MOTIONCORE: A LATENT NEURAL SIMULATOR FOR VIDEO GENERATION WITH RIGID-BODY DYNAMICS

Anonymous authors
Paper under double-blind review

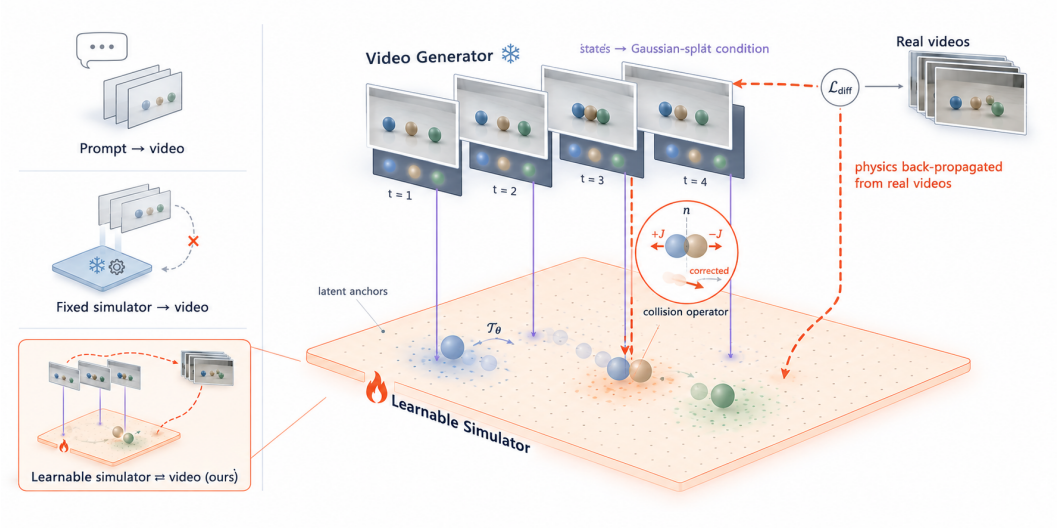


Figure 1: **Three ways to obtain physics in video generation.** Left: a video generator trained on prompts and videos alone must learn dynamics implicitly from pixels (top); a fixed simulator supplies explicit dynamics but receives no signal from the video (middle); MotionCore learns a simulator coupled to the video generator (bottom). Right: the learnable simulator advances object states in a latent space with the operator, and its collision operator applies equal and opposite impulses along the contact normal. The predicted states are rendered into a spatially aligned condition for the frozen video generator at every frame, and the diffusion loss on real videos isback-propagated into the simulator.

ABSTRACT

Video generation models can produce highly realistic videos while generating physically inconsistent object motion. This limitation suggests that visual modeling alone may not provide a transferable representation of physical dynamics. We introduce MotionCore, a framework for learning physical dynamics via a learnable partial differential equation solver and focuses on rigid-body motion. We introduce a latent neural simulators learning to predict the evolution of object-level physical states from observed motion. Its predictions are then converted into spatially aligned latent controls for a controllable video diffusion model. The generator serves as a visual interface: it provides appearance and scene detail, while the learned dynamics model specifies the physical trajectory. To preserve this trajectory through the interface, MotionCore combines a diffusion-matching objective with a low-capacity decodability constraint on object identity and position. We evaluate MotionCore on public physical-consistency benchmarks and on paired data with known trajectories. The latter setting measures obedience to a specified physical rollout, while the former evaluates broader physical and semantic consistency. Results indicate that MotionCore improves physical consistency and trajectory obedience in the evaluated rigid-body settings, while maintaining comparable visual quality. Ablation studies examine the roles of spatial alignment, temporal grouping, and decodability supervision.

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.

1 INTRODUCTION

Video generation models have made rapid progress in visual quality. They can synthesize detailed scenes, realistic appearances, and complex motion. Yet visual realism does not imply physical correctness. Generated objects may accelerate without a clear cause, pass through one another, or follow trajectories that conflict with collisions and gravity. This limitation is not restricted to older or small-scale models. Recent state-of-the-art systems, including Seedance and MiniMax Hailuo, can generate highly realistic videos but still struggle to maintain physically consistent object dynamics over time. Recent benchmarks, including VideoPhy, PhyGenBench, Physics-IQ, and VideoPhy-2, report a persistent gap between visual quality and physical consistency (Bansal et al., 2025; Meng et al., 2025; Motamed et al., 2026; Bansal et al., 2026). These observations suggest that a strong visual prior alone is insufficient for reliable physical prediction. More broadly, improving visual modeling capacity does not automatically provide a transferable representation of object dynamics.

A physical model should predict how a system evolves when its initial state, object properties, or interactions change. Hand-designed simulators provide an explicit way to model such evolution. They require, however, detailed descriptions of object geometry, materials, contacts, and governing rules. These descriptions are effective for predefined systems, but they are costly to construct and difficult to extend to the diversity of real-world scenes. A learned physical model offers a complementary approach. It can extract regularities from observations without requiring a complete symbolic description of every object and interaction. Such models may still fall short of high-fidelity simulation in the wild. They nevertheless provide a practical route toward learning physical dynamics from large-scale visual observations.

Videos are a natural source of such observations. They contain object trajectories, interactions, and changes in motion under diverse conditions. Many of these signals can be estimated from RGB frames and used to supervise dynamics learning. This suggests a modular view of video generation. A neural physical model learns how the observed system evolves. A pretrained video generator then acts as a visual tool that renders the predicted state with realistic appearance and scene detail. In this view, the generator does not need to discover all physical dynamics from appearance alone. It provides the visual interface through which a learned physical representation becomes an observable video.

In this paper, we introduce MotionCore, a framework for video-grounded rigid-body dynamics learning. MotionCore uses a latent neural simulator to model the evolution of object-level physical states. The neural simulator predicts future object states from observed motion and produces intermediate latent features that describe the evolving physical system.

MotionCore connects this learned dynamics model to a video diffusion model. A lightweight adapter converts the predicted states into spatially aligned control maps on the latent video grid. This design reduces the burden on the generator to decode a numerical token into a location in the image. The adapter also preserves the four-frame temporal grouping used by the video VAE. Motion within a compressed latent frame is therefore represented explicitly rather than inferred from a single aggregated feature. The generator remains fixed and provides appearance, texture, and visual detail, while the dynamics model provides the physical trajectory.

2 RELATED WORK

Neural operators in Latent Space. Neural operators learn maps between function spaces and can therefore transfer across spatial discretizations. DeepONet represents nonlinear operators with branch and trunk networks Lu et al. (2021), whereas the Fourier Neural Operator performs global kernel integration in the spectral domain Li et al. (2021). Latent formulations reduce the cost of evolving high-dimensional physical fields by learning a compact solution manifold or an encoder–operator–decoder factorization. NOMAD uses a nonlinear decoder for finite-dimensional solution manifolds Seidman et al. (2022). Latent Spectral Models and the Latent Neural Operator learn the principal transformation in a compact space Wu et al. (2023); Wang & Wang (2024), while latent DeepONet compresses both input and output functions before operator learning Kontolati et al. (2024). Geometry-aware variants map irregular meshes or particles to latent grids or tokens Li et al. (2023); Alkin et al. (2024), and AROMA retains spatially organized latent sequences and learns their temporal evolution with a conditional Transformer and a diffusion objective Serrano et al. (2024).

More recent models use state-space dynamics for long-range propagation Tiwari et al. (2025) or flow-matching autoregression for generative PDE evolution Koupaï et al. (2025). AROMA and ENMA are especially relevant because they combine compressed physical states with generative dynamics, but their outputs remain numerical PDE fields. MotionCore instead places a PDE-informed solution operator inside video latent denoising, using the operator state to guide the generation of detailed visual frames.

Simulator-based physical video generation. One line of work obtains motion from an explicit simulator and transfers it to a generative visual model. MotionCraft warps diffusion noise latents with optical flow computed from a physical simulation Montanaro et al. (2024). PhysGen infers scene properties, simulates rigid-body motion, and renders the result with video diffusion Liu et al. (2024), while PSIVG reconstructs a dynamic scene and places a simulator in the generation loop Foo et al. (2026). These pipelines provide interpretable control, but their behavior is coupled to reconstructed object states, simulator assumptions, or test-time optimization. A complementary line learns physical control or supervision from data. PISA combines simulated-video fine-tuning with reward-based post-training for free fall Li et al. (2025). PointVid regularizes generation with 3D point trajectories Chen et al. (2025), Force Prompting conditions videos on local or global force fields Gillman et al. (2025), and VLIPP converts vision-language reasoning into coarse motion plans Yang et al. (2025). VideoREPA instead distills pairwise spatiotemporal relations from a video representation model Zhang et al. (2025). These methods inject useful motion variables, plans, or preferences, but do not learn the governing field evolution as a reusable solution operator.

3 PRELIMINARIES

3.1 COMPACT REPRESENTATIONS OF PHYSICAL DYNAMICS

A physical trajectory records one realization of a dynamical system, whereas its governing equation describes an entire family of realizations. For a finite-dimensional state $\mathbf{x}(t) \in \mathbb{R}^d$, an ordinary differential equation (ODE) takes the form

$$\mathrm{d}\mathbf{x}(t)\backslash\mathrm{d}t = \mathbf{f}\big(\mathbf{x}(t);\phi\big), \quad \mathbf{x}(0) = \mathbf{x}_0, \tag{1}$$

where $\mathbf{f}$ specifies the instantaneous evolution and ϕ collects system attributes such as material parameters, external forces, and constraints. Changing $\mathbf{x}_0$ or ϕ produces a different trajectory without changing the underlying law $\mathbf{f}$. The ODE thus separates dynamics shared by a class of systems from the state of a particular instance.

If physical states are instead distributed over a spatial domain Ω, there produces another kind of governing equation. Let $\mathbf{s}(\boldsymbol{\xi}, t) \in \mathbb{R}^d$ denote such a field at location $\boldsymbol{\xi} \in \Omega$. Its evolution can be written as a partial differential equation (PDE) where $\mathcal{F}_\phi$ may contain spatial derivatives and interactions, and $\mathcal{B}_\phi$ specifies boundary conditions in Eq equation 2 where we take Dirichlet boundary condition for example:

$$\partial_t \mathbf{s}(\boldsymbol{\xi}, t) = \mathcal{F}_\phi[\mathbf{s}(\cdot, t)](\boldsymbol{\xi}), \quad \mathcal{B}_\phi[\mathbf{s}] = \mathbf{g}(\mathbf{s}, \mathbf{t}), \quad \mathbf{s}(\boldsymbol{\xi}, 0) = \mathbf{s}_0(\boldsymbol{\xi}), \tag{2}$$

An ODE is the corresponding evolution law for a finite set of state variables; conversely, spatial discretization turns a PDE into a coupled ODE system. We therefore treat PDEs and ODEs as a compact rule that maps the current state to its temporal change. Together with the initial and boundary conditions, this rule generates physical trajectories across configurations instead of encoding each trajectory separately. This matters for physical video generation. RGB frames entangle the underlying evolution with visual appearance. Governing equations concentrate the transferable part of the process in the physical state and its evolution rule. They consequently provide a more direct supervision target for learning dynamics than visual similarity between individual videos.

3.2 NEURAL OPERATOR

The distinction above changes what a dynamics model should learn. Rather than reconstructing one observed trajectory, the model should reuse a single evolution rule as the initial state, physical parameters, or boundary conditions change. Operator makes this precise by treating the quantities involved as functions (Lu et al., 2021; Li et al., 2021). An operator maps a function in one space

to a function in another, and a neural operator is a learnable operator. A neural operator composes layers that each act on functions. In its common form,

$$\mathcal{S}_\theta = Q \circ \sigma(W_L + \mathcal{K}_L) \circ \cdots \circ \sigma(W_1 + \mathcal{K}_1) \circ P \tag{3}$$

where $\mathcal{S}_\theta$ is the learned operator, P lifts the input pointwise to channels, W_ℓ is a pointwise linear map, $\mathcal{K}_\ell$ is a kernel integral operator with a learnable kernel κ_ℓ, σ is a pointwise nonlinearity, and Q projects back to the output channels. Time can be treated as an additional coordinate of the domain.

