PHYSDUO: LEARNING PAIRWISE PHYSICS FOR COMPOSITIONAL MULTI-OBJECT SIMULATION

Anonymous authors
Paper under double-blind review

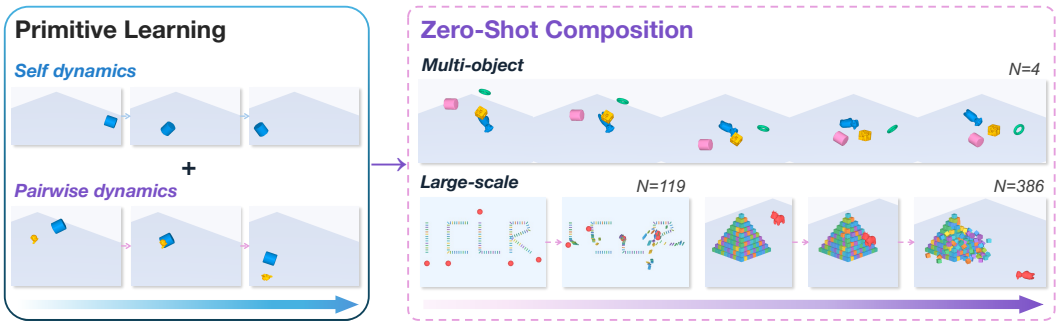

Figure 1: We propose **PhysDuo**, which learns self dynamics from single-object trajectories and pairwise contact dynamics from two-object interactions, then composes these learned primitives to simulate multi-object scenes with substantially more objects without additional training.

ABSTRACT

Generalizing learned rigid-body dynamics across physical timesteps and object counts remains challenging. We introduce PhysDuo, a compositional point-cloud simulator that decomposes rigid-body dynamics into two complementary components: per-object self dynamics and pairwise contact response. A timestep-conditioned self-dynamics model autoregressively predicts each object's motion under external forces and floor contact, using timestep conditioning and temporally scaled motion residuals. With this model frozen, a pairwise contact model learns contact-induced corrections to the predicted motion of two objects. A linear momentum constraint couples their translational corrections, while a symmetric contact gate suppresses updates for non-interacting pairs. Both models reconstruct rigid motion from updated anchor positions and maintain angular velocity as an explicit state. During multi-object inference, the shared contact model is applied sequentially to object pairs using their latest states, allowing subsequent interactions to incorporate earlier contact responses. This composition enables the same learned self and pairwise dynamics to be reused across scenes with varying object counts without changing the model parameters. We evaluate PhysDuo across physical timesteps, rollout horizons, and object counts, and further demonstrate zero-shot composition in scenes containing substantially more objects than seen during training. Ablations examine timestep conditioning, contact modeling, and pairwise composition.

1 INTRODUCTION

Predicting how objects move and interact is fundamental to robotics, graphics, and physically grounded content generation. Learned simulators model particle, mesh, and object dynamics through shared relational computations (Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020; Pfaff et al., 2021). Recent approaches incorporate geometric contact representations for rigid bodies (Allen et al., 2023b; Wei & Fink, 2025). Transformer-based models further support mesh-free rigid dynamics (Dou et al., 2026) and world-coordinate vertex trajectory generation (Chen et al., 2026). These advances raise a practical question: *how can dynamics learned from simple interactions be reused across scenes with more objects and different physical time intervals?*

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.

Reusing learned dynamics requires accounting for both scene composition and the duration of each prediction step. As object count grows, contacts become coupled: an object's response to one interaction changes the state from which it encounters another. Thus, simply applying independently predicted pairwise interactions in parallel does not capture the sequential dependence among contacts. Changing the timestep also changes the motion accumulated over each prediction interval. Compositional models already share pairwise interactions across object configurations (Chang et al., 2017), and EGODE (Yuan et al., 2024) separates continuous evolution from collision events. RigidFormer also conditions rigid-body prediction on the integration timestep (Dou et al., 2026). However, existing learned simulators typically learn free motion and inter-object interaction within a common multi-object prediction model. We instead explicitly factorize these two sources of motion during training: learning individual motion from single-object trajectories and contact-induced corrections from two-object trajectories, then composing the learned primitives in larger scenes.

We introduce PhysDuo, a compositional simulator that factorizes rigid-body motion into reusable self-dynamics and pairwise contact components (Figure 2). The self-dynamics model predicts motion under external forces and floor contact, independently of other objects. After training the self-dynamics component, we freeze it and train a shared pairwise model to predict only the motion correction induced by inter-object contact. At inference, the shared contact model is repeatedly applied to interacting pairs, with each correction immediately updating the state used by subsequent interactions. The same learned self and pairwise components can therefore be composed in scenes containing more objects than those used during training, without introducing object-count-dependent parameters.

Our self-dynamics model adapts a spatiotemporal point-cloud architecture (Wang et al., 2025) to autoregressive prediction with a bounded state history. Continuous timestep conditioning specifies the physical prediction interval, while position residuals relative to constant-velocity extrapolation and angular velocity residuals are scaled according to their temporal dependence. The contact model jointly represents both objects' geometry and motion to predict surface-anchor corrections and angular velocity changes. A linear momentum constraint couples the translational corrections, and a symmetric contact gate controls whether each pairwise update is applied. Both components recover rigid transformations from their anchor predictions to preserve object rigidity during rollout.

We evaluate PhysDuo on MOVi (Greff et al., 2022) and additional scenes with more objects than those used in training, examining temporal resolution, rollout horizon, and object count. Comparisons and ablations assess timestep conditioning, pairwise contact modeling, and sequential composition of multiple objects.

Our main contributions are:

- A compositional training formulation that factorizes rigid-body dynamics into self motion learned from single-object trajectories and contact-induced corrections learned from two-object trajectories, enabling the same learned primitives to be reused in multi-object scenes.
- An autoregressive self-dynamics model with continuous physical-timestep conditioning and temporally scaled position and angular-velocity residuals, allowing one model to operate across multiple simulation step sizes.
- A shared pairwise contact model with anchor-based rigid reconstruction, momentum-constrained translational corrections, and symmetric contact gating, which is sequentially composed using the latest object states at multi-object inference.

2 Related Work

Learned physical simulation. HRN and DPI-Net (Mrowca et al., 2018; Li et al., 2019) use hierarchical particle representations for interacting rigid and deformable materials, while GNS and MeshGraphNets (Sanchez-Gonzalez et al., 2020; Pfaff et al., 2021) learn particle and mesh dynamics through message passing. TIE (Shao et al., 2022) models interactions through attention without explicitly storing edge features, and constraint-based simulators (Rubanova et al., 2022) obtain future states by iteratively optimizing learned constraints. Graph networks can also learn discontinuous rigid contact dynamics (Allen et al., 2023a). FIGNet (Allen et al., 2023b) models mesh-face interactions, whereas HOPNet (Wei & Fink, 2025) combines higher-order topological representations with physically motivated message passing. HCMT (Yu et al., 2024) propagates collision effects

through hierarchical mesh Transformers for flexible bodies; SDF-Sim (Rubanova et al., 2024) uses implicit shapes for scalable rigid-body simulation. Point-cloud approaches employ multiscale continuous convolutions (Kim & Fuxin, 2024) or, in RigidFormer (Dou et al., 2026), object-centric attention and compact anchors, with Kabsch alignment enforcing rigidity and timestep conditioning supporting variable step sizes. We also represent motion through surface anchors, but separately learn single-object dynamics and pairwise contact corrections.

Relational and compositional dynamics. Interaction Networks (Battaglia et al., 2016) and the graph-network framework (Battaglia et al., 2018) establish shared object–relation computations for physical reasoning. The Neural Physics Engine (Chang et al., 2017) factorizes dynamics into pairwise interactions for generalization across object counts and scene configurations. Related approaches extend graph models to prediction, system identification, and control (Sanchez-Gonzalez et al., 2018) or infer latent interaction graphs from trajectories (Kipf et al., 2018). Physical inductive biases include gravity-aware subequivariance (Han et al., 2022a) and learned pairwise interactions coupled with deterministic physical updates (Han et al., 2022b). EGODE (Yuan et al., 2024) separates continuous evolution from collision events, while differentiable contact models augment Lagrangian and Hamiltonian networks (Zhong et al., 2021). PhysDuo differs in how the dynamics are factorized and learned: self-dynamics are trained from single-object trajectories and then frozen, while a shared contact model learns residual interaction dynamics from two-object trajectories. At inference, these corrections compose sequentially using the latest object states, reusing the same contact model across scene sizes.

Generative models of physical dynamics. Diffusion (Ho et al., 2020) and flow matching (Lipman et al., 2023) provide frameworks for conditional generation. Diffuser (Janner et al., 2022) models trajectories for planning, while DiT (Peebles & Xie, 2023) provides a scalable transformer backbone for diffusion. PhysCtrl (Wang et al., 2025) models point trajectories conditioned on material properties and external forces for controllable video generation. PhysiFormer (Chen et al., 2026) uses a diffusion transformer to generate rigid and elastic trajectories in world coordinates. We adapt PhysCtrl's spatiotemporal point-cloud architecture to autoregressive prediction with bounded history, continuous physical-timestep conditioning, and explicit angular velocity prediction, while handling contact through the separately trained pairwise model.

Training for autoregressive rollout. Using predicted histories at inference can shift the input distribution relative to training with ground-truth states. Scheduled sampling (Bengio et al., 2015) introduces predicted contexts, while Professor Forcing (Lamb et al., 2016) adversarially aligns teacher-forced and free-running recurrent dynamics. Diffusion Forcing (Chen et al., 2024) trains causal denoising with independent token noise levels, and Self Forcing (Huang et al., 2025) trains autoregressive video diffusion on self-generated sequences with sequence-level supervision. For physical prediction, GNS (Sanchez-Gonzalez et al., 2020) uses noisy training states, while PDE-Refiner (Lippe et al., 2023) improves rollout accuracy through multistep denoising refinement. Our contact model uses short histories generated by the frozen self-dynamics model as fixed conditioning, exposing contact prediction to motion errors encountered during autoregressive simulation.

