liu. FlexAM: Flexible Appearance-Motion Decomposition for Versatile Video Generation Control. *arXiv preprint arXiv:2602.13185*, February 2026.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. *arXiv preprint arXiv:2010.02502*, October 2022.

Kiwhan Song, Boyuan Chen, Max Simchowitz, Yilun Du, Russ Tedrake, and Vincent Sitzmann. History-Guided Video Diffusion. In *ICML*, volume 267, 2025.

Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. *ACM TOG*, 32(4):1–10, 2013.

Xiyang Tan, Ying Jiang, Xuan Li, Zeshun Zong, Tianyi Xie, Yin Yang, and Chenfanfu Jiang. PhysMotion: Physics-Grounded Dynamics From a Single Image. *arXiv preprint arXiv:2411.17189*, November 2024.

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 *ECCV*, volume 15062, pp. 1–18. Springer Nature Switzerland, February 2024.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IROS*, pp. 5026–5033. IEEE, 2012.

Chen Wang, Chuhao Chen, Yiming Huang, Zhiyang Dou, Yuan Liu, Jiatao Gu, and Lingjie Liu. PhysCtrl: Generative Physics for Controllable and Physics-Grounded Video Generation. *arXiv preprint arXiv:2509.20358*, November 2025a.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual Geometry Grounded Transformer. In *Proc. CVPR*, pp. 5294–5306. IEEE, June 2025b.

Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *Proc. CVPR*, pp. 5261–5271, 2025c.

Lilian Welschinger, Yilin Liu, Zican Wang, and Niloy Mitra. Learning to solve pdes on neural shape representations. *arXiv preprint arXiv:2512.21311*, 2025.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. PhysGaussian: Physics-Integrated 3D Gaussians for Generative Dynamics. In *Proc. CVPR*, pp. 4389–4398. IEEE, June 2024.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum. Clevrer: Collision events for video representation and reasoning. *arXiv preprint arXiv:1910.01442*, 2019.

Jiahao Zhan, Zizhang Li, Hong-Xing Yu, and Jiajun Wu. PerpetualWonder: Long-Horizon Action-Conditioned 4D Scene Generation. *arXiv preprint arXiv:2602.04876*, February 2026.

Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y. Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T. Freeman. PhysDreamer: Physics-Based Interaction with 3D Objects via Video Generation. In *ECCV*, volume 15060, pp. 388–406. Springer Nature Switzerland, 2025.

Licheng Zhong, Hong-Xing Yu, Jiajun Wu, and Yunzhu Li. Reconstruction and Simulation of Elastic Objects with Spring-Mass 3D Gaussians. In *ECCV*, volume 15060, pp. 407–423. Springer Nature Switzerland, July 2024.

Siyuan Zhou, Hejun Wang, Hu Cheng, et al. Physinone: Visual physics learning and reasoning in one suite. In *Proc. CVPR*, 2026.

Zeshun Zong, Xuan Li, Minchen Li, Maurizio M. Chiaramonte, Wojciech Matusik, Eitan Grinspun, Kevin Carlberg, Chenfanfu Jiang, and Peter Yichen Chen. Neural stress fields for reduced-order elastoplasticity and fracture. In *SIGGRAPH Asia Conference*. Association for Computing Machinery, 2023.

A IMPLEMENTATION DETAILS

A.1 VOXEL INTERPOLATION

Let $z^{(i)}_{t,u,v} \in \mathbb{R}^{1408}$ denote the V-JEPA feature of patch (u, v) in view i at time t. Using the estimated depth and camera parameters, we unproject each patch to a 3D location $P^{(i)}_{t,u,v}$ and aggregate its feature into a fixed voxel grid. The grid bounds are determined from the initial observations and kept unchanged during prediction. The current implementation uses one camera per sample and expresses the grid in that camera's coordinate system.

Each point contributes to its eight neighbouring grid locations using trilinear weights. For voxel j, the aggregated feature is

$$H_t(j) = \frac{\sum_{i,u,v} w_j(P^{(i)}_{t,u,v})\, z^{(i)}_{t,u,v}}{\max\Big(\sum_{i,u,v} w_j(P^{(i)}_{t,u,v}), \delta\Big)}, \tag{5}$$

where w_j is the trilinear weight and δ is a small numerical constant. The sum runs over the views used for the sample; in our single-camera setup, there is only one view. Voxels with no contribution have zero features.

We store the full voxel state as $Z_t = [H_t, O_t, U_t]$, where O_t denotes occupancy and U_t denotes unobserved or occluded space. The prepared geometry targets are binary and mutually exclusive. Predicted geometry is continuous, with $O_t \geq 0$ and $U_t \geq 0$.

A.2 FLOW-MATCHING OBJECTIVE

We model future voxel states using conditional flow matching. Each state $Z_t = [H_t, O_t, U_t]$ contains voxel features, occupancy, and unseen-space values. Given the clean future state Z_{t+1}, we sample Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ and an interpolation time $\tau \sim \mathcal{U}(0, 1)$, and construct:

$$Z_\tau = (1 - \tau)Z_{t+1} + \tau\epsilon. \tag{6}$$

Here t indexes physical time, while τ controls the interpolation between the clean state and noise. The target velocity is:

$$v^\star = \epsilon - Z_{t+1}. \tag{7}$$

Conditioned on the available history, force inputs $u_{1:t+1}$, and physical time interval Δt, the predictor estimates:

$$v_\theta = f_\theta(Z_{1:t}, Z_\tau, u_{1:t+1}, \Delta t, \tau)\,. \tag{8}$$

We train with the squared velocity error:

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}\Big[\|v_\theta - (\epsilon - Z_{t+1})\|_2^2\Big]\,, \tag{9}$$

where the expectation is over training sequences, noise, and interpolation times. The loss includes the feature, occupancy, and unseen channels, so all three are learned as continuous velocities.

At inference, we initialize the future state with Gaussian noise and integrate the predicted velocity from $\tau = 1$ to $\tau = 0$, keeping the conditioning inputs fixed. After integration, we constrain the geometry channels to $O \geq 0$ and $U \geq 0$. The generated state is then appended to the history to predict the next timestep.

A.3 LATENT PROJECTION

Given a generated voxel state $\hat{Z}_{t+1} = [\hat{H}_{t+1}, \hat{O}_{t+1}, \hat{U}_{t+1}]$, we render a 2D latent map for camera $\Pi^{(i)}$. We use hard first-hit projection for final rendering and soft projection for the training loss. Each output location corresponds to a latent patch.

Ray-based depth projection. For final rendering, we cast one perspective ray per latent patch:

$$\mathbf{q}_{u,v}(d) = \mathbf{c} + d\left(\frac{u - c_x}{f_x}\mathbf{r} + \frac{c_y - v}{f_y}\mathbf{u} + \mathbf{f}\right), \tag{10}$$

where $\mathbf{c}$ is the camera centre, $\mathbf{r}, \mathbf{u}, \mathbf{f}$ are its right, up, and forward axes, and the intrinsics are scaled to latent-grid resolution. d denotes camera-forward depth. We sample N uniformly spaced depths between d_{near} and d_{far}, the minimum and maximum forward depths of the bounding-box corners, with d_{near} clamped to zero. An empty interval produces no hits.

Let j_k be the nearest voxel-grid location to sample $\mathbf{q}_{u,v}(d_k)$. We select the first in-bounds sample whose occupancy meets the threshold:

$$k^\star = \min\{k : \mathbf{q}_{u,v}(d_k) \text{ is in bounds},\ \hat{O}_{t+1}(j_k) \geq \theta_{\text{occ}}\}, \qquad \hat{z}^{(i)}_{t+1}(u,v) = \hat{H}_{t+1}(j_{k^\star}). \tag{11}$$

The selected depth is $d_{k^\star}$. Rays with no hit have zero projected features and are marked invalid. This discrete visibility selection is used without gradient tracking.