NOs have following properties that enable it to effectively learn mappings between functions. (i) Initial conditions, coefficients, and boundary data enter as inputs rather than as fixed settings, so one set of parameters represents a whole family of solutions. (ii) The same parameters apply across compatible discretizations: inputs can be sampled on different grids, outputs can be queried at any resolution, and predictions converge as the discretization is refined. This transfer concerns different samplings of the same instance, and extrapolation to unseen physical regimes remains a separate question. (iii) A trained NO evaluates a new instance in one forward pass, amortizing the cost of a numerical solve.

Evolving a full-resolution field at every step can nevertheless be expensive. Latent neural operators (Wang & Wang, 2024) reduce this cost by factorizing the operator into three parts: an encoder that maps the input to a compact latent representation, an operator that evolves this representation, and a decoder that returns the result to physical space. Because the decoder can be queried at arbitrary points, the latent form keeps the resolution independence of the second property while moving the costly evolution into a smaller space. Section 4 adopts this factorization for structured physical states and analyzes what the latent representation must preserve for the approximation to hold.

4 Latent Neural Simulator

4.1 Problem Setting

We study rigid-body video generation with physical prediction separated from visual rendering. A semantic condition $\mathbf{c}$, typically a text prompt with the first frame, specifies visual appearance and initial condition. A physical model receives an observed state history and predicts the future trajectory at the video timestamps T, and a video generator renders that trajectory. Motion is thus the responsibility of the physical model, and appearance that of the generator.

A rigid body has pose $\mathbf{q}_i \in SE(3)$ and spatial velocity $\boldsymbol{\nu}_i \in \mathbb{R}^6$. Smooth motion follows the Newton–Euler equations, an ODE in the sense of Eq. 1; and at contact the velocities jump. With rigid geometry and attributes fixed over the horizon, the motion is a hybrid system (Featherstone, 2008; Anitescu & Potra, 1997), which defines the solution operator

$$\mathcal{S}^\dagger : \mathcal{X} \times \Phi \longrightarrow \mathrm{D}\big([0,T]; \mathcal{X}^{(N)}\big), \tag{4}$$

where $\mathcal{X}$ is the state space, Φ hold the attributes, N represents the number of rigid bodies, the Skorokhod space D admits the velocity discontinuities caused by impacts. This is the coefficient-to-solution setting of operator learning in Section 3.2. Just as a permeability field determines the pressure of a Darcy flow (Li et al., 2021), a coefficient field and an initial state determine a rigid-body trajectory, and one operator covers the entire family of scenes. Learning $\mathcal{S}^\dagger$ therefore means learning the shared law of rigid-body motion rather than any single trajectory.

We learn the operator on an encoded state. An encoder maps the observed history to a latent code, an neural operator evolves the code, and a decoder returns the predicted trajectory (Wu et al., 2023; Wang & Wang, 2024). Because the token has a fixed size independent of the number and shape of objects, one operator can serve environments that an explicit simulator would have to set up one by one. Theorem 4.1 makes this precise: the error is controlled by how well the token preserves the evolution and by how simple the induced latent operator is.

4.2 Theoretical Boundary

Inspired by the function-space analysis of latent nonlinear operators between Hilbert spaces(Yang & Shi, 2026), we formulate the upper boundary of latent neural operator approximation. We model

the physical flow with a latent neural operator (Wang & Wang, 2024),

$$\mathcal{L}_N = \mathcal{D} \circ \mathcal{T}_N \circ \mathcal{E}, \quad \mathcal{L}_N(u) = \hat{\mathbf{s}}_{1:T}, \tag{5}$$

where $\mathcal{E} : \mathcal{U} \to \mathcal{Z}$ encodes the observed history, $\mathcal{T}_N : \mathcal{Z} \to \mathcal{Z}$ evolves the physics in a learned latent space, and $\mathcal{D} : \mathcal{Z} \to \mathcal{V}$ decodes the predicted evolution. We assume $\mathcal{U}$ and $\mathcal{V}$ are separable real Hilbert spaces. Learning physics in this formulation means minimizing approximation error to the PDE-induced flow map over the distribution of physical systems. The error has two sources: how well a finite network represents one latent transition which we denote as $\varepsilon_{\text{approx}} = \left\|\mathcal{S}^\dagger - \mathcal{D} \circ \mathcal{T}_N \circ \mathcal{E}\right\|_{L^q_\rho(\mathcal{V})}$., and how one-step errors accumulate over the rollout which is derived in A.4. We bound each in turn.

We assume that the ideal latent dynamics belongs to a variation space, whose norm quantifies the functional complexity of the latent transition and controls its finite-width approximation rate. The definition of this variation space is shown in A.2. Write $\overline{\mathcal{Z}} := \mathbb{R} \oplus \mathcal{Z}$ and $\bar{z} := (1, z)$, and let $B_{\overline{\mathcal{Z}}}$ be the closed unit ball of $\overline{\mathcal{Z}}$. A width-N latent ridge operator $\mathcal{T}_N(z) = \sum_{j=1}^N b_j\, \sigma(\langle \bar{z}, \bar{h}_j \rangle)$ with $\bar{h}_j \in B_{\overline{\mathcal{Z}}}$ is a finite instance of an infinite-width operator class; writing $\bar{h}_j = (\alpha_j, a_j)$ recovers the biased feature $b_j\, \sigma(\alpha_j + \langle a_j, z \rangle)$. Then we give the finite-width approximation boundary.

Theorem 4.1 (Finite-width approximation of latent dynamics)**.** *Let $2 \le q < \infty$, let $\sigma : \mathbb{R} \to \mathbb{R}$ be L_σ-Lipschitz, and let μ be a probability measure on $\mathcal{Z}$ with $\int_{\mathcal{Z}} (1 + \|z\|^2_{\mathcal{Z}})^{q/2}\, \mathrm{d}\mu(z) < \infty$. Define*

$$\Gamma_{\sigma,q}(\mu) := \Big(\int_{\mathcal{Z}} \big[|\sigma(0)| + L_\sigma (1 + \|z\|^2_{\mathcal{Z}})^{1/2} \big]^q \, \mathrm{d}\mu(z) \Big)^{1/q}. \tag{6}$$

If $\mathcal{T}^\dagger \in \mathbb{V}_{\text{lat}}(\mathcal{Z}; \mathcal{Z})$, then for every $N \ge 1$ and $\varepsilon > 0$ there is a width-N latent ridge operator $\mathcal{T}_N$ with $\sum_{j=1}^N \|b_j\|_{\mathcal{Z}} \le \|\mathcal{T}^\dagger\|_{\mathbb{V}_{\text{lat}}} + \varepsilon$ and

$$\|\mathcal{T}^\dagger - \mathcal{T}_N\|_{L^q_\mu(\mathcal{Z})} \le \frac{C_q\, \Gamma_{\sigma,q}(\mu)\, \big(\|\mathcal{T}^\dagger\|_{\mathbb{V}_{\text{lat}}} + \varepsilon \big)}{\sqrt{N}}, \tag{7}$$

where $C_q > 0$ depends only on q. If the variation norm of $\mathcal{T}^\dagger$ is attained by a representing measure, the bound holds with $\varepsilon = 0$.

The bound separates representation error from finite-width operator error. ε measures how much physics-relevant information can be preserved by the encoder–decoder pair. The other items measures the complexity of the resulting latent dynamics and decreases as $N^{-1/2}$. Consequently, compression is useful only when it both retains the target evolution and makes that evolution simpler to approximate. The guarantee requires neither an injective encoder nor an inverse map, and its constant has no explicit dependence on the latent dimension. It is a conditional approximation result: representation learning must still produce a small latent variation norm. Detailed assumptions, the proof, and useful corollaries are given in Appendix A.

5 Framework of MotionCore

5.1 Overview

MotionCore predicts a rigid-body scene frame by frame from one initial frame and renders the prediction with a video generator. The overall architecture is shown in Fig 2.

Scene encoder. The encoder turns a frame into the latent geometry state. It is an occupancy autoencoder in the style of 3DShape2VecSet (Zhang et al., 2023a) and Perceiver (Jaegle et al., 2021). Its encoder embeds points (x, y, z, t) sampled inside the occupied volume with Fourier features (Tancik et al., 2020). The decoder returns an occupancy logit at any query point (Mescheder et al., 2019). Applied to single frames, all points carry $t = 0$, and the rollout starts from $\widehat{\mathbf{z}}_0$.

Latent transition. The latent transition advances the geometry by one frame. The adapter $\mathcal{A}_{\text{c}}$ first writes the object states into the latent: the 256 slots cross-attend to one token per object. The latent neural operator $\mathcal{T}$ (Wang & Wang, 2024) then aggregates the slots into latent modes by physics cross-attention, evolves the modes with attention blocks, and maps them back.

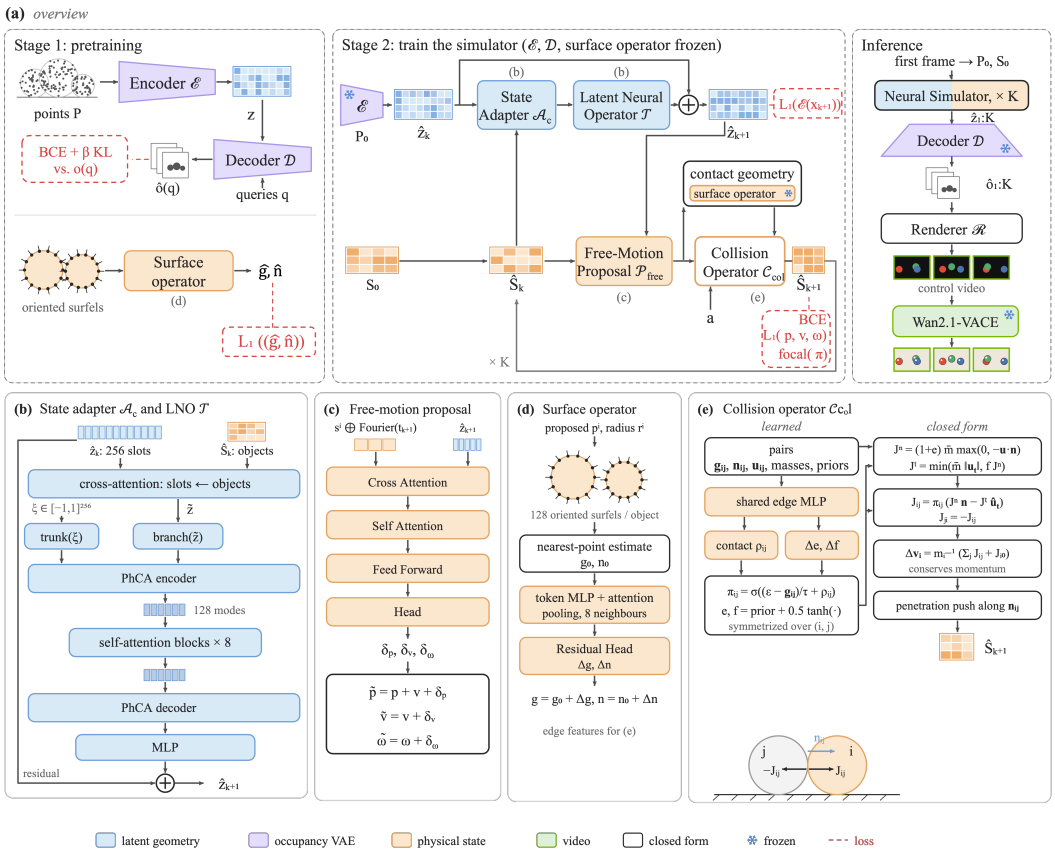


Figure 2: **Overview of MotionCore.** (a) Stage 1 pretrains the occupancy autoencoder and the surface operator. Stage 2 freezes them and trains the neural simulator on free-running rollouts of K frames. At inference the simulator rolls out the scene, and the decoded occupancy is rendered into a control video for the frozen Wan2.1-VACE. (b) State adapter and latent neural operator. (c) Free-motion proposal. (d) Surface operator. (e) Collision operator: learned contact probabilities and coefficients enter a closed-form impulse update. Snowflakes mark frozen modules, and dashed red boxes mark losses.

