# Generalizable Physics Simulation through Compositional Energy Minimization

**Alexandru Oarga**
University of Barcelona

**Yilun Du**
Harvard University

## Abstract

Learning the dynamics of interacting physical systems is a central challenge in machine learning for the physical sciences. Prevailing architectures such as Graph Neural Networks (GNNs) and feedforward models often fail to explicitly capture the compositional nature of physical laws. Instead of modeling the forces that govern a system, these networks learn to approximate aggregate state transitions. As a result, they lack a causal understanding of the underlying physical interactions, leading to poor generalization in out-of-distribution systems with varying numbers of entities or novel configurations. In this work, we introduce Compositional Potential Minimization (CPM), a framework that casts simulation as an energy minimization procedure over a composition of learned force potentials. Each force component is represented as a separate, interpretable energy landscape, and interaction dynamics emerge from their addition. Inference in CPM is framed as a global energy minimization problem over an arbitrary number of composed functions. Because CPM learns to disentangle separate forces during training, it enables seamless generalization at test time to larger-scale systems and unseen configurations through straightforward compositions of the learned energy components. Our experiments show that CPM significantly outperforms previous state-of-the-art GNN-based solutions in 2D, 3D, and particle-based simulations. Project website can be found at: https://alexoarga.github.io/compositional_physics_simulation/

## 1 Introduction

Physics simulation plays a crucial role across many scientific and engineering disciplines. Humans are naturally able to reason about physical systems by understanding their underlying mechanisms and identifying the individual components of a system, as well as the interactions between these components [6, 48]. This understanding enables us to predict how these objects will behave in different conditions or novel scenarios [50, 52].

Recently, multiple neural physics simulators have been proposed that have shown strong performance in modeling physical dynamics. However, most of these simulators are based on feedforward neural networks, which tend to perform poorly when evaluated in out-of-distribution settings [37, 42]. Instead of learning to disentangle individual components and their interactions, these models rely on suboptimal heuristics that approximate the training data but fail to generalize beyond it, which results in a lack of causal understanding of the physical simulation [44, 36].

In this work, we present Compositional Potential Minimization (CPM), a new approach to learning physics simulators. Instead of learning a monolithic feedforward simulator, we express physical interactions as the composition of multiple energy functions. We parametrize the forces acting on each object or particle using learned energy functions. Interaction forces—in particular, additive forces—are modeled using additive compositions of these energy functions. In other words, in this

Corresponding author: ydu@seas.harvard.edu

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

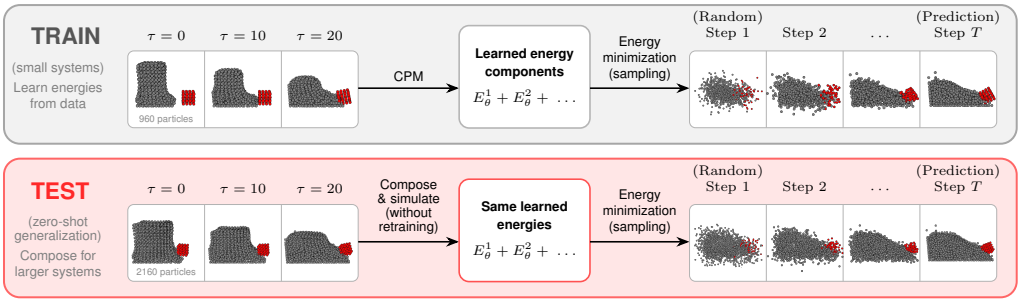



Figure 1: **Compositional Generalizable Physics Simulation.** We express physical interactions as a composition of learned energy functions. During training, these energy components are learned from small systems. At test time, these learned energies are combined without retraining to model larger systems than those seen during training. Minimizing the composed energy enables zero-shot simulation and prediction in larger systems with an arbitrary number of interacting components.

formulation, the energy functions learn physical potentials. These learned energy functions are trained without explicit supervision of individual forces, so the model learns to disentangle the forces of a system from next-state supervision alone. As a consequence of this disentanglement, our approach can seamlessly compose additional particles at test time, enabling zero-shot generalization to larger systems containing more particles than those seen during training.

We evaluate our methods on 2D physics simulations, 3D physics environments, and particle-based simulations. Our results show that our approach outperforms well-known state-of-the-art Graph Neural Network (GNN)-based physics simulators in in-distribution settings. Moreover, our method significantly outperforms these baselines in out-of-distribution scenarios where the number of interaction components exceeds that observed during training. This demonstrates the strong generalization capabilities of our approach, which arise from its ability to compose learned energy functions to scale to larger and more complex systems.

Our contributions are threefold. First, we propose CPM, a compositional framework that casts physics simulation as the minimization of a composed energy landscape, where potentials are combined additively to model interactions. Second, this compositional structure enables generalization to systems larger than those seen during training, with inference scaling to an arbitrary number of components without retraining. To improve sampling from complex composed landscapes, we propose refining sampling with iterative gradient descent. Finally, we empirically validate our method through a set of physical simulation tasks, outperforming foundational and state-of-the-art learnable physics simulators in generalization.

## 2 Related Work

**Neural Physics Simulators.** Neural physics simulators have emerged as a data-driven alternative to explicit numerical solvers, learning physical dynamics directly from observations. One major line of work builds on Graph Neural Networks (GNNs), learning interactions between physical entities by representing objects as nodes and interactions as edges over a graph [5, 46, 15]. Dynamic modification of the graph structure allows modeling time-varying interactions between the modeled entities [39, 29]. More recent extensions adapt these ideas to unstructured meshes [22] and hierarchical representations designed to capture long-range dependencies [7, 39]. A complementary line of work imposes geometric symmetries, especially translation and rotation invariance, to improve generalization [54, 56, 47]. Despite strong in-distribution performance, these approaches model interactions as a monolithic feedforward process, effectively approximating state transitions rather than disentangling the underlying forces. This limits their ability to generalize to novel configurations [33, 55].

**Physics-Informed and Structured Dynamics Models.** A separate family of methods incorporates physical priors directly into the model architecture, instead of learning dynamics from data. Approaches such as Hamiltonian Neural Networks (HNNs) [24] and Lagrangian Neural Networks (LNNs) [17] incorporate energy conservation principles by construction. Neural ODEs [16, 32] and symbolic regression approaches [18, 12] provide a principled framework for deriving the equations governing a system. While these methods offer strong inductive biases, they are typically designed

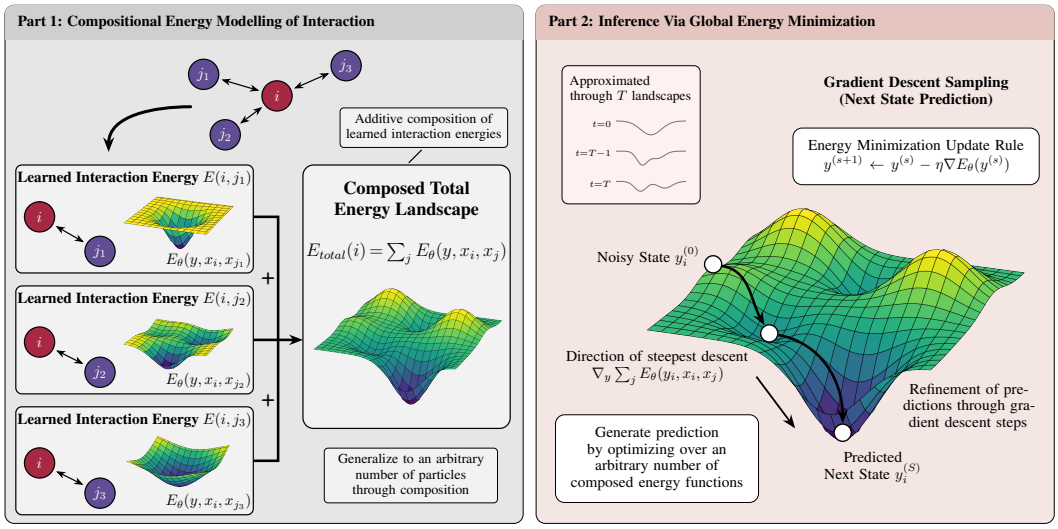



Figure 2: **Compositional Potential Minimization (CPM).** (Left) CPM models interactions acting on object $i$ by learning a pairwise potential $E_\theta(y, x_i, x_j)$ for each object. The total potential $\sum_j E_\theta(y, x_i, x_j)$ is obtained via additive composition of these pairwise potentials, forming a single composed energy landscape. This compositional structure allows CPM to generalize to systems with an arbitrary number of interacting components. (Right) Inference in CPM is cast as minimization of the composed potential. Starting from a noisy state, we follow the gradient $\nabla_y E_\theta(y)$ via iterative gradient descent steps, progressively refining the prediction until convergence to the next state.

for small closed systems with a fixed number of degrees of freedom. CPM extends this by enabling compositionality, allowing the global energy landscape to be constructed dynamically at test time.

**Molecular Dynamics Simulations.** In molecular dynamics, machine-learned interatomic potentials aim to combine the accuracy of quantum-mechanical calculations with the speed of classical empirical potentials [21, 9]. A common design choice is to decompose the total energy into a sum of local contributions [9, 4, 45], often using descriptors of an atom's neighborhood that are invariant to translation, rotation, and permutation [4, 8]. Some methods instead predict forces directly rather than differentiating a predicted energy [11, 13]. Like these approaches, CPM relies on an additive, local decomposition of interactions. However, CPM is not fitted to quantum-mechanical energies or forces. Instead, it learns to disentangle the individual forces of a system without explicit supervision.