3 METHOD

PhysDuo decomposes rigid-body dynamics into per-object self dynamics and pairwise contact response (Figure 2). After defining the problem, we introduce the timestep-conditioned self-dynamics model (Sec. 3.2) and the pairwise contact model (Sec. 3.3). We then describe their two-stage training and sequential composition for multi-object inference (Sec. 3.4).

3.1 PROBLEM SETTING

We consider N rigid objects, each represented at step t by a surface point cloud $x_t^i \in \mathbb{R}^{P\times 3}$ with fixed point identities and an angular velocity $\omega_t^i \in \mathbb{R}^3$. Given each object's history $\mathcal{H}_t^i$ of at most H states and physical conditions c_i (mass, friction, restitution, gravity, external forces, and floor height), we predict the scene state after a physical interval Δt.

In this paper, we formulate scene dynamics as per-object self dynamics followed by pairwise contact corrections. A shared self-dynamics model f_θ predicts each object's motion under external forces

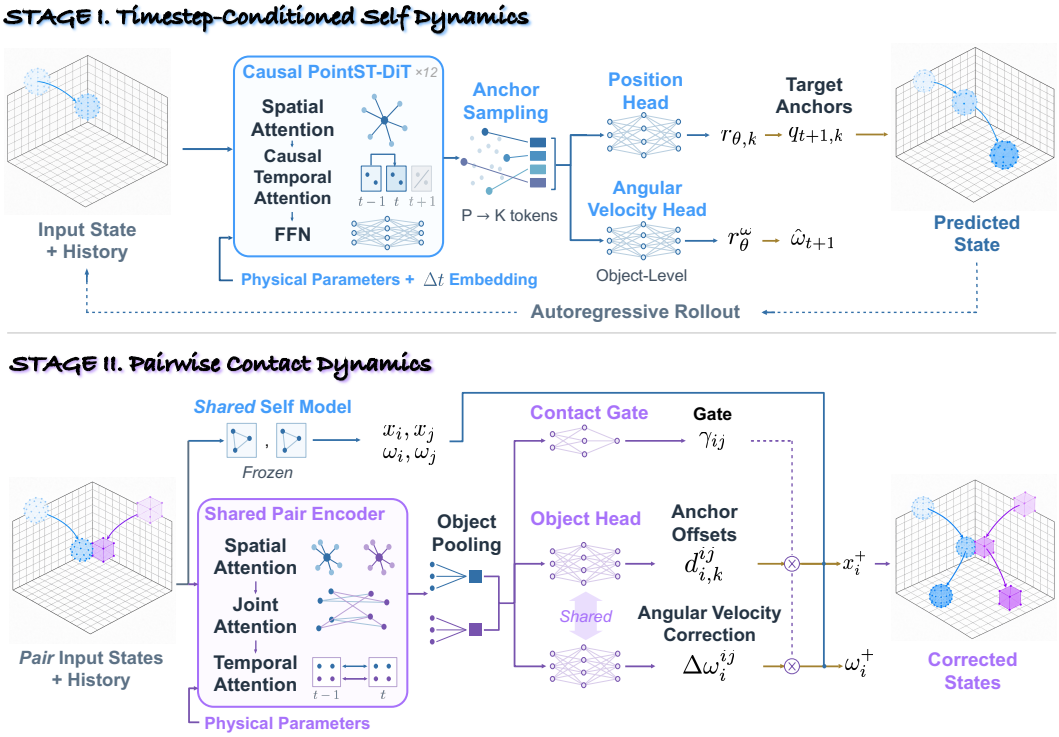

Figure 2: **Pipeline of PhysDuo. Stage I** learns timestep-conditioned self dynamics, predicting anchor residuals relative to constant-velocity extrapolation and updating angular velocity. **Stage II** freezes the self model and learns pairwise contact corrections, with a linear momentum constraint on translational corrections and a contact gate controlling their interaction. The updated point positions and angular velocities are fed back as history for autoregressive rollout.

and floor contact, while a contact solver $\mathcal{S}_\phi$ composes pairwise corrections:

$$\{(\hat{x}^i_{t+1}, \hat{\omega}^i_{t+1})\}_{i=1}^N = \mathcal{S}_\phi\Big(\{f_\theta(\mathcal{H}^i_t, c_i, \Delta t)\}_{i=1}^N, \{\mathcal{H}^{ij}_t, c_i, c_j\}_{i<j}, \Delta t\Big), \tag{1}$$

where $\mathcal{H}^{ij}_t$ denotes the joint history of objects i and j, and $t+1$ denotes the next simulation step.

In this formulation, f_θ is shared across objects, and $\mathcal{S}_\phi$ applies a shared contact model to each pair, keeping the learned parameters independent of object count. Both the self-dynamics and contact models predict residuals at K anchors for rigid motion reconstruction, along with an object-level angular velocity residual.

3.2 Timestep-Conditioned Self Dynamics

Conditional autoregressive dynamics. As illustrated in Stage I of Figure 2, the self-dynamics model predicts each object's motion under external forces and floor contact, independently of other objects. We adapt the spatiotemporal architecture of PhysCtrl (Wang et al., 2025) into a causal PointST-DiT with rotary temporal positional encoding (Su et al., 2024) for autoregressive prediction. Shared across objects, the model predicts each object's next point positions and angular velocity, conditioned on its most recent H states, physical attributes c_i, and physical timestep Δt. During rollout, predicted states are appended to the history, retaining at most H states. We train the model with flow matching (Lipman et al., 2023), where the flow time τ is distinct from the physical timestep Δt. We omit the object index i below for clarity.

Continuous timestep conditioning. To accommodate different physical time intervals, we condition the model on the normalized timestep $\rho = \Delta t / \Delta t_{\text{ref}}$, where $\Delta t_{\text{ref}} > 0$ is a chosen reference time interval used for normalization. We encode the timestep through a learnable MLP f_ϕ:

$$e_{\Delta t} = f_\phi(\log \rho) - f_\phi(0). \tag{2}$$

This embedding is added to the flow-time embeddings of both the current and historical frames. Subtracting $f_\phi(0)$ ensures that the conditioning vanishes at the reference interval. The continuous

parameterization allows a single model to represent different physical prediction intervals within a common conditioning space, rather than introducing separate parameters for each timestep.

Anchor-based motion prediction. We express the position prediction as a residual relative to a constant-velocity extrapolation. For consecutive states separated by the same physical interval Δt, the reference configuration is $x_{t+1}^{\text{ext}} = 2x_t - x_{t-1}$. This removes the dominant constant-velocity component from the learned prediction, so that the network only needs to model acceleration-induced deviations due to external forces and floor contact. Gravity is incorporated through the physical conditioning of the model.

We parameterize the residual using K surface anchors selected from the initial point cloud x_0 by farthest-point sampling (FPS) (Qi et al., 2017). Their indices $\{a_k\}_{k=1}^K$ remain fixed throughout the trajectory to preserve temporal correspondence. For each anchor, the position predictor estimates a residual $r_{\theta,k} \in \mathbb{R}^3$, expressed at the reference time scale. The target anchor positions are

$$q_{t+1,k} = x_{t+1,a_k}^{\text{ext}} + \rho^2 r_{\theta,k}, \qquad k = 1, \dots, K. \tag{3}$$

The quadratic scaling reflects the timestep dependence of acceleration-induced displacement.

We recover the next object state by fitting a rigid transformation from the current anchors to their predicted targets. Specifically, the Kabsch algorithm (Kabsch, 1976) estimates a rotation $R_t \in \mathrm{SO}(3)$ and a translation $t_t \in \mathbb{R}^3$ that minimize the sum of squared anchor alignment errors:

$$\begin{aligned} (R_t, t_t) &= \mathrm{Kabsch}\left(\{x_{t,a_k}\}_{k=1}^K, \{q_{t+1,k}\}_{k=1}^K\right), \\ \hat{x}_{t+1,p} &= R_t x_{t,p} + t_t, \qquad p = 1, \dots, P. \end{aligned} \tag{4}$$

Applying the same transformation to all surface points preserves the rigidity of the predicted object.

Angular velocity prediction. An auxiliary predictor estimates an angular velocity residual $r_\theta^\omega \in \mathbb{R}^3$ at the reference time scale:

$$\hat{\omega}_{t+1} = \omega_t + \rho r_\theta^\omega. \tag{5}$$

The linear scaling reflects the integration of angular acceleration. The predicted angular velocity is included in the state history for subsequent autoregressive predictions.

3.3 Pairwise Contact Dynamics

As illustrated in Stage II of Figure 2, the pairwise model predicts residual motion induced by inter-object contact on top of the frozen self-dynamics prediction. A shared contact model updates both objects in each pair, while a contact gate suppresses corrections for noninteracting pairs.

Pairwise representation. We encode each object's point positions, displacements, and angular motion, conditioned on its physical attributes. Spatial attention first captures the geometry of each object, followed by joint attention over both objects and temporal attention across successive states. The point features are then pooled separately for each object to obtain representations conditioned on the pair. The encoders and prediction heads share parameters across both objects and all pairs.

Momentum-constrained anchor corrections. For each object in a pair (i, j), the model predicts displacements at K anchors and an angular velocity correction. Following the self-dynamics parameterization, these are scaled by ρ^2 and ρ, respectively.