Soft latent projection and projection loss For training, we replace the hard visibility selection with front-to-back compositing. Projection supervision is applied to the estimated clean endpoint

$$\bar{Z}_{t+1} = Z_\tau - \tau v_\theta. \tag{12}$$

For each camera ray, we intersect the ray with the fixed bounding box and sample the intersecting interval at the midpoints of 32 equal segments. Rays that miss the volume have zero opacity and zero projected features.

For a given ray, let h_k and o_k be the features and occupancy obtained by trilinear interpolation at sample k, ordered from near to far. We convert occupancy to segment opacity as:

$$\alpha_k = 1 - (1 - o_k)^{\Delta s_k/\ell}, \qquad T_k = \prod_{m<k} (1 - \alpha_m), \qquad w_k = T_k \alpha_k, \tag{13}$$

where Δs_k is the segment length and ℓ is the minimum voxel spacing. Occupancy is capped slightly below one for numerical stability. This length correction avoids increasing opacity solely by subdividing a segment with constant occupancy. Out-of-bounds samples contribute zero opacity. The projected features and accumulated opacity are

$$\hat{z} = \frac{\sum_k w_k H_k}{\max(\hat{M}, \delta)}, \qquad \hat{M} = \sum_k w_k, \tag{14}$$

where δ is a small numerical constant. Normalizing the feature weights avoids attenuating feature magnitudes on partially opaque rays; the opacity map itself is not normalized. The unseen channel does not contribute directly to opacity or feature compositing.

Let h_p denote the reference feature encoded from the future RGB video and transformed into the same PCA space, where p indexes the rendered patches. Let $M_p \in [0,1]$ be the fraction of valid depth observations in that patch and temporal tubelet, set to zero when the pooled 3D point lies outside the fixed volume. This is a depth-validity target, not an object segmentation mask. We use

$$\mathcal{L}_{\text{featproj}} = \frac{\sum_p M_p \|\hat{z}_p - z_p\|_2}{D\left(\sum_p M_p + \delta\right)}, \qquad \mathcal{L}_{\text{occproj}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} (\hat{M}_p - M_p)^2, \tag{15}$$

where D is the feature dimension and $\mathcal{P}$ contains all rendered patches, including those with zero reference validity. For each predicted frame r, the loss combines the volumetric velocity losses with the two projection terms:

$$\mathcal{L}_r = \mathcal{L}_{\text{feat},r} + \lambda_O \mathcal{L}_{O,r} + \lambda_U \mathcal{L}_{U,r} + \lambda_{\text{fp}} \mathcal{L}_{\text{featproj},r} + \lambda_{\text{op}} \mathcal{L}_{\text{occproj},r}. \tag{16}$$

The full training objective averages $\mathcal{L}_r$ over the predicted frames and takes the expectation over training windows and sampled noise, as specified in Eq. (4). We use $\lambda_{\text{fp}} = \lambda_{\text{op}} = 1$.

A.4 Model Architecture and Training Details

We use a spatial–temporal transformer with five blocks and a hidden dimension of 128. Each active voxel of the 25^3 grid contains 1408 feature channels and two binary occupancy and unseen channels. We select occupied locations from the context and expand this set with two iterations of a $3 \times 3 \times 3$ neighbourhood dilation. A shared linear layer maps the resulting 1410-channel voxel vectors to

128-dimensional tokens. The same spatial locations are used across the context and noisy target frames.

We add spatial and temporal positional embeddings and an embedding of the force associated with each frame. Force and physical-time condition tokens are available to the spatial attention, while the noise level τ controls adaptive LayerNorm. Each block applies local spatial attention over a $5 \times 5 \times 5$ neighbourhood, a feed-forward network with hidden dimension 512 and GELU, and temporal self-attention at each voxel location. Attention uses eight heads, and the feed-forward hidden dimension is four times the token dimension. In our model, temporal attention includes the context and noisy target with a causal mask.

A small convolutional branch uses the latest history state concatenated with its difference from the preceding state. A $1 \times 1 \times 1$ convolution first maps these inputs to hidden features at each voxel. Learned linear projections of the force inputs, physical prediction interval, and noise levels of the two history states are added to these features and shared across spatial locations. A depthwise $3 \times 3 \times 3$ convolution then gathers local spatial information independently within each hidden channel. SiLU activations are applied before and after this spatial convolution, followed by a $1 \times 1 \times 1$ output projection to the joint state dimensions. We add this branch's output to the transformer's output to predict a joint velocity field for voxel features, occupancy, and unseen space.

Starting from two observed latent states, we train over 8 prediction steps. At each step, a state generated by integrating from fresh noise is appended to the history, with gradients retained through the rollout. The separate endpoint estimate obtained from a GT-interpolated noisy target is used only for supervision, including the soft projection losses; it is not used as generated history.

We also corrupt history states at frame-dependent noise levels and provide those levels explicitly to the model. Active voxel regions are determined from the uncorrupted observed or generated history. At inference, initial observations remain clean, while generated history can receive the small corruption used by the stabilized sampling configuration. We use AdamW with learning rate 3×10^{-4}, weight decay 10^{-4}, and a ten-update linear warm-up followed by a constant learning rate. Batch size is one, with activation checkpointing and bfloat16 model computation; state updates and losses remain in float32.

A.5 Dataset and testing

We generated 1000 sequences of multiple simple rigid objects in a room bouncing and colliding with MuJoCo (Todorov et al., 2012) of 3–6 rigid objects (spheres, cubes, and cylinders) interacting in a walled 8×5×3 m room, simulated at 30 fps with a 2 ms physics timestep. Each scene contains a 2.5 s active rollout (38 latent frames at 30 fps), during which a randomized 3D impulse force is applied to either a single target object (local scope, 50 %) or all objects simultaneously (global scope, 50 %). Objects vary in size, mass, color, and initial airborne state (1–3 objects are in flight at the start). Each sequence is rendered from 5 fixed cameras at 480×480 resolution. For the PhysInOne (Zhou et al., 2026) dataset, we only found the validation dataset with 200 videos from the publicly available PhysInOne repository, each rendered from 12 stationary cameras at 1120×1120 and 30 fps with per-frame ground-truth depth. Each trajectory spans about 150 frames (5 s) and involves two or three simultaneously active physics phenomena drawn from a large vocabulary, including rigid-body collisions, liquid dynamics, and wind-driven motion. The PhysGaia (Kim et al., 2026) benchmark covers four continuum-material categories in 17 scenes: smoke/gas, fluid/liquid, cloth/textile, and viscoelastic solids. Each scene is rendered from 4 views at 720×960 and 24 fps; sequences are 240 frames (10s). No ground-truth depth is available, so depth is estimated with MoGe (Wang et al., 2025c). For all datasets, we used 5% of the data for testing, and specifically hand-picked the ones with fluid and smoke interactions.

We acknowledge that evaluating in V-JEPA space may favor methods whose representations are closer to V-JEPA-style features. However, we argue that latent-space evaluation provides a meaningful complementary criterion for video generation and prediction. A physically plausible generated video should not only match the target at the pixel level or satisfy high-level VLM judgments, but should also remain close to the target in a temporally structured video representation space. We therefore measure prediction error in V-JEPA feature space. V-JEPA is a natural choice because its embeddings are learned through predictive video objectives and are temporally aligned, allowing the metric to compare dynamic scene evolution rather than isolated frame appearance.