**Video Prediction.** Autoregressive video generation [53, 30, 2] and diffusion-based video generation [27, 10], trained on internet-scale data, have demonstrated remarkable capabilities in realistic visual dynamics. Several works attempt to equip video generation models with physical dynamics reasoning [14, 1, 40]. Others try to leverage video priors for 3D simulation [28, 41]. However, despite their fidelity, these large-scale video models have been shown to lack a grounded understanding of the underlying physics, and consequently fail to generalize reliably to out-of-distribution settings [31, 3].

**Energy-Based Models.** Our work is related to Energy-Based Models (EBMs) [20, 51, 34], initially proposed to model probabilistic distributions [49, 25, 23, 35]. In this work, rather than modeling a data distribution, we train energy functions to represent the individual force potentials governing a system. Physical simulation is then performed by minimizing the composed energy landscape, yielding an inference procedure that supports compositional generalization.

# 3 Method

## 3.1 Composition of Physical Potentials

Let $D = \{(X, Y)\}$ be a training dataset representing physical simulations, where $X = \{x_1, \ldots, x_N\} \in \mathbb{R}^{N \times d}$ represents the current states of $N$ objects and $Y = \{y_1, \ldots, y_N\} \in \mathbb{R}^{N \times d}$ denotes their states at the next timestep, having $x_i, y_i \in \mathbb{R}^d$. For each object $i$, we define $\mathcal{Z}_i = \{j : j \neq i, j \text{ interacts with } i\}$ as the set of interacting objects.

Rather than learning a transition function for the entire scene, we model the physics of object $i$ via a learned pairwise potential $E_\theta : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, parametrized as an EBM. Given the set of interacting objects $\mathcal{Z}_i$ for object $i$, the total potential acting on object $i$ is then defined through additive composition of these pairwise contributions:

$$E_{total}(y) = \sum_{j \in \mathcal{Z}_i} E_\theta(y, x_i, x_j) \tag{1}$$

This formulation treats interactions as independent forces that can be combined, providing the basis of our compositional approach. Since every interaction is parametrized by the same potential function $E_\theta$, any number of object interactions can be modeled while remaining permutation invariant.

### 3.2 Physics Prediction as Energy Minimization

In the CPM framework, the future state $\hat{y}_i$ is not directly regressed, but instead discovered by solving for the state that minimizes the joint energy:

$$\hat{y}_i = \arg\min_{y_i} \sum_{j \in \mathcal{Z}_i} E_\theta(y_i, x_i, x_j) \tag{2}$$

We can solve this optimization problem via iterative gradient descent:

$$y_i^{(k+1)} = y_i^{(k)} - \eta \sum_{j \in \mathcal{Z}_i} \nabla_y E_\theta(y_i^{(k)}, x_i, x_j) \tag{3}$$

where $y_i^{(k)}$ is the state of object $i$ at the $k$-th iteration, $y_i^{(0)}$ is the initial state drawn from a noise distribution, and $\eta$ is the step size. That is, the next state is obtained by jointly optimizing the individual contribution of each interacting neighbor $j$ of object $i$.

### 3.3 Learning Compositional Potentials

To train the potential function $E_\theta$, we cast learning as compositional energy minimization over a sequence of $T$ noise levels. Concretely, we train a sequence of $T$ energy-based models $E_\theta(\cdot, t)$, optimized using the energy-based denoising diffusion training objective of [19]. At each timestep $t$, we train the gradient of $E_\theta(\cdot, t)$ to point toward a clean target state $y_i$. That is, the composed landscape is trained to denoise a corrupted input $y_i^*$. Formally, given a ground-truth label $y_i$ and a Gaussian-corrupted state $y_i^* = \sqrt{1-\sigma_t^2}\, y_i + \sigma_t\, \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$, we define the diffusion training objective as:

$$\mathcal{L}_{\text{MSE}}(\theta) = \mathbb{E}_{y_i, \epsilon \sim \mathcal{N}(0,I)} \left[ \left\| \epsilon - \sigma_t \sum_{j \in \mathcal{Z}_i} \nabla_{y_i} E_\theta(y_i^*, x_i, x_j, t) \right\|^2 \right] \tag{4}$$

with $E_\theta(y_i^*, x_i, x_j, t)$ being an explicitly defined scalar function[1], and $\sigma_t$ a fixed noise schedule.

This formulation allows us to supervise the gradient of the energy function at each optimization step $t$, instead of having to backpropagate through a sequence of $T$ steps. Consequently, we learn an energy gradient that maps a noisy state to the target distribution, and thus perform inference by following the energy gradient. To generate outputs, we can use, for example, the update rule given in (3).

### 3.4 Improving Sampling with Gradient Descent

Under CPM, minimizing energy corresponds to generating a prediction for the future state. Existing methods utilize MCMC sampling methods to sample from this landscape, starting from a noisy initialization (usually Gaussian noise), and following a transition distribution. A common approach for MCMC sampling of energy landscapes is Unadjusted Langevin Dynamics (ULA) [20, 43], formally defined as:

$$y_i^{(k+1)} = y_i^{(k)} - \eta \sum_{j \in \mathcal{Z}_i} \nabla_y E_\theta(y_i^{(k)}, x_i, x_j) + \sqrt{2\eta}\xi, \quad \xi \sim \mathcal{N}(0, 1) \tag{5}$$

[1] Empirically, in this work we use the same function as in the original paper [19], that is, $E_\theta(x_t, t) = \|s_\theta(x_t, t)\|^2$, where $s_\theta(x_t, t)$ is a vector-output neural network. For simplicity, we omit the dependency on $t$ from the energy notation when clear from context.

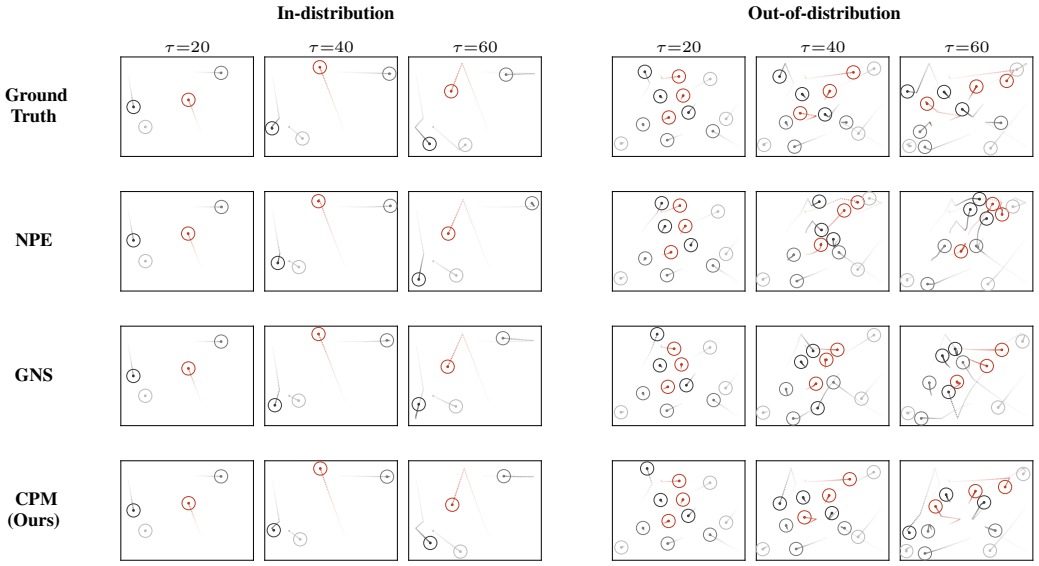



Figure 3: **Qualitative Results of 2D Simulation.** We compare our method with the selected baselines in both in-distribution and out-of-distribution settings at rollout timesteps $\tau$=20, $\tau$=40, and $\tau$=60. Our method produces higher-quality predictions in both settings. In-distribution, all methods produce physically consistent trajectories, with CPM achieving the closest agreement with the ground truth. NPE and GNS deteriorate in the out-of-distribution setting.

where $y_i^{(k)}$ is the state of object $i$ at the $k$-th iteration, and $\eta$ is the step size. This is fundamentally gradient descent with added noise to avoid local minima.

However, as the number of interacting components increases, the composed energy landscape becomes more complex, making ULA sampling inefficient. Since CPM has direct access to the energy gradient, we propose to augment the MCMC chain with $S$ inner gradient descent steps at each timestep $t$. Before transitioning to the next energy landscape $E(\cdot, t-1)$, we first refine the current sample by running $S$ gradient descent steps on $E(\cdot, t)$. The converged sample is then used as the starting point for the next timestep. The full procedure is given in Algorithm 1.

**Algorithm 1** Inference with Gradient Descent Sampling

**Input:** $T$ optimization steps, $S$ gradient descent steps, $\xi \sim \mathcal{N}(0, I)$

Initialize $y^{(T)} \sim \mathcal{N}(0, I)$
**for** $t = T, \ldots, 1$ **do**
  $\tilde{y}^{(0)} \leftarrow y^{(t)}$
  **for** $s = 1, \ldots, S$ **do**
    $\tilde{y}^{(s)} \leftarrow \tilde{y}^{(s-1)} - \eta \nabla E_\theta(\tilde{y}^{(s-1)}, t)$
  **end for**
  $y^{(t)} \leftarrow \tilde{y}^{(S)}$
  $y^{(t-1)} \leftarrow y^{(t)} - \sigma_t \nabla E_\theta(y^{(t)}, t) + \sigma_t \xi$
**end for**
**return** $\mathbf{y}^{(0)}$

### 3.5 Generalizable Physical Interactions

A key property of CPM is its ability to generalize at test time to systems larger than those seen during training. In our setting, we represent interaction as the additive composition of the same potential function $E_\theta$. As a direct consequence, scaling to an arbitrary number of components requires no retraining. In this setting, inference becomes:

$$\hat{y}_i = \arg\min_y \sum_{j=1}^{P} E_\theta(y, x_i, x_j) \tag{6}$$

where $P$ is an arbitrary number of objects larger than $N$.