Free-motion proposal. The proposal predicts where each object moves if no contact occurs. Each object token, formed from $\mathbf{s}^i$ and a Fourier embedding of t_{k+1}, cross-attends to $\widehat{\mathbf{z}}_{k+1}$ and then to the other objects. A head outputs bounded residual corrections $\boldsymbol{\delta} = c\tanh(\cdot)$, and $\tilde{\mathbf{p}}^i = \mathbf{p}^i + \mathbf{v}^i + \boldsymbol{\delta}^i_{\mathbf{p}}$, $\tilde{\mathbf{v}}^i = \mathbf{v}^i + \boldsymbol{\delta}^i_{\mathbf{v}}$, $\tilde{\boldsymbol{\omega}}^i = \boldsymbol{\omega}^i + \boldsymbol{\delta}^i_{\boldsymbol{\omega}}$,.

Collision operator. The collision operator $\mathcal{C}_{\mathrm{col}}$ corrects the proposal at contacts. For the proposed positions, gaps g_{ij} and unit normals $\mathbf{n}_{ij}$ (from j to i) are computed for every object pair. A pretrained surface operator, frozen here, adds gap and normal estimates from oriented surface samples as edge features. With $\mathbf{u}_{ij} = \tilde{\mathbf{v}}^i - \tilde{\mathbf{v}}^j$, its tangential part $\mathbf{u}^{\mathrm{t}}_{ij}$, and $\bar{m}_{ij} = (m_i^{-1} + m_j^{-1})^{-1}$,

$$J^{\mathrm{n}}_{ij} = (1 + e_{ij})\,\bar{m}_{ij}\max\big(0, -\mathbf{u}_{ij}\cdot\mathbf{n}_{ij}\big), \quad J^{\mathrm{t}}_{ij} = \min\big(\bar{m}_{ij}\|\mathbf{u}^{\mathrm{t}}_{ij}\|,\, f_{ij}J^{\mathrm{n}}_{ij}\big),$$
$$\mathbf{J}_{ij} = \pi_{ij}\big(J^{\mathrm{n}}_{ij}\mathbf{n}_{ij} - J^{\mathrm{t}}_{ij}\hat{\mathbf{u}}^{\mathrm{t}}_{ij}\big), \quad \mathbf{v}^i = \tilde{\mathbf{v}}^i + m_i^{-1}\Big(\textstyle\sum_{j\neq i}\mathbf{J}_{ij} + \mathbf{J}_{i0}\Big), \tag{8}$$

The normal impulse reverses the closing speed with restitution e_{ij}, the tangential impulse stays in the Coulomb cone of friction f_{ij}, and equal and opposite impulses conserve linear momentum between bodies for any mass ratio. Overlapping pairs are separated along $\mathbf{n}_{ij}$, which resolves resting contacts without closing speed. An edge network predicts the contact probability $\pi_{ij} = \mathrm{sigmoid}\big((\epsilon - g_{ij})/\tau_{\mathrm{c}} + \rho_{ij}\big)$, a geometric logit plus a learned residual ρ_{ij}, and bounded corrections to e_{ij} and f_{ij} around priors from ϕ. Both are symmetrized over each pair, which keeps the exchange equal and

opposite. The corrected state re-enters the latent through $\mathcal{A}_{\mathrm{c}}$ at the next step, so a collision also changes the predicted geometry.

Video interface. The interface turns the predicted latents into a condition for the video generator. The renderer decodes $\widehat{\mathbf{z}}_k$ with the decoder of the neural simulator, projects the occupied geometry with the calibrated camera, separates it into instances matched across frames, and draws each in its own label colour.

5.2 Training

The modules are trained in stage-wise pattern, each frozen before the next, and the simulator is never trained through the video objective: with the physical loss active the diffusion gradient leaves it unchanged, and without it the simulator drifts toward more motion and nearly doubles its short-horizon error (Appendix C).

Stage 1 pretrains the surface operator on exact gap and normal targets, and the occupancy autoencoder on space–time segments with the binary cross-entropy loss (Mescheder et al., 2019).

Stage 2 freezes both and trains $\mathcal{A}_{\mathrm{c}}$, $\mathcal{T}$, $\mathcal{P}_{\mathrm{free}}$, and $\mathcal{C}_{\mathrm{col}}$ jointly on free-running rollouts of K frames from a single initial frame, with

$$\mathcal{L} = \frac{1}{K}\sum_{k=1}^{K}\big(\lambda_z\mathcal{L}_{\mathrm{lat},k} + \lambda_o\mathcal{L}_{\mathrm{occ},k} + \lambda_s\mathcal{L}_{\mathrm{state},k} + \lambda_c\mathcal{L}_{\mathrm{con},k}\big). \tag{9}$$

The latent term $\mathcal{L}_{\mathrm{lat}}$ is a smooth-L_1 loss between $\widehat{\mathbf{z}}_k$ and the posterior mean $\boldsymbol{\mu}_k$ of the ground-truth frame. The occupancy term $\mathcal{L}_{\mathrm{occ}}$ is a binary cross-entropy on query points (Mescheder et al., 2019). The state term $\mathcal{L}_{\mathrm{state}}$ applies the smooth-L_1 loss to the normalized position, velocity, and angular velocity, including the collision response. The contact term $\mathcal{L}_{\mathrm{con}}$ is a focal loss (Lin et al., 2017) on per-object, pairwise, and floor contact labels, which supervises π_{ij}. Because no step is teacher-forced, $\mathcal{L}$ scores the accumulated error described by Theorem A.4, not only single-step error.

Stage 3 needs no training for the rendered interface. The learned adapter $\mathcal{A}_\theta$ (6.7×10^4 parameters), which we study as an ablation for states without renderable geometry, splats object attributes and operator features at the projected centres and is trained with the flow-matching loss of the frozen generator on ground-truth states. Loss weights and optimization settings are in Appendix E.

6 Experiments

6.1 Setup

Data. The simulator is trained on rigid-body trajectories from MOVi-Spheres (Wei & Fink, 2025) and Billiards (Zhang et al., 2024). Their held-out scenes carry the state-space measurements of Section 6.1; Appendix E lists every split. For video generation we use two benchmarks: a subset of Physics-IQ-Verified (Motamed et al., 2026), a general benchmark of physical reasoning, from which we select the scenes whose prompts describe rigid-body motion such as collision, rolling, sliding, and free fall and finally get 81 five-second videos; and RigidBench (Jain & Wu, 2026), a public benchmark built for rigid-body dynamics, which provides 100 two-second videos.

Metrics. Physics-IQ (Motamed et al., 2026; Rädsch et al., 2026) extracts motion masks from both videos. ST-IoU measures whether motion occurs at the right place and time, and MSE measures pixel fidelity. The Physics-IQ score aggregates these to a 100-point scale. For RigidBench (Jain & Wu, 2026) we report per-object mask IoU, the mask-centroid error (L2) and the bidirectional Chamfer distance, both normalized by image height (Li et al., 2025). We also report the 2D trajectory error of tracked points (ATE), the 3D error of the reconstructed object positions (ATE-3D), and the scale-invariant depth error (SI-MSE).

Baselines. We choose general-purpose models of comparable scale CogVideoX-5B-I2V (Yang et al., 2024), HunyuanVideo-1.5-480P-I2V (Team, 2025), MiniMax-H3 (MiniMax, 2026), Wan2.2-TI2V-5B (Wan Team, 2025), Cosmos-Predict-2.5 2B (NVIDIA, 2026) to complete the quality axis. Baseline numbers are copied from the cited papers where benchmark, split, and backbone match, and re-run otherwise.

Implementation. Our generator is Wan 2.1-VACE-1.3B (Wan Team, 2025; Jiang et al., 2025) with base weights. Sampling uses 30 steps at 832×480 with classifier-free guidance 5.0 on the text and none on the physical condition; clips have 81 frames at 16 fps on Physics-IQ and 21 frames on MOVi-Spheres. The simulator is frozen after Stage 2, advances one block per step, and is decoded at every video timestamp. All experiments are conducted on 8 Nvidia H100 GPUs. Remaining details are in Appendix E.

6.2 Main Results

Table 1: **Physical plausibility on the two rigid-body benchmarks.** The Physics-IQ-Verified block uses the subset of scenes whose prompts describe rigid-body motion. — : not evaluated.

		Physics-IQ-Verified subset			RigidBench (Jain & Wu, 2026)					
Method	#Params	Score ↑	ST-IoU ↑	MSE ↓	IoU↑	L2↓	Cham.↓	ATE↓	ATE-3D↓	SI-MSE↓
CogVideoX-5B-I2V	5B	30.59	.297	.0046	.146	.222	.361	.186	2.63	.0349
HunyuanVideo1.5-480P-I2V	8.3B	30.08	.151	.0057	.170	.211	.345	.181	2.19	.0372
MiniMax-H3	33B	**39.36**	.236	.0048	.202	.198	.328	.186	2.00	.0375
Wan2.2-TI2V	5B	24.70	.165	.0057	.183	.220	.370	.200	2.64	.0288
Cosmos-Predict-2.5 2B	2B	27.05	.190	.0060	.171	.300	.527	.282	2.84	.0331
MotionCore	1.3B	26.2	.253	.0070	—	—	—	—	—	—

On the Physics-IQ-Verified subset, MotionCore reaches a score of 24.53 with a 1.3B generator (Table 1), on par with Wan2.2-TI2V-5B (24.70), a newer model of the same family as our backbone with about four times as many parameters, and above its own backbone without the physical condition (21.94). The models above it, Cosmos-Predict-2.5-2B (27.05), CogVideoX-5B (30.59), HunyuanVideo-1.5 (30.08), and MiniMax-H3 (39.36), are trained for physical AI or are 4 to 27 times larger. The generator in MotionCore is a 1.3B model, so appearance terms of the score, in particular MSE, are bounded by its rendering quality. The simulator changes where and when objects move, which the ST-IoU component measures: it rises from .189 for the backbone without the condition to .253, second only to CogVideoX-5B, while MSE improves from .0088 to .0070. On the 21 ball scenes, where every moving object is handed to the simulator, the score rises from 16.2 to 28.9 (Table 2).

6.3 Ablations

Table 2 asks three questions of the full model, changing one component per row.

Q1: is a simulator needed? Removing the simulator leaves the generator with the first frame and the prompt. The MOVi-Spheres hit rate falls from 0.910 to 0.087, and on the Physics-IQ ball scenes the score falls from 28.9 to 16.2 and the spatial IoU from .218 to .124. Without explicit dynamics the generator adds and removes objects instead of moving them (Figure 4 in Appendix D).

Q2: why a learnable simulator? We replace the simulator at inference by a hand-written rigid-body integrator (gravity, floor restitution, Coulomb friction with rolling, pairwise impulses) that receives the same inputs, including the ground-truth masses and materials on MOVi-Spheres and the same default values on Physics-IQ. MOVi-Spheres is itself produced by a rigid-body engine, so the integrator's states are closer to the ground truth than the simulator's (0.48 against 0.74 radii at 0.67 s). The videos conditioned on them are nevertheless slightly less accurate: per-scene hit rate 0.895 against 0.910, a paired difference of 0.015 ± 0.008 ($n = 24$). On real videos, where materials are unknown, the fixed integrator lowers the ball-scene score from 28.9 to 20.2 (paired -8.6 ± 3.6) and the spatial IoU from .218 to .118.

Q3: can the physics be edited, and must the generator change? Rotating every object's initial horizontal velocity by 90° inside the simulator changes the video accordingly (Figure 5 in Appendix D): the hit rate against the edited states is 0.943, as high as for unedited rollouts (0.906), while agreement with the unedited ground truth drops to 0.280. Keeping the generator frozen, i.e. without the low-rank update of its control branch, gives the same hit rate on MOVi-Spheres (0.910) but a lower Physics-IQ score (27.1 against 28.9) and spatial IoU (.187 against .218).

Table 2: **Three questions about the pipeline.** Each row changes one component of the full model and keeps the generator weights, first frame, prompt, and seed. MOVi-Spheres: 24 held-out scenes; hit rate within one object diameter against the ground truth (GT) and against the states fed to the generator (fed), ± standard error over scenes; simulator error is the centre error of the fed states at 0.67 s in object radii. Physics-IQ: the 21 ball scenes of the rigid-body subset, one seed, official scorer.