Table 4: Latent prediction results under different camera-depth evaluation protocols. Each entry reports **2D / 3D latent L2 loss** (↓). Columns specify the complete evaluation condition: single-camera or multi-camera prediction with either ground-truth or estimated depth. N/A indicates that the dataset does not provide ground-truth depth, when available. The GT depth columns for CLEVRER are tested with our synthetic dataset, see Appendix A.

Data	Method	Single-camera		Multi-camera	
		GT depth	estimated depth	GT depth	estimated depth
CLEVRER (Yi et al., 2019)	CogVideoX	2.98 ± 0.68 / 0.22 ± 0.12	2.98 ± 0.68 / 0.56 ± 0.13	3.87 ± 0.58 / 1.50 ± 0.11	4.14 ± 0.49 / 1.18 ± 0.07
	PhysGen	1.91 ± 0.08 / 0.17 ± 0.02	2.97 ± 0.19 / 0.56 ± 0.03	3.57 ± 0.17 / 1.89 ± 0.11	4.12 ± 0.49 / 1.24 ± 0.19
	PhysGaussian	4.13 ± 0.08 / 0.56 ± 0.01	4.39 ± 0.08 / 0.55 ± 0.03	4.34 ± 0.09 / 1.18 ± 0.07	4.21 ± 0.09 / 1.14 ± 0.08
	PhysCtrl	2.96 ± 0.37 / 0.35 ± 0.05	2.84 ± 0.23 / 0.57 ± 0.03	3.32 ± 0.31 / 1.01 ± 0.07	3.21 ± 0.20 / 1.15 ± 0.08
	2D baseline	1.39 ± 0.28 / 0.32 ± 0.08	1.53 ± 0.71 / 0.42 ± 0.12	3.19 ± 0.37 / 1.20 ± 0.27	3.69 ± 0.59 / 1.10 ± 0.11
	Ours-GT	**0.98 ± 0.07 / 0.02 ± 0.00**	**1.01 ± 0.10** / 0.39 ± 0.01	**1.12 ± 0.09 / 0.82 ± 0.05**	2.51 ± 0.11 / 0.91 ± 0.05
	Ours-estimate	1.26 ± 0.13 / 0.25 ± 0.03	**1.01 ± 0.23 / 0.36 ± 0.04**	2.50 ± 0.15 / 0.99 ± 0.09	**2.25 ± 0.09 / 0.90 ± 0.09**
PhysInOne (Zhou et al., 2026)	CogVideoX	3.02 ± 0.32 / 0.53 ± 0.16	3.08 ± 0.33 / 0.75 ± 0.20	4.05 ± 0.31 / 1.01 ± 0.11	4.00 ± 0.39 / 1.00 ± 0.19
	PhysGen	3.47 ± 0.59 / 0.52 ± 0.15	3.61 ± 0.27 / 0.71 ± 0.19	3.98 ± 0.37 / 0.87 ± 0.20	4.12 ± 0.91 / 1.02 ± 0.49
	PhysGaussian	3.98 ± 0.22 / 1.02 ± 0.22	3.95 ± 0.89 / 0.89 ± 0.20	4.57 ± 0.21 / 0.90 ± 0.15	4.62 ± 0.23 / 1.04 ± 0.21
	PhysCtrl	4.24 ± 0.37 / 0.66 ± 0.31	4.23 ± 0.51 / 0.81 ± 0.21	4.57 ± 0.35 / 0.84 ± 0.17	4.34 ± 0.38 / 0.98 ± 0.21
	2D baseline	2.99 ± 0.13 / 0.55 ± 0.19	2.25 ± 0.23 / 0.82 ± 0.19	3.98 ± 0.11 / 0.99 ± 0.12	4.14 ± 0.38 / 1.00 ± 0.50
	Ours-GT	**0.43 ± 0.05 / 0.06 ± 0.01**	3.08 ± 0.17 / 0.63 ± 0.14	**2.62 ± 0.10 / 0.49 ± 0.07**	2.62 ± 0.17 / 0.51 ± 0.14
	Ours-estimate	3.02 ± 0.17 / 0.53 ± 0.14	**1.67 ± 0.13 / 0.42 ± 0.14**	3.51 ± 0.17 / 0.66 ± 0.14	**2.59 ± 0.21 / 0.49 ± 0.11**
PhysGaia (Kim et al., 2026)	CogVideoX	N/A	3.71 ± 0.72 / 0.96 ± 0.32	N/A	3.72 ± 0.72 / 1.57 ± 0.47
	PhysGen	N/A	4.89 ± 0.50 / 1.15 ± 0.32	N/A	5.23 ± 0.52 / 1.68 ± 0.54
	PhysGaussian	N/A	3.98 ± 1.16 / 0.96 ± 0.33	N/A	3.95 ± 1.21 / 1.58 ± 0.48
	PhysCtrl	N/A	4.25 ± 1.05 / 0.97 ± 0.32	N/A	4.25 ± 1.09 / 1.58 ± 0.48
	2D baseline	N/A	2.16 ± 0.92 / 0.95 ± 0.61	N/A	4.97 ± 0.99 / 1.47 ± 0.22
	Ours-GT	N/A	1.84 ± 0.78 / 0.47 ± 0.21	N/A	2.10 ± 0.50 / 1.15 ± 0.35
	Ours-estimate	N/A	**1.66 ± 0.50 / 0.41 ± 0.21**	N/A	**1.99 ± 0.51 / 1.00 ± 0.09**

B Additional experiments

B.1 Latent prediction

Generally, the multi-camera errors in Table 4 are higher than the single-camera ones because more angles are introduced to calculate loss, while the model and baselines' input is mostly single-camera. Leaving out multi-camera PhysGaussian, our model shows a comparatively small increase when moving from single-camera to multi-camera evaluation, indicating some cross-view consistency despite using single-camera information during training. CogVideoX and PhysGen are competitive in some single-camera settings, especially on simpler rigid-body scenes, e.g., CLEVRER. But both CogVideoX and PhysGen degrade when moving to multi-camera evaluation. This shows that although CogVideoX is trained to reduce implicit embedding loss, and PhysGen follows physical laws, they are 2D methods and would have a large drop in performance for 3D consistency. Ours-GT performed the best with the overall loss, but since the dynamics is trained on GT, the 3D feature loss is consistently higher than Ours-estimate.

B.2 Flow estimation

Experiment settings. We evaluate the predicted video latents against ground-truth trajectories by comparing their latent flow consistency. We compare both the 2D and 3D flow predictions. This tests dynamics by evaluating predicted video latents against ground-truth trajectories using latent flow consistency. For baseline methods, each video frame generated is encoded into feature space with the VJEPA encoder and projected into 3D space for 3D evaluation. For our models, we directly generate the latent according to past frames, and projected to 2D for 2D evaluations. Given two consecutive latent frames, we estimate a per-patch displacement field by normalizing and comparing the cosine distance within a local window in the next frame. For each trajectory, we compute consecutive flow pairs from both the predicted and ground-truth latent sequences, then measure the total error accumulated between them by calculating their displacement differences. A lower error indicates that the predicted video exhibits motion patterns, in 2D or 3D, more consistent with the ground truth. The updated flow evaluation is reported on CLEVRER, PhysGaia, PhysInOne, and the 30 zero-shot samples from the unseen Kubric MOVI-C dataset. For these experiments, we report Ours-Estimate, our longer-horizon model trained with diffusion forcing and history-guided diffusion. For 3D flow loss, we omit the 2D baseline as it requires an extra z channel that cannot be supplied.