In CPM, each potential function $E_\theta$ is evaluated independently, and the global dynamics emerge from their sum. Consequently, zero-shot generalization emerges from jointly optimizing the individual potential functions. Minimizing the composed energy simultaneously accounts for all interactions, allowing the predicted next state to reflect the interaction with an arbitrary number of neighbors.

| | Similar Distribution | | Larger Distribution | |
|---|---|---|---|---|
| **Model** | **MSE Position** | **MSE Velocity** | **MSE Position** | **MSE Velocity** |
| NPE | $0.3739 \pm 0.6333$ | $1.1372 \pm 1.4008$ | $1.2163 \pm 0.5887$ | $3.2941 \pm 1.3289$ |
| GNS | $0.3971 \pm 0.6177$ | $1.1374 \pm 1.2619$ | $0.9206 \pm 0.4403$ | $2.6434 \pm 1.0584$ |
| CPM (Ours) | $\mathbf{0.1648 \pm 0.3344}$ | $\mathbf{0.7537 \pm 1.1207}$ | $\mathbf{0.6245 \pm 0.3362}$ | $\mathbf{2.1156 \pm 0.9692}$ |

Table 1: **2D Simulation Evaluation.** We compare our method against widely used neural physics simulators from the literature. We report mean $\pm$ std position and velocity MSE for both in-distribution with 4 balls and out-of-distribution with 12–14 balls. Our method achieves a lower MSE in both settings. CPM results were sampled using $S$=10.

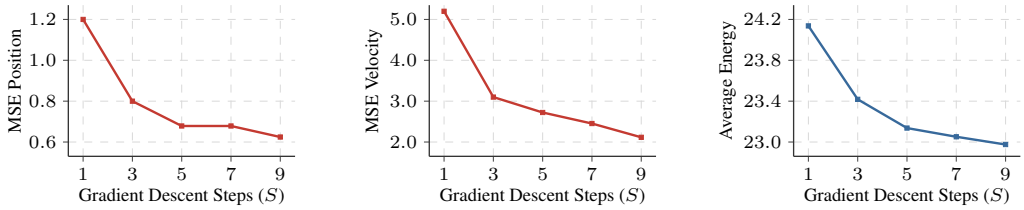



Figure 4: **Gradient Descent Steps Ablation.** Ablation study on the number of gradient descent steps performed per timestep in the 2D out-of-distribution setting. We report position MSE, velocity MSE, and average energy of the sampled solutions across simulation timesteps. Performance improves and energy decreases as the number of gradient descent steps increases.

# 4 Experiments

## 4.1 2D Simulation

**Data.** We consider a world consisting of an arbitrary number of balls of uniform mass in a bounded space. We follow a setup similar to the one in [15]. We randomly generate 30,000 trajectories, each with 60 timesteps, and a number of balls $k$=4. For the out-of-distribution setting, we generate 100 trajectories with $k \in [12, 14]$. All environments have a resolution of $600 \times 800$ pixels, and the velocities are randomly sampled in the range $[0, 60]$ pixels/second. In each trajectory, balls do not have global visibility. Instead, they can only observe other balls within a radius of 240 pixels.

**Baselines.** We compare our method with foundational and widely used physical GNN-based simulation engines. These include Neural Physics Engine (NPE) [15] and Graph Network-based Simulators (GNS) [46].

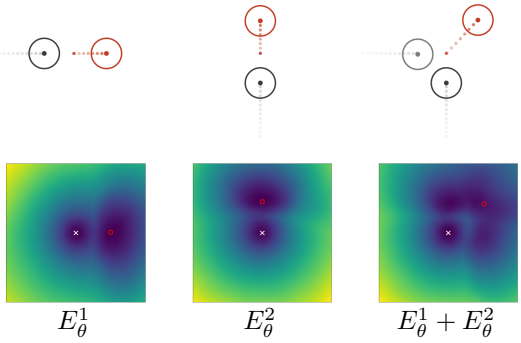



Figure 5: **Potential Maps Visualization.** Each column depicts the potential $E_\theta$ of the red particle induced by the interaction with neighboring particles. (Left) Potential $E_\theta^1$ arising from the interaction of a single neighbor moving horizontally. (Center) Potential $E_\theta^2$ arising from the interaction of a single neighbor moving vertically. (Right) Composed potential $E_\theta^1 + E_\theta^2$ when both neighbors interact simultaneously with the red particle. The minimum of the composed landscape corresponds to the predicted movement.

**Quantitative Results.** In Table 1, we report the MSE between the predicted and ground-truth position and velocity, averaged over 60-timestep rollouts, for both in-distribution and out-of-distribution settings. In-distribution, CPM outperforms all baselines on both metrics, achieving a position MSE of 0.1648 compared with 0.3739 MSE for NPE and a velocity MSE of 0.7537 compared with 1.1372 MSE for NPE. This difference is more significant out-of-distribution, where our method reaches a position MSE of 0.6245 compared to 1.2163 MSE for NPE and 0.9206 MSE for GNS.

**Qualitative Results.** In Figure 3 we show trajectory predictions for all methods in an in-distribution setting (4 balls) and an out-of-distribution setting (12 balls). While all methods produce consistent trajectories in-distribution, CPM achieves visibly higher accuracy in the simulation. In the out-of-distribution setting, the baselines degrade substantially. NPE trajectories collapse into unphysical curved paths, while GNS suffers from a loss of precision that compounds over the rollout. As a direct

consequence of the compositional nature of CPM, the simulation remains stable over an arbitrary number of interacting objects. In Figure 5 we provide an example of the composition of learned potentials in a 3-particle system.

**Ablation Studies.** In Figure 4, we ablate the effect of the number of gradient descent steps performed per timestep in the out-of-distribution setting. We report position MSE, velocity MSE, and mean energy of sampled solutions averaged over timesteps. Performance improves with more steps, converging after $S$=7 steps. Notably, lower energy correlates with better predictive accuracy, suggesting that the gradient descent successfully refines samples from the composed potential.

### 4.2 2D Simulation Beyond Elastic Pairwise Interactions

The elastic pairwise setting above tests compositional scaling over object count. We further stress-test CPM under two physically harder interaction regimes that break assumptions implicit in that setup: **(i)** higher-order potentials, where interactions are not pairwise-decomposable, and **(ii)** anisotropic interactions, where interactions depend on orientation and direction. We follow the same experimental setup and baselines as in Section 4.1, with the simulation setting adapted to the new interaction regimes. Simulation, composition, and training details are in Appendices A.2 and A.3.

**Higher-Order Potentials.**

We generate 2D simulations in which balls interact through both elastic collisions and smoothed particle hydrodynamics (SPH) forces. For this setting, we extend CPM with a higher-order potential that is composed additively with the pairwise potentials. Appendix B.2 reports the quantitative results: CPM achieves the lowest average position MSE in both settings, with 0.1093 in-distribution (compared to 0.2320 for GNS and 0.3202 for NPE) and 1.4438 out-of-distribution (compared to 1.7376 for GNS and 1.8029 for NPE).

**Anisotropic Interactions.**

We generate 2D simulations in which balls interact through elastic collisions as well as anisotropic interactions, i.e., interactions that depend on both the relative direction between balls and their orientations. For this setting, we extend CPM so that the pairwise potentials are additionally conditioned on direction and orientation. Appendix B.3 reports the quantitative results: CPM achieves the lowest average position MSE in both settings, with 0.2730 in-distribution (compared to 0.3949 for GNS and 0.4619 for NPE) and 0.7633 out-of-distribution (compared to 1.1424 for GNS and 1.4786 for NPE). Similarly, CPM achieves the lowest average angular velocity MSE, with 1.9371 in-distribution (compared to 2.2466 for GNS and 2.5269 for NPE) and 9.4770 out-of-distribution (compared to 10.5536 for GNS and 12.1953 for NPE).

### 4.3 3D Simulation

**Data.** We adopt the same 3D environment as in [38], which simulates the motion and interaction of three rigid cubes. The cubes are arranged in a vertical stack and released from above, colliding with each other as they fall. Each cube consists of 64 particles. Following the setup of [38], the gravitational acceleration is not constant; instead, for each simulation, it is sampled uniformly from the range $[-15.0, -5.0]$. For each rollout, we consider 40 timesteps.

**Baselines.** We compare with well-established GNN-based physics simulators from the literature, including Graph Network-based Simulators (GNS) [46], Dynamic Particle Interaction Networks (DPI) [39], equivariant graph methods such as Equivariant Graph Neural Networks (EGNN) [47], and state-of-the-art equivariant methods that explicitly disentangle gravity such as Subequivariant Graph Neural Networks (SGNN) [26].