Let $d_{i,k}^{ij} \in \mathbb{R}^3$ denote the scaled displacement of anchor k on object i. The mean anchor displacement $\bar{d}_i^{ij} = \frac{1}{K}\sum_{k=1}^K d_{i,k}^{ij}$ defines a translational correction $u_i^{ij} = \bar{d}_i^{ij}/\Delta t$. To enforce zero net pairwise translational impulse in the predicted anchor corrections, we remove their shared mass-weighted velocity component:

$$\begin{aligned} \bar{u}^{ij} &= \frac{m_i u_i^{ij} + m_j u_j^{ij}}{m_i + m_j}, \\ \Delta v_i^{ij} &= u_i^{ij} - \bar{u}^{ij}, \qquad \Delta v_j^{ij} = u_j^{ij} - \bar{u}^{ij}. \end{aligned} \tag{6}$$

The resulting corrections satisfy $m_i \Delta v_i^{ij} + m_j \Delta v_j^{ij} = 0$. This constraint removes pairwise center-of-motion drift while leaving relative contact motion to be predicted by the network. We then reconstruct the anchor displacements as

$$\tilde{d}_{i,k}^{ij} = d_{i,k}^{ij} - \bar{d}_i^{ij} + \Delta t\, \Delta v_i^{ij}, \qquad k = 1, \dots, K, \tag{7}$$

with the same construction applied to object j.

Contact gating and rigid reconstruction. A symmetric contact gate γ_{ij} is predicted from the pairwise representations. We use a soft gate during training and a binary gate at inference. The binary decision combines the predicted contact probability with a scale-relative proximity test, preventing spurious corrections between clearly separated objects.

Let x_i and ω_i denote the object's current state before the pairwise update, and let $a_{i,k}$ index its anchors. The corrected anchor targets are $x_{i,a_{i,k}} + \gamma_{ij}\tilde{d}^{ij}_{i,k}$. We recover the updated point positions x_i^+ using the same Kabsch reconstruction as in Eq. (4), and update angular velocity as

$$\omega_i^+ = \omega_i + \gamma_{ij}\Delta\omega_i^{ij}. \tag{8}$$

Both objects are updated in the same manner, with their current states preserved when $\gamma_{ij} = 0$.

3.4 Training and Multi-Object Inference

Two-stage training. We first train the self-dynamics model on single-object trajectories with varying physical timesteps. For flow time $\tau \in (0, 1)$, we define $x_\tau = (1-\tau)x_{t+1} + \tau\epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$, and minimize

$$\mathcal{L}_{\text{self}} = \underbrace{\mathbb{E}\left[\left\| \frac{x_\tau - \hat{x}^{\text{self}}_{t+1}}{\tau} - (\epsilon - x_{t+1}) \right\|_2^2\right]}_{\mathcal{L}_{\text{FM}}} + \lambda^s_R\mathcal{L}^s_R + \lambda^s_\omega\mathcal{L}^s_\omega + \lambda^s_{\text{cons}}\mathcal{L}^s_{\text{cons}}. \tag{9}$$

Alongside flow matching, $\mathcal{L}^s_R$ and $\mathcal{L}^s_\omega$ supervise rotation and angular velocity, while $\mathcal{L}^s_{\text{cons}}$ enforces consistency between the rotations implied by the predicted point motion and angular velocity.

We then freeze the self-dynamics model and train the pairwise contact model on two-object contacts:

$$\mathcal{L}_{\text{pair}} = \lambda_x\mathcal{L}_x + \lambda_g\mathcal{L}_{\text{gate}} + \underbrace{\lambda_v\mathcal{L}_{\Delta v} + \lambda_{\Delta\omega}\mathcal{L}_{\Delta\omega}}_{\mathcal{L}_{\text{contact}}}. \tag{10}$$

Here, $\mathcal{L}_x$ supervises corrected point positions, $\mathcal{L}_{\text{gate}}$ supervises contact prediction, and $\mathcal{L}_{\text{contact}}$ supervises contact-induced changes in linear and angular velocity. Training emphasizes contact frames and includes short model-generated rollout histories. Loss definitions and training details are provided in Appendix A.2.

Autoregressive rollout. We generate trajectories by repeatedly predicting self dynamics and applying pairwise contact corrections over each physical interval Δt. The corrected state of each object is appended to its history, retaining at most H recent states to condition the next prediction.

Multi-object inference. At each simulation step, the self-dynamics model predicts trajectories for all N objects over Δt. For each candidate pair, the swept trajectories define a time of impact (TOI) in $[0, 1]$, the predicted first-contact time within the interval. As illustrated in Figure 3, we repeatedly select the smallest TOI and apply the shared contact model using the latest point positions and angular velocities. Corrections are committed immediately, after which TOIs for pairs incident to either updated object are recomputed before the next selection. This lets later contacts incorporate earlier responses and composes the two-object model over varying object counts.

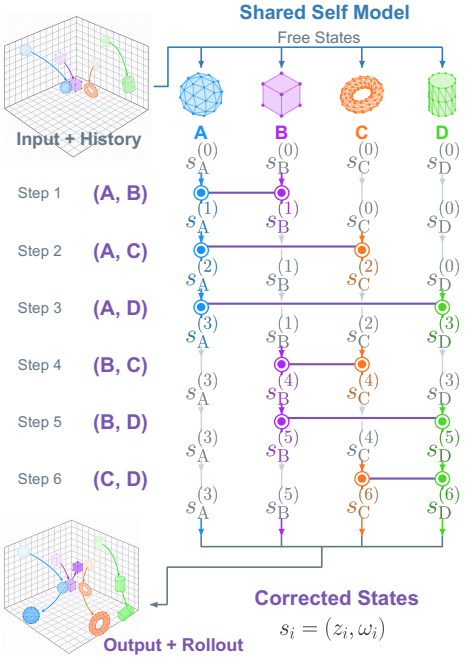

Figure 3: **Sequential contact composition.** Pairwise corrections are committed immediately, so each subsequent interaction is conditioned on the latest states produced by earlier contacts.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We evaluate on MOVi-A, MOVi-B, and MOVi-S using the splits established by HOPNet (Wei & Fink, 2025). For training, we generate two Kubric/PyBullet datasets from MOVi-B assets: *Single* contains one object and isolates self dynamics, whereas *Pair* contains two-object interactions. The self-dynamics model is trained on *Single*, while the interaction model is trained on either *Pair* or the target MOVi dataset. Our transfer setting trains the interaction model only on *Pair* and evaluates it directly on multi-object MOVi scenes, without target-dataset training. We compare against HOPNet (Wei & Fink, 2025), RigidFormer (Dou et al., 2026), and PhysiFormer (Chen et al., 2026).

We report position RMSE of object centers, geodesic orientation RMSE, and volumetric IoU (vIoU) (Wang et al., 2025) over autoregressive rollouts at matched physical horizons; lower RMSE and higher vIoU are better. Exact timestep and horizon conventions and metric implementations are provided in Appendix A.6. Further dataset and stress-test details are in Appendix A.7.

4.2 COMPARISONS WITH BASELINES

Cross-dataset transfer. Table 1 demonstrates the transfer advantage of our factorization. Trained only on *Pair*, PhysDuo consistently improves position accuracy and vIoU over reimplemented baselines across MOVi-A/B/S, including beyond early pre-contact horizons. With in-domain training, the gap narrows, though PhysDuo remains competitive in geometric overlap at early and intermediate horizons. These results suggest that pairwise corrections transfer across object-shape distributions while preserving complete object geometry.

Temporal resolution and rollout stability. Table 2 evaluates longer timesteps on MOVi-B. The Pair-trained model consistently outperforms the reimplemented RigidFormer at timesteps 5 and 10 across early and intermediate horizons, showing robust operation across simulation step sizes. The largest gains are in geometric overlap, while RMSE differences narrow at the longest horizons. This suggests that the proposed decomposition mitigates autoregressive geometric drift despite accumulated contact-timing errors. Corresponding in-domain MOVi-B results and longer-timestep results on MOVi-A and MOVi-S are reported in Appendix A.5.

Qualitative and out-of-distribution results. Figure 5 shows that our Pair-trained model better preserves contact response and coherent motion across MOVi-A, MOVi-B, and MOVi-S. Additional in-domain, *Single*, and *Pair* results are provided in Appendix A.7. Figure 4 shows, without fine-tuning, a Kubric calf asset named *Spot* striking cube pyramids with unseen scales, masses, and object counts, probing composition under dense cascading contacts.

4.3 ABLATIONS

We evaluate controlled ablations at a common timestep on *Single* and *Pair*, varying one component per row to allow the self-dynamics and contact mechanisms to be interpreted separately.

Self dynamics. Table 3 (a) isolates rigid reconstruction, angular-velocity prediction, and auxiliary geometric losses. Individual ablations occasionally improve short-horizon metrics, but the full model achieves better long-horizon geometric and rotational consistency. Removing anchor-based rigid reconstruction causes the clearest degradation, confirming that predicting a small set of anchors and fitting a rigid transform is more stable than unconstrained point motion. Angular-velocity prediction and the three geometric losses are complementary, aligning point trajectories with object-level rotation. Additional anchor-count and history-length variants are reported in Appendix A.3.

Pairwise contact. Table 3 (b) shows that the pairwise interaction module is essential after contact. Suppressing the pairwise solver or contact gate substantially reduces geometric overlap, demonstrating the importance of selective contact corrections even when the underlying single-object predictor is strong. Removing angular correction or contact-specific supervision also reduces post-contact consistency. Overall, the full model achieves the strongest vIoU while maintaining comparable or

Table 1: **Baseline comparison on MOVi-A, MOVi-B, and MOVi-S at timestep 1.** Each cell reports position RMSE (m) ↓ / orientation RMSE (deg) ↓ / vIoU (%) ↑, in that order at the indicated raw-frame horizon. NA denotes an unreported value. For baseline methods, † denotes results from our reimplementations; unmarked results are reported by the original baselines. Ranking is per metric component within each training block: 1st, 2nd.