Table 5: 2D latent-flow estimation error across the evaluation datasets. Lower values indicate greater consistency between the predicted and ground-truth 2D motion. Kubric MOVI-C is evaluated zero-shot. Best results are shown in bold and second-best results are underlined.

Method	CLEVRER ↓	PhysGaia ↓	PhysInOne ↓	Kubric MOVI-C ↓
CogVideoX	0.492 ± 0.031	0.824 ± 0.100	0.753 ± 0.065	0.667 ± 0.094
PhysGen	$\underline{0.315 \pm 0.022}$	0.737 ± 0.105	0.637 ± 0.110	0.738 ± 0.133
PhysGaussian	0.407 ± 0.007	0.682 ± 0.134	$\underline{0.405 \pm 0.083}$	0.836 ± 0.126
PhysCtrl	0.651 ± 0.017	0.778 ± 0.127	0.694 ± 0.084	0.633 ± 0.091
2D baseline	0.383 ± 0.020	$\underline{0.534 \pm 0.160}$	0.435 ± 0.055	$\underline{0.563 \pm 0.109}$
Ours-Estimate	$\mathbf{0.273 \pm 0.009}$	$\mathbf{0.464 \pm 0.118}$	$\mathbf{0.373 \pm 0.031}$	$\mathbf{0.488 \pm 0.151}$

Table 6: 3D latent-flow estimation error across the evaluation datasets. Lower values indicate greater consistency between the predicted and ground-truth 3D motion. Kubric MOVI-C is evaluated zero-shot. Best results are shown in bold and second-best results are underlined.

Method	CLEVRER ↓	PhysGaia ↓	PhysInOne ↓	Kubric MOVI-C ↓
CogVideoX	0.6188 ± 0.0682	0.7011 ± 0.0190	0.7627 ± 0.0679	0.7353 ± 0.0253
PhysGen	0.6827 ± 0.0250	0.7007 ± 0.0309	0.7480 ± 0.0700	0.7214 ± 0.0215
PhysGaussian	0.7277 ± 0.0061	0.7227 ± 0.0612	$\underline{0.7064 \pm 0.0996}$	0.7900 ± 0.0722
PhysCtrl	$\underline{0.5335 \pm 0.0966}$	$\underline{0.6325 \pm 0.0913}$	0.7113 ± 0.1048	$\underline{0.6909 \pm 0.0991}$
Ours-Estimate	$\mathbf{0.3564 \pm 0.0146}$	$\mathbf{0.5640 \pm 0.0445}$	$\mathbf{0.5370 \pm 0.0313}$	$\mathbf{0.5194 \pm 0.0520}$

Results. We report the mean 2D and 3D flow errors across the three evaluation datasets and the zero-shot Kubric MOVI-C dataset in Tables 5 and 6. Ours-Estimate obtains the lowest 2D and 3D flow error across all four datasets, including the unseen Kubric MOVI-C samples. The 2D baseline remains competitive in several 2D evaluations but is consistently outperformed by our 3D model, supporting the contribution of the 3D projection beyond optimizing the same latent-space objective. The zero-shot Kubric result further evaluates whether the learned dynamics transfer to scenes outside the training datasets. We also show a visualization of the trajectory from the optical flow head in Section 4.2 (not the per-patch displacement field) in Figure 4.

B.3 Occupancy evaluation

Experiment settings. In addition to flow, we directly evaluate the predicted 3D occupancy against the ground-truth occupied voxels. We report occupancy IoU as the primary geometric measure and the ratio between predicted and ground-truth occupied volume as an additional diagnostic. Higher IoU indicates better spatial agreement. A mass ratio closer to one is informative primarily for approximately closed, mass-conserving objects; it need not remain constant for phenomena such as smoke. Moreover, a mass ratio close to one does not imply that the occupied volume is placed correctly, so we interpret it jointly with occupancy IoU. We evaluate Ours-Estimate using occupancy thresholds of 0.4, 0.5, and 0.6. The same evaluation is performed on CLEVRER, PhysGaia, PhysInOne, and the zero-shot Kubric MOVI-C samples. For the baseline video generators, the occupancy is extracted from the 3D voxels that are unprojected using estimated depth.

Results. Ours-Estimate obtains substantially higher occupancy IoU than the baselines across all four datasets, including the unseen Kubric MOVI-C samples. The mass-ratio results should not be treated as a standalone ranking: a ratio close to one may coincide with incorrect spatial occupancy, while non-conservative phenomena such as smoke need not preserve total occupied volume. We therefore use occupancy IoU as the primary geometric measure and interpret mass ratio jointly with it.

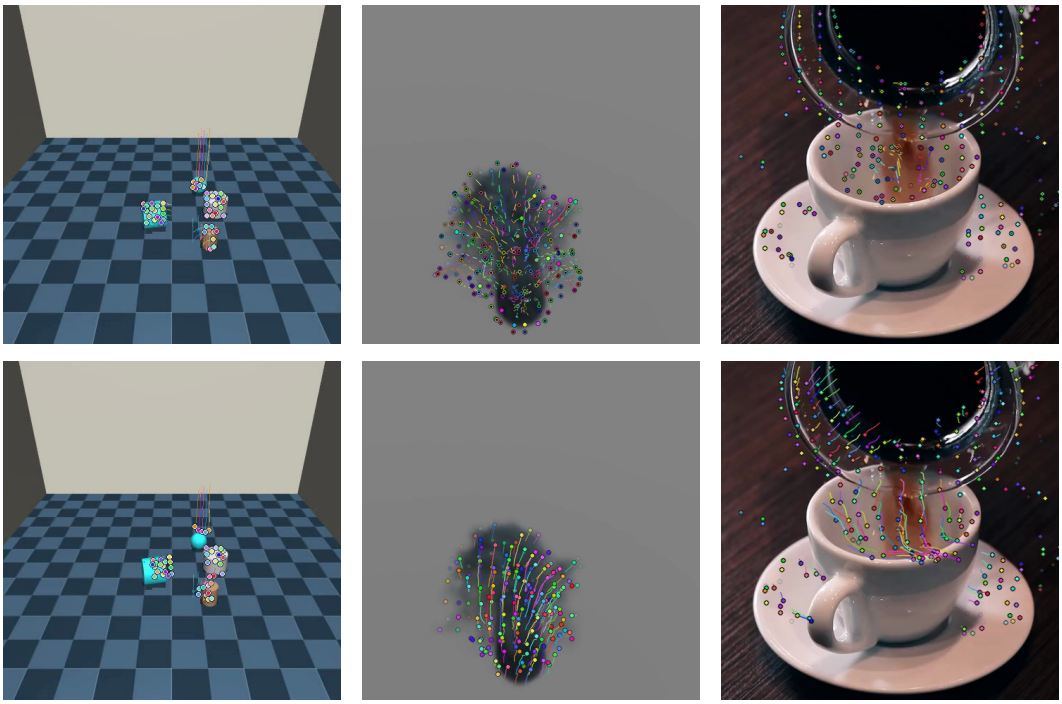


Figure 4: Qualitative comparison of 2D flow extraction in Section 4.2. Ground-truth flow is shown on the top, and our estimated flow on the bottom, with flow arrows overlaid on the V-JEPA features for visualization. Our voxel dynamics model is not trained with an additional 2D flow loss, so this comparison illustrates how the learned representation can be repurposed for motion estimation. See supplemental webpage for videos.

Table 7: Occupancy-based evaluation. Each entry reports occupancy IoU and mass ratio. Higher IoU is better, while a mass ratio closer to one is better. Our model is evaluated using three different occupancy thresholds. Best results are shown in bold and second-best results are underlined.