**Quantitative Results.** In Table 2 we report MSE averaged over 500 rollouts between the ground-truth and predicted velocities across different training set sizes and simulation timesteps. CPM consistently outperforms state-of-the-art GNN-based baselines in all settings. In the low-data regime (200 training samples), CPM achieves an MSE of 0.30 compared to 0.72 for SGNN at $\tau$=20, and 1.59 compared to 2.44 at $\tau$=40. In the largest training set (5,000 samples), the gap widens further, with CPM reaching 0.03 compared to 0.20 for SGNN at $\tau$=20, and 0.36 compared to 1.01 at $\tau$=40. This highlights how the decomposition of forces into separate potentials of our CPM framework enables more data-efficient learning of physical dynamics.

| | \|Train\| = 200 | | \|Train\| = 500 | | \|Train\| = 1000 | | \|Train\| = 5000 | |
|---|---|---|---|---|---|---|---|---|
| | $\tau$=20 | $\tau$=40 | $\tau$=20 | $\tau$=40 | $\tau$=20 | $\tau$=40 | $\tau$=20 | $\tau$=40 |
| GNS | 2.40±1.32 | 6.79±3.43 | 1.89±0.87 | 5.41±2.78 | 1.09±0.70 | 3.38±2.15 | 0.65±0.43 | 2.38±1.66 |
| DPI | 1.71±0.87 | 5.37±2.86 | 1.48±0.78 | 4.67±2.55 | 1.24±0.72 | 3.97±2.35 | 0.52±0.36 | 2.40±1.53 |
| EGNN | 1.89±1.34 | 5.66±2.81 | 1.07±0.71 | 3.94±2.20 | 0.95±0.44 | 2.72±1.75 | 0.78±0.45 | 2.63±1.72 |
| SGNN | 0.72±0.60 | 2.44±1.74 | 0.39±0.23 | 1.47±1.26 | 0.31±0.17 | 1.09±0.93 | 0.20±0.14 | 1.01±0.97 |
| CPM (Ours) | **0.30**±0.35 | **1.59**±1.30 | **0.14**±0.39 | **1.03**±1.28 | **0.06**±0.10 | **0.65**±0.91 | **0.03**±0.05 | **0.36**±0.49 |

Table 2: **3D Simulation Evaluation.** We compare our method with state-of-the-art neural physics simulators. We report mean ± std MSE ($\times 10^{-2}$) averaged over 500 trajectories for different training set sizes and simulation timesteps ($\tau$=20 and $\tau$=40). Our method achieves the lowest MSE across all settings. With CPM, we report results obtained with $S$=7.

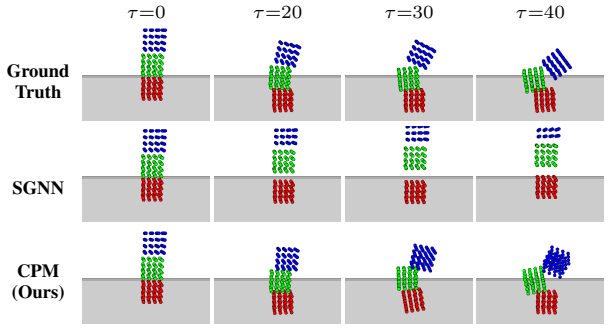



Figure 6: **Qualitative Results of 3D Simulation.** We show a qualitative comparison of the rollout simulation of our method and SGNN with 200 training samples and at timesteps $\tau$=0, $\tau$=20, $\tau$=30, and $\tau$=40. CPM maintains a coherent and physically consistent simulation, suggesting a more faithful modeling of underlying physical interactions.

**Qualitative Results.** In Figure 6 we show trajectory predictions in the low-data regime (200 training trajectories). In this data regime, SGNN fails to predict meaningful dynamics, producing an unrealistic trajectory. CPM, despite being trained on the same limited data, is able to maintain a coherent simulation.

**Ablation Studies.** In Figure 7 we ablate the number of gradient descent steps performed per timestep during sampling at $\tau$=20 and $\tau$=40 in the largest training dataset case (5,000 training samples). We observe that prediction quality improves as the number of steps increases, reducing the MSE at $\tau$=20 from 0.11 with $S$=1 to 0.03 with $S$=7. Generally, we observe that more gradient descent steps correlate with lower MSE, indicating that the iterative refinement helps improve the accuracy of the simulation.

### 4.4 Particle Simulation

**Data.** We use the BoxBath setting from [39], where a block of fluid interacts with a rigid cube. The rigid cube position is randomly initialized in the scene. We generated 1,000 rollouts with 150 time steps using the NVIDIA FleX simulator. The fluid is represented using 960 particles and the rigid cube is represented using 64 particles. For out-of-distribution evaluation, we generated 30 additional rollouts with fluids of 1,500 and 2,160 particles each. We keep the rigid-cube particle count at 64 for all the rollouts.

**Baselines.** We compare our method with well-established physics simulators from the literature that have demonstrated state-of-the-art performance in the BoxBath setting, including Graph Network-based Simulators (GNS) [46] and Dynamic Particle Interaction Networks (DPI) [39].

**Quantitative Results.** In Table 3 we report the position MSE between ground-truth and predicted particle positions in the BoxBath setting for different numbers of fluid particles, including the in-distribution case (960 particles) and out-of-distribution cases (1,500 and 2,160 particles). In-distribution, our method outperforms all baselines, achieving an MSE of 0.0117 compared to 0.0147 for DPI and 0.0163 for GNS. This demonstrates that iterative energy minimization produces more accurate predictions than single-pass feedforward approaches. Out-of-distribution, the advantage of CPM becomes more pronounced. DPI diverges at larger particle counts, with unbounded simulation errors. GNS reaches an MSE of 0.0139 at 1,500 particles and 0.0247 at 2,160 particles. CPM

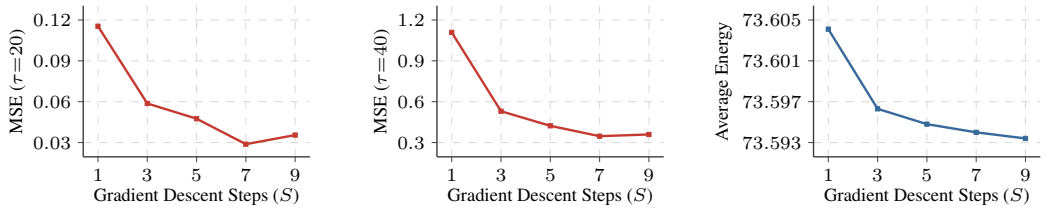



Figure 7: **Gradient Descent Steps Ablation.** Ablation study of the number of gradient descent steps performed per timestep. We report MSE ($\times 10^{-2}$) averaged over 500 trajectories at different simulation timesteps. Additionally, we report the energy of the sampled solutions, averaged over simulation timesteps. Performance improves with increasing number of gradient descent steps.

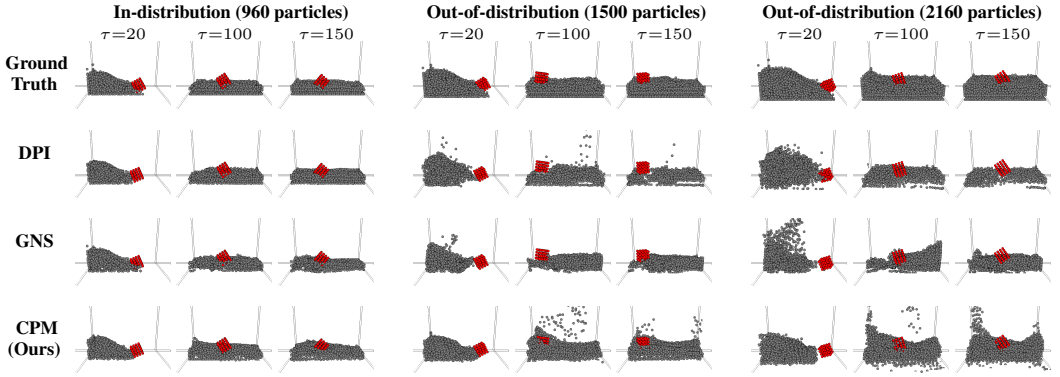



Figure 8: **Qualitative Results of Particle Simulation.** We show a comparison of our method with well-established physics simulators in the BoxBath setting at timesteps $\tau$=20, $\tau$=100, and $\tau$=150, across in-distribution (960 particles) and out-of-distribution (1,500 and 2,160 particles) settings. CPM maintains physically consistent particle dynamics across all settings, generalizing zero-shot to particle counts beyond those seen during training.

| | Similar Distribution | Larger Distribution | |
|---|---|---|---|
| **Model** | 960 Particles | 1500 Particles | 2160 Particles |
| DPI | 0.0147 ± 0.0128 | —† | —† |
| GNS | 0.0163 ± 0.0089 | 0.0139 ± 0.0090 | 0.0247 ± 0.0151 |
| CPM (Ours) | **0.0117 ± 0.0109** | **0.0100 ± 0.0088** | **0.0103 ± 0.0078** |

Table 3: **Particle Simulation Evaluation.** We compare our method with established physics simulators in the BoxBath setting. We report mean ± std position MSE averaged over 150 simulation timesteps for different numbers of fluid particles, including the in-distribution case (960 particles) and out-of-distribution cases (1,500 and 2,160 particles). Our method achieves the lowest MSE across all settings, demonstrating strong generalization to larger particle counts. For CPM, we report results obtained by sampling with $S$=5 steps. †DPI simulations diverge in the out-of-distribution cases, producing unbounded MSE values.

remains stable across both settings, achieving an MSE of 0.0100 and 0.0103, respectively. This demonstrates how the compositional approach enables zero-shot generalization to a much larger number of particles.

**Qualitative Results.** In Figure 8, we provide a qualitative comparison of generated rollouts in the BoxBath setting for both in-distribution and out-of-distribution particle counts. Generally, all methods produce reasonable simulations in-distribution. Out-of-distribution, the baselines exhibit clear physical failures. In the out-of-distribution case, DPI quickly becomes unstable, producing noisy and diverging predictions. Additionally, in both DPI and GNS, particles fail to stack naturally, resulting in a compressed mass that does not spread evenly across the floor of the scene. CPM, by contrast, preserves the shape of the particle mass throughout the rollout, maintaining stable dynamics and physically consistent particle interactions even as the number of particles increases.

**Ablation Studies.** In Appendix B we show that in both in-distribution and out-of-distribution, increasing the number of gradient steps improves the accuracy of the solutions, highlighting the ability to refine sample solutions. In-distribution, MSE is reduced from 0.0113 to 0.0098 with $S$=10. Out-of-distribution, it is reduced from 0.0144 to 0.0100 MSE with $S$=5.

