PHYSIPLAY: LEARNING MULTI-AGENT PHYSICS-BASED GAME SIMULATION IN WORLD SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent work showed that multi-object future dynamics can be generated as a single denoising process directly over 3D world-space mesh vertex trajectories, without a latent space or hand-imposed physical priors such as rigidity. Such models are bidirectional: all conditioning is fixed before sampling, so they cannot respond to actions that arrive during a rollout. We present PHYSIPLAY, which keeps this world-space formulation and makes it interactive. PHYSIPLAY adapts a bidirectional teacher into a block-causal generator that produces four-frame chunks from a fixed-size window of recent history and a persistent geometry reference, and binds keyboard actions to individual scene objects through global per-frame modulation and object-local FiLM. Actions therefore directly drive the controlled objects and propagate through contact to other objects. We demonstrate PHYSIPLAY on two-player 3D pong with balls spanning six materials and multiple geometries, conditioned on physical-property vectors. Because the output is explicit geometry, physical quantities and behaviors such as rigidity, momentum, and deformation can be directly inspected or measured; we quantitatively evaluate trajectory accuracy, rigidity, and momentum over rollouts three times longer than the bidirectional teacher's training window and ablate the causal adaptation. By operating in view-invariant world coordinates with persistent object representations, PHYSIPLAY takes a step toward learned interactive simulators for multi-agent control.

1 INTRODUCTION

Interactive world models have become increasingly capable of supporting real-time gameplay. Neural game engines map key presses to the next frame of a running game (Valevski et al., 2025; Alonso et al., 2024), Genie (Bruce et al., 2024) learns an action space from unlabeled video rather than being given one, and recent multi-agent systems bind separate action streams to separate entities so that several players act on one world at once (Liu et al., 2026a; Pondaven et al., 2026; Hu et al., 2026). What these models return, however, is a picture of the consequence rather than the physical state. Object permanence is not guaranteed and view dependency can yield inconsistencies.

PhysiFormer (Chen et al., 2026) showed that multi-object dynamics can be generated as a single denoising process over world-space mesh vertex trajectories. Its output vertex trajectories directly represent 3D geometry, which can be measured against physical quantities and rendered from any viewpoint. However, the fixed-length trajectory is denoised at once, and all conditioning must be fixed before the first frame exists. A continuous game inverts both properties: input arrives while the simulation is running and shapes future dynamics.

We present PHYSIPLAY, which keeps the world-space diffusion formulation and makes it interactive by simulating a two-player 3D pong game where the ball varies across six materials and multiple geometries. PHYSIPLAY generates four-frame chunks along a block-causal temporal axis from a fixed-size window of recent history, where action conditioning is supplied incrementally per block. Keys drive a paddle, the paddle strikes the ball, and the ball's trajectory, deformation, and later contacts change as a consequence. Because the state is shared rather than rendered per player, multiple agents interact with the same predicted world state. PHYSIPLAY is built in three stages: (1) a bidirectional dynamics prior initialized from Chen et al. (2026), (2) action and material conditioning, and (3) causal adaptation. We evaluate rollouts of up to 243 frames, three times the 81-frame window of the bidirectional teacher.

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.

Our contributions are as follows. (1) We adapt world-space trajectory diffusion from a fixed-length predictor into a block-causal chunked generator with a fixed-size context and a persistent cross-attention geometry reference. (2) We introduce object-level action conditioning for 3D dynamics, allowing actions applied to selected objects to influence the motion of other, unactuated objects through physical interactions.

2 Related Work

Numerical and Differentiable Simulation. Physical dynamics are classically solved by numerical integration, the discretization distinguishing finite-element (Zienkiewicz et al., 2013), position-based (Macklin & Müller, 2013), material-point (Jiang et al., 2016), particle-based (Desbrun & Gascuel, 1996), and mass–spring (Liu et al., 2013) formulations, with dedicated treatments of contact (Bouaziz et al., 2023) and time integration (Baraff & Witkin, 2023); a later strand exposes gradients through the solver itself (Hu et al., 2019; Jatavallabhula et al., 2021). Each is written against a particular constitutive law, so different materials require explicit constitutive models and corresponding solver configurations with differing inference costs, complex scenes invite instability or failure, and all demand a fully specified physical state. PhysiPlay is instead a single learned model spanning several materials, probabilistic where a solver is deterministic.

Learned Simulation. Data-driven simulators replace the solver with a network. Physics-informed networks (Raissi et al., 2019) write constraints into the loss; convolutional models suit regular grids (Guo et al., 2016; Bhatnagar et al., 2019); graph networks (Scarselli et al., 2008) advance particle systems by message passing (Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020a), extended to mesh discretizations (Pfaff et al., 2020), to face-based and higher-order interactions for contact-rich motion (Allen et al., 2022; Wei & Fink, 2025), to hierarchical structure with attention (Yu et al., 2024), and to attention in place of explicit edges (Shao et al., 2022). Two recurring challenges remain, even for recent transformer-based methods. Stepping a deterministic predictor repeatedly lets residuals compound into drift (Sanchez-Gonzalez et al., 2020a), and accuracy is often bought with explicit structure: supplied connectivity, learned signed-distance geometry (Rubanova et al., 2024), an integrator retained in the loop beside a learned corrector with a separate model per dynamics category (Wang et al., 2026), rigidity imposed by projection onto the rigid-body manifold (Dou et al., 2026), or a learned latent space between model and physical state (Zhang et al., 2026). Instead, PhysiPlay denoises trajectories over raw vertex coordinates without hand-imposed physical constraints such as rigidity, relying only on mesh topology, object grouping, and a material vector, and covers six materials with one model.

Diffusion-based 4D Simulation. Recent work treats future dynamics as generation in world space: PhysiFormer (Chen et al., 2026) denoises future mesh vertex trajectories directly in 3D coordinates from initial positions, velocities and material type, PhysCtrl (Wang et al., 2025) models 3D point trajectories conditioned on material parameters and forces to drive image-to-video synthesis of single objects, and others reconstruct 4D surfaces from sparse or partial point-cloud sequences by latent diffusion (Cao et al., 2024). Denoising a whole trajectory at once avoids the error accumulation of one-step regressors but fixes the horizon and sets conditioning before sampling. PhysiPlay keeps coordinate-space diffusion but makes the temporal axis causal, so conditioning arrives incrementally.

Interactive and Playable World Models. Neural game engines condition frame prediction on key presses (Valevski et al., 2025) or game-specific actions (Alonso et al., 2024), while Genie (Bruce et al., 2024) learns a latent action space from unlabeled video without action supervision. Other interfaces combine camera navigation with textual event control (Team et al., 2026; Gao et al., 2026). Multi-agent models associate actions with individual entities through per-agent action conditioning and coordinated video streams (Liu et al., 2026a), persistent 2D subject states (Pondaven et al., 2026), or joint conditioning on multiple players' controls (Hu et al., 2026). These methods predict video, sometimes augmented with subject states; PhysiPlay instead predicts material-conditioned vertex trajectories in world coordinates, with actions assigned to individual objects in a shared 3D scene, so objects remain represented regardless of visibility and every viewpoint renders the same state.