(a) MOVi-A	50	100	200	300	400
Train on Pair					
HOPNet†	0.120 / 5.68 / 77.7	0.566 / 19.09 / 28.1	1.787 / 65.55 / 4.4	2.624 / 99.56 / 2.3	3.489 / 112.92 / 1.4
RigidFormer†	0.066 / 3.34 / 85.1	0.274 / 13.73 / 59.0	0.990 / 57.76 / 17.9	1.987 / 99.75 / 3.9	2.885 / 115.00 / 2.1
PhysiFormer†	0.901 / 5.83 / 14.4	1.566 / 20.22 / 3.9	2.372 / 68.21 / 2.3	3.251 / 100.64 / 1.3	3.963 / 113.48 / 0.9
Ours	0.046 / 2.59 / 92.8	0.202 / 11.08 / 77.9	0.860 / 54.69 / 30.5	1.653 / 103.96 / 10.2	2.397 / 115.51 / 5.2
HOPNet	0.054 / 5.64 / NA	0.196 / 18.83 / NA	NA	NA	NA
HOPNet†	0.043 / 3.44 / 71.9	0.164 / 13.62 / 51.2	0.574 / 51.46 / 28.2	1.177 / 89.27 / 13.3	1.808 / 103.74 / 7.4
RigidFormer	0.049 / 5.06 / NA	0.177 / 18.32 / NA	NA	NA	NA
RigidFormer†	0.050 / 2.95 / 88.3	0.213 / 12.26 / 64.6	0.932 / 58.11 / 19.9	2.671 / 102.82 / 2.9	5.906 / 116.69 / 0.6
PhysiFormer†	0.712 / 4.37 / 2.17	1.376 / 13.56 / 8.1	2.428 / 56.70 / 2.4	3.266 / 93.36 / 1.3	4.000 / 105.90 / 1.1
Ours	0.039 / 2.67 / 92.7	0.155 / 11.71 / 78.7	0.734 / 56.85 / 31.4	1.616 / 104.78 / 10.0	2.562 / 116.36 / 4.6

(b) MOVi-B	50	100	200	300	400
Train on Pair					
HOPNet†	0.104 / 4.97 / 73.2	0.514 / 17.99 / 24.6	1.697 / 63.42 / 3.3	2.456 / 107.35 / 1.7	3.280 / 115.81 / 1.1
RigidFormer†	0.054 / 3.04 / 84.8	0.219 / 11.81 / 58.0	0.878 / 54.85 / 15.8	1.819 / 101.55 / 2.6	2.537 / 113.33 / 2.2
PhysiFormer†	0.895 / 6.13 / 10.8	1.540 / 21.45 / 2.8	2.368 / 68.77 / 1.1	3.294 / 107.15 / 0.6	3.985 / 112.90 / 0.4
Ours	0.043 / 2.65 / 91.8	0.169 / 10.51 / 77.4	0.777 / 52.46 / 26.5	1.519 / 100.56 / 7.9	2.223 / 116.04 / 4.1
HOPNet	0.047 / 4.91 / NA	0.176 / 17.91 / NA	NA	NA	NA
HOPNet†	0.037 / 3.49 / 71.7	0.144 / 13.47 / 48.3	0.546 / 52.40 / 23.2	1.136 / 94.75 / 9.2	1.747 / 107.89 / 4.7
RigidFormer	0.050 / 3.97 / NA	0.161 / 15.33 / NA	NA	NA	NA
RigidFormer†	0.054 / 3.04 / 85.4	0.196 / 11.37 / 60.2	0.800 / 52.40 / 18.7	1.738 / 98.27 / 4.6	2.920 / 112.37 / 1.5
PhysiFormer†	0.723 / 4.74 / 15.8	1.387 / 13.87 / 5.4	2.404 / 58.15 / 1.5	3.282 / 104.75 / 0.7	3.997 / 111.44 / 0.6
Ours	0.041 / 2.67 / 91.7	0.146 / 10.63 / 77.5	0.679 / 53.06 / 26.9	1.393 / 102.87 / 8.2	2.148 / 116.33 / 4.2

(c) MOVi-S	50	100	200	300	400
Train on Pair					
HOPNet†	0.116 / 4.41 / 77.1	0.547 / 14.68 / 26.3	1.795 / 59.47 / 3.7	2.585 / 105.25 / 1.7	3.478 / 115.81 / 0.9
RigidFormer†	0.062 / 2.66 / 86.1	0.260 / 10.15 / 60.3	0.943 / 51.31 / 19.5	1.927 / 103.33 / 3.1	2.704 / 114.10 / 2.5
PhysiFormer†	0.928 / 5.26 / 10.6	1.593 / 17.29 / 2.3	2.392 / 63.26 / 1.4	3.312 / 106.24 / 0.7	4.065 / 112.27 / 0.5
Ours	0.045 / 2.02 / 92.6	0.194 / 7.59 / 78.6	0.806 / 49.46 / 33.8	1.580 / 103.48 / 10.4	2.340 / 116.80 / 4.7
HOPNet	0.034 / 4.05 / NA	0.124 / 13.68 / NA	NA	NA	NA
HOPNet†	0.030 / 2.85 / 75.2	0.115 / 10.37 / 53.9	0.411 / 44.75 / 32.0	0.889 / 91.23 / 15.4	1.429 / 105.93 / 7.8
RigidFormer	0.026 / 3.00 / NA	0.099 / 11.19 / NA	NA	NA	NA
RigidFormer†	0.039 / 2.22 / 91.2	0.158 / 8.60 / 72.0	0.755 / 48.53 / 24.6	2.229 / 101.13 / 3.1	5.147 / 118.47 / 0.5
PhysiFormer†	0.683 / 3.68 / 18.9	1.324 / 9.18 / 5.4	2.373 / 48.55 / 1.4	3.226 / 97.90 / 0.8	4.010 / 108.06 / 0.5
Ours	0.047 / 2.08 / 92.2	0.191 / 7.89 / 78.4	0.803 / 50.36 / 33.6	1.592 / 104.26 / 10.2	2.389 / 118.99 / 4.6

Table 2: **Pair-trained comparison on MOVi-B at longer timesteps.** All methods are trained on *Pair*. Each cell reports position RMSE (m) ↓ / orientation RMSE (deg) ↓ / vIoU (%) ↑.

Timestep	Method	50	100	200	300	400
5	RigidFormer†	0.045 / 2.52 / 86.0	0.213 / 11.19 / 58.0	0.807 / 53.29 / 18.7	1.475 / 95.99 / 7.3	2.099 / **107.59** / 4.3
	Ours	**0.032 / 2.10 / 92.2**	**0.141 / 9.28 / 76.6**	**0.644 / 49.35 / 27.4**	**1.349 / 94.77 / 8.9**	**2.000** / 109.09 / **4.9**
10	RigidFormer†	0.037 / 2.00 / 87.6	0.207 / 10.19 / 58.3	0.821 / 51.40 / 17.9	1.529 / **95.07** / 6.1	2.190 / 108.46 / 3.6
	Ours	**0.026 / 1.61 / 93.2**	**0.138 / 8.35 / 76.7**	**0.656 / 49.22 / 27.2**	**1.378** / 95.36 / **7.7**	**2.045 / 107.95 / 3.9**

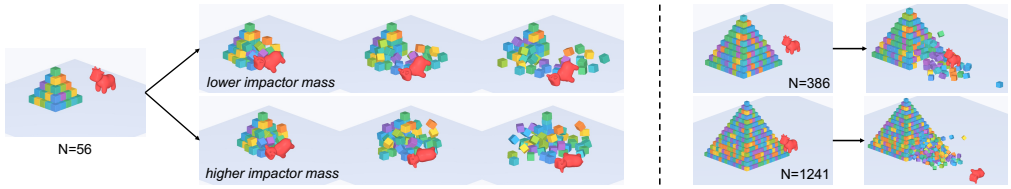

Figure 4: **Extreme out-of-distribution generalization.** The Kubric calf asset named *Spot* strikes cube pyramids with unseen scales, masses, object counts, and impact regimes, without fine-tuning.

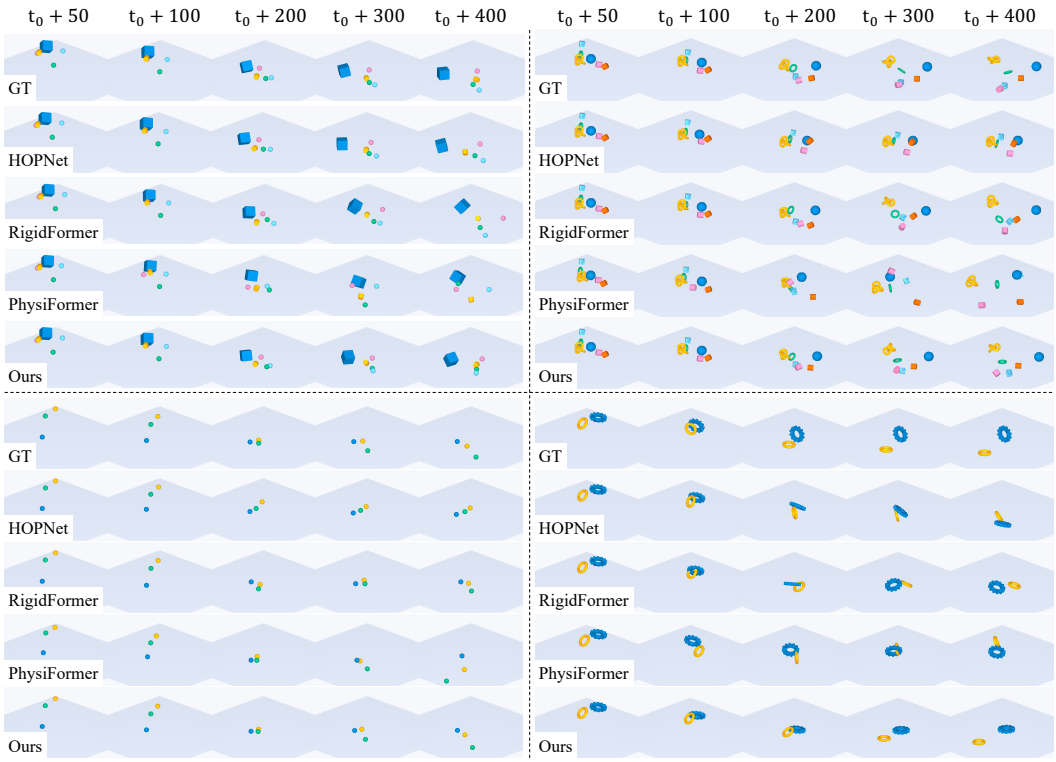

Figure 5: **Comparison on MOVi-A, MOVi-B, MOVi-S, and Pair.** All methods are trained on *Pair*.