		MOVi-Spheres			Physics-IQ (ball scenes)		
	Variant	Hit (GT) ↑	Hit (fed) ↑	Sim. err. ↓	Score ↑	Spatial IoU ↑	W. spatial IoU ↑
	MotionCore (learnable simulator)	**0.910**±0.024	0.906	0.74	**28.9**	**.218**	**.115**
Q1	w/o simulator (first frame + text)	0.087±0.007	—	—	16.2	.124	.044
Q2	fixed simulator (hand-written integrator)	0.895±0.024	0.917	**0.48**	20.2	.118	.059
Q3	edited: initial velocity rotated by 90°	0.280±0.020	0.943	—	—	—	—
	frozen generator (no LoRA on the control branch)	0.910±0.018	0.938	0.74	27.1	.187	.086

Table 3: **Physics-to-video interface** with ground-truth states and the same frozen generator, scored by hit rate within one object diameter and median error in diameters.

Interface	Hit ↑	Median ↓
Tokens + cross-attention	0.52	1.05
Learned adapter	0.84	0.179
Rendered (ours)	**0.85**	**0.045**

Position must be carried by feature location. With the same ground-truth states and the same frozen generator, a tokenized state read through cross-attention reaches a hit rate of 0.52, while the two spatially aligned conditions reach 0.84 and 0.85. The rendered condition, which trains no parameters, also has the lowest median error at 0.045 diameters. Attempts to repair the token interface with absolute position codes, guidance on the physical condition, or a loss re-weighted towards objects do not raise its hit rate above 0.45, against 0.43 without repair . A condition read through cross-attention must be converted from a number into an image location inside the backbone, and the frozen backbone does not learn that conversion from a diffusion loss.

Stage-wise training and a condition for the whole clip. Training the simulator jointly with the diffusion loss does not help. The stage-wise pipeline leads the co-trained one by 0.054 ± 0.017 in per-scene hit rate ($n = 32$), and without the physical loss the diffusion gradient nearly doubles the simulator's short-horizon error, from 0.56 to 0.96 (Appendix C). The condition must also be present for the whole clip. On an 81-frame CLEVRER scene, supplying it throughout keeps the median position error at 0.083 scene units, whereas supplying it for the first five frames only lets the generator drift to 1.164, worse than the 0.687 of linear extrapolation .

7 Conclusion

We factorized physics-aware video generation into a dynamics term and an appearance term and assigned each to the model that can be supervised for it. MotionCore learns the dynamics term with a latent neural simulator trained in physical state space, and renders its rollout into a spatially aligned condition for a frozen video diffusion model, which supplies the appearance term without further training. However, the simulator is trained on rigid bodies, is accurate only on the support of its training distribution, and enforces no conservation law. The generator is frozen, so its visual prior is preserved but never adapted. Obedience is bounded by the simulator, because the generator renders faithfully what it is given, including physical error. Identity binding under a strong positional condition remains unsolved for every condition we tried. The conditions that perform best consume geometry such as size, mesh, and pose, which a state estimator must supply outside a simulator; on public benchmarks the accuracy of that estimator therefore enters every result.

AI USE STATEMENT

We used generative AI tools to assist with English-language polishing, code development and debugging, and discussions related to theoretical formulations and derivations. AI-generated suggestions were used only as research assistance; all mathematical claims, derivations, code, and experimental implementations were independently checked and verified by the authors. The authors take full responsibility for the final content of this work.

REFERENCES

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers: A framework for efficiently scaling neural operators. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0793.

Mihai Anitescu and Florian A. Potra. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics*, 14(3):231–247, 1997. doi: 10.1023/A:1008292328909.

Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. VideoPhy: Evaluating physical commonsense for video generation. In *International Conference on Learning Representations*, 2025.

Hritik Bansal, Clark Peng, Yonatan Bitton, Roman Goldenberg, Aditya Grover, and Kai-Wei Chang. VideoPhy-2: A challenging action-centric physical commonsense evaluation in video generation. In *International Conference on Learning Representations*, 2026.

Yunuo Chen, Junli Cao, Vidit Goel, Sergei Korolev, Chenfanfu Jiang, Jian Ren, Sergey Tulyakov, and Anil Kag. Towards physical understanding in video generation: A 3d point regularization approach. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3438.

Roy Featherstone. *Rigid Body Dynamics Algorithms*. Springer, New York, NY, 2008. doi: 10.1007/978-1-4899-7560-7.

Lin Geng Foo, Mark He Huang, Alexandros Lattas, Stylianos Moschoglou, Thabo Beeler, and Christian Theobalt. Physical simulator in-the-loop video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4301–4311, 2026.

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. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-3448.

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.

Andrew Jaegle, Felix Gimeno, Andy Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, 2021.

Swarnim Jain and Shangzhe Wu. Rigidbench: Evaluating rigid-body physics in video generation models, 2026.

Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. Vace: All-in-one video creation and editing. *arXiv preprint arXiv:2503.07598*, 2025.

Katiana Kontolati, Somdatta Goswami, George Em Karniadakis, and Michael D. Shields. Learning nonlinear operators in latent spaces for real-time predictions of complex dynamics in physical systems. *Nature Communications*, 15:5101, 2024. doi: 10.1038/s41467-024-49411-w.

Armand Kassaï Koupaï, Lise Le Boudec, Louis Serrano, and Patrick Gallinari. ENMA: Tokenwise autoregression for continuous neural PDE operators. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Chenyu Li, Oscar Michel, Xichen Pan, Sainan Liu, Mike Roberts, and Saining Xie. Pisa experiments: Exploring physics post-training for video diffusion models by watching stuff drop. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 35685–35709, 2025.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3d PDEs. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-1556.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision*, 2017.

Shaowei Liu, Zhongzheng Ren, Saurabh Gupta, and Shenlong Wang. PhysGen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision*, pp. 360–378, 2024.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3:218–229, 2021. doi: 10.1038/s42256-021-00302-5.

Fanqing Meng, Jiaqi Liao, Xinyu Tan, Wenqi Shao, Quanfeng Lu, Kaipeng Zhang, Yu Cheng, Dianqi Li, Yu Qiao, and Ping Luo. Towards world simulator: Crafting physical commonsense-based benchmark for video generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 43781–43806, 2025.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

MiniMax. Minimax h3. `https://github.com/MiniMax-AI/MiniMax-H3`, 2026. Open-weight omni-modal audio-video generation model. Accessed: 2026-09-25.

Antonio Montanaro, Luca Savant Aira, Emanuele Aiello, Diego Valsesia, and Enrico Magli. Motioncraft: Physics-based zero-shot video generation. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3914.

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*, 2026.

NVIDIA. World simulation with video foundation models for physical ai, 2026.

Tim Rädsch, Yuki M. Asano, Hilde Kuehne, Stefan Bauer, Priyank Jaini, Robert Geirhos, and Carsten T. Lüth. Physics-IQ verified, 2026.

Jacob H. Seidman, Georgios Kissas, Paris Perdikaris, and George J. Pappas. NOMAD: Nonlinear manifold decoders for operator learning. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Louis Serrano, Thomas X Wang, Etienne Le Naour, Jean-Noël Vittaut, and Patrick Gallinari. AROMA: Preserving spatial structure for latent PDE modeling with local neural fields. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0431.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, 2020.

Tencent Hunyuan Foundation Model Team. Hunyuanvideo 1.5 technical report, 2025.

Karn Tiwari, Niladri Dutta, N M Anoop Krishnan, and Prathosh Ap. Latent mamba operator for partial differential equations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59639–59667, 2025.

Wan Team. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-1042.

Amaury Wei and Olga Fink. Integrating physics and topology in neural networks for learning rigid body dynamics. *Nature Communications*, 16(1), 2025. ISSN 2041-1723. doi: 10.1038/s41467-025-62250-7.

Haixu Wu, Tengge Hu, Huakun Luo, Jianmin Wang, and Mingsheng Long. Solving high-dimensional PDEs with latent spectral models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 37417–37438, 2023.

Jia-Qi Yang and Lei Shi. Variation spaces for encoder–decoder neural operators: Approximation and generalization, 2026.

Xindi Yang, Baolu Li, Yiming Zhang, Zhenfei Yin, Lei Bai, Liqian Ma, Zhiyong Wang, Jianfei Cai, Tien-Tsin Wong, Huchuan Lu, and Xu Jia. VLIPP: Towards physically plausible video generation with vision and language informed physical prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12360–12370, 2025.

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.

Biao Zhang, Jiapeng Tang, Matthias Nießner, and Peter Wonka. 3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models. *ACM Transactions on Graphics*, 42(4), 2023a. doi: 10.1145/3592442.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023b.

Qianru Zhang, Zheng Wang, Cheng Long, and Siu-Ming Yiu. Billiards sports analytics: Datasets and tasks, 2024.

Xiangdong Zhang, Jiaqi Liao, Shaofeng Zhang, Fanqing Meng, Xiangpeng Wan, Junchi Yan, and Yu Cheng. VideoREPA: Learning physics for video generation through relational alignment with foundation models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-4091.

A More Proof

A.1 Setting

Let $\mathcal{X}$ and $\mathcal{Z}$ be separable real Hilbert spaces. $\mathcal{X}$ holds the state of one block. $\mathcal{Z}$ is the latent space. The encoder $\mathcal{E} : \mathcal{X} \to \mathcal{Z}$ is measurable, and the decoder $\mathcal{D} : \mathcal{Z} \to \mathcal{X}$ is $L_{\mathcal{D}}$-Lipschitz. Throughout, $2 \le q < \infty$. A trajectory $x_0, \dots, x_K$ is drawn from the data distribution, so its encoded states $z_k := \mathcal{E}(x_k)$ are random with laws $\mu_k := \mathrm{Law}(z_k)$. For a random variable Y with values in a Hilbert space $\mathcal{H}$ we write $\|Y\|_{L^q} := (\mathbb{E}\|Y\|_{\mathcal{H}}^q)^{1/q}$, and for an operator $\mathcal{T} : \mathcal{Z} \to \mathcal{Z}$ and a probability measure μ on $\mathcal{Z}$ we write $\|\mathcal{T}\|_{L^q_\mu(\mathcal{Z})} := (\int_{\mathcal{Z}} \|\mathcal{T}(z)\|_{\mathcal{Z}}^q \,\mathrm{d}\mu(z))^{1/q}$. The two notations agree along the trajectory: $\|\mathcal{T}(z_k)\|_{L^q} = \|\mathcal{T}\|_{L^q_{\mu_k}(\mathcal{Z})}$.

A.2 Latent Variation Space

Let $\sigma : \mathbb{R} \to \mathbb{R}$ be L_σ-Lipschitz. Write $\overline{\mathcal{Z}} := \mathbb{R} \oplus \mathcal{Z}$ and $\bar{z} := (1, z)$, so that $\|\bar{z}\|_{\overline{\mathcal{Z}}} = (1 + \|z\|_{\mathcal{Z}}^2)^{1/2}$, and let $B_{\overline{\mathcal{Z}}}$ be the closed unit ball of $\overline{\mathcal{Z}}$.

Definition A.1 (Latent variation space)**.** An operator $\mathcal{T} : \mathcal{Z} \to \mathcal{Z}$ belongs to $\mathbb{V}_{\mathrm{lat}}(\mathcal{Z}; \mathcal{Z})$ if

$$\mathcal{T}(z) = \int_{B_{\overline{\mathcal{Z}}}} \sigma\big(\langle \bar{z}, \bar{h} \rangle\big) \,\mathrm{d}\Lambda(\bar{h}) \tag{10}$$

for a $\mathcal{Z}$-valued Borel measure Λ of bounded variation. Its variation norm is

$$\|\mathcal{T}\|_{\mathbb{V}_{\mathrm{lat}}} := \inf_{\Lambda} |\Lambda|(B_{\overline{\mathcal{Z}}}), \tag{11}$$

the infimum being over all measures that represent $\mathcal{T}$ in equation 10.