From Bidirectional to Causal Generation. Teacher forcing (Williams & Zipser, 1989) conditions predictions on ground-truth history during training, introducing a train–test mismatch. Diffusion

forcing (Chen et al., 2024) uses independent per-token noise levels, allowing a causal model to roll out beyond its training horizon. Self forcing (Huang et al., 2025) trains on the model's own outputs to account for the distribution of accumulated prediction errors. Rolling forcing (Liu et al., 2026b) jointly denoises multiple frames with progressively increasing noise levels. Causal forcing (Zhu et al., 2026) addresses the mismatch in distillation from a bidirectional teacher to an autoregressive student. Flex-forcing (Ma et al., 2026) unifies bidirectional and autoregressive regimes through chunking along the temporal and denoising dimensions. These paradigms were developed for video; PHYSIPLAY draws on them for generation in 3D vertex space, with the noise schedule tuned to the signal-to-noise ratio of the data.

Physics-Grounded Video Generation. A further family imports dynamics from a simulator, reconstructing a 3D representation and simulating before rendering (Xie et al., 2023; Zhong et al., 2024; Xie et al., 2025b; Jiang et al., 2024), estimating physics parameters by distilling motion from video models (Zhang et al., 2024), delegating dynamics to a simulator and appearance to a video model (Liu et al., 2024; Chen et al., 2025; Tan et al., 2026; Xie et al., 2025a), or adopting force as the conditioning signal (Gillman et al., 2025). These works return video; PhysTwin (Jiang et al., 2025) retains interactivity by fitting spring–mass models and Gaussian splats to sparse video, keeping an explicit physical model in the loop and inheriting its assumptions. PHYSIPLAY keeps an interactive state without a simulator in the loop, emitting 3D geometry that can be rendered from any viewpoint and advanced under new actions.

3 METHOD

A playable simulation has to do two things at once: carry a scene forward under its own mechanics, and absorb whatever the players do to it while that is happening. PHYSIPLAY treats this as sampling from a conditional distribution over future 4D geometry, a few frames at a time as control arrives.

Notation. A scene consists of one ball and two paddles of varying geometries, all represented as object meshes in 3D space. We write N for the number of vertices, $X_t \in \mathbb{R}^{N\times 3}$ for their world-space positions at frame t, V_t for the corresponding per-vertex velocities. Time is discretized at 120 Hz, and a window of T frames is $X \in \mathbb{R}^{T\times N\times 3}$. Player input over that window is $A \in \{0,1\}^{T\times 8}$, ordered `W,A,S,D,UP,LEFT,DOWN,RIGHT`, with the first four keys bound to one paddle and the last four to the other; by convention A_t is the input that advances frame $t-1$ to frame t. Each material is a vector $y \in \mathbb{R}^7$ (section 3.2). We write $c = (X_0, V_0, y, A)$ for the full conditioning.

Pipeline. PHYSIPLAY is built in three stages, each initialized from the last. Stage 1 (section 3.1) is a bidirectional, material-conditioned model for future dynamics prediction, initialized from PhysiFormer. Stage 2 (section 3.2) adds action conditioning while the model is still bidirectional, and extends training to all six materials. Stage 3 (section 3.3) is a bidirectional-to-causal adaptation for autoregressive chunked generation.

3.1 STAGE 1: BIDIRECTIONAL DYNAMICS PRIOR

We initialize from PhysiFormer, which already predicts future mesh vertex trajectories in world coordinates from an initial position and velocity. Its weights therefore arrive with a prior over physically plausible motion—inertia, gravity, contact, and the difference between rigid and elastic response. PhysiFormer's factorized spatial, temporal, and object-level attention provides a natural initialization for PHYSIPLAY.

Generative Formulation. The model represents $p(X \mid c)$ for a window $X \in \mathbb{R}^{T\times N\times 3}$ and conditioning c. Let x be a trajectory drawn from it and ϵ a noise vector of matching shape, drawn from $\mathcal{N}(0, \sigma_n^2 I)$ with $\sigma_n = 0.1$. Training mixes the two into $z_\tau = \tau x + (1-\tau)\epsilon$ according to $\tau \in [0,1]$ (Lipman et al., 2022; Liu et al., 2023; Albergo & Vanden-Eijnden, 2023), so z_τ holds pure noise at $\tau = 0$ and pure data at $\tau = 1$. Its derivative in τ is the flow velocity $v(z_\tau, x, \tau) = (x - z_\tau)/(1-\tau)$ (unrelated to the vertex velocity V). Following JiT (Li & He, 2026) we learn it through a network $x_\theta(z_\tau, c, \tau)$ that emits the trajectory instead of the velocity, set $v_\theta = (x_\theta(z_\tau, c, \tau) - z_\tau)/(1-\tau)$,

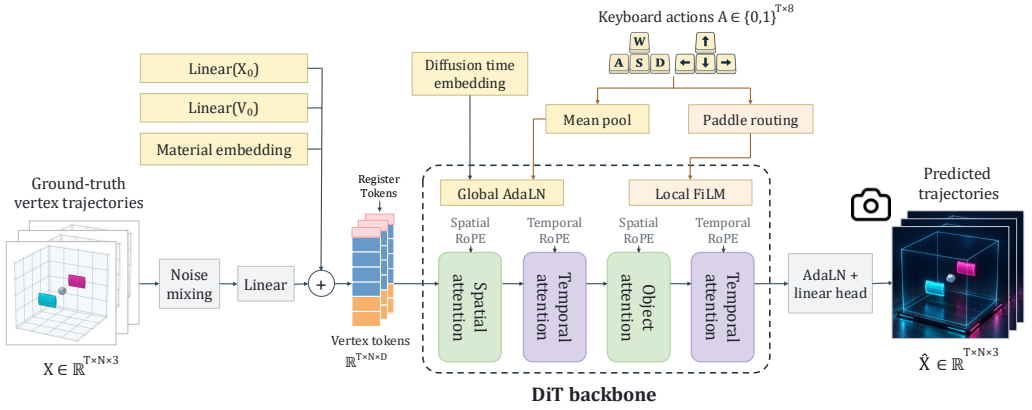


Figure 1: **Stage 2 architecture.** The action- and material-conditioned bidirectional model, over a window of $T = 81$ frames. Noised vertex trajectories are linearly embedded into tokens, conditioned additively on the first-frame position X_0, velocity V_0, and the material vector, then processed by the factorized DiT backbone before a linear head returns the predicted trajectory $\hat{X}$. The keyboard stream A enters on two paths: it is pooled across the scene and joined with the diffusion time embedding to form the global AdaLN modulation applied to every token. It is also routed per paddle to drive a local FiLM layer (Perez et al., 2018) acting only on that paddle's tokens.