Table 3: **Ablations of self and pairwise dynamics at timestep 10.** Each cell reports position RMSE (m) ↓ / orientation RMSE (deg) ↓ / vIoU (%) ↑ at raw-frame horizons 100 and 200. Ranking is per metric component: 1st, 2nd. $\mathcal{S}_\phi$ denotes the pairwise correction solver and γ_{ij} the contact gate.

(a) *Self*	100	200
w/o Anchor–Kabsch	0.003 / 0.19 / 97.0	0.078 / 13.65 / 54.4
w/o ω	0.003 / 0.11 / 98.1	0.065 / 13.30 / 57.4
w/o $\mathcal{L}^s_{\text{cons}}$	0.002 / 0.10 / 98.6	0.143 / 19.93 / 43.1
w/o $\mathcal{L}^s_R$	0.001 / 0.09 / 99.5	0.120 / 17.70 / 47.7
w/o $\mathcal{L}^s_\omega$	0.002 / 0.09 / 98.5	0.088 / 16.97 / 53.2
Ours	0.002 / 0.07 / 98.6	0.064 / 12.64 / 58.4

(b) *Pair*	100	200
w/o $\mathcal{S}_\phi$	0.497 / 25.53 / 38.9	2.208 / 79.75 / 9.9
w/o γ_{ij}	0.356 / 25.98 / 35.0	0.813 / 80.95 / 10.6
w/o $\Delta\omega$	0.170 / 25.53 / 39.2	0.643 / 79.75 / 11.0
w/o $\mathcal{L}_{\text{contact}}$	0.178 / 25.73 / 38.3	0.656 / 81.81 / 10.4
Ours	0.167 / 25.49 / 54.9	0.648 / 79.59 / 16.5

improved translational and rotational error across post-contact horizons. Comparisons with reimplemented baselines across timesteps are provided in Appendix A.4.

5 Conclusion

We presented a compositional framework for rigid-body simulation that learns self motion and pairwise contact dynamics from simple single- and two-object trajectories, then reuses these learned primitives in multi-object scenes. Combining timestep-conditioned self motion with a shared residual contact model allows the same learned dynamics to be reused across scene configurations and simulation step sizes. Our evaluation over extended rollouts highlights the importance of assessing both collision response and subsequent translational and rotational drift.

Limitations and future work. The current formulation is limited to rigid bodies and remains susceptible to rotational drift and interaction-order effects under simultaneous contacts. Future work will improve coupled-contact modeling and extend the framework to non-rigid dynamics through deformable representations and material-dependent motion models.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish writing and for research ideation or execution. We have not used generative AI tools for retrieval and discovery, to draft sections of the paper, or to generate synthetic datasets. Proving mathematical claims are not applicable to this work. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Kelsey R. Allen, Tatiana Lopez-Guevara, Yulia Rubanova, Kim Stachenfeld, Alvaro Sanchez-Gonzalez, Peter Battaglia, and Tobias Pfaff. Graph network simulators can learn discontinuous, rigid contact dynamics. In *Conference on Robot Learning (CoRL)*, volume 205, pp. 1157–1167, 2023a.

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. In *International Conference on Learning Representations (ICLR)*, 2023b.

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261, 2018.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Michael B. Chang, Tomer Ullman, Antonio Torralba, and Joshua B. Tenenbaum. A compositional object-based approach to learning physical dynamics. In *International Conference on Learning Representations (ICLR)*, 2017.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *Advances in Neural Information Processing Systems*, volume 37, pp. 24081–24125, 2024. doi: 10.52202/079017-0759.

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

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.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J. Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: A scalable dataset generator. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3749–3761, 2022.

Jiaqi Han, Wenbing Huang, Hengbo Ma, Jiachen Li, Josh Tenenbaum, and Chuang Gan. Learning physical dynamics with subequivariant graph neural networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 26256–26268, 2022a. doi: 10.52202/068431-1904.

Zhichao Han, David S. Kammer, and Olga Fink. Learning physics-consistent particle interactions. *PNAS Nexus*, 1(5):pgac264, 2022b. doi: 10.1093/pnasnexus/pgac264.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems*, volume 38, pp. 167283–167308, 2025. doi: 10.52202/085713-5576.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning (ICML)*, volume 162, pp. 9902–9915, 2022.

Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A*, 32(5):922–923, 1976. doi: 10.1107/S0567739476001873.

Chanho Kim and Li Fuxin. Object dynamics modeling with hierarchical point cloud-based representations. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20977–20986, 2024.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning (ICML)*, volume 80, pp. 2688–2697, 2018.

Alex Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B. Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. In *International Conference on Learning Representations (ICLR)*, 2019.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Phillip Lippe, Bas Veeling, Paris Perdikaris, Richard Turner, and Johannes Brandstetter. PDE-Refiner: Achieving accurate long rollouts with neural PDE solvers. In *Advances in Neural Information Processing Systems*, volume 36, pp. 67398–67433, 2023. doi: 10.52202/075280-2946.

Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li Fei-Fei, Joshua B. Tenenbaum, and Daniel L. K. Yamins. Flexible neural representation for physics prediction. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *International Conference on Computer Vision (ICCV)*, pp. 4195–4205, 2023.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations (ICLR)*, 2021.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Yulia Rubanova, Alvaro Sanchez-Gonzalez, Tobias Pfaff, and Peter Battaglia. Constraint-based graph network simulator. In *International Conference on Machine Learning (ICML)*, volume 162, pp. 18844–18870, 2022.

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 *Advances in Neural Information Processing Systems*, volume 37, pp. 125809–125838, 2024. doi: 10.52202/079017-3997.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning (ICML)*, volume 80, pp. 4470–4479, 2018.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8459–8468. PMLR, 2020.

Yidi Shao, Chen Change Loy, and Bo Dai. Transformer with implicit edges for particle-based physics simulation. In *European Conference on Computer Vision (ECCV)*, 2022.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

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. In *Advances in Neural Information Processing Systems*, volume 38, pp. 167907–167932, 2025. doi: 10.52202/085713-5598.

Amaury Wei and Olga Fink. Integrating physics and topology in neural networks for learning rigid body dynamics. *Nature Communications*, 16(1):6867, 2025. doi: 10.1038/s41467-025-62250-7.

Youn-Yeol Yu, Jeongwhan Choi, Woojin Cho, Kookjin Lee, Nayong Kim, Kiseok Chang, ChangSeung Woo, Ilho Kim, SeokWoo Lee, Joon Young Yang, Sooyoung Yoon, and Noseong Park. Learning flexible body collision dynamics with hierarchical contact mesh transformer. In *International Conference on Learning Representations (ICLR)*, pp. 23367–23394, 2024.

Jingyang Yuan, Gongbo Sun, Zhiping Xiao, Hang Zhou, Xiao Luo, Junyu Luo, Yusheng Zhao, Wei Ju, and Ming Zhang. EGODE: An event-attended graph ODE framework for modeling rigid dynamics. In *Advances in Neural Information Processing Systems*, volume 37, pp. 59093–59118, 2024. doi: 10.52202/079017-1885.

Yaofeng Desmond Zhong, Biswadip Dey, and Amit Chakraborty. Extending Lagrangian and Hamiltonian neural networks with differentiable contact models. In *Advances in Neural Information Processing Systems*, volume 34, pp. 21910–21922, 2021.

Supplementary Material for "PhysDuo: Learning Pairwise Physics for Compositional Multi-Object Simulation"

A Appendix

A.1 Implementation Details

Self-dynamics network. Each object is represented by $P = 128$ surface points, and the self-dynamics model uses at most $H = 8$ historical states. The backbone contains 12 causal PointST-DiT blocks with a feature dimension of 512 and 8 attention heads. Temporal positions are represented using rotary positional embeddings. The physical timestep is encoded relative to $\Delta t_{\text{ref}} = 0.0417$ seconds by the continuous timestep embedding in Eq. (2).

The position head predicts residual motion at $K = 4$ anchors selected by farthest-point sampling. A Kabsch fit between the current and predicted anchors produces the rigid transformation applied to the complete point cloud. A separate head predicts the object-level angular velocity. Both position and angular predictions are scaled according to the physical timestep, as described in Sec. 3.2.

Pairwise interaction network. The pairwise model has a feature dimension of 256 and uses 8 attention heads. It contains three attention stages that operate over points within each object, points across the two objects, and the most recent two states, respectively. The resulting point features are pooled per object and decoded into $K = 4$ anchor corrections, an angular-velocity correction, and a symmetric contact gate. All interaction parameters are shared across objects and object pairs.

We train on physical timesteps $\{1, 5, 10\}$ and use the same continuous timestep conditioning in both models. During multi-object inference, the self-dynamics predictions are computed in parallel, after which the pairwise model is applied once to every unordered object pair in a time-of-impact-driven dynamic ordering. Each correction is written immediately and is therefore available to subsequent pairwise updates.

A.2 Training Objectives and Optimization

We detail the loss terms introduced in Sec. 3.4. For a point cloud x, let $c(x)$ denote its centroid, $\tilde{x} = x - c(x)$ its centered coordinates, and $s(x)$ the diagonal length of its axis-aligned bounding box. We write $\boldsymbol{\phi}(A, B) = \mathrm{Log}(R(A, B))^{\vee}$ for the rotation vector obtained by Kabsch alignment between the centered point clouds A and B.