The representation equation 10 is a one-hidden-layer network of unbounded width. Each point $\bar{h} = (\alpha, a) \in B_{\overline{\mathcal{Z}}}$ is a hidden unit $z \mapsto \sigma(\alpha + \langle a, z \rangle)$, and $\mathrm{d}\Lambda(\bar{h})$ is its output weight, a vector in $\mathcal{Z}$. The total variation $|\Lambda|(B_{\overline{\mathcal{Z}}})$ is then the total size of the output weights, and the variation norm is the smallest such total over all networks that compute $\mathcal{T}$. It measures how simple $\mathcal{T}$ is for this model class rather than how many parameters represent it: the constraint $\|\bar{h}\|_{\overline{\mathcal{Z}}} \le 1$ fixes the scale of the hidden units, so the norm cannot be lowered by shrinking output weights while enlarging input weights. A finite network belongs to the space. For $N \ge 1$, a width-N latent ridge operator is

$$\mathcal{T}_N(z) = \sum_{j=1}^{N} b_j \,\sigma\big(\langle \bar{z}, \bar{h}_j \rangle\big), \qquad \bar{h}_j \in B_{\overline{\mathcal{Z}}}, \quad b_j \in \mathcal{Z}, \tag{12}$$

which is represented by the atomic measure $\Lambda = \sum_j b_j \delta_{\bar{h}_j}$ and therefore has $\|\mathcal{T}_N\|_{\mathbb{V}_{\mathrm{lat}}} \le \sum_j \|b_j\|_{\mathcal{Z}}$. Writing $\bar{h}_j = (\alpha_j, a_j)$ recovers the biased feature $b_j \,\sigma(\alpha_j + \langle a_j, z \rangle)$. The space is the latent-to-latent case of the variation spaces for encoder–decoder operators of Yang & Shi (2026).

The second quantity describes the distribution μ of the encoded states. If $\int_{\mathcal{Z}} (1 + \|z\|_{\mathcal{Z}}^2)^{q/2} \,\mathrm{d}\mu(z) < \infty$, set

$$\Gamma_{\sigma,q}(\mu) := \Big(\int_{\mathcal{Z}} \big[|\sigma(0)| + L_\sigma (1 + \|z\|_{\mathcal{Z}}^2)^{1/2} \big]^q \,\mathrm{d}\mu(z) \Big)^{1/q}. \tag{13}$$

The bracket bounds the output of a single hidden unit at z, since $|\langle \bar{z}, \bar{h} \rangle| \le \|\bar{z}\|_{\overline{\mathcal{Z}}}$ for $\bar{h} \in B_{\overline{\mathcal{Z}}}$, and $\Gamma_{\sigma,q}(\mu)$ is its L^q average over the encoded states. Minkowski's inequality gives

$$\Gamma_{\sigma,q}(\mu) \le |\sigma(0)| + L_\sigma \, \overline{M}_q(\mu), \qquad \overline{M}_q(\mu) := \Big(\int_{\mathcal{Z}} (1 + \|z\|_{\mathcal{Z}}^2)^{q/2} \,\mathrm{d}\mu(z) \Big)^{1/q}, \tag{14}$$

and for ReLU, where $\sigma(0) = 0$ and $L_\sigma = 1$, $\Gamma_{\sigma,q}(\mu) \le \overline{M}_q(\mu)$. The quantity therefore grows with how far the encoded states spread from the origin, and it is finite exactly when this q-th moment is.

A.3 Finite-width approximation

Theorem A.2 (Finite-width approximation of latent dynamics)**.** *Let μ be a probability measure on $\mathcal{Z}$ with $\overline{M}_q(\mu) < \infty$, and let $\mathcal{T}^\dagger \in \mathbb{V}_{\mathrm{lat}}(\mathcal{Z}; \mathcal{Z})$. For every $N \ge 1$ and $\varepsilon > 0$ there is a width-N*

latent ridge operator $\mathcal{T}_N$ *with* $\sum_{j=1}^N \|b_j\|_{\mathcal{Z}} \le \|\mathcal{T}^\dagger\|_{\mathbb{V}_{\mathrm{lat}}} + \varepsilon$ *and*

$$\|\mathcal{T}^\dagger - \mathcal{T}_N\|_{L^q_\mu(\mathcal{Z})} \le \frac{C_q\, \Gamma_{\sigma,q}(\mu)\, \big(\|\mathcal{T}^\dagger\|_{\mathbb{V}_{\mathrm{lat}}} + \varepsilon\big)}{\sqrt{N}}, \tag{15}$$

where $C_q > 0$ *depends only on* q*. If the infimum in equation 11 is attained, the bound holds with* $\varepsilon = 0$*.*

The proof uses the Marcinkiewicz–Zygmund inequality in Hilbert space: for independent, mean-zero, $\mathcal{Z}$-valued random variables $Y_1, \dots, Y_N$ with finite q-th moments,

$$\mathbb{E}\Big\| \sum_{j=1}^N Y_j \Big\|_{\mathcal{Z}}^q \le B_q\, \mathbb{E}\Big(\sum_{j=1}^N \|Y_j\|_{\mathcal{Z}}^2 \Big)^{q/2}, \tag{16}$$

where B_q depends only on q. Because the inequality holds in every Hilbert space with the same constant, no dimension of $\mathcal{Z}$ enters C_q.

Proof. Fix $\varepsilon > 0$ and choose a measure Λ representing $\mathcal{T}^\dagger$ with $V := |\Lambda|(B_{\overline{\mathcal{Z}}}) \le \|\mathcal{T}^\dagger\|_{\mathbb{V}_{\mathrm{lat}}} + \varepsilon$. If $V = 0$, then $\mathcal{T}^\dagger = 0$ and $\mathcal{T}_N = 0$ satisfies the claim, so assume $V > 0$. Because Hilbert spaces have the Radon–Nikodym property, Λ admits a polar decomposition $\mathrm{d}\Lambda(\bar{h}) = \lambda(\bar{h})\, \mathrm{d}|\Lambda|(\bar{h})$ with $\|\lambda(\bar{h})\|_{\mathcal{Z}} = 1$ for $|\Lambda|$-almost every $\bar{h}$. With the probability measure $\nu := |\Lambda|/V$,

$$\mathcal{T}^\dagger(z) = V\, \mathbb{E}_{\bar{h}\sim\nu}\big[\lambda(\bar{h})\, \sigma(\langle \bar{z}, \bar{h}\rangle)\big]. \tag{17}$$

Draw $\bar{h}_1, \dots, \bar{h}_N$ independently from ν and set

$$\mathcal{T}_N(z) := \frac{V}{N} \sum_{j=1}^N \lambda(\bar{h}_j)\, \sigma(\langle \bar{z}, \bar{h}_j\rangle). \tag{18}$$

This is a width-N latent ridge operator with $b_j = V\lambda(\bar{h}_j)/N$, so $\sum_j \|b_j\|_{\mathcal{Z}} = V$.

Fix $z \in \mathcal{Z}$. By the Cauchy–Schwarz inequality and the Lipschitz continuity of σ,

$$\big|\sigma(\langle \bar{z}, \bar{h}_j\rangle)\big| \le g(z) := |\sigma(0)| + L_\sigma(1 + \|z\|_{\mathcal{Z}}^2)^{1/2}. \tag{19}$$

The variables $Y_j(z) := V\lambda(\bar{h}_j)\sigma(\langle \bar{z}, \bar{h}_j\rangle) - \mathcal{T}^\dagger(z)$ are independent and have mean zero. By equation 19, the first term of $Y_j(z)$ has norm at most $Vg(z)$, and so does its expectation $\mathcal{T}^\dagger(z)$; hence $\|Y_j(z)\|_{\mathcal{Z}} \le 2Vg(z)$. Since $\mathcal{T}_N(z) - \mathcal{T}^\dagger(z) = N^{-1}\sum_j Y_j(z)$, equation 16 yields

$$\mathbb{E}\big\|\mathcal{T}_N(z) - \mathcal{T}^\dagger(z)\big\|_{\mathcal{Z}}^q \le \frac{B_q}{N^q}\big(N\,(2Vg(z))^2\big)^{q/2} = \frac{C_q^q\, V^q\, g(z)^q}{N^{q/2}}, \qquad C_q := 2B_q^{1/q}. \tag{20}$$

The integrand is jointly measurable in $(z, \bar{h}_1, \dots, \bar{h}_N)$, so integrating over $z \sim \mu$ and applying Fubini's theorem gives

$$\mathbb{E}\,\|\mathcal{T}_N - \mathcal{T}^\dagger\|_{L^q_\mu(\mathcal{Z})}^q \le \frac{C_q^q\, V^q\, \Gamma_{\sigma,q}(\mu)^q}{N^{q/2}}. \tag{21}$$

At least one realization of $\bar{h}_1, \dots, \bar{h}_N$ therefore satisfies $\|\mathcal{T}_N - \mathcal{T}^\dagger\|_{L^q_\mu(\mathcal{Z})} \le C_q V \Gamma_{\sigma,q}(\mu)/\sqrt{N}$, and $V \le \|\mathcal{T}^\dagger\|_{\mathbb{V}_{\mathrm{lat}}} + \varepsilon$ proves equation 15. If the norm is attained, the same argument applies with $V = \|\mathcal{T}^\dagger\|_{\mathbb{V}_{\mathrm{lat}}}$. □

A.4 Rollout bound

The rollout applies a learned transition $\widehat{\mathcal{T}} : \mathcal{Z} \to \mathcal{Z}$ to its own output. To compare it with the data, fix any reference transition $\mathcal{T}^\dagger : \mathcal{Z} \to \mathcal{Z}$.

Definition A.3 (Rollout quantities)**.** For $0 \le k \le K$, the closure defect, one-step error, and reconstruction error are

$$\eta_k := \big\|z_{k+1} - \mathcal{T}^\dagger(z_k)\big\|_{L^q}, \quad \delta_k := \big\|\mathcal{T}^\dagger - \widehat{\mathcal{T}}\big\|_{L^q_{\mu_k}(\mathcal{Z})}, \quad r_k := \big\|x_k - \mathcal{D}(z_k)\big\|_{L^q}, \tag{22}$$

where η_k is defined for $k < K$.

Each quantity isolates one component of the simulator. The closure defect depends only on the data, the encoder, and $\mathcal{T}^\dagger$. It vanishes exactly when $z_{k+1} = \mathcal{T}^\dagger(z_k)$ almost surely, that is, when the encoded states evolve as a closed system under one latent map. It is positive whenever the code omits information on which the next block depends, such as velocity when a block is a single frame. The one-step error compares $\mathcal{T}^\dagger$ with the learned transition on the distribution of encoded states at step k; these distributions change with k because the scene evolves. The reconstruction error depends only on the autoencoder.

Theorem A.4 (Long-horizon latent rollout bound)**.** *Let $\widehat{z}_0$ be any initial latent state and let $\widehat{z}_{k+1} = \widehat{\mathcal{T}}(\widehat{z}_k)$ and $\widehat{x}_k = \mathcal{D}(\widehat{z}_k)$. Suppose that $\widehat{\mathcal{T}}$ is $L_{\mathcal{T}}$-Lipschitz on a set containing every z_k and every $\widehat{z}_k$. Then for every $1 \le k \le K$,*

$$\|z_k - \widehat{z}_k\|_{L^q} \le L_{\mathcal{T}}^k \, \|z_0 - \widehat{z}_0\|_{L^q} + \sum_{s=0}^{k-1} L_{\mathcal{T}}^{k-1-s} \, (\eta_s + \delta_s), \tag{23}$$

$$\|x_k - \widehat{x}_k\|_{L^q} \le r_k + L_{\mathcal{D}} \, \|z_k - \widehat{z}_k\|_{L^q}. \tag{24}$$

Proof. Let $e_k := \|z_k - \widehat{z}_k\|_{L^q}$. Inserting $\mathcal{T}^\dagger(z_k)$ and $\widehat{\mathcal{T}}(z_k)$ gives, pointwise,

$$\|z_{k+1} - \widehat{z}_{k+1}\|_{\mathcal{Z}} \le \|z_{k+1} - \mathcal{T}^\dagger(z_k)\|_{\mathcal{Z}} + \|\mathcal{T}^\dagger(z_k) - \widehat{\mathcal{T}}(z_k)\|_{\mathcal{Z}} + \|\widehat{\mathcal{T}}(z_k) - \widehat{\mathcal{T}}(\widehat{z}_k)\|_{\mathcal{Z}}. \tag{25}$$

Both z_k and $\widehat{z}_k$ lie in the set on which $\widehat{\mathcal{T}}$ is $L_{\mathcal{T}}$-Lipschitz, so the last term is at most $L_{\mathcal{T}}\|z_k - \widehat{z}_k\|_{\mathcal{Z}}$. Taking L^q norms, applying Minkowski's inequality, and using $\|\mathcal{T}^\dagger(z_k) - \widehat{\mathcal{T}}(z_k)\|_{L^q} = \delta_k$ (Appendix A.1) gives

$$e_{k+1} \le \eta_k + \delta_k + L_{\mathcal{T}} \, e_k. \tag{26}$$

Iterating equation 26 from $k = 0$ proves equation 23. For the decoded prediction,

$$\|x_k - \widehat{x}_k\|_{\mathcal{X}} \le \|x_k - \mathcal{D}(z_k)\|_{\mathcal{X}} + \|\mathcal{D}(z_k) - \mathcal{D}(\widehat{z}_k)\|_{\mathcal{X}} \le \|x_k - \mathcal{D}(z_k)\|_{\mathcal{X}} + L_{\mathcal{D}}\|z_k - \widehat{z}_k\|_{\mathcal{Z}}, \tag{27}$$

and taking L^q norms proves equation 24. □

Substituting equation 23 into equation 24 gives

$$\|x_k - \widehat{x}_k\|_{L^q} \le r_k + L_{\mathcal{D}} L_{\mathcal{T}}^k \, \|z_0 - \widehat{z}_0\|_{L^q} + L_{\mathcal{D}} \sum_{s=0}^{k-1} L_{\mathcal{T}}^{k-1-s} (\eta_s + \delta_s). \tag{28}$$

The rollout starts from $\widehat{z}_0 = \mathcal{E}(x_0) = z_0$, so the middle term vanishes and equation 28 reduces to Theorem A.4. The bound holds for every $\mathcal{T}^\dagger$. With $\mathcal{T}^\dagger = \widehat{\mathcal{T}}$, every δ_k is zero and η_k is the one-step error of the learned transition on encoded ground truth.