## 5 Limitations and Conclusion

**Limitations.** While CPM demonstrates strong generalization across a range of physical simulation settings, several limitations remain. First, inference requires iterative gradient descent over composed landscapes. During inference, each of the composed potentials is evaluated independently. Increasing the number of interacting components introduces additional computational costs. Future work could explore strategies to accelerate sampling, such as progressive distillation, compressing multiple iterative steps into a single pass, or through learned sparse interaction graphs that reduce the number of pairwise terms. Second, although it has been shown that CPM can be extended to model higher-order interactions, hierarchical or long-range compositions for still more complex systems remain open. Finally, CPM has not yet been extended to control tasks, where the composed energy landscape could accommodate additional objectives as energy terms.

**Conclusion.** In this work, we introduce CPM, a framework that models physics simulation as a minimization of a composed energy landscape. By expressing physical interactions as additive potentials, CPM generalizes zero-shot to systems larger than those seen during training. To improve sampling from composed functions, we propose augmenting MCMC sampling with gradient descent steps. Our approach outperforms foundational and state-of-the-art GNN-based physics simulators in 2D (including elastic, higher-order, and anisotropic interactions), 3D, and particle simulation scenarios.

## Acknowledgments

This work has been made possible in part by a gift from the Chan Zuckerberg Initiative Foundation to establish the Kempner Institute for the Study of Natural and Artificial Intelligence at Harvard University.

## References

[1] Rick Akkerman, Haiwen Feng, Michael J Black, Dimitrios Tzionas, and Victoria Fernández Abrevaya. Interdyn: Controllable interactive dynamics with video diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12467–12479, 2025. 3

[2] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021. 3

[3] Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. Videophy: Evaluating physical commonsense for video generation. *arXiv preprint arXiv:2406.03520*, 2024. 3

[4] Albert P Bartók, Risi Kondor, and Gábor Csányi. On representing chemical environments. *Physical Review B—Condensed Matter and Materials Physics*, 87(18):184115, 2013. 3

[5] 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. 2

[6] Peter W Battaglia, Jessica B Hamrick, and Joshua B Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the national academy of sciences*, 110(45):18327–18332, 2013. 1

[7] Daniel Bear, Chaofei Fan, Damian Mrowca, Yunzhu Li, Seth Alter, Aran Nayebi, Jeremy Schwartz, Li F Fei-Fei, Jiajun Wu, Josh Tenenbaum, et al. Learning physical graph representations from visual scenes. *Advances in Neural Information Processing Systems*, 33:6027–6039, 2020. 2

[8] Jörg Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *The Journal of chemical physics*, 134(7), 2011. 3

[9] Jörg Behler and Michele Parrinello. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Physical review letters*, 98(14):146401, 2007. 3

[10] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 3

[11] Venkatesh Botu and Rampi Ramprasad. Learning scheme to predict atomic forces and accelerate materials simulations. *Physical Review B*, 92(9):094306, 2015. 3

[12] Beatriz R Brum, Luiza Lober, Isolde Previdelli, and Francisco A Rodrigues. Discovering equations from data: symbolic regression in dynamical systems. *Journal of Physics: Complexity*, 7(1):012001, 2026. 2

[13] Marco Caccin, Zhenwei Li, James R Kermode, and Alessandro De Vita. A framework for machine-learning-augmented multiscale atomistic simulations on parallel supercomputers. *International Journal of Quantum Chemistry*, 115(16):1129–1139, 2015. 3

[14] Qinglong Cao, Ding Wang, Xirui Li, Yuntian Chen, Chao Ma, and Xiaokang Yang. Teaching video diffusion model with latent physical phenomenon knowledge. *arXiv preprint arXiv:2411.11343*, 2, 2024. 3

[15] Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016. 2, 6

[16] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018. 2

[17] Miles Cranmer, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho. Lagrangian neural networks. *arXiv preprint arXiv:2003.04630*, 2020. 2

[18] Miles Cranmer, Alvaro Sanchez Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering symbolic models from deep learning with inductive biases. *Advances in neural information processing systems*, 33:17429–17442, 2020. 2

[19] Yilun Du, Conor Durkan, Robin Strudel, Joshua B Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Sussman Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *International conference on machine learning*, pages 8489–8510. PMLR, 2023. 4

[20] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *Advances in neural information processing systems*, 32, 2019. 3, 4

[21] MW Finnis. Concepts for simulating and understanding materials at the atomic scale. *MRS bulletin*, 37(5):477–484, 2012. 3

[22] Meire Fortunato, Tobias Pfaff, Peter Wirnsberger, Alexander Pritzel, and Peter Battaglia. Multiscale meshgraphnets. *arXiv preprint arXiv:2210.00612*, 2022. 2

[23] Ruiqi Gao, Erik Nijkamp, Diederik P Kingma, Zhen Xu, Andrew M Dai, and Ying Nian Wu. Flow contrastive estimation of energy-based models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7518–7528, 2020. 3

[24] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. *Advances in neural information processing systems*, 32, 2019. 2

[25] Fredrik K Gustafsson, Martin Danelljan, Goutam Bhat, and Thomas B Schön. Energy-based models for deep probabilistic regression. In *European Conference on Computer Vision*, pages 325–343. Springer, 2020. 3

[26] Jiaqi Han, Wenbing Huang, Hengbo Ma, Jiachen Li, Josh Tenenbaum, and Chuang Gan. Learning physical dynamics with subequivariant graph neural networks. *Advances in Neural Information Processing Systems*, 35:26256–26268, 2022. 7, 16

[27] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022. 3

[28] Tianyu Huang, Haoze Zhang, Yihan Zeng, Zhilu Zhang, Hui Li, Wangmeng Zuo, and Rynson WH Lau. Dreamphysics: Learning physics-based 3d dynamics with video diffusion priors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 3733–3741, 2025. 3

[29] Zijie Huang, Yizhou Sun, and Wei Wang. Coupled graph ode for learning interacting system dynamics. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 705–715, 2021. 2

[30] Longbin Ji, Xiaoxiong Liu, Junyuan Shang, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. Videoar: Autoregressive video generation via next-frame & scale prediction. *arXiv preprint arXiv:2601.05966*, 2026. 3

[31] Bingyi Kang, Yang Yue, Rui Lu, Zhijie Lin, Yang Zhao, Kaixin Wang, Gao Huang, and Jiashi Feng. How far is video generation from world model: A physical law perspective. *arXiv preprint arXiv:2411.02385*, 2024. 3

[32] Patrick Kidger. On neural differential equations. *arXiv preprint arXiv:2202.02435*, 2022. 2

[33] Boris Knyazev, Graham W Taylor, and Mohamed Amer. Understanding attention and generalization in graph neural networks. *Advances in neural information processing systems*, 32, 2019. 2

[34] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006. 3

[35] Yann LeCun, Sumit Chopra, M Ranzato, and F-J Huang. Energy-based models in document recognition and computer vision. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 1, pages 337–341. IEEE, 2007. 3

[36] Ming Lei, Shufan Wu, and Christophe Baehr. Hcp-dcnet: A hierarchical causal primitive dynamic composition network for self-improving causal understanding. *arXiv preprint arXiv:2603.12305*, 2026. 1

[37] Shiqian Li, Ruihong Shen, Junfeng Ni, Chang Pan, Chi Zhang, and Yixin Zhu. Learning physics-grounded 4d dynamics with neural gaussian force fields. In *The Fourteenth International Conference on Learning Representations*, 2026. 1

[38] Yunzhu Li, Toru Lin, Kexin Yi, Daniel Bear, Daniel Yamins, Jiajun Wu, Joshua Tenenbaum, and Antonio Torralba. Visual grounding of learned physical models. In *International conference on machine learning*, pages 5927–5936. PMLR, 2020. 7, 16

[39] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018. 2, 7, 8, 16

[40] Fangfu Liu, Hanyang Wang, Shunyu Yao, Shengjun Zhang, Jie Zhou, and Yueqi Duan. Physics3d: Learning physical properties of 3d gaussians via video diffusion. *arXiv preprint arXiv:2406.04338*, 2024. 3

[41] Zhuoman Liu, Weicai Ye, Yan Luximon, Pengfei Wan, and Di Zhang. Unleashing the potential of multi-modal foundation models and video diffusion for 4d dynamic physical scene simulation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 11016–11025, 2025. 3

[42] Manish Mehta et al. Causalphysics: Unifying semantic reasoning, physical dynamics, and counterfactual simulation in world models. In *ICLR 2026 the 2nd Workshop on World Models: Understanding, Modelling and Scaling*, 2026. 1

[43] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. On the anatomy of mcmc-based maximum likelihood learning of energy-based models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5272–5280, 2020. 4

[44] Carlota Parés-Morlans, Michelle Yi, Claire Chen, Sarah A Wu, Rika Antonova, Tobias Gerstenberg, and Jeannette Bohg. Causal-pik: Causality-based physical reasoning with a physics-informed kernel. *arXiv preprint arXiv:2505.22861*, 2025. 1

[45] Evgeny V Podryabinkin and Alexander V Shapeev. Active learning of linearly parametrized interatomic potentials. *Computational Materials Science*, 140:171–180, 2017. 3

[46] 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*, pages 8459–8468. PMLR, 2020. 2, 6, 7, 8

[47] Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. In *International conference on machine learning*, pages 9323–9332. PMLR, 2021. 2, 7

[48] Kevin A Smith, Peter W Battaglia, and Edward Vul. Different physical intuitions exist between tasks, not domains. *Computational Brain & Behavior*, 1(2):101–118, 2018. 1

[49] Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021. 3

[50] Felix A Sosa, Samuel J Gershman, and Tomer D Ullman. Blending simulation and abstraction for physical reasoning. *Cognition*, 254:105995, 2025. 1

[51] Yee Whye Teh, Max Welling, Simon Osindero, and Geoffrey E Hinton. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, 4(Dec):1235–1260, 2003. 3

[52] Haoliang Wang, Khaled Jedoui, Rahul Venkatesh, Felix Jedidja Binder, Josh Tenenbaum, Judith E Fan, Daniel Yamins, and Kevin A Smith. Probabilistic simulation supports generalizable intuitive physics. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 46, 2024. 1

[53] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling autoregressive video models. *arXiv preprint arXiv:1906.02634*, 2019. 3

[54] Liming Wu, Zhichao Hou, Jirui Yuan, Yu Rong, and Wenbing Huang. Equivariant spatio-temporal attentive graph networks to simulate physical dynamics. *Advances in Neural Information Processing Systems*, 36:45360–45380, 2023. 2

[55] Gilad Yehudai, Ethan Fetaya, Eli Meirom, Gal Chechik, and Haggai Maron. From local structures to size generalization in graph neural networks. In *International Conference on Machine Learning*, pages 11975–11986. PMLR, 2021. 2

[56] Chaohao Yuan, Maoji Wen, Ercan Engin Kuruoglu, Yang Liu, Jia Li, Tingyang Xu, Deli Zhao, Hong Cheng, and Yu Rong. Non-stationary equivariant graph neural networks for physical dynamics simulation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 2

## Appendix Overview

The Appendix is organized as follows: Section A describes the experimental settings, including hyperparameters and training details for the elastic 2D, higher-order, anisotropic, 3D, and particle settings; Section B presents additional quantitative results and ablations, including elastic 2D simulation (Section B.1), higher-order potentials (Section B.2), anisotropic interactions (Section B.3), and particle simulation (Section B.4).