Self-dynamics objective. The flow-matching loss is supplemented by rotation, angular velocity, and consistency losses. Let $\phi_t = \boldsymbol{\phi}(x_t, x_{t+1})$ and $\hat{\phi}_t = \boldsymbol{\phi}(x_t, \hat{x}^{\text{self}}_{t+1})$. The rotation and angular velocity losses are

$$\begin{aligned} \mathcal{L}^s_R &= \mathbb{E}\left[\|\hat{\phi}_t - \phi_t\|_2^2\right], \\ \mathcal{L}^s_\omega &= \mathbb{E}\left[\left\|\hat{\omega}_{t+1} - \frac{\phi_t}{\Delta t}\right\|_2^2\right]. \end{aligned} \tag{11}$$

The consistency loss penalizes discrepancies between the rotations encoded by the two predictions:

$$\mathcal{L}^s_{\text{cons}} = \mathbb{E}\left[\frac{1}{P}\sum_{p=1}^{P}\left\|\frac{\tilde{\hat{x}}^{\text{self}}_{t+1,p} - \mathrm{Exp}(\mathrm{sg}(\hat{\omega}_{t+1})\Delta t)\,\tilde{x}_{t,p}}{r_t \Delta t}\right\|_2^2\right], \tag{12}$$

where $r_t^2 = P^{-1}\sum_p \|\tilde{x}_{t,p}\|_2^2$, Exp maps a rotation vector to a rotation matrix, and sg treats its argument as constant during differentiation. Thus, consistency supervision acts on the point prediction using the angular velocity estimate as its reference.

Pairwise contact objective. For each pair (i, j), the corrected point positions are supervised by

$$\mathcal{L}_x = \frac{1}{2} \sum_{o \in \{i,j\}} \mathbb{E}\left[\frac{\|\hat{x}_{o,t+1} - x_{o,t+1}\|_F^2}{P\, s(x_{o,t+1})^2} \right]. \tag{13}$$

The soft contact gate γ_{ij} is trained with weighted binary cross-entropy:

$$\mathcal{L}_{\text{gate}} = -\mathbb{E}[w_+ y_{ij} \log \gamma_{ij} + w_-(1 - y_{ij}) \log(1 - \gamma_{ij})], \tag{14}$$

where $y_{ij} \in [0, 1]$ is the contact target and w_+, w_- balance positive and negative supervision.

The contact target combines changes in relative motion with spatial proximity:

$$\begin{aligned} q_t &= \frac{\|\Delta u_t^{\text{rel}}\|_2}{s_{ij,t}} \exp\left(-\frac{d_{ij,t}^{\text{box}}}{\kappa s_{ij,t}} \right), \\ y_{ij,t} &= \max\left(\bar{q}_t^{\alpha},\, \eta \bar{q}_{t-1}^{\alpha},\, \eta \bar{q}_{t+1}^{\alpha} \right), \qquad \bar{q}_t = \frac{q_t}{\max_{t'} q_{t'}}. \end{aligned} \tag{15}$$

Here, Δu_t^{rel} is the temporal change in relative centroid displacement, $d_{ij,t}^{\text{box}}$ is the distance between the object bounding boxes, and $s_{ij,t}$ is their mean scale. The neighboring terms extend supervision to frames adjacent to contact.

Contact-induced velocity supervision. Velocity correction targets are defined relative to the frozen self-dynamics predictions. For $o \in \{i, j\}$, let

$$\begin{aligned} u_o &= \frac{c(x_{o,t+1}) - c(\hat{x}_{o,t+1}^{\text{self}})}{\Delta t}, \\ \Delta v_o^* &= u_o - \frac{m_i u_i + m_j u_j}{m_i + m_j}, \\ \Delta \omega_o^* &= \frac{\boldsymbol{\phi}(x_{o,t}, x_{o,t+1}) - \boldsymbol{\phi}(x_{o,t}, \hat{x}_{o,t+1}^{\text{self}})}{\Delta t}. \end{aligned} \tag{16}$$

The linear targets satisfy the same mass-weighted balance constraint as the predicted translational corrections. Using the smooth-L_1 penalty ℓ_β, we define

$$\begin{aligned} \mathcal{L}_{\Delta v} &= \frac{\mathbb{E}\left[\frac{y_{ij}}{2} \sum_{o \in \{i,j\}} \ell_\beta\left(\frac{(\widehat{\Delta v}_o - \Delta v_o^*)\Delta t}{s_{ij}} \right) \right]}{\mathbb{E}[y_{ij}] + \varepsilon}, \\ \mathcal{L}_{\Delta \omega} &= \frac{\mathbb{E}\left[\frac{y_{ij}}{2} \sum_{o \in \{i,j\}} \ell_\beta\left((\widehat{\Delta \omega}_o - \Delta \omega_o^*)\Delta t \right) \right]}{\mathbb{E}[y_{ij}] + \varepsilon}. \end{aligned} \tag{17}$$

Here, $\widehat{\Delta v}_o$ and $\widehat{\Delta \omega}_o$ denote corrections before contact gating, and $\varepsilon > 0$ stabilizes normalization. Weighting by y_{ij} emphasizes contact events without modulating this supervision by the predicted gate.

Optimization. We train PhysDuo on NVIDIA A100 GPUs in two stages. First, the self-dynamics model is optimized on *Single* for 40,000 steps. Its checkpoint then initializes the self-dynamics branch of the pairwise model, which remains frozen while the interaction module is trained on *Pair* for 15,000 steps. Both stages use AdamW with an initial learning rate of 10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.999)$, and weight decay 10^{-2}. We apply cosine learning-rate decay after 500 warm-up steps and use a per-GPU batch size of 32. Training examples cycle through physical timesteps $\{1, 5, 10\}$ in round-robin order. During pairwise training, collision-aware temporal sampling allocates 25%, 50%, and 25% of examples to pre-impact, impact, and post-impact windows, respectively. Loss weights are

$$(\lambda_R^s, \lambda_\omega^s, \lambda_{\text{cons}}^s) = (0.035, 0.04, 0.002), \qquad (\lambda_x, \lambda_g, \lambda_v, \lambda_{\Delta\omega}) = (1, 0.2, 2, 1).$$

We use $\beta = 0.02$ for the smooth-L_1 penalty and $(\kappa, \alpha, \eta) = (0.03, 1.5, 0.25)$ for contact targets.

Pairwise training emphasizes contact events and gradually transitions from ground-truth to predicted gates. Short histories generated by the frozen self-dynamics model are progressively introduced without backpropagation through the history generation.

Table A: **Complete self-dynamics ablation on *Single* at timestep 10.** Each cell reports position RMSE (m) ↓ / orientation RMSE (deg) ↓ / vIoU (%) ↑ at the indicated raw-frame horizon. Unless stated otherwise, variants use four anchors, history length $H = 8$, Kabsch reconstruction, angular-velocity prediction, and all self-dynamics losses. Best values for each metric component are bolded.

Variant	100	200	300	400
w/o Anchors & Kabsch	0.003 / 0.19 / 97.0	0.078 / 13.65 / 54.4	0.260 / 46.87 / 24.3	0.393 / 68.97 / 16.7
w/o Anchors	0.002 / **0.07** / 98.2	0.202 / 20.40 / 34.4	0.410 / 72.20 / 15.8	0.599 / 101.11 / 10.9
$K = 8$	0.002 / 0.10 / 97.9	0.141 / 16.57 / 43.8	0.420 / 61.14 / 15.7	0.666 / 85.92 / 9.6
$H = 2$	0.002 / 0.11 / 98.5	0.150 / 18.83 / 41.5	0.395 / 68.32 / 15.4	0.603 / 94.20 / 9.9
$H = 4$	0.003 / 0.14 / 98.2	0.143 / 18.51 / 42.8	0.401 / 63.74 / 15.3	0.604 / 86.21 / 9.5
$H = 16$	0.002 / 0.10 / 99.0	0.172 / 21.34 / 39.1	0.578 / 71.71 / 9.6	0.944 / 93.34 / 5.4
w/o ω Prediction	0.003 / 0.11 / 98.1	0.065 / 13.30 / 57.4	**0.225** / 45.30 / 26.4	0.343 / 72.68 / 19.0
w/o $\mathcal{L}^s_{\text{cons}}$	0.002 / 0.10 / 98.6	0.143 / 19.93 / 43.1	0.389 / 70.26 / 16.3	0.615 / 94.19 / 10.5
w/o $\mathcal{L}^s_R$	**0.001** / 0.09 / **99.5**	0.120 / 17.70 / 47.7	0.365 / 62.13 / 16.4	0.569 / 89.52 / 8.9
w/o $\mathcal{L}^s_\omega$	0.002 / 0.09 / 98.5	0.088 / 16.97 / 53.2	0.302 / 66.87 / 20.8	0.436 / 92.90 / 14.3
Ours	0.002 / **0.07** / 98.6	**0.064 / 12.64 / 58.4**	0.232 / **42.47 / 26.6**	**0.337 / 59.70 / 19.6**

Table B: **Reimplemented-baseline comparison on *Pair*.** Each cell reports position RMSE (m) ↓ / orientation RMSE (deg) ↓ / vIoU (%) ↑ at the indicated raw-frame horizon. † marks our reimplementation. Best values for each metric component are bolded.