A.5 Corollaries used in Section 4.2

The first corollary shows that one sampled operator controls the one-step error at every step of the rollout and satisfies the Lipschitz assumption of Theorem A.4 with an explicit constant.

Corollary A.5 (One operator for the whole rollout)**.** *Let $\mathcal{T}^\dagger \in \mathbb{V}_{\mathrm{lat}}(\mathcal{Z}; \mathcal{Z})$, let each μ_k satisfy $\overline{M}_q(\mu_k) < \infty$, and let $\bar{\mu} := K^{-1} \sum_{k=0}^{K-1} \mu_k$. For every $N \ge 1$ and $\varepsilon > 0$ there is a width-N latent ridge operator $\mathcal{T}_N$ such that, for every $0 \le k \le K-1$,*

$$\|\mathcal{T}^\dagger - \mathcal{T}_N\|_{L^q_{\mu_k}(\mathcal{Z})} \le \frac{K^{1/q} \, C_q \, \Gamma_{\sigma,q}(\bar{\mu}) \, \big(\|\mathcal{T}^\dagger\|_{\mathbb{V}_{\mathrm{lat}}} + \varepsilon\big)}{\sqrt{N}}, \tag{29}$$

and $\mathcal{T}_N$ is Lipschitz on $\mathcal{Z}$ with $\mathrm{Lip}(\mathcal{T}_N) \le L_\sigma\big(\|\mathcal{T}^\dagger\|_{\mathbb{V}_{\mathrm{lat}}} + \varepsilon\big)$.

Proof. The average $\bar{\mu}$ satisfies $\overline{M}_q(\bar{\mu}) < \infty$, so Theorem A.2 applied to $\bar{\mu}$ gives $\mathcal{T}_N$ satisfying equation 15 in $L^q_{\bar{\mu}}(\mathcal{Z})$. For each k,

$$\int_{\mathcal{Z}} \|\mathcal{T}^\dagger - \mathcal{T}_N\|_{\mathcal{Z}}^q \, \mathrm{d}\mu_k \le \sum_{j=0}^{K-1} \int_{\mathcal{Z}} \|\mathcal{T}^\dagger - \mathcal{T}_N\|_{\mathcal{Z}}^q \, \mathrm{d}\mu_j = K \, \|\mathcal{T}^\dagger - \mathcal{T}_N\|^q_{L^q_{\bar{\mu}}(\mathcal{Z})}, \tag{30}$$

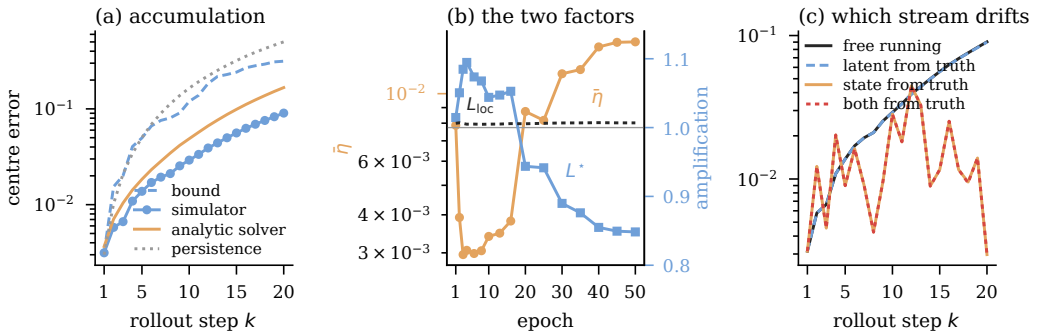


Figure 3: **The rollout bound of Theorem A.4, measured term by term** on held-out MOVi-Spheres scenes. (a) Centre error against rollout horizon, with the bound evaluated from the measured per-step defects and the amplification L_{loc}. (b) The bound's two factors across 50 epochs: the mean per-step defect $\bar{\eta}$ (left, log scale) rises fivefold after epoch 6 while the amplification implied by the observed accumulation, $L^\star$ (right), falls through one. The perturbation rate L_{loc} stays flat and tracks neither. (c) Substituting the encoded truth into one factor of the latent state at every step. Correcting the occupancy latent leaves the rollout unchanged, while correcting the object state collapses it to the per-step defect.

which gives equation 29. For the Lipschitz bound, write $\bar{h}_j = (\alpha_j, a_j)$ with $\|a_j\|_{\mathcal{Z}} \le 1$. Then

$$\|\mathcal{T}_N(z) - \mathcal{T}_N(z')\|_{\mathcal{Z}} \le \sum_{j=1}^{N} \|b_j\|_{\mathcal{Z}}\, L_\sigma\, |\langle a_j, z - z'\rangle| \le L_\sigma \Big(\sum_{j=1}^{N} \|b_j\|_{\mathcal{Z}}\Big) \|z - z'\|_{\mathcal{Z}}, \tag{31}$$

and $\sum_j \|b_j\|_{\mathcal{Z}} \le \|\mathcal{T}^\dagger\|_{\mathbb{V}_{\text{lat}}} + \varepsilon$. □

The second corollary describes the residual parameterization, in which the network represents only the change between consecutive blocks.

Corollary A.6 (Residual transition)**.** *Let $\mathcal{R}^\dagger := \mathcal{T}^\dagger - \mathrm{I}$ belong to $\mathbb{V}_{\text{lat}}(\mathcal{Z}; \mathcal{Z})$, and let μ satisfy $\overline{M}_q(\mu) < \infty$. For every $N \ge 1$ and $\varepsilon > 0$ there is a width-N latent ridge operator $\mathcal{T}_N$ such that $\widehat{\mathcal{T}} := \mathrm{I} + \mathcal{T}_N$ satisfies*

$$\|\mathcal{T}^\dagger - \widehat{\mathcal{T}}\|_{L^q_\mu(\mathcal{Z})} \le \frac{C_q\, \Gamma_{\sigma,q}(\mu)\, \big(\|\mathcal{R}^\dagger\|_{\mathbb{V}_{\text{lat}}} + \varepsilon\big)}{\sqrt{N}}, \qquad \mathrm{Lip}(\widehat{\mathcal{T}}) \le 1 + L_\sigma\big(\|\mathcal{R}^\dagger\|_{\mathbb{V}_{\text{lat}}} + \varepsilon\big). \tag{32}$$

Proof. Apply Theorem A.2 to $\mathcal{R}^\dagger$ and note that $\mathcal{T}^\dagger - \widehat{\mathcal{T}} = \mathcal{R}^\dagger - \mathcal{T}_N$. The Lipschitz bound of Corollary A.5, applied to $\mathcal{T}_N$, gives $\mathrm{Lip}(\mathcal{T}_N) \le L_\sigma(\|\mathcal{R}^\dagger\|_{\mathbb{V}_{\text{lat}}} + \varepsilon)$, and $\mathrm{Lip}(\mathrm{I} + \mathcal{T}_N) \le 1 + \mathrm{Lip}(\mathcal{T}_N)$. □

The same quantity, $\|\mathcal{R}^\dagger\|_{\mathbb{V}_{\text{lat}}}$, thus controls both the one-step error and the amplification factor $L_{\mathcal{T}}$ of Theorem A.4. When consecutive blocks differ little in the latent coordinates, a residual transition is accurate at small width and amplifies errors only slightly.

B Measurements of the Latent Neural Simulator

We evaluate on held-out MOVi-Spheres scenes, none of which appear in training. Each episode is a 20-step free-running rollout in which the true state enters at the first frame only. Scenes hold 3 to 10 spheres of radius 0.35 to 0.70, and errors are distances between predicted and true object centres in scene units, where the mean radius is 0.54.

Accuracy on held-out scenes. The analytic solver of Table 4 integrates gravity and resolves sphere–sphere and sphere–floor impulses with the restitution and friction the dataset records, and it beats

Table 4: **The simulator against explicit alternatives**, on 200 held-out windows of 20 frames. Error is the centre distance in scene units, averaged over the rollout and at its end. Contact F1 scores predicted against true contact events under one geometric criterion applied to all four rows, so it reads the raw predicted geometry rather than any learned contact head.

	Error ↓	Error at k=20 ↓	Sphere IoU ↑	Occ. IoU ↑	Contact F1 ↑
Latent neural simulator	**0.0384**	**0.0908**	**0.901**	**0.892**	0.508
Analytic impulse solver	0.0654	0.1669	0.855	0.846	**0.651**
Ballistic (gravity, no contact)	0.2264	0.6248	0.693	0.705	0.597
Persistence (frozen state)	0.2049	0.5008	0.670	0.673	0.597

the ballistic arm by $3.5\times$. The simulator cuts its 20-step error by 46%, from 0.1669 to 0.0908. The one column it loses is contact F1, 0.508 against 0.651. Its own contact head scores 0.691 on the same validation split, so the learned classifier recovers what the predicted geometry misses rather than the geometry being right in the first place. The solver applies one impulse per frame while the reference engine resolves contact over sub-steps, so it marks the weaker end of the classical range rather than its best.

The rollout bound has two factors, and they can be measured separately. Taking $\mathcal{T}^\dagger = \widehat{\mathcal{T}}$ in Definition A.3 sets every δ_k to zero and leaves η_k, the one-step error of the learned transition from the encoded true state, so equation 28 reduces to $\|x_k - \widehat{x}_k\| \le \sum_{s<k} L_{\mathcal{T}}^{k-1-s}\eta_s$ once the per-step physical error absorbs r_k and $L_{\mathcal{D}}$. We measure η_s at every $s < 20$ over 649 objects and estimate the amplification two ways: L_{loc}, the growth rate of a 10^{-3} perturbation of the initial position, and $L^\star$, the value that makes the bound tight at $k = 20$. Figure 3a shows the bound against the measured accumulation at epoch 50, which it holds with a factor of 3.5 to spare.

Training buys contraction rather than per-step accuracy. Over 50 epochs the two factors move in opposite directions (Figure 3b). The mean per-step defect $\bar{\eta}$ falls to 0.0030 by epoch 6 and then rises fivefold to 0.0148, while $L^\star$ falls from 1.095 at epoch 4 to 0.848 at epoch 50, crossing one between epochs 16 and 20. The rollout error follows their combination rather than either factor alone, e_{20} falling from 0.124 to 0.090 across exactly the span in which the per-step defect is getting worse, so a single-step validation loss would report that half of training as a regression. The local rate explains none of this: L_{loc} sits between 1.0042 and 1.0075 at every epoch, and used in the bound it is violated early (2.57 times the bound at epoch 4) and slack late (0.287 at epoch 50), as a local linearisation must be when it stands in for a constant the theorem requires to hold uniformly on a set containing both trajectories.