Method	CLEVRER: IoU ↑ / mass ratio ∼ 1	PhysGaia: IoU ↑ / mass ratio ∼ 1
CogVideoX	0.4168 ± 0.0962 / 1.1006 ± 0.0806	0.0846 ± 0.0536 / $\mathbf{0.7661 \pm 0.4417}$
PhysGen	0.5341 ± 0.1057 / $\mathbf{1.0165 \pm 0.0205}$	0.1197 ± 0.0767 / 1.3260 ± 0.8176
PhysGaussian	0.3450 ± 0.0058 / 0.8298 ± 0.0175	0.1076 ± 0.0903 / 0.6329 ± 0.7830
PhysCtrl	0.1478 ± 0.0197 / 0.6849 ± 0.0163	0.1129 ± 0.0489 / $\underline{0.6874 \pm 0.4790}$
Ours-Estimate (0.4 threshold)	$\underline{0.9119 \pm 0.0384}$ / $\underline{1.0789 \pm 0.0361}$	0.7059 ± 0.0797 / 1.3834 ± 0.1076
Ours-Estimate (0.5 threshold)	$\underline{0.9119 \pm 0.0384}$ / $\underline{1.0789 \pm 0.0361}$	$\underline{0.7061 \pm 0.0798}$ / 1.3827 ± 0.1076
Ours-Estimate (0.6 threshold)	$\mathbf{0.9120 \pm 0.0384}$ / $\underline{1.0789 \pm 0.0361}$	$\mathbf{0.7064 \pm 0.0798}$ / 1.3821 ± 0.1075

Method	PhysInOne: IoU ↑ / mass ratio ∼ 1	Kubric MOVI-C: IoU ↑ / mass ratio ∼ 1
CogVideoX	0.0880 ± 0.0183 / 0.2612 ± 0.0403	0.0835 ± 0.0634 / 0.5623 ± 0.3523
PhysGen	0.2155 ± 0.0212 / $\underline{1.2152 \pm 0.5123}$	0.4243 ± 0.0422 / $\mathbf{1.0233 \pm 0.9457}$
PhysGaussian	0.1151 ± 0.0147 / $\mathbf{0.7853 \pm 0.2199}$	0.2043 ± 0.0323 / 0.5352 ± 0.2334
PhysCtrl	0.1000 ± 0.0115 / 0.7008 ± 0.0458	0.1290 ± 0.0720 / $\underline{0.6982 \pm 0.9228}$
Ours-Estimate (DF), 0.4	0.7538 ± 0.0409 / 1.3089 ± 0.0643	0.5286 ± 0.0454 / 1.6276 ± 0.6049
Ours-Estimate (DF), 0.5	$\underline{0.7541 \pm 0.0408}$ / 1.3101 ± 0.0646	$\underline{0.5287 \pm 0.0454}$ / 1.6119 ± 0.5947
Ours-Estimate (DF), 0.6	$\mathbf{0.7543 \pm 0.0408}$ / 1.3095 ± 0.0644	$\mathbf{0.5288 \pm 0.0454}$ / 1.5951 ± 0.5843

Table 8: Physics IQ scores in four criteria.

Method	Spatiotemporal IoU ↑	Spatial IoU ↑	Weighted Spatial IoU ↑	MSE ↓	score ↑
CogVideoX	0.1782	0.3835	0.2273	0.0259	0.237
PhysGen	0.2110	0.2212	0.1199	0.0210	0.163
PhysGaussian	**0.2894**	0.3599	0.2161	0.0162	0.272
PhysCtrl	0.2725	0.3939	0.2504	0.0248	0.281
Wan-move (no tracks)	0.1373	**0.4648**	0.2521	0.0256	0.259
Ours-Estimate	0.2643	**0.5566**	**0.3752**	**0.0152**	**0.384**

B.4 RGB EVALUATIONS

B.4.1 PHYSICS IQ BENCHMARK

B.4.2 SIGNAL-TO-NOISE RATIO

In order to get back to pixels, we decode predicted latents to RGB in two ways: (i) a scene-specific lightweight V-JEPA-to-video head trained on the CLEVRER-style data, and (ii) a trajectory flow-conditioned video (Chu et al., 2025) generator driven by our predicted flow. We report PSNR against reference videos (Table 9); see supplemental for the qualitative videos. We omit FVD, as the number of decoded clips is too small for a meaningful distribution-level estimate.

Table 9: RGB reconstruction fidelity PSNR (dB) of decoded predictions.

Method	CLEVRER ↑	PhysInOne ↑	PhysGaia ↑	Kubric ↑
CogVideoX	16.40 ± 0.53	$\mathbf{20.46 \pm 4.07}$	16.96 ± 7.11	17.19 ± 3.00
PhysGen	24.71 ± 2.30	19.17 ± 3.39	16.45 ± 8.71	16.36 ± 1.48
PhysGaussian	25.87 ± 1.64	19.92 ± 3.52	15.99 ± 5.94	17.87 ± 3.11
PhysCtrl	18.21 ± 5.14	19.63 ± 2.19	16.80 ± 5.95	17.12 ± 3.04
2D baseline	26.42 ± 1.48	17.20 ± 1.36	17.02 ± 6.27	17.30 ± 1.67
Ours-Estimate (decoder)	$\mathbf{29.83 \pm 2.38}$	18.59 ± 1.34	$\mathbf{18.39 \pm 6.78}$	$\mathbf{17.89 \pm 2.54}$
Ours-Estimate (wan-move)	22.85 ± 2.62	14.14 ± 4.30	18.57 ± 3.83	15.00 ± 2.00

Results. Decoded predictions preserve object identity and motion, giving the best PSNR among compared methods by 5 dB. The video-from-flow path produces sharper textures at comparable motion accuracy, showing the predicted dynamics are precise enough to drive a larger high-fidelity generator.

B.5 ADDITIONAL ABLATIONS

Effect of depth or geometry estimators. In the development of our pipeline, we have tested different depth estimators for videos or multiple frames, and we show the results in Table 10. We have tested a static scene reconstruction model, VGGT (Wang et al., 2025b), the combination of depth estimator (Chen et al., 2025b) and camera estimator (Leroy et al., 2024), and a depth and camera estimator, MoGe (Wang et al., 2025c). The choice of estimator has a substantial impact on all metrics, showing that the lifting process is highly sensitive to the quality of the underlying geometric estimates. Once 2D features are lifted into 3D, geometric errors directly translate into incorrect voxel occupancy and degraded feature alignment. We found that MoGe performs best across all metrics, and thus used it for our main results.

Table 10: Effect of the depth estimator choice.

Method	**IoU % ↑**	**2D feat ↓**	**3D feat ↓**
VidDepA;Mast3r	38.64	3.19	1.76
VGGT	61.03	2.19	0.88
MoGe	87.13	1.78	0.61

Number of views. We test the effect of view coverage by varying both the number of input views in Table 11. We test our model on different numbers of input views at inference time, then compute the

IoU and feature loss for single-camera and multiple-camera settings. In the multi-camera evaluation setting, increasing the view coverage has a limited effect on IoU. We suspect that this is because, although the voxel construction gains higher quality for multiple camera angles as input (suggested in the single camera column), there are more entries for noise and loss aggregates; thus, the effects cancel out. This still supports the use of diverse camera viewpoints when constructing the voxel state, as they reduce occlusion ambiguity and encourage a more complete scene representation.