Timestep	Method	50	100	200	300	400
1	HOPNet†	0.025 / 1.65 / 90.0	0.227 / 28.00 / 47.6	0.802 / 81.57 / 14.1	1.401 / 105.38 / **6.1**	1.958 / 114.84 / **3.5**
	RigidFormer†	0.014 / 0.59 / 95.1	**0.169 / 23.26** / 53.2	0.702 / **77.13** / 13.7	1.715 / **100.78** / 2.2	2.289 / **110.59** / 1.8
	PhysiFormer†	0.252 / 2.54 / 44.2	0.449 / 28.23 / 24.4	0.853 / 89.03 / 10.6	1.310 / 109.69 / 4.8	**1.649** / 115.51 / **3.5**
	Ours	**0.010 / 0.51 / 97.1**	0.174 / 25.03 / **53.4**	**0.657** / 82.03 / **15.5**	**1.139** / 113.58 / **6.1**	1.728 / 125.37 / 2.8
5	RigidFormer†	0.013 / 0.61 / 94.6	**0.119 / 22.96 / 57.5**	**0.456 / 75.03 / 21.2**	**0.870 / 92.19 / 10.7**	**1.263 / 103.82 / 6.8**
	Ours	**0.008 / 0.46 / 97.5**	0.163 / 24.79 / 54.8	0.611 / 79.97 / 16.7	0.993 / 104.43 / 8.8	1.399 / 111.76 / 5.1
10	RigidFormer†	0.014 / 0.58 / 94.0	**0.117 / 22.13 / 57.5**	**0.468 / 74.16 / 21.6**	**0.882 / 95.62 / 11.3**	**1.251 / 103.49 / 7.7**
	Ours	**0.008 / 0.45 / 97.9**	0.159 / 24.80 / 54.9	0.612 / 79.03 / 16.5	0.991 / 98.32 / 8.5	1.336 / 107.65 / 5.6

A.3 Ablation tables

Table A complements the main ablation with all controlled anchor-count and history-length variants. As in Table 3, all ablations use timestep $s = 10$ and raw-frame horizons 100, 200, 300, and 400.

A.4 Pairwise baseline reimplementations

Table B evaluates our pairwise model over timesteps $s \in \{1, 5, 10\}$. At the 50-frame horizon, our model ranks first in position error, orientation error, and vIoU at every timestep. Its short-horizon accuracy is also highly consistent as the timestep increases. This stability supports the effectiveness of continuous timestep conditioning and shows that the same factorized self-and-contact model operates reliably across temporal resolutions. At $s = 1$, our model further achieves the best position accuracy at frames 200 and 300 and the best geometric overlap through frame 300. Although all methods accumulate error over extended autoregressive rollouts, the repeated early-horizon advantage demonstrates robust local dynamics and rigid-geometry preservation.

A.5 Longer-timestep baseline results

Table C reports in-domain MOVi-B results at timesteps 5 and 10, while Tables D and E evaluate longer timesteps on MOVi-A and MOVi-S over frames 50, 100, 200, 300, and 400.

Table C: **In-domain longer-timestep comparison on MOVi-B.** All methods are trained on MOVi-B. Each cell reports position RMSE (m) ↓ / orientation RMSE (deg) ↓ / vIoU (%) ↑ at the indicated raw-frame horizon. NA denotes an unreported value; † marks our reimplementation. Best metric values are bolded.

Timestep	Method	50	100	200	300	400
5	RigidFormer	0.040 / 3.06 / NA	0.136 / 13.55 / NA	NA / NA / NA	NA / NA / NA	NA / NA / NA
	RigidFormer†	0.040 / 2.48 / 87.4	0.159 / 10.22 / 63.9	**0.590** / 50.61 / 24.8	**1.141 / 94.11 / 10.3**	**1.656 / 106.02 / 6.4**
	Ours	**0.031 / 2.12 / 92.9**	**0.121 / 9.42 / 80.0**	0.611 / **49.55 / 29.6**	1.323 / 95.51 / 9.3	1.970 / 108.30 / 5.2
10	RigidFormer	0.029 / **1.51** / NA	0.115 / 10.85 / NA	NA / NA / NA	NA / NA / NA	NA / NA / NA
	RigidFormer†	0.029 / 1.91 / 89.9	0.140 / 9.44 / 66.2	**0.579** / 49.48 / 24.4	**1.144 / 90.97 / 9.2**	**1.685 / 104.26 / 5.0**
	Ours	**0.023** / 1.62 / **94.2**	**0.112 / 8.45 / 80.8**	0.618 / **49.17 / 28.6**	1.347 / 95.74 / 8.0	2.004 / 108.82 / 4.2

Table D: **Longer-timestep comparison on MOVi-A.** Each cell reports position RMSE (m) ↓ / orientation RMSE (deg) ↓ / vIoU (%) ↑ at the indicated raw-frame horizon. NA denotes an unreported value; † marks our reimplementation. Best metric values within each training block are bolded.

Timestep	Train	Method	50	100	200	300	400
5	*Pair*	RigidFormer†	0.056 / 2.77 / 86.3	0.268 / 12.80 / 58.7	0.937 / 55.54 / 21.0	1.730 / **92.53** / 7.6	2.505 / **108.99** / 3.8
		Ours	**0.038 / 2.02 / 92.8**	**0.160 / 9.91 / 78.2**	**0.699 / 52.04 / 31.0**	**1.451** / 100.50 / **10.3**	**2.161** / 110.23 / **5.8**
	MOVi-A	RigidFormer	0.035 / 3.33 / NA	**0.148** / 15.08 / NA	NA / NA / NA	NA / NA / NA	NA / NA / NA
		RigidFormer†	0.035 / 2.26 / 90.6	**0.148** / 10.84 / 71.4	**0.606 / 51.74** / 30.2	**1.273 / 89.86** / 10.0	**1.894 / 105.12** / 5.2
		Ours	**0.033 / 2.07 / 93.9**	0.153 / **10.24 / 81.1**	0.720 / 53.55 / **32.8**	1.496 / 101.92 / **10.7**	2.234 / 111.96 / **5.8**
10	*Pair*	RigidFormer†	0.046 / 2.08 / 87.8	0.257 / 11.87 / 59.0	0.950 / 54.95 / 20.3	1.761 / **90.99** / 6.9	2.548 / **107.29** / 3.6
		Ours	**0.033 / 1.38 / 93.7**	**0.161 / 8.99 / 77.7**	**0.713 / 52.41 / 30.3**	**1.469** / 102.24 / **9.1**	**2.193** / 113.15 / **4.9**
	MOVi-A	RigidFormer	**0.018 / 1.47** / NA	**0.118** / 11.93 / NA	NA / NA / NA	NA / NA / NA	NA / NA / NA
		RigidFormer†	0.025 / 1.62 / 92.4	0.129 / 9.77 / 73.2	**0.531 / 50.19 / 32.8**	**1.136 / 88.74 / 11.8**	**1.716 / 102.95 / 6.2**
		Ours	0.023 / **1.47 / 95.3**	0.140 / **9.76 / 81.4**	0.715 / 56.39 / 31.6	1.506 / 105.32 / 9.2	2.228 / 115.49 / 4.9

Table E: **Longer-timestep comparison on MOVi-S.** Each cell reports position RMSE (m) ↓ / orientation RMSE (deg) ↓ / vIoU (%) ↑ at the indicated raw-frame horizon. NA denotes an unreported value; † marks our reimplementation. Best metric values within each training block are bolded.

Timestep	Train	Method	50	100	200	300	400
5	*Pair*	RigidFormer†	0.053 / 2.18 / 87.2	0.247 / 9.90 / 60.1	0.861 / **48.32** / 23.4	1.537 / **98.34 / 9.2**	**2.214 / 113.22 / 4.8**
		Ours	**0.037 / 1.62 / 92.8**	**0.148 / 7.27 / 79.1**	**0.651** / 48.60 / **33.1**	**1.469** / 105.33 / 7.9	2.270 / 116.69 / 3.5
	MOVi-S	RigidFormer	**0.021** / 2.11 / NA	**0.086** / 9.58 / NA	NA / NA / NA	NA / NA / NA	NA / NA / NA
		RigidFormer†	0.030 / 1.73 / 92.5	0.116 / 7.96 / 74.9	**0.452 / 43.15 / 35.7**	**0.978 / 93.89 / 12.1**	**1.503 / 109.99 / 6.0**
		Ours	0.037 / **1.66 / 93.6**	0.168 / **7.58 / 81.1**	0.741 / 50.02 / 34.8	1.578 / 105.87 / 8.2	2.406 / 116.55 / 3.5
10	*Pair*	RigidFormer†	0.047 / 1.86 / 88.0	0.240 / 9.36 / 60.3	0.857 / **47.46** / 22.3	1.550 / **96.89 / 7.8**	**2.243 / 110.25 / 3.7**
		Ours	**0.033 / 1.23 / 93.5**	**0.146 / 6.56 / 79.1**	**0.668** / 49.27 / **32.1**	**1.516** / 107.75 / 6.6	2.351 / 115.47 / 2.7
	MOVi-S	RigidFormer	**0.014 / 1.09** / NA	**0.076** / 7.47 / NA	NA / NA / NA	NA / NA / NA	NA / NA / NA
		RigidFormer†	0.023 / 1.39 / 94.0	0.103 / 7.36 / 76.6	**0.398 / 42.00 / 38.3**	**0.838 / 90.77 / 16.6**	**1.297 / 106.61 / 8.6**
		Ours	0.030 / 1.26 / **94.6**	0.156 / **6.81 / 81.7**	0.722 / 51.35 / 34.0	1.590 / 109.84 / 6.7	2.442 / 117.23 / 2.6

In the Pair-trained setting, our model achieves lower position error and higher vIoU at every MOVi-A horizon for both timesteps, while also reducing orientation error through frame 200. The advantage therefore persists under both a change in shape distribution and a coarser temporal resolution. On MOVi-S, the same model retains lower position error through frame 300, higher vIoU through frame 200, and lower orientation error through frame 100 at both timesteps, demonstrating a consistent transfer pattern across a second object distribution. The in-domain results also show strong early geometric overlap, with our model attaining the highest vIoU at frames 50 and 100 in every block and at all five horizons on MOVi-A for $s = 5$. Overall, the shared Pair-trained model transfers without target-specific interaction training and preserves complete rigid geometry particularly well over early and intermediate rollout horizons.

A.6 Evaluation Protocol and Metric Details

Following RigidFormer, we evaluate physical timesteps $s \in \{1, 5, 10\}$. A model prediction advances the state by s raw frames, corresponding to $\Delta t_s = s\delta t$ with $\delta t \approx 0.00417\,\mathrm{s}$. We generate an independent autoregressive rollout for each timestep and evaluate all methods at matched raw-frame horizons $h \in \{50, 100, 200, 300, 400\}$. Thus, comparisons within a column use the same elapsed physical time and do not mix rollouts produced at different temporal resolutions.