Only one of the two coupled streams accumulates. The latent state is a product of an occupancy code and an explicit object state, so substituting the encoded truth into one factor at every step isolates the other (Figure 3c). Re-encoding the occupancy latent from ground truth moves the 20-step error from 0.0903 to 0.0899. Resetting the object state instead collapses the rollout to its per-step defect regardless of what the latent does, so accumulation runs through the object-state factor alone. The occupancy latent is nonetheless read: a latent taken from a different scene costs 17% at $k = 20$ and a zero latent costs 93%. It also drifts, to 0.80 of its own norm by step 20, and correcting that drift buys almost nothing. Sensitivity to the latent is therefore strongly anisotropic, which is one concrete reason a single $L_{\mathcal{D}}$ over the whole latent space leaves the bound conservative.

What the remaining defect is made of. At epoch 50 the per-step defect is close to deterministic rather than heavy-tailed: at $s = 11$ its mean is 0.0447, its median 0.0446 and its ninetieth percentile 0.0478. Every object in a scene is displaced by nearly the same vector, almost entirely horizontal, and its size depends on which frame the step starts from, 0.0031 at $s = 0$ and 0.0030 at $s = 19$ against 0.0447 at $s = 11$. The defect is a frame-indexed rigid translation that cancels along the model's own trajectory and appears only when a true state is injected, which is the dependence of μ_k on k that Definition A.3 isolates. It also locates the residual error away from the contact response, since error at ball–ball contacts (0.0458), at floor contacts (0.0369) and in free flight (0.0383) differs by at most 24%.

These measurements are a single training run, and $L^\star$ is fitted to the observed accumulation rather than certified.

C CO-TRAINING

We tested whether making the operator trainable under the video objective helps, with the physical loss weighted by λ. At $\lambda = 1$ the operator is pinned by the physical loss: its held-out state error is unchanged (e_{12} $0.56 \to 0.56$, end-of-horizon $5.55 \to 5.57$ sphere radii) even when its learning rate is raised tenfold, and the pipeline reduces to the frozen case. At $\lambda = 0$ the diffusion gradient does move the operator, but in the wrong direction for physics: short-horizon error nearly doubles ($0.56 \to 0.96$) while obedience is unchanged. Across seeds the frozen pipeline is ahead of the co-trained arm by 0.054 ± 0.017 in per-scene hit rate ($n = 32$). The diffusion loss carries too little signal about physical variables to supervise a dynamics model: validation losses of all interface variants agree to three decimals.

D ADDITIONAL FIGURES

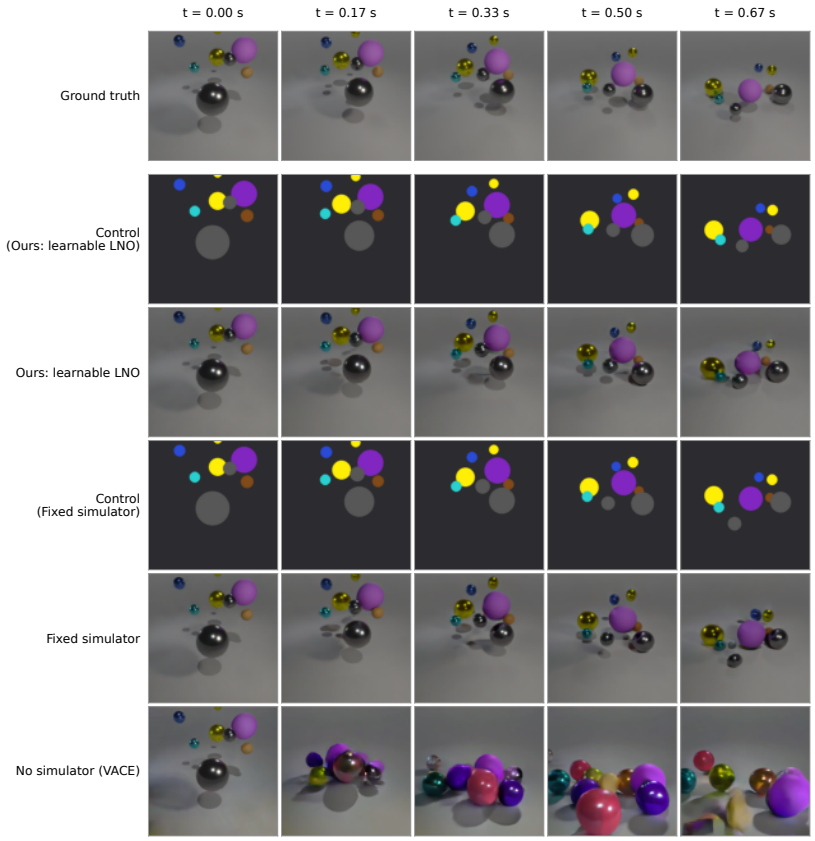


Figure 4: **Q1 and Q2 on a MOVi-Spheres scene.** Rows: ground truth; the condition from the learnable simulator and the resulting video; the condition from the fixed integrator and its video; the generator without a simulator. Columns share the time stamp. Without a simulator the generator invents new objects within $0.17\,\mathrm{s}$. The scene is the one with the largest per-scene hit-rate gap between the two simulators (all 24 scenes in the supplementary material).

The figures below plot numbers already given in the tables of Section 6.

E IMPLEMENTATION DETAILS OF MOTIONCORE

Data. MOVi-Spheres (Wei & Fink, 2025; Greff et al., 2022) contains 1200 Kubric scenes with 3 to 10 spheres of varying radius, mass, restitution, and friction, simulated at 1200 Hz under gravity $10\,\mathrm{m/s^2}$ and stored at 240 Hz for 2 s. Scenes are split 80/10/10 into training, validation, and test

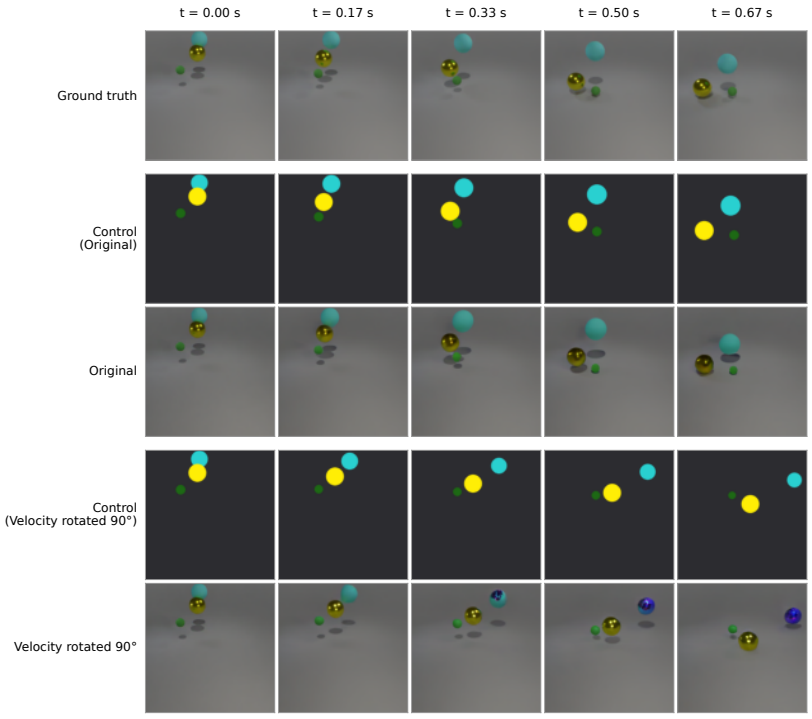


Figure 5: **Q3: editing the physics.** The same first frame is rolled out by the simulator with the original initial velocities (upper block) and with every object's horizontal velocity rotated by 90° (lower block); the generated videos follow the edited trajectories.

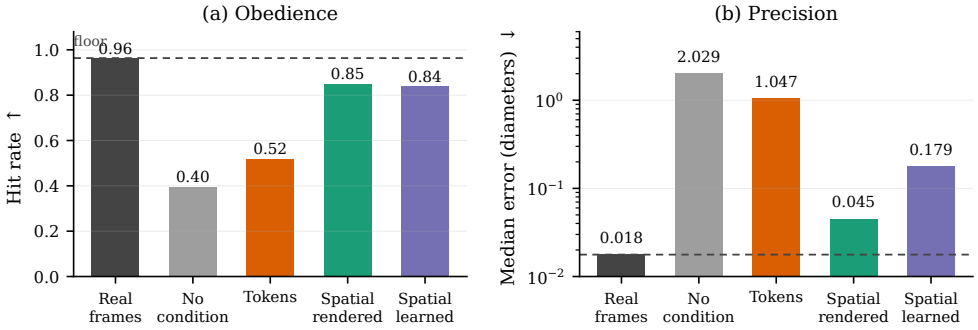


Figure 6: **The form of the condition, not its capacity, decides whether a frozen generator obeys it.** Dashed lines are the measurement floor obtained by running real frames through the same pipeline. Note the logarithmic axis in (b).

sets with a fixed seed. We subsample the trajectories to 60 Hz and cut them into non-overlapping windows of 21 frames: the simulator receives the first frame and predicts the next $K = 20$, matching the 21-frame clip of the generator. A pair or floor contact label is positive for a transition when the minimum gap over the 240 Hz frames between the two sampled frames is at most 0.02. Coordinates are mapped from the scene box $[-5, 5]^2 \times [0, 5]$ to $[-1, 1]^3$. Each object state $\mathbf{s}^i = (\mathbf{p}^i, \mathbf{v}^i, \boldsymbol{\omega}^i, r^i) \in \mathbb{R}^{10}$ stores the normalized position, the velocity as a normalized displacement per frame, the angular velocity multiplied by the frame interval, and the radius normalized by the mean box extent. Mass, restitution, and friction are read from the scene metadata and enter only the collision operator.

Point sampling. The encoder input $\mathbf{P}$ contains $N_p = 2048$ points drawn uniformly inside the spheres, with the same number of points for every object regardless of its size. The query set $\mathbf{Q}$ contains $N_q = 2048$ points. Half are uniform over the scene box. The other half lie near objects:

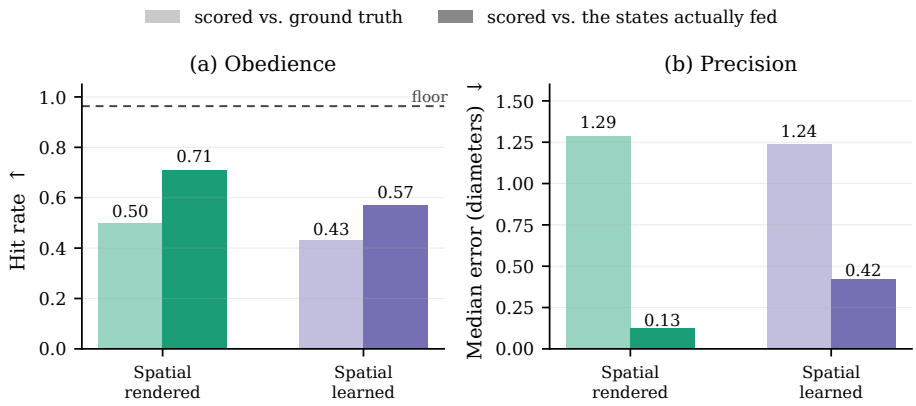


Figure 7: **Two references separate the operator's error from the generator's obedience.** The same videos are scored against ground truth (pale) and against the states actually fed to the generator (solid). The gap between the two is the operator's error as it appears on screen.