Table 11: Effect of the number of views and view-angle span on 2D/3D feature loss.

views	**Angle**	**multi-camera / single-camera**		
		IoU % ↑	**2D feat ↓**	**3D feat ↓**
1	$0°$	71.87 / 80.00	2.99 / 2.23	1.47 / 0.61
2	$< 45°$	71.07 / 87.62	2.82 / 2.22	1.48 / 0.53
max	$> 90°$	71.50 / 90.25	2.18 / 1.43	1.52 / 0.32

Quality vs computation for voxel grid size. We studied the effect of voxel grid resolution on reconstruction quality and training memory in Table 12. We show the 2D and 3D feature loss, in addition to the occupancy IoU. As expected, increasing the grid resolution consistently improves all quality metrics, indicating that the voxel representation benefits from finer spatial discretizations. But higher voxel resolution also introduces a clear memory-quality trade-off. We speculate that this gain in quality is due to the gaps between the V-JEPA latent patch size 30 and the grid size. Unfortunately, higher grid sizes cannot be tested due to limited memory.

Table 12: Trade-off between quality and computation under different voxel grid resolutions.

Voxel Grid Size	**Train Mem. ↓**	**IoU ↑**	**2D feature Loss ↓**	**3D feature Loss ↓**
10^3	1.44 GB	51.74%	2.75	1.60
15^3	4.70 GB	65.14%	2.63	1.55
20^3	11.06 GB	69.79%	2.02	0.99
25^3	23.45 GB	87.13%	1.78	0.61

Projection loss. We compare the model's performance *with* and *without* training with our projection loss in Table 13. The results show that the projection loss is a key component of the proposed representation learning objective. Although it is computationally expensive to add, it corresponds to a large improvement in both geometric overlap and feature consistency. Without this constraint, the model obtains a weaker voxel representation, suggesting that volumetric supervision from voxels from monocular views alone is insufficient for reliable feature reconstruction.

Table 13: Effect of the projection loss on reconstruction quality and memory usage.

Proj.	**Mem. ↓**	**IoU% ↑**	**2D feat ↓**	**3D feat ↓**
w/o	100%	69.47	3.24	1.75
w/	110%	87.13	1.78	0.61

Additional voxel channels. We also tested the importance of our occupancy channels in Table 14, under the monocular setting. The feature-only variant performs poorly, with a 2D feature loss of 4.97 and a 3D feature loss of 2.42. This is intuitive because without a depth/occupancy indicator, the whole voxel grid would be filled with active voxels; thus, the 3D features cannot learn accurate geometry information, and the projection would be blocked by the voxels nearest to the camera. Adding an `occupancy` channel substantially improves the representation. The further added `observed` channel indicates that parts of the unoccupied voxels might not be empty, but unseen. This further improves our model's performance considerably.

Table 14: Effect of choice of channels.

Voxel State	**# ch.**	**IoU% ↑**	**2D feat ↓**	**3D feat ↓**
Features only	1408	N/A	4.97	2.42
Features with occupancy channels	1409	67.52	2.96	1.02
Features, occupancy and observed channels	1410	87.13	1.78	0.61

Occupancy and occlusion losses. We ablate the occupancy and occlusion losses to measure their contribution to spatial consistency and motion prediction. All variants in this comparison are trained for 5,000 steps, and we report occupancy IoU together with 2D and 3D flow error. Removing either loss degrades the full model, while removing both produces the largest 2D-flow error. These results support the contribution of both terms to the predicted occupancy and motion.

Is a generative model necessary? In Table 15 we evaluate the importance of a generative model. Because of partial observation and lack of exact material parameters, our prediction task is indeed reasoning over a distribution. Hence, the generative model performs better.

Table 15: Deterministic regression vs. flow-matching variants (5,000 steps, 24-frame sequences). DF: diffusion forcing; HG: history guidance.

Method	Occupancy IoU ↑	2D flow error ↓	3D flow error ↓
Binary occupancy target	0.330	0.724	0.733
Deterministic regression	0.208	0.822	0.762
Conditional (DF)	0.572	0.700	**0.680**
Vanilla guidance (DF + HG)	**0.709**	**0.685**	**0.655**
Fractional guidance (DF + HG)	<u>0.660</u>	<u>0.687</u>	0.694

Table 16: Ablation of the occupancy and occlusion losses after 5,000 training steps. We report occupancy IoU together with 2D and 3D flow-estimation error.

Configuration	Occupancy IoU ↑	2D flow error ↓	3D flow error ↓
Full model	**0.709**	**0.685**	**0.655**
Without occupancy loss	0.572	0.854	<u>0.713</u>
Without occlusion loss	<u>0.580</u>	<u>0.837</u>	0.804
Without both losses	0.572	0.946	0.829

C Downstream tasks

C.1 Latent-to-video decoding

We render predicted dynamics using a trainable convolutional mapper followed by a frozen pretrained image VAE decoder. The mapper translates spatial V-JEPA features into VAE latents; rendering is deterministic and does not involve a diffusion denoising model. V-JEPA and the VAE remain frozen during mapper training.

Inputs and appearance conditioning. For camera view i, let $\hat{z}_{t+1}^{(i)} \in \mathbb{R}^{1408\times30\times30}$ denote the predicted feature map after projection into that view. Appearance conditioning uses only the first observed RGB frame of the corresponding view, resized to 240×240. Writing this image as $x_0^{(i)} \in [0,1]^{3\times240\times240}$, we compute a fixed anchor

$$a^{(i)} = s\,\mu_E(2x_0^{(i)} - 1), \qquad s = 0.18215, \tag{17}$$

where μ_E is the posterior mean of the frozen VAE encoder. We use the pretrained `stabilityai/sd-vae-ft-mse` AutoencoderKL. The anchor has shape $4 \times 30 \times 30$ and is

reused at every time step. Camera parameters enter the upstream feature projection, rather than the mapper itself.

Mapper and RGB decoding. The mapper concatenates the features and appearance anchor along channels:

$$\tilde{y}_{t+1}^{(i)} = g_\theta\Big([\hat{z}_{t+1}^{(i)}, a^{(i)}]\Big). \tag{18}$$

Its input therefore has 1412 channels. A 1×1 convolution maps these to 256 channels, followed by six residual blocks at unchanged spatial resolution. Each block applies GroupNorm with 32 groups, SiLU, a 3×3 convolution, GroupNorm, SiLU, and a second 3×3 convolution, with an identity skip connection. All convolutions in these blocks have 256 input and output channels, stride one, and padding one. The output head is GroupNorm, SiLU, and a 3×3 convolution producing four channels. The mapper has approximately 7.46 million trainable parameters and contains no temporal convolutions or attention layers. Its output is an absolute scaled VAE latent, not a residual added to the appearance anchor.

The frozen VAE decoder D converts the mapped latent directly to RGB:

$$\hat{x}_{t+1}^{(i)} = \mathrm{clip}_{[0,1]}\left(\frac{D(\tilde{y}_{t+1}^{(i)}/s) + 1}{2}\right). \tag{19}$$

Each latent state is decoded independently, yielding native 240×240 images. Larger exported videos are spatially resized from these images. Where the source frame rate exceeds the latent frame rate, our recent export pipeline linearly interpolates decoded RGB frames between latent timestamps and holds the last decoded frame through the remaining tail.

Mapper training. We train on paired cached V-JEPA maps and temporally aligned source RGB frames, resizing RGB targets to 240×240. The first-frame anchor is computed separately for each training view. Let $y_t = s\mu_E(2x_t - 1)$, let Δ denote a consecutive-frame difference, and define $\rho(u, v) = \mathrm{mean}\sqrt{(u-v)^2 + 10^{-6}}$. The selected base model uses

$$\begin{aligned}\mathcal{L} = \rho(\tilde{y}, y) + \rho(\hat{x}, x) + 0.5\ \mathrm{LPIPS}_{\mathrm{AlexNet}}(\hat{x}, x) \\ + 0.5\,\rho(\Delta\hat{x}, \Delta x) + 0.02\ \mathrm{TV}(\hat{x}),\end{aligned} \tag{20}$$

where TV is the sum of mean absolute horizontal and vertical image differences. LPIPS receives images normalized to $[-1, 1]$. We use batches of six two-state clips, AdamW with learning rate 2×10^{-4} and weight decay 10^{-4}, gradient-norm clipping at 1, and bfloat16 automatic mixed precision. Image losses backpropagate through the frozen VAE decoder into the mapper. We select checkpoints by validation PSNR; the deployed base checkpoint is from zero-indexed epoch 8.