and minimize

$$\mathcal{L}(\theta;\tau) = \mathbb{E}_{x,\epsilon} \left\| v_\theta(z_\tau, c, \tau) - v(z_\tau, x, \tau) \right\|^2 = \mathbb{E}_{x,\epsilon} \left\| \frac{x_\theta\big(\tau x + (1-\tau)\epsilon,\, c,\, \tau\big) - x}{1-\tau} \right\|^2, \quad (1)$$

drawing τ from a logit-normal law, $\mathrm{logit}(\tau) \sim \mathcal{N}(\mu, \sigma^2)$ with $\mu = -0.8$ and $\sigma = 0.8$. Generation integrates $dz_\tau/d\tau = v_\theta(z_\tau, c, \tau)$ from $\tau = 0$ to 1 with a 25-step Heun integrator. We follow PhysiFormer's x-prediction parameterization.

Diffusion Architecture. We adopt the PhysiFormer DiT-L backbone, with diffusion over vertex tokens. Blocks cycle through spatial attention within each frame, object-level attention within each object, and temporal attention along each vertex's trajectory, reducing complexity from $\mathcal{O}(T^2N^2)$ to $\mathcal{O}(TN^2 + NT^2)$. Without explicit object-identity embeddings, the architecture generalizes to more objects than seen during training. Rotary embeddings (Su et al., 2024) are 1D over frames in temporal blocks and conditioned on 3D coordinates in spatial blocks, while sixteen register tokens (Darcet et al., 2024) carry global context across the factorization. Trained on 3D geometries, PhysiFormer also generalizes well to complex shapes with substantially more vertices per object than seen during training. We retain this backbone to preserve these advantages.

3.2 Stage 2: Bidirectional Action-Conditioned Game Simulation

Starting from the pre-trained model of section 3.1, we augment it in two ways: an action-conditioning path injects the players' input into the backbone, and PhysiFormer's binary rigid/elastic material embedding is dropped in favor of a vector over seven physical properties, so that one set of weights spans a range of materials rather than two classes. The network architecture is depicted in fig. 1.

Material Conditioning. The ball takes one of six materials—metal, rubber, jello, foam, putty, and paste—each represented by a vector $y = (E, \nu, \rho, \zeta, e, \mu, \kappa) \in \mathbb{R}^7$ of Young's modulus, Poisson's ratio, density, material damping, restitution, friction, and plastic creep. The values are the simulator's design constants (table 4, section A). Because the material properties span substantially different scales, we first compress the positive, unbounded quantities before standardization: $\phi(y) = \big(\log_{10} E,\ \nu,\ \log_{10} \rho,\ \log(1+\zeta),\ e,\ \mu,\ \log(1+\kappa)\big)$. We then standardize $\phi(y)$ with its mean and standard deviation over the six presets, embed it with a two-layer MLP, and add the result to the ball's vertex tokens. The paddles are rigid and receive no material conditioning. We progressively

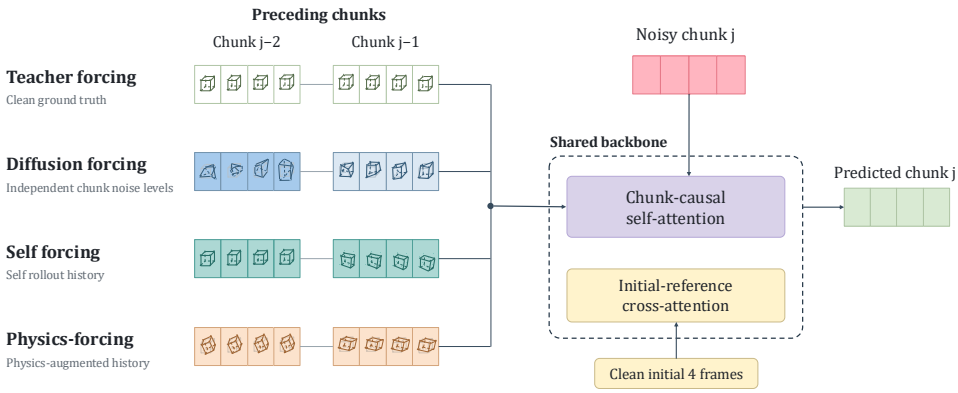


Figure 2: **Stage 3: bidirectional-to-causal adaptation.** The converged Stage 2 model is adapted to generate in four-frame chunks. Each denoiser call sees the two preceding chunks—eight completed frames—together with the noisy target chunk, and cross-attends to the clean first four frames of the episode as a fixed geometry reference. The backbone is shared across training regimes; only the provenance of the preceding-chunk context changes. Teacher forcing supplies clean ground truth, diffusion forcing (Chen et al., 2024) adds an independent noise level per chunk, self forcing (Huang et al., 2025) uses the model's own rollout to mitigate error accumulation, and physics forcing supplies a structured, geometrically perturbed history.

introduce new materials into the training data in the order listed in table 4, adding each once the training loss on previously introduced materials has plateaued.

Action Conditioning. Each paddle can move along a plane with two degrees of freedom specified with eight key states in A_t: `WASD` drives the front paddle and the arrow keys the back paddle, so the two players act concurrently on separate geometry within one scene. The action vector enters the network along two paths. Globally, A_t is pooled across the scene and joined with the diffusion-timestep embedding to produce an AdaLN modulation. Locally, the action vector is routed per paddle into a FiLM layer (Perez et al., 2018) that scales and shifts only the tokens of the paddle the corresponding half of A_t controls, so each player's input is applied to their own object rather than being averaged into a scene-wide signal.

Work on converting full-sequence video generators into streaming ones finds that a strong bidirectional teacher is an important starting point for the causal student (Huang et al., 2025; Zhu et al., 2026; Gao et al., 2026), so we spend most of the training budget here and treat section 3.3 as an adaptation of a converged model rather than training from scratch under a causal constraint.

3.3 Stage 3: Causal Adaptation

Causal Architecture. The bidirectional teacher is adapted into a chunked causal generator (fig. 2) that produces frames in blocks of four. Each denoiser call sees the eight most recently completed frames—the previous two chunks—together with the four noisy target frames, and attention is *block-causal*: the target chunk attends to itself bidirectionally and exclusively to all history within the window. Temporal positions are assigned locally over the twelve-frame input rather than by absolute frame number to allow generalized rollout beyond the training horizon.