## A Experimental Settings

### A.1 2D Simulation

**Dataset Setup.** For all baselines, out of the 30,000 generated trajectories, we randomly select 3,000 for validation and use the remainder for training. For simulation, we use Euler steps and resolve elastic wall and inter-ball collisions using impulse-based collision response with positional correction to prevent overlap.

**Compositional Approach.** We compose a self-potential model, which models the ball's own motion independently of its neighbors, and a pairwise potential, which models the interaction with each neighboring ball. The total potential acting on each ball is the composition of the self-potential with one instance of the shared pairwise potential per neighbor.

**Experimental Setup.** For both self-potential and pairwise potential models, we train a 6-layer MLP with skip connections and each layer having layer normalization, 3 linear layers of dimensions 128, 256, 128, followed by a ReLU activation. The self-potential takes as input the absolute position and velocity of the current particle. The pairwise potential takes as input the absolute positions and velocities of each interacting pair. We train for 200 epochs with the AdamW optimizer, learning rate of $10^{-4}$, and batch size 2048. For scheduled noise, we used a linear schedule with $T$=30 timesteps. Training requires approximately 12 hours on a single NVIDIA A10 GPU (24GB). In Table 4 we report the average inference wall-clock time in seconds for a simulation timestep across different numbers of balls. For CPM, we observe no significant difference in wall-clock time when scaling from 4 balls (0.7309s on average) to 14 balls (0.7864s on average).

| Num. Balls | GNS | CPM |
|---|---|---|
| 4 | 0.0143 | 0.7309 |
| 6 | 0.0147 | 0.7770 |
| 8 | 0.0138 | 0.7687 |
| 10 | 0.0144 | 0.7867 |
| 12 | 0.0136 | 0.7752 |
| 14 | 0.0137 | 0.7864 |

Table 4: **2D Simulation Inference Time.** Average inference wall-clock time in seconds per simulation timestep across different numbers of balls.

**Baselines.** For both NPE and GNS, we train for 50 epochs using the AdamW optimizer, learning rate of $10^{-4}$ and batch size 2048. For NPE, we used MLPs of 6 layers and hidden dimension 128. For GNS, we used 6 message passing steps with a hidden dimension of 128 and 2-layer MLPs.

### A.2 2D Simulation with Higher-Order Interactions

**Dataset Setup.** For all baselines, out of the 30,000 generated trajectories, we randomly select 3,000 for validation and use the remainder for training. For simulation, we use Euler steps and resolve elastic wall and inter-ball collisions using impulse-based collision response with positional correction to prevent overlap. Beyond elastic collisions, balls interact through smoothed particle hydrodynamics forces: local density is computed with a cubic spline kernel (smoothing length $h$=$2r$, where $r$ is the ball radius), pressure follows the nonlinear equation of state $P = k(\rho^\gamma - \rho_0^\gamma)$ with $k$=5, $\gamma$=4, $\rho_0$=1, and each ball is accelerated by the resulting pressure gradient together with a viscosity term ($\mu$=0.3). Because density aggregates over all neighbors within $2h$, these forces are non-additive and cannot be decomposed into independent pairwise interactions.

**Compositional Approach.** To model self-potential and pairwise interactions, we follow the approach introduced in Appendix A.1. However, pairwise potentials alone are not sufficient to capture the higher-order interactions induced by the SPH forces. We therefore extend the system in (1) with an additional potential defined over triplets of objects, which allows the model to represent interactions beyond the pairwise level. Formally, let $\mathcal{Z}_i$ denote the set of objects interacting with object $i$

(excluding $i$ itself). The total potential acting on object $i$ is obtained by additively composing self, pairwise, and higher-order contributions:

$$E_i(y) = \underbrace{\sum_{j \in \mathcal{Z}_i} E_\theta^{(1)}(y, x_i, x_j)}_{\text{Pairwise interaction}} + \underbrace{\sum_{\substack{j,k \in \mathcal{Z}_i \\ j<k}} E_\theta^{(2)}(y, x_i, x_j, x_k)}_{\text{Higher-order interaction}}, \tag{7}$$

where $E_\theta^{(2)}(y, x_i, x_j, x_k)$ is the higher-order potential acting on object $i$ due to its joint interaction with objects $j$ and $k$. For simplicity, we omit the self-potential term in (7).

**Experimental Setup.** For self-potential, pairwise, and higher-order models, we train a 4-layer MLP with skip connections and each layer having layer normalization, 3 linear layers of dimensions 128, 128, 128, followed by a ReLU activation. The self-potential takes as input the absolute position and velocity of the current particle. The pairwise and higher-order models take as input the absolute positions and velocities of each ball. We train for 200 epochs with the AdamW optimizer, learning rate of $10^{-4}$, and batch size 2048. For scheduled noise, we used a linear schedule with $T$=30 timesteps. Training requires approximately 24 hours on a single NVIDIA A10 GPU (24GB).

**Baselines.** We use the same baselines and settings as introduced in Appendix A.1, trained on the higher-order trajectories.

### A.3 2D Simulation with Anisotropic Interactions

**Dataset Setup.** For all baselines, out of the 30,000 generated trajectories, we randomly select 3,000 for validation and use the remainder for training. For simulation, we use Euler steps and resolve elastic wall and inter-ball collisions using impulse-based collision response with positional correction to prevent overlap. Balls are initialized with zero angular velocity. Beyond elastic collisions, each ball carries a magnetic dipole moment $\mathbf{m}_i = |m|(\cos\theta_i, \sin\theta_i)$ with $|m| = 1$, orientation $\theta_i$, and angular velocity $\omega_i$. The field at ball $i$ is the sum of a uniform external field $\mathbf{B}_{\text{ext}} = (0.5, 0)$ and the point-dipole fields of all other balls, $\mathbf{B}_i = \mathbf{B}_{\text{ext}} + \sum_{j \neq i} \kappa \left(3(\mathbf{m}_j \cdot \hat{\mathbf{r}}_{ij})\hat{\mathbf{r}}_{ij} - \mathbf{m}_j\right)/\|\mathbf{r}_{ij}\|^3$, with coupling strength $\kappa = 10^8$. Each ball experiences the torque $\mathcal{T}_i = \mathbf{m}_i \times \mathbf{B}_i$, giving the angular acceleration $\alpha_i = \mathcal{T}_i/I$ with moment of inertia $I = 50$, together with the translational dipole–dipole force $\mathbf{F}_i$ from the other dipoles, which decays as $1/r^4$. The angular velocity is additionally subject to weak linear damping with coefficient 0.1. For all balls we assume the same mass $M = 1$. The resulting interactions are anisotropic and depend on the orientations of both balls. Collisions are frictionless and do not alter $\theta$ or $\omega$.

**Compositional Approach.** As in the isotropic setting introduced in Appendix A.1, we model interactions using a self-potential and a pairwise potential. In this case, the simulation is also anisotropic, meaning that interactions also depend on orientation and direction of the balls. To model this, we extend the system in (1) with orientation and angular velocity terms for each interacting ball. Formally, the total potential acting on object $i$ is obtained by:

$$E_i(y) = \sum_{j \in \mathcal{Z}_i} E_\theta(y, x_i, x_j, \theta_i, \theta_j, \omega_i, \omega_j) \tag{8}$$

where $\theta_i$ and $\theta_j$ are the orientations of balls $i$ and $j$, and $\omega_i$ and $\omega_j$ are the angular velocities. For simplicity, we omit the self-potential term in (8).

**Experimental Setup.** For both self-potential and pairwise models, we train a 4-layer MLP with skip connections and each layer having layer normalization, 3 linear layers of dimensions 128, 128, 128, followed by a ReLU activation. The self-potential takes as input the absolute position and velocity of the current particle. The pairwise model takes as input the absolute positions, velocities, orientations, and angular velocities of each ball. The system is trained to recover velocity, change in orientation, and angular velocity. We train for 200 epochs with the AdamW optimizer, learning rate of $10^{-4}$, and batch size 2048. For scheduled noise, we used a linear schedule with $T$=30 timesteps. Training requires approximately 16 hours on a single NVIDIA A10 GPU (24GB).

**Baselines.** We use the same baselines and settings as introduced in Appendix A.1, trained on the anisotropic trajectories. The baselines are extended to also receive the orientation and angular velocity as input.

### A.4 3D Simulation

**Dataset Setup.** We use the data and experimental settings from [38].

**Compositional Approach.** We compose two potential models: one that captures gravity interaction exclusively, and an E(3)-equivariant, object-aware potential model adapted from [26] to model interactions between cubes. This decomposition allows cube interactions to be modeled without relaxing the equivariance constraints to accommodate gravitational forces.