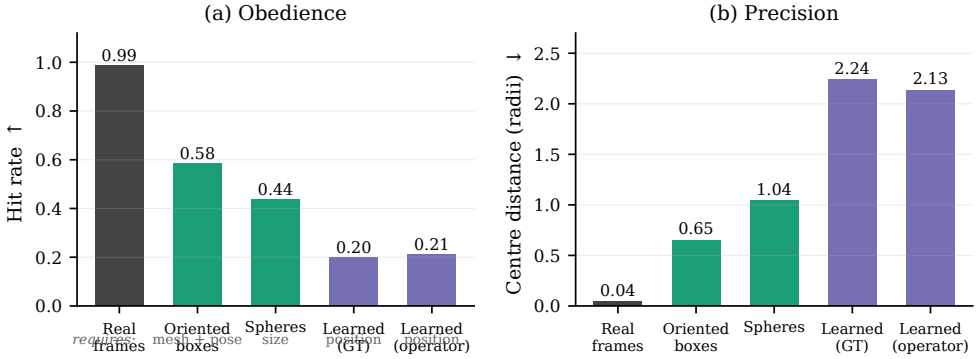


Figure 8: **On non-spherical objects with occlusion, obedience follows how much geometry the condition carries.** The label under each bar states the geometric information the condition consumes.

half of these inside the object volume and half on its surface, perturbed by isotropic Gaussian noise with standard deviation $\sigma_s = 0.05$. Each frame is sampled independently. The autoencoder is pretrained on space–time segments of 16 frames with the time coordinate normalized to $[-1, 1]$; for a single frame, all points carry $t = 0$.

Occupancy autoencoder. Fourier features with eight octaves map each four-dimensional point to width 256. The encoder uses one cross-attention block from 256 learned query tokens to the points, followed by eight pre-norm self-attention blocks with eight heads and feed-forward ratio 4. Two linear heads produce $\boldsymbol{\mu}$ and $\log \boldsymbol{\sigma}^2$ of size 256×32. The decoder projects the tokens to width 256, applies eight self-attention blocks, and lets each query cross-attend to the result before an MLP predicts its occupancy logit. The reported autoencoder is trained with $\beta = 0.001$.

Latent neural operator. Two-layer GELU MLPs project the reference coordinates $\boldsymbol{\xi}$, evenly spaced in $[-1, 1]$, and the layer-normalized slot features to width 128. The physics cross-attention computes $\mathbf{W} \in \mathbb{R}^{256\times128}$ from the projected coordinates. A softmax over slots aggregates the 256 slots into 128 modes, eight attention blocks with four heads act on the modes, and a softmax over modes distributes the result back to the slots, where a two-layer MLP with a zero-initialized output layer returns the latent update. Since $\boldsymbol{\xi}$ is shared by all samples, $\mathbf{W}$ does not depend on the input.

State adapter and free-motion proposal. The adapter $\mathcal{A}_c$ projects each object state with a two-layer MLP and the latent slots with a linear layer to width 128, applies one four-head cross-attention

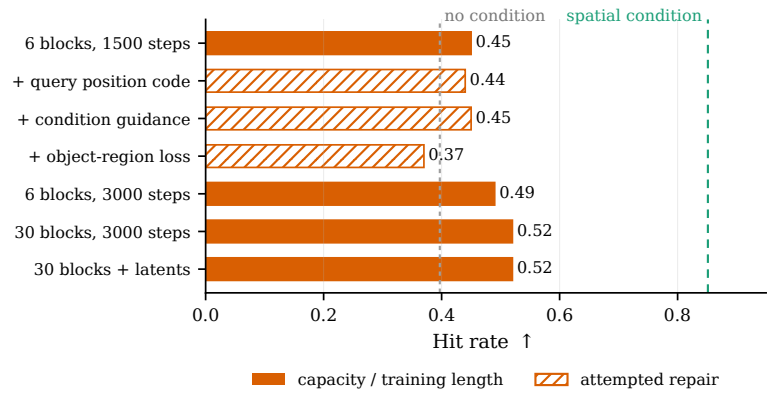


Figure 9: **A token interface cannot be repaired by making the signal easier to find.** Hatched bars are attempted repairs; solid bars vary capacity or training length. The two dashed lines are the unconditioned model and a spatially aligned condition.

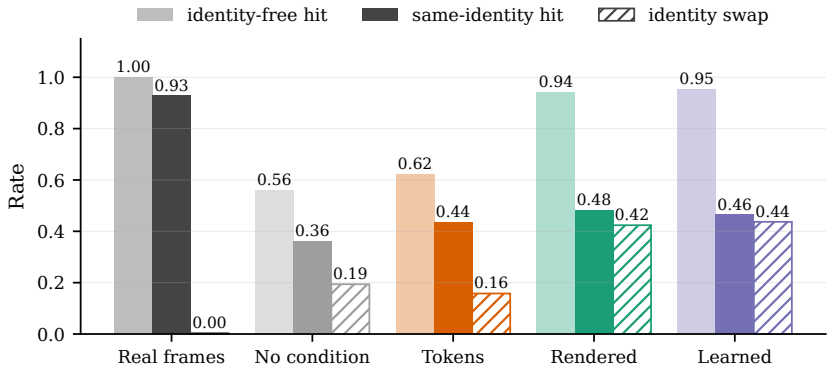


Figure 10: **Position and identity fail independently.** Pale bars ignore which object is which, solid bars require the correct object, hatched bars count correctly placed objects carrying the wrong identity.

from slots to objects with absent objects masked, and maps the normalized sum back to the latent width as a residual. The proposal head $\mathcal{P}_{\text{free}}$ adds an eight-band Fourier embedding of the target time to each object token and applies cross-attention to $\hat{\mathbf{z}}_{k+1}$, masked self-attention among objects, and a feed-forward block, each followed by a residual connection and layer normalization, all at width 128 with four heads. Its zero-initialized linear output gives the corrections of free-motion proposal, bounded by 0.05 for position and 0.25 for linear and angular velocity in normalized units. The radius is copied from the initial state. The collision operator does not change $\boldsymbol{\omega}$, so spin changes from friction are learned only by this head.

Surface operator. Each object is represented by 128 oriented surface samples placed on a Fibonacci lattice. For every object, the eight nearest samples of the other objects give a geometric gap and normal. Their relative positions and normals, scaled by the object radius, are encoded by a shared MLP of width 128, pooled by learned attention weights, and mapped to a gap correction bounded by 0.25 and a normal correction bounded by 0.5; both output layers are initialized to zero. The operator is pretrained on exact gap and normal targets with a smooth-L_1 gap loss weighted towards near and contacting queries and a cosine normal loss, reaching contact F1 0.934 against 0.840 for the uncorrected estimate on validation windows. It is frozen during Stage 2.

Collision operator. Contact gaps and normals for the proposed positions are computed in closed form for sphere pairs and against the floor plane; the surface-operator estimates enter as features. Each contact edge is described by 21 features: the gap, the normal, the relative velocity and its

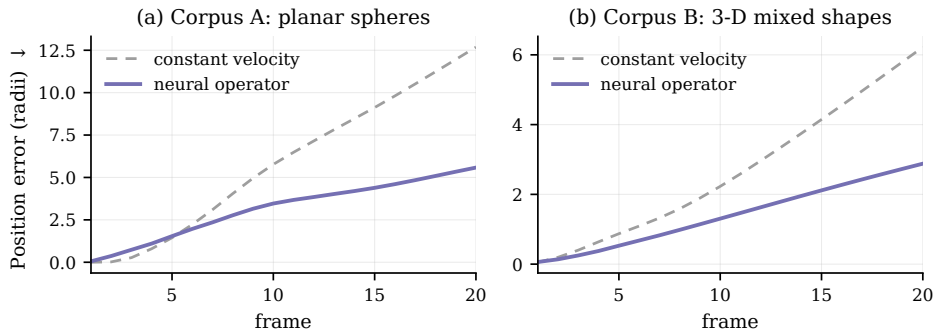


Figure 11: **The neural operator against a constant-velocity baseline** on held-out scenes of both corpora, in units of object radius.

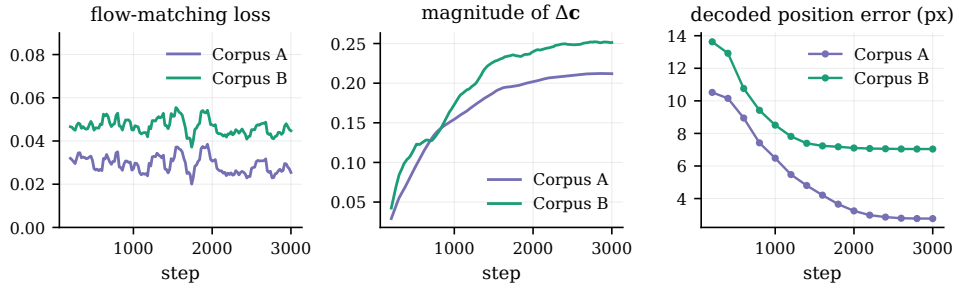


Figure 12: **Training the learned adapter** on both corpora. The zero-initialised condition opens as training proceeds (centre), and the decodability constraint tightens (right); the flow-matching loss itself does not separate the variants, which is why obedience is judged by measurement after generation rather than by validation loss.

normal and tangential speeds, both radii, both inverse masses and the reduced mass, the restitution and friction priors, the surface-operator gaps and their alignment with the contact normal, and an indicator for the static floor. A two-layer GELU edge network of width 128, shared by all edges, encodes them, and zero-initialized linear heads give ρ_{ij} and the coefficient corrections, so the operator starts as exact geometry with the material priors of the data. The contact probability uses tolerance $\epsilon = 0.02$ and temperature $\tau_c = 0.02$. The restitution and friction priors are the geometric means of the per-object values for body pairs and the object's own value against the floor; the learned corrections enter through $0.5 \tanh(\cdot)$, restitution is clipped to $[0, 1]$, and friction to $[0, 2]$. The velocity correction of each object is clipped to norm 50, which leaves momentum conservation exact whenever the clip is inactive. Overlaps are removed by moving each body along the normal by half of the penetration depth, weighted by its share of the inverse mass, and positions additionally follow half of the velocity correction over one frame.

Loss weights. The occupancy temperature is $\tau_o = 0.02$, and the state losses use smooth-L_1 with $\beta = 0.02$. The weights of Eq. 9 are $\lambda_z = \lambda_o = 1$, $\lambda_{\mathbf{p}} = \lambda_{\mathbf{v}} = \lambda_{\boldsymbol{\omega}} = 1$, and $\lambda_c = 0.2$ for the averaged object and pair contact terms and, separately, for the floor contact term. All contact terms use focal parameters $\alpha = 0.75$ and $\gamma = 2$.

Learned adapter. The adapter embeds object attributes and the operator features $\mathbf{M}_{1:T}$, splats them at the projected object centres on the latent video grid, and applies a zero-initialized 1×1 projection to the control channels (Zhang et al., 2023b). The video VAE compresses frames in groups $\{0\}, \{1, \ldots, 4\}, \{5, \ldots, 8\}, \ldots$, and the adapter concatenates the four sub-frame maps of each group along the channel axis before the projection. A 1×1 convolution on the pre-projection feature map produces one heat map per object and sub-frame, and a soft-argmax decodes a position

$\widehat{\mathbf{p}}_{t,i}$ supervised by

$$\mathcal{L}_{\text{pos}} = \frac{1}{|\mathcal{V}|} \sum_{(t,i)\in\mathcal{V}} \text{SmoothL1}\left(\widehat{\mathbf{p}}_{t,i}, \pi(\mathbf{c}_{t,i})\right) \tag{33}$$

over valid frame–object pairs $\mathcal{V}$. With the video VAE latent $\mathbf{z}$, flow time $s \in [0,1]$, $\mathbf{z}_s = (1-s)\mathbf{z}+s\boldsymbol{\epsilon}$, and the frozen velocity network $\mathcal{G}_{\text{v}}$, only θ is optimized with

$$\mathcal{L}_{\text{vid}}(\theta) = \left\|\mathcal{G}_{\text{v}}(\mathbf{z}_s, s, \mathbf{c}) - (\boldsymbol{\epsilon} - \mathbf{z})\right\|_2^2 + \lambda_p \mathcal{L}_{\text{pos}}, \qquad \lambda_p = 0.1. \tag{34}$$

The adapter receives the ground-truth states of each paired sample, and the position head is discarded after training.

Optimization. All stages use AdamW with weight decay 10^{-4} and gradient clipping at norm 1. The autoencoder is trained for 200 epochs with learning rate 4×10^{-5} and batch size 64. The surface operator is trained with learning rate 3×10^{-4}, batch size 64, and a cosine schedule on 6000 training windows, with early stopping after at least 8 epochs. The dynamics stage uses learning rate 10^{-4} with a cosine schedule and a total batch size of 128 for 100 epochs, and the checkpoint with the best validation occupancy IoU is kept. Training runs with data parallelism on eight GPUs.