Dataset-specific adaptation. For the PhysGaia-adapted checkpoint, we fine-tune the mapper for five epochs using the first eight latent states of the stationary training views and their cached RGB previews resized to the decoder resolution. We retain the base objective, use batch size four and learning rate 5×10^{-5}, and select on the corresponding validation split. For the liquid-transfer example, a separate adaptation uses 15 other fluid scenes for training and three for validation, excluding the target scene from both. It runs 400 steps with batches of two two-state clips and learning rate 5×10^{-5}. Its objective is VAE-latent MSE plus RGB L1 plus 0.25 times temporal-difference L1; selection uses validation RGB L1. Neither procedure trains a diffusion model, constructs a single-image augmentation bank, nor applies token dropout.

C.2 Motion-guided video generation

We also use the predicted latent dynamics to guide video generation with Wan-Move Chu et al. (2025). The pipeline has three steps: we estimate optical flow from consecutive latent maps, turn the flow into point trajectories, and use these trajectories to guide Wan-Move. We give Wan-Move an initial RGB image, a text prompt, and the trajectory coordinates and visibility masks.

Video features and latent-to-flow mapping. For real videos, we resize pairs of consecutive RGB frames to 480×480, apply ImageNet normalization, and encode each pair with frozen V-JEPA. This gives a latent map $z_k \in \mathbb{R}^{1408 \times 30 \times 30}$, which we assign to the second frame of the pair.

The flow head first applies LayerNorm across the channels of each spatial token, without learned scale or bias. It then uses a shared 1×1 convolution to mix and reduce the 1408 channels to 48, followed by GELU and a 3×3 convolution with 48 channels. Let $a_k = P(z_k)$ denote these compressed features. The head concatenates $[a_k, a_{k+1}, a_{k+1} - a_k]$, giving 144 channels, and applies a 3×3 convolution to 64 channels, eight-group GroupNorm, GELU, and two residual blocks. Each block applies a 3×3 convolution, eight-group GroupNorm, GELU, and another 3×3 convolution, then adds its input and applies GELU. A final 3×3 convolution produces 32 channels, which PixelShuffle with a factor of four rearranges into a two-channel 120×120 displacement field. All 3×3 convolutions use stride one and padding one. The selected checkpoints use only the two consecutive feature maps and their difference. The output values are in token units, so we multiply them by four to express displacement in pixels on the flow grid. Each displacement covers the interval between two latent timestamps.

Optical reference and trajectory integration. We use OpenCV DIS to estimate optical flow from adjacent RGB frames as an apparent-motion reference. We select moving points across a 30×30 grid using forward–backward consistency and image evidence, and keep at most 150 tracks for the sparse videos. All compared methods start from the same points. We build trajectories by sampling each flow field at the current point location:

$$p_{k+1} = p_k + \text{bilinear}(u_k, p_k). \tag{21}$$

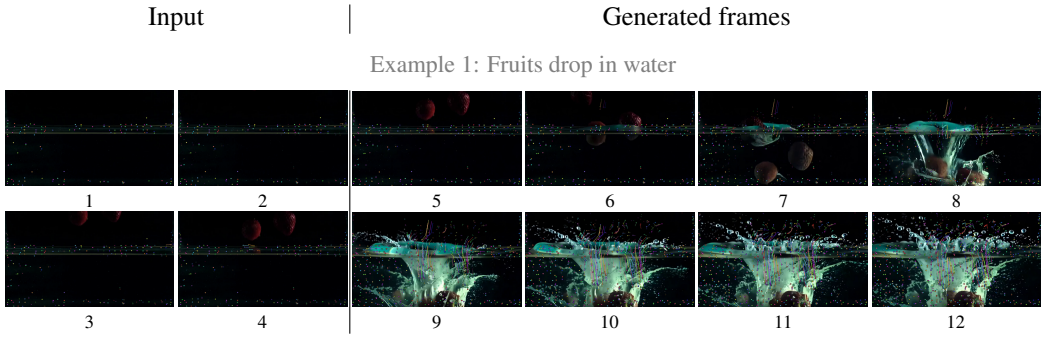
We scale coordinates consistently between the flow grid and image, interpolate positions between latent timestamps, and stop tracks that leave the image or lose valid support.

Wan-Move conditioning and generation. We give Wan-Move the initial RGB image, a short essential text prompt for it to understand the scene, and the selected trajectories with visibility masks. We keep only tracks that are valid at the start of the clip and resample them to 81 timestamps. For the real-video examples, this stretches the selected motion over the output duration. The multiview pipeline also supports preserving the original timing and holding the final positions for the remaining frames.

We use the pretrained Wan-Move 14B 480P checkpoint without further fine-tuning, generating 81 frames at 16 fps with 40 UniPC steps, schedule shift 3, guidance scale 5, and seed 42. Output dimensions follow the input aspect ratio within a 480×832 pixel area budget. These videos show how the estimated motion can guide generation; they do not directly measure the accuracy of the predictor or flow head.

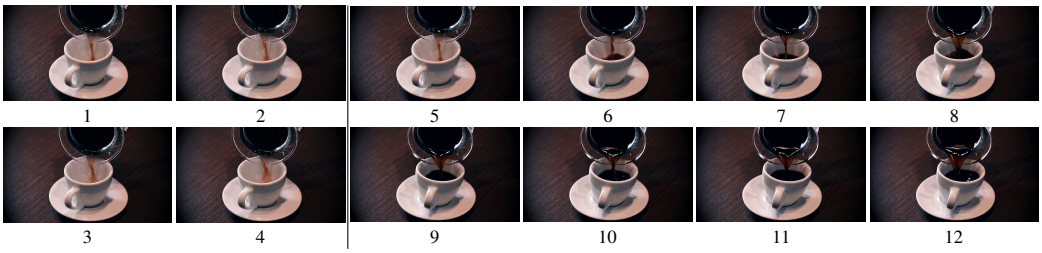
Results The video results are shown in Figure 5. See supplemental webpage for accompanying videos.

Figure 5: Input frames and the corresponding generated sequences.