Position RMSE is computed from object centers of mass in meters. Orientation RMSE uses the geodesic distance between predicted and ground-truth rotations and is reported in degrees. vIoU measures volumetric overlap between the predicted and ground-truth rigid geometry and is reported as a percentage. All three metrics are computed per evaluation frame before aggregation over the test scenes.

The early horizons retain direct comparability with prior work, while the later horizons probe collision response and subsequent error accumulation. In the MOVi evaluation split, a substantial portion of first contacts occurs after the earliest evaluation window, with many contacts beginning between raw frames 100 and 199. Reporting through frame 400 therefore distinguishes accurate pre-contact extrapolation from stable post-contact dynamics.

A.7 Additional Qualitative Results

The supplementary qualitative set complements Figure 5 with dedicated *Single* and *Pair* rollouts and additional in-domain MOVi sequences. Figure A focuses on the first two settings: *Single* isolates self-dynamics drift, while *Pair* additionally exposes collision response and interaction error over the full temporal evolution. Figure B presents in-domain MOVi rollouts, enabling a direct qualitative assessment when the interaction model is trained on the corresponding target distribution.

Extreme out-of-distribution stress-test setup. Figure 4 presents the stress-test rollouts in the main paper. We construct the scenes from the same Kubric asset library but use compositions far outside the training distribution. Cubes are stacked into pyramids of increasing height, and the Kubric calf asset named *Spot* with varied scale and mass is launched into the structure. The model is applied without fine-tuning. This design changes object count, scene topology, and impact regime simultaneously, providing a deliberately severe test of compositional generalization.

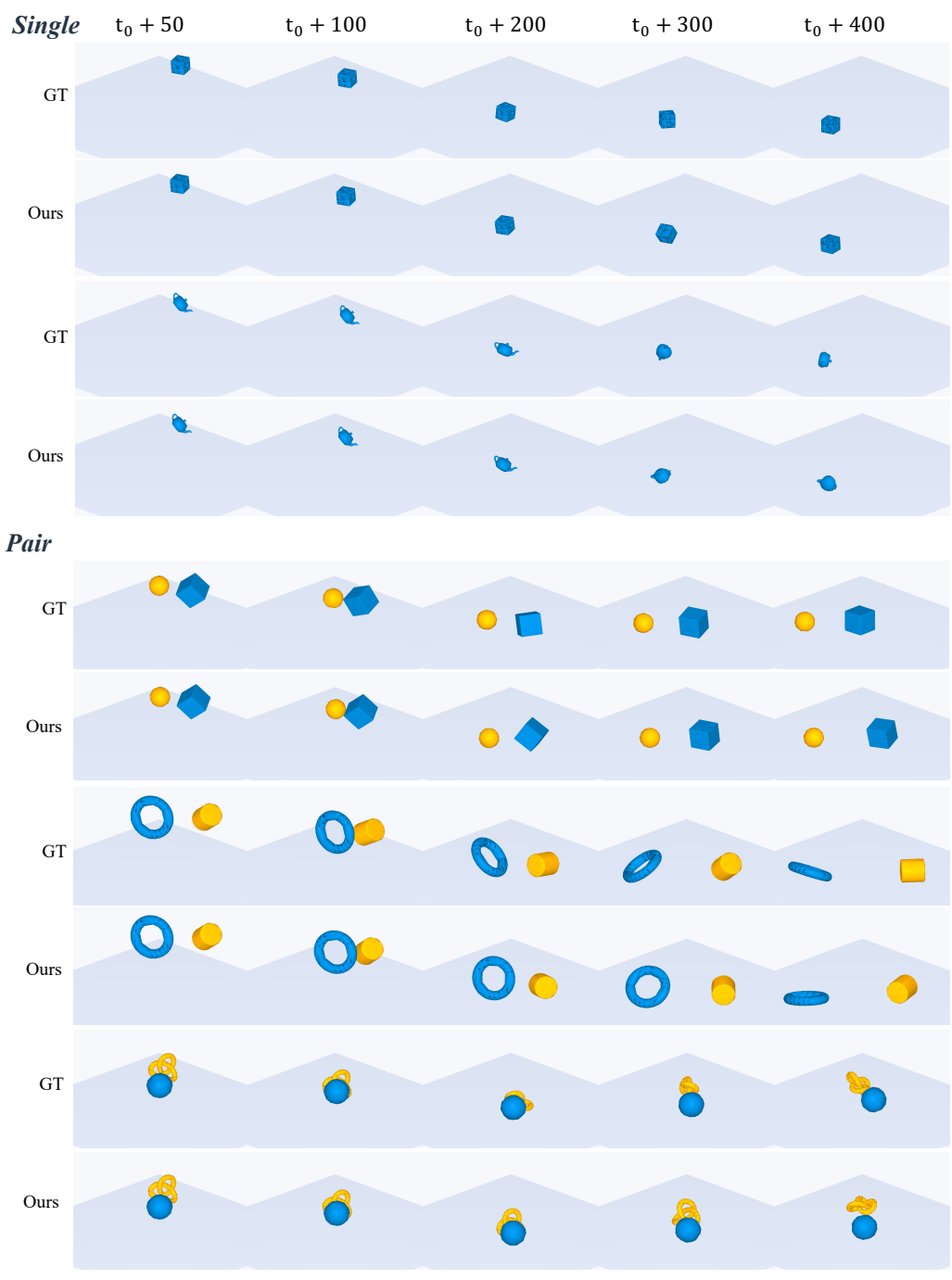

Figure A: **Qualitative results on *Single* and *Pair*.** Autoregressive predictions are shown alongside ground truth at the displayed rollout horizons. *Single* isolates self dynamics, whereas *Pair* evaluates the composition of self dynamics with learned pairwise contact corrections.

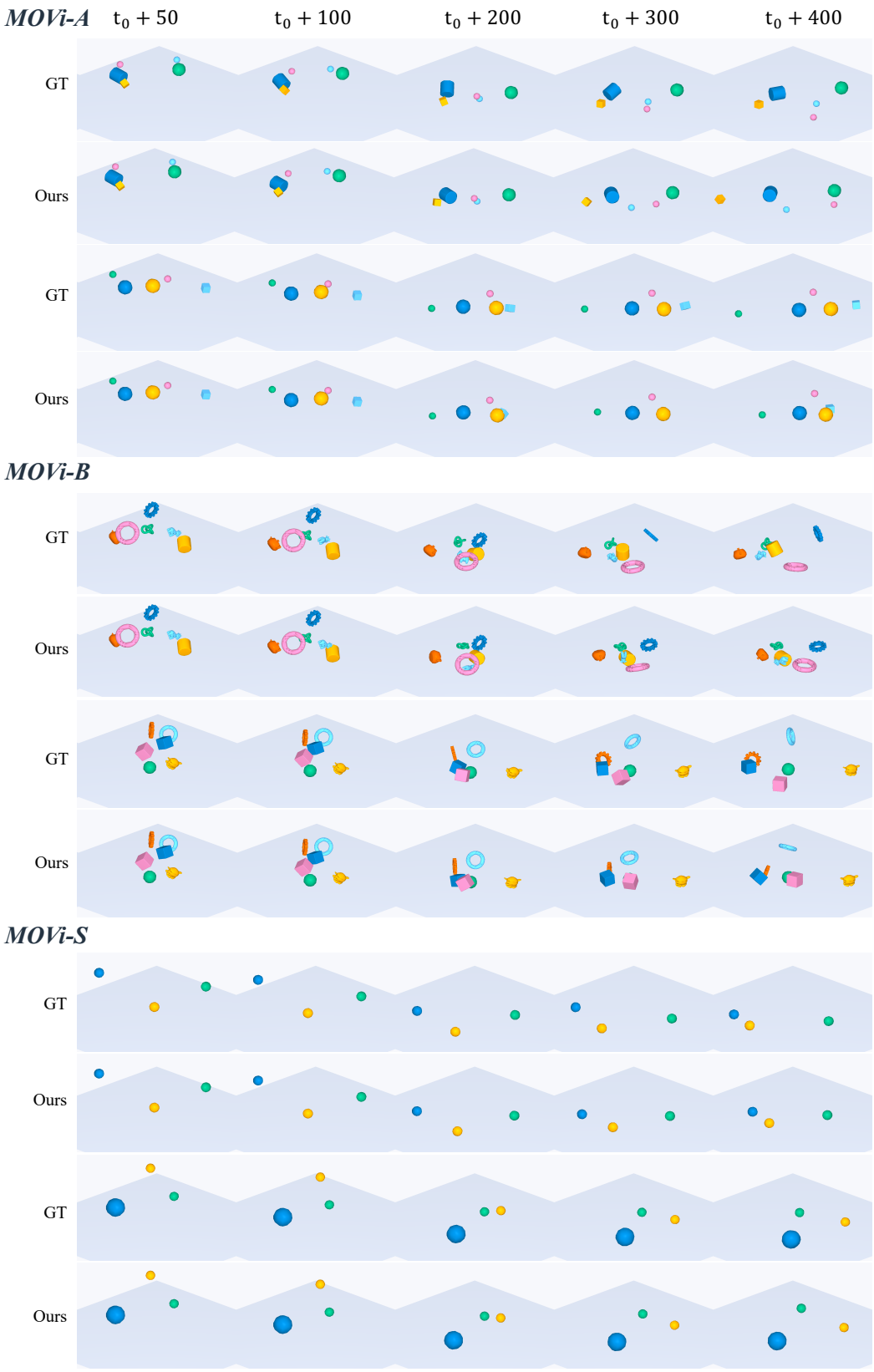

Figure B: **In-domain qualitative results on MOVi.** Autoregressive predictions are shown alongside ground truth for the displayed MOVi sequences and rollout horizons.