**Experimental Setup.** For the gravity potential, we train a 6-layer MLP with skip connections, where each layer consists of layer normalization followed by three linear layers of hidden dimension 128 and a ReLU activation. For the interaction potential, we train an E(3)-equivariant, object-aware message-passing network with 4 message-passing steps, hidden dimension 200, and SiLU activations. The gravity potential takes a particle's velocity as input, while the interaction potential takes both positions and velocities. For training set sizes of 200, 500, 1,000, and 5,000, we train for 400, 400, 150, and 75 epochs, respectively. We train using the AdamW optimizer, learning rate of $10^{-4}$, and batch size 16. For scheduled noise, we used a linear schedule with $T$=30 timesteps. Training takes approximately 2 days in the low-data regime and up to 4 days for the largest dataset, on a single NVIDIA A10 GPU (24GB).

**Baselines.** For SGNN hyperparameters and all other baselines, we refer the reader to [38].

### A.5 Particle Simulation

**Dataset Setup.** We follow the setting of [39] to generate 1,000 BoxBath training samples, and use a random 10% split for validation.

**Compositional Approach.** For each particle we compose a self-potential to capture each particle's individual motion, and pairwise potentials to model interactions with neighboring particles. If a particle interacts with the rigid body, a separate potential is composed. Similarly, wall interactions are modeled through an additional potential when particles are close to boundaries. To capture higher-order effects such as density, we further include a long-range potential.

**Experimental Setup.** All local interaction potentials are parametrized by a 6-layer MLP with skip connections, where each layer consists of layer normalization followed by three linear layers of hidden dimension 128, 256, 128 and a ReLU activation. The long-range potential is modeled using a 10-layer message-passing network with hidden dimension 128. For pairwise potentials, the inputs are relative positions and velocities. All models receive as input the type of each particle (particle or rigid body). The wall interaction model takes the relevant axis-aligned position and velocity as input, while all other models use absolute positions and velocities. During training, the velocities of the rigid particles are predicted separately. However, during inference, we follow the same approach as [39] and the predicted velocities are aggregated into rigid body transformations using the Kabsch algorithm. For all models, instead of predicting a single next step, we predict the next 5 simulation steps. For scheduled noise, we used a linear schedule with $T$=30 timesteps. Training proceeds in two phases. First local potentials are trained using the AdamW optimizer, learning rate of $3 \cdot 10^{-4}$, and batch size 4 for 200 epochs. In the second phase, we freeze these models, introduce the long-range potential, and fine-tune the full system using the AdamW optimizer, learning rate of $3 \cdot 10^{-4}$, and batch size 4 for 10 epochs. Training takes approximately 3 days, with an additional 1 day for finetuning, on a single NVIDIA A10 GPU (24GB). In Table 5, for CPM, wall-clock time increases from 0.8846 s on average with 960 particles to 1.3489 s with 2160 particles. This is expected, as the number of particles more than doubles and the system grows in scale. We also observe an overhead relative to GNS, which is expected given the additional optimization steps performed at inference time by CPM.

| Num. Particles | GNS | CPM |
|---|---|---|
| 960 | 0.0129 | 0.8846 |
| 1500 | 0.0121 | 0.9625 |
| 2160 | 0.0141 | 1.3489 |

Table 5: **Particle Simulation Inference Time.** Average inference wall-clock time in seconds per simulation timestep across different numbers of particles.

**Baselines.** For both DPI and GNS we use the same hyperparameters reported in the original works for the BoxBath scenario.

| Num. Balls | GNS | CPM |
|---|---|---|
| 4 | $0.3971 \pm 0.6177$ | $0.1648 \pm 0.3344$ |
| 6 | $0.4410 \pm 0.4665$ | $0.3117 \pm 0.4374$ |
| 8 | $0.5826 \pm 0.5152$ | $0.4413 \pm 0.5205$ |
| 10 | $0.6489 \pm 0.4221$ | $0.5320 \pm 0.4508$ |
| 12 | $0.8061 \pm 0.4430$ | $0.7537 \pm 0.5400$ |
| 14 | $1.0515 \pm 0.4533$ | $0.8087 \pm 0.4558$ |

Table 6: **2D Simulation Performance vs. Number of Balls.** We report mean $\pm$ std position MSE averaged across simulation trajectories for numbers of balls ranging from 4 (in-distribution) to 6–14 (out-of-distribution). We compare with GNS baseline trained on the same dataset. CPM consistently achieves lower MSE across all settings. CPM results were sampled using $S$=10.

# B Additional Results

## B.1 2D Simulation

**Quantitative Results.** Table 6 reports the performance of CPM and the GNS baseline on the 2D simulation task as the number of balls increases. CPM consistently achieves a lower MSE than GNS across all settings. For the largest number of balls, CPM attains an average error of $0.8087$, compared to $1.0515$ for GNS.

**Ablation Studies.** Framing physics simulation as training an energy-based model (EBM) and performing energy minimization allows sampled solutions to be refined directly through gradient descent. In this setting, the improvement of CPM could stem from this refinement ability rather than from the compositional approach. To show that composition is a key factor in the success of CPM, we conduct an ablation study on the composition of potentials. To this end, we take the baseline model and train it as an EBM, i.e., we train it to output a scalar energy whose gradient with respect to the input points in the direction that removes noise from the input. Formally, we train the model with the following loss:

$$\mathcal{L}_{\text{MSE}}(\theta) = \mathbb{E}_{y_i, \epsilon \sim \mathcal{N}(0,I)} \left[ \left\| \epsilon - \sigma_t \nabla_{y_i} E_\theta(y_i^*, x_i, t) \right\|^2 \right], \tag{9}$$

where $y_i^* = \sqrt{1-\sigma_t^2}\, y_i + \sigma_t\, \epsilon$ is a noise-corrupted version of the ground-truth sample $y_i$ with $\epsilon \sim \mathcal{N}(0, I)$, $\sigma_t$ is the fixed noise schedule at time $t$, and $E_\theta$ is an energy model with a GNS architecture. This objective allows us to use the sampling procedure in Algorithm 1 without the model being compositional, as CPM is.

Table 7 reports the results of this ablation. We report the MSE between the predicted and ground-truth positions and velocities, averaged over 100 simulations with 60-timestep rollouts, in both the in-distribution and out-of-distribution settings. We compare CPM against the GNS baseline trained as an EBM (GNS-EBM). For GNS-EBM, we additionally vary the number of gradient descent steps per timestep and report results for $1$, $5$, and $10$ steps. As with CPM, more gradient descent steps during sampling reduce the prediction error of GNS-EBM: the average position MSE decreases from 1.7931 with 1 step to 0.9552 with 5 steps, and the velocity MSE decreases from 4.3303 to 2.4977. However, despite this improvement, GNS-EBM does not reach the performance of CPM. This is expected: GNS-EBM is trained to predict ground-truth samples, but nothing in its construction guarantees generalization to a larger number of interacting objects, whereas CPM is compositional by construction. We therefore conclude that the compositional structure of CPM is an essential contributor to its performance gains.

## B.2 2D Simulation with Higher-Order Interactions

**Quantitative Results.** In Table 8, we report the MSE between the predicted and ground-truth position and velocity, averaged over 100 simulations with 60-timestep rollouts each, for both in-distribution and out-of-distribution settings. In-distribution, CPM achieves the lowest position MSE of 0.1093 compared to 0.3202 for NPE and 0.2320 for GNS. Similarly, in out-of-distribution, the lowest position error is obtained by CPM with 1.4438 compared to 1.7376 for GNS and 1.8029 for NPE. In Table 9, we provide a detailed breakdown of how the error of CPM and GNS evolves with an increasing number of balls in the simulation, ranging from 4 (in-distribution) to 6–14 (out-of-distribution). CPM

| Model | Composition | Similar Distribution | | Larger Distribution | |
|---|---|---|---|---|---|
| | | MSE Position | MSE Velocity | MSE Position | MSE Velocity |
| GNS-EBM ($S$=1) | ✗ | 1.7931 ± 1.3447 | 4.3303 ± 2.7965 | 2.8754 ± 0.9150 | 8.0056 ± 1.9922 |
| GNS-EBM ($S$=5) | ✗ | 0.9552 ± 1.1461 | 2.4977 ± 2.3608 | 2.3624 ± 1.0402 | 5.1660 ± 1.9131 |
| GNS-EBM ($S$=10) | ✗ | 1.0028 ± 1.2190 | 2.5233 ± 2.2798 | 2.3995 ± 1.0388 | 5.1341 ± 1.7329 |
| CPM ($S$=10) | ✓ | **0.1648 ± 0.3344** | **0.7537 ± 1.1207** | **0.6245 ± 0.3362** | **2.1156 ± 0.9692** |

Table 7: **2D Simulation Composition Ablation.** Ablation of the composition of potentials on the 2D simulation task. We compare CPM against a non-compositional EBM with a GNS architecture (GNS-EBM). We report the MSE between predicted and ground-truth positions and velocities, averaged over 100 simulations with 60-timestep rollouts, in both the in-distribution and out-of-distribution settings. The compositional approach outperforms the non-compositional EBM regardless of the number of gradient descent steps per timestep.

| Model | Similar Distribution | | Larger Distribution | |
|---|---|---|---|---|
| | MSE Position | MSE Velocity | MSE Position | MSE Velocity |
| NPE | 0.3202 ± 0.4207 | 1.1741 ± 1.0442 | 1.8029 ± 0.9474 | 11.1952 ± 11.5877 |
| GNS | 0.2320 ± 0.3336 | 0.8044 ± 0.7045 | 1.7376 ± 1.0196 | **10.7748 ± 11.6633** |
| CPM (Ours) | **0.1093 ± 0.0937** | **0.7898 ± 0.7437** | **1.4438 ± 1.0348** | 10.8065 ± 11.2284 |