The additive X_0, V_0 embeddings of the bidirectional model assume that the prediction window begins at a ground truth state, which no longer holds under rolling generation. We therefore remove both embedders and replace them with a per-vertex cross-attention reference built from the clean vertex positions of the episode's first four frames. Each position is split into the object's centroid and the vertex's offset from it, separating where an object is from what it looks like, alongside a finite-difference velocity and a learned tag identifying which of the four frames it came from. Every vertex then attends to its own four reference tokens in each subsequent chunk. The reference supplies a stable identity for each object's undeformed geometry no matter how far the rollout has run, while

the recent history supplies the dynamic state. Action conditioning is retained but global AdaLN is applied per frame: the four actions of a target chunk are supplied together, and each modulates its own frame. Because attention within a chunk is bidirectional, generation is causal at the level of chunks rather than individual frames.

Training Recipe. From each 81-frame trajectory we expand every valid target start $s = 8, \ldots, 77$, giving 70 training windows per trajectory, and score the loss only on the four target frames $s, \ldots, s+3$. Teacher forcing (TF) (Williams & Zipser, 1989) supplies clean ground-truth history. Diffusion forcing (DF) (Chen et al., 2024) supplies the same frames corrupted to a noise level drawn independently per four-frame chunk, so the model learns to denoise a target from history that is itself imperfect. Self forcing (SF) (Huang et al., 2025) supplies the model's own generated history, with gradients stopped through the rollout that produced it, so the loss trains the current prediction rather than the history it conditions on. The three therefore form a progression—clean, corrupted, self-generated—that narrows the gap to inference, where the history is always the model's own output. Noise levels follow the Stage 1 schedule, and the reference frames are never noised. We additionally introduce *physics forcing*, to inject structured geometric perturbations, such as small rotations and translations, so that the model learns to recover from coherent state errors rather than independent noise.

We begin from the teacher's EMA weights and proceed in four stages: $30{,}030$ updates at DF80/TF20; 980 at DF40/TF20/SF40, a further $4{,}200$ at the same mixture with a reduced learning rate; and a final $1{,}120$ at that mixture with physics forcing applied to 20% of the examples within each regime.

4 Experiments

4.1 Dataset

The game is two-player 3D pong in a $[-1, 1]^3$ court, with one ball and two rigid paddles, simulated with the Godot XPBD solver at $120\,\mathrm{Hz}$. Ball geometry is drawn from fifteen convex templates of 4–20 vertices and paddle geometry from five profiles of 6–24 vertices, giving 75 pairings and scenes of 16–68 vertices; each configuration draws a ball radius from $\mathcal{U}(0.06, 0.14)$, a launch speed from $\mathcal{U}(0.5, 2.0)$, and one of the six materials of table 4. A center-of-mass speed servo applies one uniform velocity correction per step so rallies do not decay to rest. Both paddles are driven by scripted agents that predict the ball's intercept but differ in reaction delay, decision rate and prediction noise, and deliberately miss a fixed fraction of returns, so both move concurrently through much of a rally. Trajectories are generated on the fly from a session seed in continuous chunks of 81 frames, together with initial velocities, per-vertex object indices, triangle topology, and an 81×8 binary action tensor over `WASD` and the arrow keys. Soft materials destabilize under fast impacts, so launch speed is capped at 1.25 for foam and 0.8 for paste, and chunks in which the solver reports a tetrahedral inversion are regenerated.

4.2 Implementation Details

Stage 1 initializes from the PhysiFormer DiT-L checkpoint and inherits the JiT noise schedule unchanged. Stage 2 introduces materials by a curriculum in the order of table 4. All stages use AdamW (Loshchilov & Hutter, 2019) with $\beta = (0.9, 0.95)$, gradient clipping at global norm 1.0, BF16 autocast with FP32 sampling state, and an EMA of the weights with decay 0.999, and are trained on two NVIDIA H200 GPUs. Stages 1 and 2 use no weight decay, learning rate 4×10^{-5} on a cosine schedule decaying to 5% of peak, and an effective batch of 64 (8 per GPU $\times$ 2 GPUs $\times$ 4 accumulation steps). Stage 3 restarts the optimizer from step zero with weight decay 0.01, learning rate 8×10^{-5}, 1000 warmup updates, an effective batch of 128, gradient checkpointing enabled, and 140 updates per geometry configuration; the teacher's optimizer, scheduler, and RNG state are not resumed, and the continuation stages of section 3.3 lower the learning rate further. Attention runs through PyTorch scaled-dot-product attention.

4.3 Evaluation Metrics

Following PhysiFormer (Chen et al., 2026), we report three metrics; full definitions are given in section B. All metrics are computed on generated frames only, excluding the eight observed frames.

Table 1: **Comparison against video diffusion baselines.** Appearance and motion metrics over generated frames 8–80 from 100 rendered 81-frame rollouts, comparing PHYSIPLAY with Wan2.1-14B (Wan et al., 2025), Causal Forcing (Zhu et al., 2026), and LingBot-World 2.0 (Gao et al., 2026). All methods are evaluated in pixel space under the same rendering viewpoint. **Best** results are shown in bold and *second-best* in italics.

Metric	PHYSIPLAY	Causal Forcing	Wan2.1-14B	LingBot-World 2.0
LPIPS ↓	**0.00264**	*0.142*	0.267	0.204
SSIM ↑	**0.993**	*0.874*	0.674	0.636
PSNR ↑	**38.1**	20.8	18.4	*21.7*
Foreground flow EPE ↓	**0.291**	*1.08*	1.59	1.36
Trajectory AUC ↑	**0.991**	*0.865*	0.782	0.535

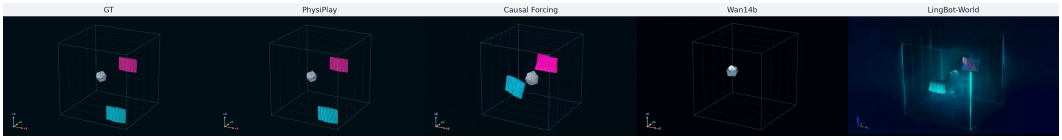


Figure 3: **Frame 80 of the 81-frame rollout.** Ground truth and each baseline, rendered under the same camera. PHYSIPLAY preserves both paddles and the ball; the baselines displace, drop, or dissolve them.

Trajectory error is the mean-square per-vertex position error against the reference (Shao et al., 2022; Sanchez-Gonzalez et al., 2020b). Because the model is generative, a rollout that diverges from the reference may still be physically valid, so this error is read alongside the physical metrics. **Rigidity error** is the residual of each object against the best rigid transform of its first-frame shape, fitted per frame by the Kabsch algorithm (Kabsch, 1976). Nothing in the architecture enforces rigidity, so this measures how well rigidity was learned; it exposes the typical failure of iterated predictors, in which a paddle shears or inflates. **Momentum-change error** compares predicted and ground-truth changes in ball momentum since the last observed frame, using the simulator's per-vertex masses and normalized by the ground-truth momentum scale. Collisions and the speed servo inject impulses by design, so this measures agreement with the reference dynamics rather than conservation.