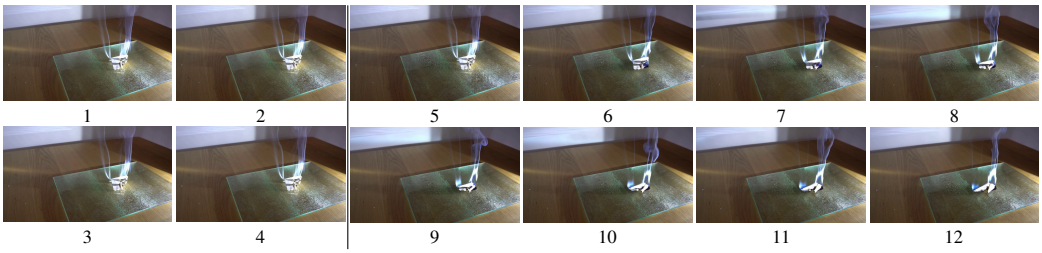


Input | Generated frames

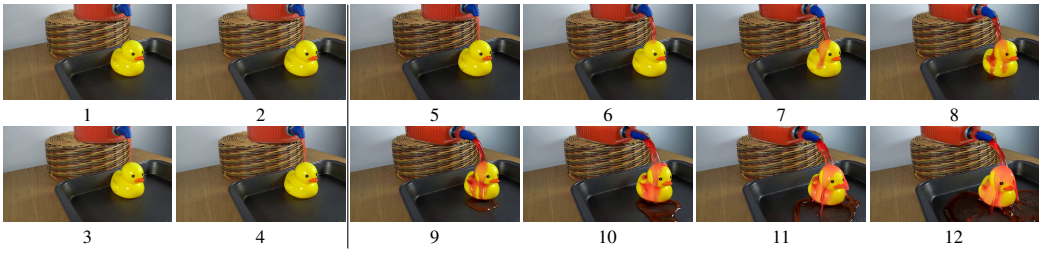
Example 2: Coffee pouring



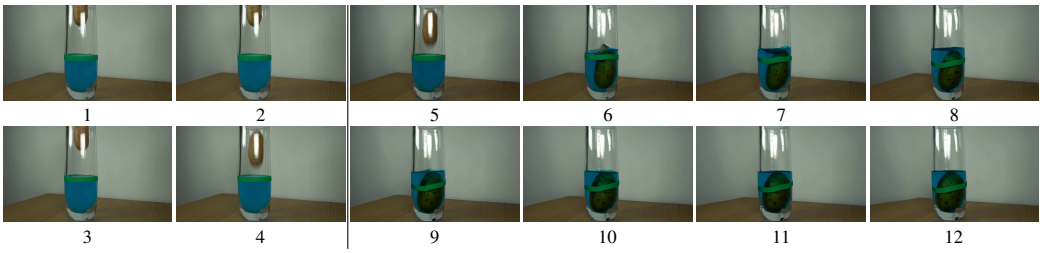
Example 3: Paper on fire



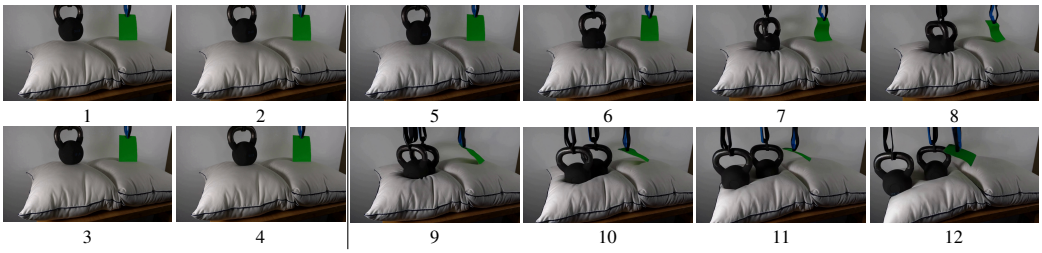
Example 4: Liquid with duck

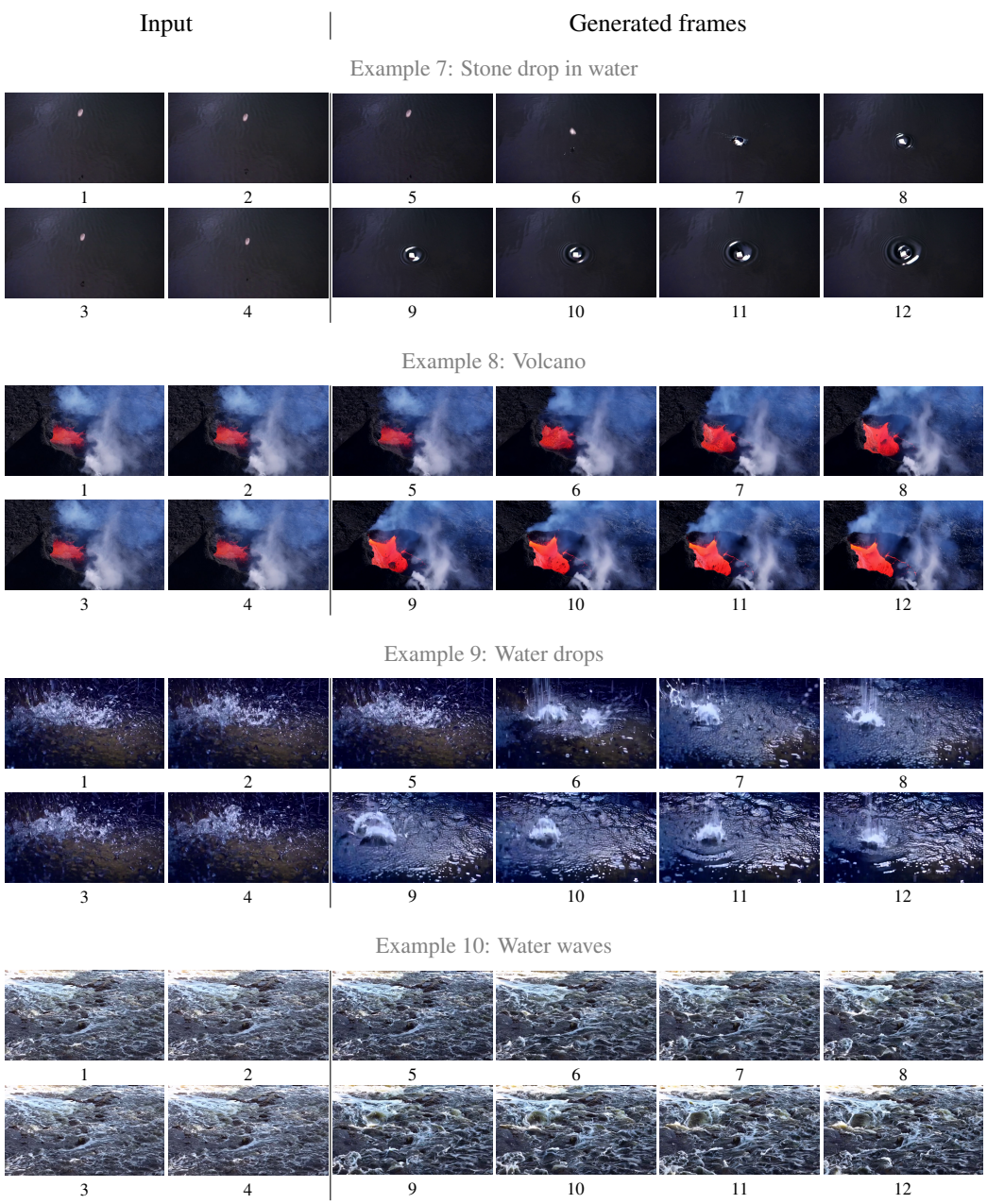


Example 5: Potato in water



Example 6: Weight on pillow





C.3 Joint training ability

Although our model is trained modularly, we can also adopt joint embedding learning, where the encoder latent is jointly trained with the predictor. However, this would require more data to generalize and reduce collapse, and would require much more compute. We hope that this is also one possible direction for future work.

C.4 Broader societal impact

Grounding generative video models in Euclidean space holds significant positive potential for physical planning, autonomous driving, and robotics, where structural and physical accuracy are paramount for safety. By simulating realistic physical outcomes from novel views, our method could accelerate safe, offline reinforcement learning. However, as with any high-fidelity video generation, there is

an inherent risk of misuse. Enhancing the physical realism of generated videos ironically increases their believability, potentially facilitating the creation of convincing misinformation or sophisticated deepfakes. Mitigating these risks will require the parallel development of robust video-forensic tools and watermarking techniques.