Table 8: **2D Higher-Order Simulation Evaluation.** We compare our method against widely used neural physics simulators from the literature. We report mean ± std position and velocity MSE for both in-distribution with 4 balls and out-of-distribution with 12–14 balls. Our method achieves a lower position MSE in both settings. CPM results were sampled using $S$=5.

| Num. Balls | GNS | CPM |
|---|---|---|
| 4 | 0.2320 ± 0.3336 | 0.1093 ± 0.0937 |
| 6 | 0.3929 ± 0.4938 | 0.2601 ± 0.3316 |
| 8 | 0.7105 ± 0.6753 | 0.5621 ± 0.6819 |
| 10 | 0.8653 ± 0.7091 | 0.7466 ± 0.6445 |
| 12 | 1.1606 ± 0.6380 | 0.9977 ± 0.7790 |
| 14 | 1.7564 ± 0.7982 | 1.4761 ± 0.7792 |

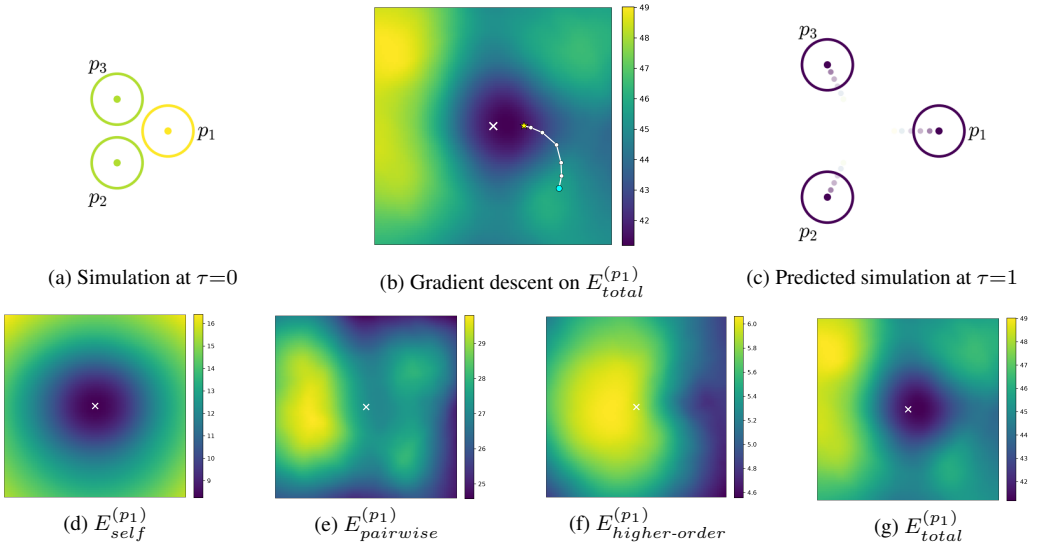
Table 9: **2D Higher-Order Simulation Performance vs. Number of Balls.** We report mean ± std position MSE averaged across simulation trajectories for numbers of balls ranging from 4 (in-distribution) to 6–14 (out-of-distribution). We compare with GNS baseline trained on the same dataset. CPM consistently achieves lower MSE across all settings. CPM results were sampled using $S$=5.

consistently achieves lower MSE across all settings. In the setting with the largest number of balls, CPM achieves an average MSE of 1.4761 compared to 1.7564 for GNS.

**Qualitative Results.** Figure 9 show the learned potentials acting on a query particle $p_1$ in a three-ball higher-order interaction setup. Starting from a zero-velocity configuration at $\tau$=0 (a), CPM performs gradient descent on the composed potential of particle $p_1$, $E^{(p_1)}_{total}$ (b) to obtain the predicted next state (c), in which $p_1$ moves to the right. Panels (d)–(f) show the learned self, pairwise, and higher-order contributions, and (g) their sum $E^{(p_1)}_{total}$, whose minimum coincides with the predicted next state.

### B.3 2D Simulation with Anisotropic Interactions

**Quantitative Results.** Table 10 reports the mean squared error (MSE) between predicted and ground-truth positions, velocities, orientations, and angular velocities. Results are averaged over 100 simulations with 60-timestep rollouts, in both in-distribution and out-of-distribution settings. CPM has the lowest position MSE in both settings: 0.2730 in-distribution (vs. 0.3949 for GNS and 0.4619 for NPE) and 0.7633 out-of-distribution (vs. 1.1424 for GNS and 1.4786 for NPE). For orientation, GNS performs slightly better in-distribution (3.0168 vs. 3.1505 for CPM), whereas CPM performs better out-of-distribution (5.1208 vs. 5.4670 for GNS). For angular velocity, CPM again gives the lowest MSE in both settings: 1.9371 in-distribution (vs. 2.2466 for GNS and 2.5269 for NPE) and 9.4770 out-of-distribution (vs. 10.5536 for GNS and 12.1953 for NPE).




(a) Simulation at $\tau$=0 (b) Gradient descent on $E^{(p_1)}_{total}$ (c) Predicted simulation at $\tau$=1

(d) $E^{(p_1)}_{self}$ (e) $E^{(p_1)}_{pairwise}$ (f) $E^{(p_1)}_{higher\text{-}order}$ (g) $E^{(p_1)}_{total}$

Figure 9: **Higher-Order Potential Maps Visualization.** We plot the learned energies acting on particle $p_1$. (a) Initial configuration at $\tau$=0, where yellow indicates higher pressure. Balls have zero initial velocities. (b) Gradient descent on the composed potential of particle $p_1$, $E^{(p_1)}_{total}$. It predicts that $p_1$ moves from current position (marked with a cross) to the right. (c) Predicted next state, obtained by the optimization in (b), which predicts that $p_1$ moves to the right. (d)–(f) Individual potentials $E^{(p_1)}_{self}$, $E^{(p_1)}_{pairwise}$, and $E^{(p_1)}_{higher\text{-}order}$. (g) Composed potential $E^{(p_1)}_{total} = E^{(p_1)}_{self} + E^{(p_1)}_{pairwise} + E^{(p_1)}_{higher\text{-}order}$, whose minimum corresponds to the predicted next state of particle $p_1$.

| | Similar Distribution | | | | Larger Distribution | | | |
|---|---|---|---|---|---|---|---|---|
| **Model** | **MSE Position** | **MSE Velocity** | **MSE Orientation** | **MSE Ang. Velocity** | **MSE Position** | **MSE Velocity** | **MSE Orientation** | **MSE Ang. Velocity** |
| NPE | 0.4619 ± 0.5688 | 1.1360 ± 0.9503 | 3.2354 ± 1.8057 | 2.5269 ± 3.2791 | 1.4786 ± 0.6378 | 3.5804 ± 1.1248 | 5.5284 ± 0.7511 | 12.1953 ± 3.0334 |
| GNS | 0.3949 ± 0.4395 | 1.2438 ± 1.2023 | **3.0168 ± 1.8775** | 2.2466 ± 2.9280 | 1.1424 ± 0.4279 | **2.9305 ± 0.9024** | 5.4670 ± 0.8887 | 10.5536 ± 2.2653 |
| CPM (Ours) | **0.2730 ± 0.4540** | **1.0396 ± 0.9058** | 3.1505 ± 1.7247 | **1.9371 ± 2.3057** | **0.7633 ± 0.3544** | 3.1131 ± 1.0570 | **5.1208 ± 0.8903** | **9.4770 ± 2.1058** |

Table 10: **2D Anisotropic Simulation Evaluation.** We compare our method against widely used neural physics simulators from the literature. We report mean ± std position, velocity, orientation, and angular velocity MSE for both in-distribution with 4 balls and out-of-distribution with 12–14 balls. CPM achieves lower position and angular velocity MSE in both settings. Results from CPM were sampled using $S$=5.

| | **Similar Distribution** | **Larger Distribution** | |
|---|---|---|---|
| **Steps** | 960 Particles | 1500 Particles | 2160 Particles |
| $S = 1$ | 0.0113 ± 0.0087 | 0.0144 ± 0.0107 | 0.0134 ± 0.0117 |
| $S = 5$ | 0.0117 ± 0.0109 | **0.0100 ± 0.0088** | **0.0103 ± 0.0078** |
| $S = 10$ | **0.0098 ± 0.0076** | 0.0149 ± 0.0107 | 0.0148 ± 0.0130 |

Table 11: **Gradient Descent Steps Ablation.** Ablation study on the number of gradient descent steps performed per timestep in the BoxBath setting. We report mean ± std position MSE averaged over 150 simulation timesteps for different particle counts. Increasing the number of steps generally improves performance. In the in-distribution settings, the best results are achieved at $S$=10. In the out-of-distribution settings, performance converges at $S$=5.

### B.4 Particle Simulation

**Ablation Studies.** In Table 11 we evaluate the effect of the number of gradient descent steps performed per timestep. Generally, increasing the number of gradient descent steps helps to refine the accuracy of the predictions. In-distribution, increasing $S$ consistently improves performance, reducing position MSE from 0.0113 at $S$=1 to 0.0098 at $S$=10. In the larger distribution case, the best results are obtained at $S$=5, with diminishing returns beyond that point.

## NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

**The checklist answers are an integral part of your paper submission.** They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Section 1.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section 5.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: Not Applicable.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section 4 and Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: See Section 4 and Appendix A.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section 4 and Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: See Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: No direct societal impact can be derived as an outcome from our work. The work proposes core machine learning contributions for improving generalization in physical simulation tasks and does not involve sensitive data or high-risk applications.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No high risk exists in the misuse of the models. Released data and models involve only physical simulation tasks.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: See Section 4 and Appendix A.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: See Section 4 and Appendix A.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: No crowdsourcing or research with human subjects experiments were performed in this work.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: No crowdsourcing or research with human subjects experiments were performed in this work.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Core method development does not involve LLMs.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.