4.4 QUALITATIVE RESULTS

Beyond the quantitative comparisons below, the supplemental webpage provides autoregressive rollouts of up to 243 frames for all six materials under matched initial states and action streams, illustrating qualitatively distinct deformation and recovery behaviors across materials.

4.5 BASELINES

Video Diffusion Models. We compare against text- and image-conditioned video generators, including Wan2.1-14B (Wan et al., 2025), Causal Forcing (Zhu et al., 2026), and LingBot-World 2.0 (Gao et al., 2026), each given an initial rendered frame of the scene and text prompts that describe the dynamics and actions as completely as possible. We evaluate several prompt formulations and use the best-performing prompt for each baseline. Their outputs are pixels, so the state metrics above cannot be computed directly. We therefore compare all four models in pixel space, on the same metal scenes and under the same camera, over 100 rollouts of 81 frames scored on the generated frames 8–80. LPIPS, SSIM, and PSNR measure appearance agreement—in perceptual features, local structure, and raw pixel error; foreground flow EPE measures optical-flow disagreement inside detected object regions, and trajectory AUC scores localization over time. None speaks to physical plausibility, which is what the state metrics of section 4.3 address. PHYSIPLAY obtains the best scores on all five metrics (table 1, fig. 3), though the video models are not specifically finetuned for this task.

Deterministic Autoregressive Methods. We train TIE (Shao et al., 2022) on our data as a representative next-state regressor over the same vertex representation. It has no action interface, so it is

Table 2: **State-space comparison against TIE and PhysCtrl.** Each block is one dataset—TIE's above, PhysCtrl's below—on which that baseline is compared against the same six-material PHYSIPLAY checkpoint; **best** per column within each block. Momentum normalization is dataset-specific, so rows are comparable only within a block.

Model	Position MSE ↓	Rigidity error ($\times 10^{-3}$) ↓	Momentum error ↓
TIE	**0.199**	38.4	**0.631**
PHYSIPLAY	0.204	**0.187**	0.712
PhysCtrl	**0.331**	0.470	**0.715**
PHYSIPLAY	0.608	**0.382**	0.742

trained on trajectories with the paddles stationary and omitted from its inputs; for the comparison we evaluate the six-material PHYSIPLAY checkpoint on the same data with idle actions, keeping the paddles present but stationary. Table 2 shows a characteristic failure mode of an iterated one-step regressor: TIE matches PHYSIPLAY on position to within 2.7%, but its rigidity error is over two hundred times larger, with the ball visibly shearing as the rollout proceeds.

Bidirectional Diffusion Models. PhysCtrl (Wang et al., 2025) generates full-sequence 3D point trajectories for single objects conditioned on material parameters and applied forces, sharing our world-space formulation. Using its officially released code, we retrain PhysCtrl on metal-only recordings of wall impacts. Each object is represented by $2{,}048$ surface points with fixed temporal correspondence, conditioned on the frame-zero geometry, initial velocity and center, and two material properties: $\log_{10} E$ and Poisson's ratio.

To accommodate PhysCtrl's single-object formulation, the training data excludes paddles. This comparison therefore evaluates dynamics prediction without paddle-based action conditioning. For consistency, we evaluate the same PHYSIPLAY checkpoint with paddles omitted from its input, a configuration unseen during its training. PhysCtrl's predicted points are mapped back to mesh vertices using the frame-zero barycentric weights and a least-squares inverse.

PHYSIPLAY achieves 18.7% lower rigidity error, whereas PhysCtrl achieves lower positional error. This result highlights PHYSIPLAY's ability to preserve object shape, key to physical plausibility, despite PhysCtrl's bidirectional full-sequence generation, which can favor long-horizon consistency.

4.6 AGENTIC SIMULATION BASELINE

With the rise of agentic capabilities in spatial reasoning, we prompt GPT-6 Astra (OpenAI, 2026) with the same initial frame and text prompts used for the video diffusion baselines. In one example, using high reasoning effort, Astra took 6 min 12 s to generate 80 future frames. The output video is nearly identical to the ground truth with stationary camera as shown in the supplemental index.html. When asked to describe its procedure, the model reported using inverse reconstruction for image-space motion estimation. We include this as an exploratory qualitative comparison rather than as a controlled, quantitative baseline.

4.7 CAUSAL ADAPTATION ABLATIONS

For controlled architectural ablations, all variants use the same metal-only teacher and the initial DF80/TF20 adaptation stage; these experiments are separate from the final six-material checkpoint. The PHYSIPLAY configuration uses four-frame chunks, eight frames of history, cross-attention to a fixed four-frame reference, per-frame AdaLN, paddle-local FiLM, and local temporal positions.

No variant performs best on every metric. We focus on the ball, whose motion emerges from contact and scene dynamics rather than being directly actuated by player inputs. PHYSIPLAY provides a favorable trade-off across trajectory accuracy, rigidity, momentum consistency, and inference cost (table 3). Single-frame chunks improve momentum tracking but are roughly four times slower, taking $145\,\mathrm{s}$ per 243-frame rollout against about $34.5\,\mathrm{s}$ for the four-frame variants, since they denoise

Table 3: **Causal adaptation ablations.** All variants adapt the same bidirectional teacher on metal-only data under the $80/20$ diffusion–teacher forcing schedule of section 3.3, evaluated on 50 held-out sequences at an identical sampling budget. Paddle rows average the two paddles, and momentum is ball-only (section B.3) as the paddles are kinematically driven. Three significant figures, lower is better; ball and paddle rows are ranked separately as they are not comparable—**best** bold, *second* italic.

		Position MSE ($\times 10^{-3}$) ↓			Rigidity error ($\times 10^{-5}$) ↓			Momentum error ↓		
Variant	Object	8–80	81–161	162–242	8–80	81–161	162–242	8–80	81–161	162–242
PHYSIPLAY	Ball	**2.19**	**165**	*596*	**1.62**	*6.97*	*9.31*	**0.352**	*0.765*	*0.887*
	Paddle	0.269	*0.752*	*2.54*	4.34	13.8	40.1	–	–	–
+ additive X_0, V_0	Ball	3.80	228	842	*2.69*	17.7	44.4	0.423	0.816	0.896
	Paddle	0.648	3.99	11.5	4.39	6.54	*7.31*	–	–	–
chunk size 1	Ball	2.88	*210*	676	5.98	17.5	12.2	*0.358*	**0.714**	**0.822**
	Paddle	*0.151*	**0.470**	**1.85**	**2.72**	**4.91**	30.7	–	–	–
full context	Ball	*2.85*	244	**584**	3.59	**5.29**	**6.97**	0.380	0.908	1.56
	Paddle	**0.0814**	50.3	163	*2.98*	*6.38*	**7.02**	–	–	–

once per frame rather than once per four, while full-context history improves rigidity at the cost of long-horizon paddle tracking and growing context.

Additive initial-state embeddings. Restoring the additive X_0, V_0 embeddings alongside the reference cross-attention is the worst variant on ball trajectory in the first and last windows, on both position and rigidity. Once the rollout passes its first window, the additive path is fed generated state, so its error is re-injected at every chunk; this supports removing it rather than keeping it beside the fixed reference. However, it improves paddle rigidity beyond the training horizon.

Chunk size. Single-frame chunks give the strictest causal factorization, at the inference cost noted above. They yield the best paddle tracking and momentum error in both extrapolation windows—a one-frame target is the easiest prediction—but are worse on ball position in every window and markedly worse on ball rigidity, suggesting that denoising four frames jointly helps the model commit to a contact and its consequences.

Context length. Full-context history gives the best ball rigidity in both extrapolation windows and the best paddle rigidity in the last, so a longer view helps shape preservation, but its paddle position collapses once the rollout passes the training horizon and its momentum error exceeds unity in the final window. Its measured wall-clock is indistinguishable from the eight-frame window (2 chunks of 4 frames) at this horizon, though its context, and hence its cost, grows with the length of the rally. We keep the eight-frame window for its constant cost and trajectory accuracy, and note that the paddle rigidity of the full model is the weakest of the four in both extrapolation windows, which the longer context and the additive anchor both improve.

5 CONCLUSION

We presented PHYSIPLAY, which makes world-space trajectory diffusion interactive. Three changes carry the result: generation in four-frame chunks along a block-causal axis from a fixed-size window; a persistent cross-attention reference in place of additive initial-state conditioning; and actions attached to named sets of vertices, where action effects can propagate to other objects. Across six materials in one set of weights, two agents act on a shared scene over rollouts three times the teacher's window, and because the output is geometry rather than pixels, rigidity and momentum are measured rather than inspected. In pixel-space comparisons with general-purpose video generators, PHYSIPLAY obtains the best scores on all five metrics; in the restricted stationary-paddle comparison with TIE, it achieves comparable position error while substantially reducing rigidity error. As a neural formulation, PHYSIPLAY may enable gradient-based action optimization under suitable continuous action parameterizations. Future work includes testing this, extending rollout horizons, achieving real-time inference, and handling richer scenes (section D).

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementation and debugging, including debugging of the data-generation pipeline; to provide feedback on experimental design and result interpretation; and to polish parts of the manuscript for clarity and readability. We also used AI-assisted coding for evaluation and visualization utilities. All AI-assisted code and suggestions were reviewed and validated by the authors, and all reported results were produced and checked using our own experimental and evaluation pipelines. We manually verified all cited sources and did not rely on generative AI to formulate mathematical claims or proofs. We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of AI.

REPRODUCIBILITY STATEMENT

The authors have included data, training, and architecture detail and plan to release model training, inference and data generation code.

REFERENCES

Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *Proc. ICLR*, 2023.

Kelsey R Allen, Yulia Rubanova, Tatiana Lopez-Guevara, William Whitney, Alvaro Sanchez-Gonzalez, Peter Battaglia, and Tobias Pfaff. Learning rigid dynamics with face interaction graph networks. *arXiv preprint arXiv:2212.03574*, 2022.

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pp. 767–778. Association for Computing Machinery, New York, NY, USA, 2023.

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29, 2016.

Saakaar Bhatnagar, Yaser Afshar, Shaowu Pan, Karthik Duraisamy, and Shailendra Kaushik. Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64(2):525–545, 2019.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: Fusing constraint projections for fast simulation. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pp. 787–797. ACM, 2023.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024.

Wei Cao, Chang Luo, Biao Zhang, Matthias Nießner, and Jiapeng Tang. Motion2vecsets: 4d latent vector set diffusion for non-rigid shape reconstruction and tracking. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20496–20506. IEEE, 2024.

Boyuan Chen, Yilun Du, Diego Martí, et al. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *NeurIPS*, 2024.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6178–6189. IEEE, 2025.

Yiming Chen, Yushi Lan, and Andrea Vedaldi. Physiformer: Learning to simulate mechanics in world space. *arXiv preprint arXiv:2606.27364*, 2026.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. *Proc. ICLR*, 2024.

Mathieu Desbrun and Marie-Paule Gascuel. Smoothed particles: A new paradigm for animating highly deformable bodies. In *Computer Animation and Simulation'96: Proceedings of the Eurographics Workshop in Poitiers, France, August 31–September 1, 1996*, pp. 61–76. Springer, 1996.

Zhiyang Dou, Minghao Guo, Haixu Wu, Doug Roble, Tuur Stuyck, and Wojciech Matusik. Rigidformer: Learning rigid dynamics using transformers. *arXiv preprint arXiv:2605.09196*, 2026.

Zelin Gao, Qiuyu Wang, Jiapeng Zhu, Jingye Chen, Zichen Liu, Qingyan Bai, Jiahao Wang, Yufeng Yuan, Hanlin Wang, Yichong Lu, Ka Leong Cheng, Haojie Zhang, Jian Gao, Tianrui Feng, Yuzheng Liu, Yao Yao, Yinghao Xu, Xing Zhu, Yujun Shen, and Hao Ouyang. Infinite worlds with versatile interactions. *arXiv preprint arXiv:2607.07534*, 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, 2025. URL `https://arxiv.org/abs/2505.19386`.

Xiaoxiao Guo, Wei Li, and Francesco Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 481–490, 2016.

Anthony Hu, Václav Volhejn, Adrien Ramanana Rahary, Chris Mulder, Aditya Makkar, Alyx Liao, Amélie Royer, Manu Orsini, Adam Jelley, Eloi Alonso, et al. Multiplayer interactive world models with representation autoencoders. *arXiv preprint arXiv:2607.05352*, 2026.

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. *arXiv preprint arXiv:1910.00935*, 2019.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

Krishna Murthy Jatavallabhula, Miles Macklin, Florian Golemo, Vikram Voleti, Linda Petrini, Martin Weiss, Breandan Considine, Jérôme Parent-Lévesque, Kevin Xie, Kenny Erleben, et al. gradsim: Differentiable simulation for system identification and visuomotor control. *arXiv preprint arXiv:2104.02646*, 2021.

Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *Acm siggraph 2016 courses*, pp. 1–52. 2016.

Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. Phystwin: Physics-informed reconstruction and simulation of deformable objects from videos. *ICCV*, 2025.

Ying Jiang, Chang Yu, Tianyi Xie, Xuan Li, Yutao Feng, Huamin Wang, Minchen Li, Henry Lau, Feng Gao, Yin Yang, et al. Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality. In *ACM SIGGRAPH 2024 conference papers*, pp. 1–1, 2024.

Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Foundations of Crystallography*, 32(5), 1976.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *CVPR*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv.cs*, abs/2210.02747, 2022.

Fangfu Liu, Kai He, Tianchang Shen, Tianshi Cao, Sanja Fidler, Yueqi Duan, Jun Gao, Igor Gilitschenski, Zian Wang, and Xuanchi Ren. Gamma-world: Generative multi-agent world modeling beyond two players. *arXiv preprint arXiv:2605.28816*, 2026a.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. In *International Conference on Learning Representations*, volume 2026, pp. 91177–91196, 2026b.

Shaowei Liu, Zhongzheng Ren, Saurabh Gupta, and Shenlong Wang. Physgen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision (ECCV)*, 2024.

Tiantian Liu, Adam W Bargteil, James F O'Brien, and Ladislav Kavan. Fast simulation of mass-spring systems. *ACM Transactions on Graphics (TOG)*, 32(6):1–7, 2013.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *Proc. ICLR*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proc. ICLR*, 2019.

Xinyin Ma, Julius Berner, Chao Liu, Arash Vahdat, Weili Nie, and Xinchao Wang. Flex-forcing: Towards a unified autoregressive and bidirectional video diffusion model. *arXiv preprint arXiv:2607.03509*, 2026.

Miles Macklin and Matthias Müller. Position based fluids. *ACM Transactions on Graphics (TOG)*, 32(4):1–12, 2013.

OpenAI. GPT-6 Astra System Card. Technical report, OpenAI, September 2026. URL `https://deploymentsafety.openai.com/gpt-6-astra`. Accessed: 2026-09-23.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proc. AAAI*, 2018.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International conference on learning representations*, 2020.

Alexander Pondaven, Ziyi Wu, Igor Gilitschenski, Philip Torr, Sergey Tulyakov, Fabio Pizzati, and Aliaksandr Siarohin. Actionparty: Multi-subject action binding in generative video games. In *European Conference on Computer Vision*, pp. 74–92. Springer, 2026.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Yulia Rubanova, Tatiana Lopez-Guevara, Kelsey R. Allen, William F. Whitney, Kimberly Stachenfeld, and Tobias Pfaff. Learning rigid-body simulators over implicit shapes for large-scale scenes and vision. In *Proc. NeurIPS*, volume k, 2024.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *ICML*, pp. 8459–8468. PMLR, 2020a.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. In *Proc. ICML*, 2020b.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

Yidi Shao, Chen Change Loy, and Bo Dai. Transformer with implicit edges for particle-based physics simulation. In *Proc. ECCV*, 2022.

Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024.

Xiyang Tan, Ying Jiang, Xuan Li, Zeshun Zong, Tianyi Xie, Yin Yang, and Chenfanfu Jiang. Physmotion: Physics-grounded dynamics from a single image. In *2026 International Conference on 3D Vision (3DV)*, pp. 806–818. IEEE, 2026.

DreamX Team, Yancheng Bai, Rui Chen, Xiangxiang Chu, Rujing Dang, Hao Dou, Bingjie Gao, Qiwen Gu, Siyu Hong, Jiachen Lei, et al. Dreamx-world 1.0: A general-purpose interactive world model. *arXiv preprint arXiv:2606.16993*, 2026.

Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. In *International Conference on Learning Representations*, volume 2025, pp. 73754–73776, 2025.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Caoliwen Wang, Minghao Guo, Siyuan Chen, Heng Zhang, Mengdi Wang, Xingyu Ni, Hanson Sun, Kunyi Wang, Zherong Pan, Kui Wu, et al. Worldparticle: Unified world simulation of lagrangian particle dynamics via transformer. *arXiv preprint arXiv:2605.15305*, 2026.

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. *Advances in Neural Information Processing Systems*, 38:167907–167932, 2025.

Amaury Wei and Olga Fink. Integrating physics and topology in neural networks for learning rigid body dynamics. *Nature Communications*, 16(1), 2025.

Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. *arXiv preprint arXiv:2311.12198*, 2023.

Tianyi Xie, Yiwei Zhao, Ying Jiang, and Chenfanfu Jiang. Physanimator: Physics-guided generative cartoon animation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10793–10804. IEEE, 2025a.

Ziyang Xie, Zhizheng Liu, Zhenghao Peng, Wayne Wu, and Bolei Zhou. Vid2sim: Realistic and interactive simulation from video for urban navigation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1581–1591. IEEE, 2025b.

Youn-Yeol Yu, Jeongwhan Choi, Woojin Cho, Kookjin Lee, Nayong Kim, Kiseok Chang, ChangSeung Woo, Ilho Kim, SeokWoo Lee, Joon Young Yang, et al. Learning flexible body collision dynamics with hierarchical contact mesh transformer. In *Proc. ICLR*, 2024.

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 *European Conference on Computer Vision*, pp. 388–406. Springer, 2024.

Yu Zhang, Yidi Shao, Wenqi Ouyang, Yushi Lan, Zhexin Liang, Chengrui Wu, Xudong Xu, and Xingang Pan. Clothtransformer: Unified latent-space transformers for scalable cloth simulation. *arXiv preprint arXiv:2605.27852*, 2026.

Licheng Zhong, Hong-Xing Yu, Jiajun Wu, and Yunzhu Li. Reconstruction and simulation of elastic objects with spring-mass 3d gaussians. *European Conference on Computer Vision (ECCV)*, 2024.

Hongzhou Zhu, Min Zhao, Guande He, Hang Su, Chongxuan Li, and Jun Zhu. Causal forcing: Autoregressive diffusion distillation done right for high-quality real-time interactive video generation. *arXiv preprint arXiv:2602.02214*, 2026.

Olgierd Cecil Zienkiewicz, Robert Leroy Taylor, and Jian Z Zhu. *The finite element method: its basis and fundamentals*. Elsevier, 2013.

APPENDIX

A MATERIAL PRESETS

Table 4 gives the exact seven-property vector behind each of the six materials referred to in section 3.2. These are the simulator's declared design constants rather than quantities fitted to trajectories: the conditioning vector and the solver that generated the motion therefore cannot disagree, and we verify this on every exported chunk by comparing the stored vector against the declared preset. Materials are introduced to training in the order listed.

Table 4: **The six material presets.** Each material is a fixed seven-property vector: Young's modulus E, Poisson's ratio ν, density ρ, material damping ζ, restitution e, friction μ, and plastic creep κ.

		E (Pa)	ν	ρ (kg m^{-3})	ζ	e	μ	κ
1	Metal	2×10^{11}	0.300	7900	0.2	0.80	0.2	0
2	Rubber	1.5×10^{4}	0.490	1100	2.0	0.90	0.8	0
3	Jello	5×10^{3}	0.495	1000	0.3	0.30	0.5	0
4	Foam	1×10^{2}	0.050	30	25.0	0.10	0.7	0
5	Putty	1×10^{3}	0.490	1000	5.0	0.15	0.7	1
6	Paste	3×10^{2}	0.490	1000	12.0	0.05	0.8	8

B EVALUATION METRIC DEFINITIONS

This section gives the full definitions summarized in section 4.3. Throughout, $X, \hat{X} \in \mathbb{R}^{T\times N\times 3}$ are the ground-truth and predicted trajectories, $\mathcal{F}$ is the mesh face set and $\mathrm{cc}(\mathcal{F})$ its connected components, and O denotes one such component. Frames are zero-indexed and Δt is taken from the recorded simulation frequency of 120 Hz. Rollouts begin from eight observed frames, so frames 0–7 are excluded from every metric and $r = 7$ denotes the last observed frame. We report three evaluation windows, $\mathcal{W} \in \{\,8\text{–}80,\ 81\text{–}161,\ 162\text{–}242\,\}$, inclusive; frame differences are taken continuously across window boundaries rather than restarting at each one. We write $\mathcal{W}_{\mathrm{all}} = \{8, \ldots, 242\}$ for their union.

B.1 TRAJECTORY MSE

Given ground-truth and predicted trajectories, the mean-square per-vertex position error is

$$\mathcal{L}_{\mathrm{MSE}}(\hat{X} \mid X) = \frac{\|\hat{X} - X\|_2^2}{TN}. \tag{2}$$

B.2 RIGIDITY PRESERVATION

For each object and frame we fit the best rigid transform from that object's first-frame configuration, with rotation $R \in SO(3)$ and translation $\boldsymbol{b} \in \mathbb{R}^3$ obtained by the Kabsch algorithm (Kabsch, 1976), and average the residual over objects and frames,

$$\mathcal{L}_{\mathrm{rigid}}(\hat{X} \mid X_0, \mathcal{F}) = \frac{1}{T}\sum_{t=1}^{T}\sum_{O\in \mathrm{cc}(\mathcal{F})}\frac{1}{|O|}\min_{(R,\boldsymbol{b})\in SE(3)}\left\|\hat{X}_{t,O} - X_{0,O}R - \boldsymbol{b}\right\|_2^{\mathcal{F}}, \tag{3}$$

where $X_{*,O}$ denotes the subset of vertices belonging to component O.

B.3 MASS-WEIGHTED MOMENTUM-CHANGE ERROR

Per-Vertex Masses. We evaluate motion using the per-vertex masses assigned by the Godot XPBD simulator rather than assuming uniform mass. For each tetrahedron e, its rest mass is computed from

the material density ρ and undeformed volume V_e^0 and distributed equally among its four vertices, so the lumped mass of vertex i is

$$m_i = \sum_{e\,:\,i\in e} \frac{\rho\, V_e^0}{4}. \tag{4}$$

These masses are generally nonuniform, because vertices participate in different numbers of tetrahedra. We recover the simulator's actual masses and map them onto the canonical vertex ordering of the exported mesh. They remain fixed throughout evaluation and are shared between the ground-truth and predicted trajectories; they are never recomputed from predicted geometry, which would let a model reduce its own error by deforming. In the evaluated dataset every XPBD particle is represented by an exported surface vertex.

Momentum Estimate. For either trajectory, ball momentum is estimated by backward finite differences,

$$\mathbf{P}(t) = \sum_i m_i \, \frac{\mathbf{x}_i(t) - \mathbf{x}_i(t-1)}{\Delta t}, \tag{5}$$

which estimates momentum over the recording interval rather than instantaneous substep momentum.

Momentum Error. Over an evaluation window $\mathcal{W}$ we measure agreement between the predicted and ground-truth momentum changes relative to the fixed reference frame $r = 7$,

$$\mathcal{L}_{\text{mom}}(\mathcal{W}) = \frac{1}{N P_{\text{RMS}}} \sum_{n=1}^{N} \sqrt{\frac{1}{|\mathcal{W}|} \sum_{t\in\mathcal{W}} \Big\| \big[\widehat{\mathbf{P}}_n(t) - \widehat{\mathbf{P}}_n(r)\big] - \big[\mathbf{P}_n(t) - \mathbf{P}_n(r)\big] \Big\|_2^2}. \tag{6}$$

The normalization is derived exclusively from ground-truth trajectories,

$$P_{\text{RMS}} = \sqrt{\frac{1}{N_{\text{GT}}} \sum_{n=1}^{N_{\text{GT}}} \frac{1}{|\mathcal{W}_{\text{all}}|} \sum_{t\in\mathcal{W}_{\text{all}}} \big\|\mathbf{P}_n(t)\big\|_2^2}, \tag{7}$$

where N_{GT} is the complete fixed evaluation dataset and N the evaluated sample set. The same normalization constant is used across models, evaluation subsets, and temporal windows, so values are comparable throughout.

Only the ball contributes to this metric: the paddles are kinematically controlled and carry no finite simulated mass. The metric therefore measures agreement with the ground-truth momentum changes, not momentum conservation, since collisions and the speed control of section 4.1 introduce external impulses by design. Lower values indicate better agreement.

C Extended Method Details

This section collects the fuller discussion of the method that is summarized in section 3.

Object-level versus viewpoint control. Nothing in the construction of section 3.2 is specific to paddles or keyboards: any control signal that can be associated with a subset of vertices can be injected the same way, which makes the scheme a general interface between an agent and scene dynamics rather than a pong-specific mechanism. Both action paths are applied per frame rather than pooled over the window, which matters once generation becomes causal and the four actions inside a chunk may differ.

Causal adaptation. The four-frame reference is encoded once at the start of the episode and is never refreshed, so no generated state ever enters it; this is what allows it to anchor each object's undeformed geometry however far the rollout has run.

D Limitations and Future Work

Long-Horizon Drift. Rollouts remain plausible to roughly three times the bidirectional teacher's 81-frame window, after which physical plausibility degrades: deformation accumulates without fully

recovering, and rally speed departs from the reference. The fixed geometry reference bounds the loss of object identity but does not eliminate drift in the dynamic state, which is carried entirely by the eight-frame history. The most direct remedy is a teacher trained on longer sequences, which would expose the prior to more of the game's structure—multi-contact rallies, wall sequences, and the slow phases between impacts—before adaptation begins, and would extend the horizon over which the causal student is interpolating rather than extrapolating.

Inference Cost. Generation solves an ODE per chunk, which at 25 Heun intervals is 49 denoiser calls for four frames. That is far below the cost of the equivalent per-frame factorization but still short of the budget a played game allows. Step distillation, or a consistency objective over the chunk-level sampler, is the natural next step and is orthogonal to everything described here.

Scope. The results are demonstrated on a single game with three objects and convex geometry. The action interface is general—any control signal associable with a subset of vertices is injected the same way—but concave contact, articulated bodies, and scenes whose object count changes during play are untested.