NEURAL-PD: A PERIDYNAMICS-INSPIRED NEURAL SIMULATOR FOR 3D FRACTURE DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting fracture in deformable objects remains challenging for neural dynamics models because cracks introduce localized, irreversible discontinuities. Peridynamics (PD) is a well-established continuum physical simulation framework whose interaction operators share the same structure as deep networks on point clouds. Inspired by Peridynamics, we present **Neural-PD**, a particle-based neural simulator that uses the interaction structure of PD as a physical inductive bias for learning deformation, contact, and fracture. Different from other particle-based neural dynamics models, Neural-PD follows Peridynamics in combining material forces and contact forces, while utilizing each type of force on a different neighborhood: material forces over the reference configuration and contact forces over the current one. An attention-based formulation inspired by PD explicitly models the strength of each bond, which helps predicting fracture and damage. Unlike traditional peridynamics, Neural-PD adopts a U-Net architecture that propagates force information across spatial scales, supporting long-range dynamics prediction at large time steps. Due to the lack of benchmark for evaluating neural dynamics models in damage and fracture scenarios, we introduce a new benchmark spanning visco-elastic deformations and multiple crack and damage scenarios. On these benchmarks, Neural-PD achieves $38\% - 93\%$ lower error than other neural dynamics models, showcasing the significant potential of incorporating particle-based physics ideas such as PD in neural dynamics modeling.

1 INTRODUCTION

Learned dynamics models that predict the evolution of complex systems directly from particle, mesh, or point-cloud states have received substantial interest in recent years because they can make predictions much faster than traditional simulation approaches. Graph-based models such as GNS and DPI-Net have demonstrated accurate, generalizable rollouts for fluids, deformable materials, and interacting rigid bodies (Sanchez-Gonzalez et al., 2020; Li et al., 2019), while hierarchical point-cloud networks and object-centric Transformers have improved performance on long-range interaction modeling and efficient rigid-body simulation (Kim & Fuxin, 2024; Kim et al., 2026; Dou et al., 2026; Han et al., 2022; Zhang et al., 2025b; Zhobro et al., 2026).

However, most of the algorithms have been restricted to cases where particles or meshes move continuously. In practice, fracture of objects is common and important: a knife splits a carrot, a spoon cracks the shell of an egg, packaging ripped open or a mug cracks when lands too hard on the ground. Fracture is a routine goal of manipulation — much of cooking, harvesting, disassembly, and surgery consists of deliberately breaking material in a controlled way. Meanwhile it is also important for understanding safe behaviors. Predicting when an object will break, and into what pieces, therefore is an important goal for neural dynamic models, especially for robotic use cases.

Physics-based numerical methods remain the standard for fracture simulation, however they have significant difficulties on this task as well. Standard finite element methods (FEM) require mesh connectivity, hence evolving cracks call for specialized formulations, such as enriched approximation spaces or auxiliary phase fields (Moës et al., 1999; Miehe et al., 2010). Material Point Method (MPM) transfers particle states through a continuous grid velocity field and therefore does not natively represent the displacement discontinuity across a crack. There is some recent work addressing

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

this issue (Sun et al., 2021; Wolper et al., 2019; Wang et al., 2026b; Huang et al., 2026), but their dynamics still come from engineered numerical solvers.

Peridynamics (PD)(Silling, 2000; Silling & Askari, 2005; Silling et al., 2007) provides a framework that represents internal forces through interactions between material points within a finite horizon. Its integral formulation allows fractures to be represented through the failure of material interactions, without needing explicit crack tracking or remeshing. Interestingly, the integral formulation of PD can be discretized into estimating functions in a point-based neighborhood, which leads to the natural idea of utilizing its physical structure to inform the model structure of a learned neural dynamics model based on point cloud deep learning.

In this paper, we introduce **Neural-PD**, a learned dynamics model inspired by Peridynamics. Neural-PD represents deformable bodies as 3D point clouds, providing a natural interface for irregular geometries and mesh-free physical states. The model architecture organizes learned interactions using neighborhoods defined in the reference configuration, following Peridynamics, and separate neighborhoods defined in the current configuration for transient contact between bodies. This separation distinguishes interactions within a material from transient interactions between different bodies. At each time step, the network predicts force-density-like quantities that are used to advance the physical state, as well as the strength of all the bonds which feed into a modified attention model that explicitly considers bond damage.

Neural-PD also differs from PD and traditional simulation in that it does not attempt to subdivide time into infinitesimal steps. It directly learns nonlinear functions that predicts the motion after a significant time step. This requires a significant amount of non-local modeling as forces need to be propagated very far from its initial location. Therefore, we utilize a U-Net formulation which allowed non-local propagation of forces across the entire object in as little as a single timestep. This allows us to simulate a dynamical scene much more quickly compared to traditional simulators.

Since there is no benchmark to quantitatively evaluate neural dynamics models on fracture and damage, we propose a new benchmark dataset to test neural dynamics models on fracture scenarios. This benchmark contains scenarios with different types of cracks and damages, as well as viscoelastic deformations without cracking/damaging. We compare neural-PD with other point-cloud, graph-based, and Transformer-based neural dynamic models. In particular, we study long-horizon rollout and generalization across different geometries, loading conditions, and crack configurations, including scenarios involving multiple cracks. Experiment results show that Neural-PD obtained $38\% - 93\%$ lower errors than other neural dynamics models, showcasing the significant potential of incorporating particle-based dynamics ideas from physics into neural dynamics modeling.

In summary, our key contributions are:

- We introduce **Neural-PD**, a learned 3D point-based dynamics model that has its architecture organized with inspirations by peridynamics, with resting-frame neighborhoods for material interactions and current-frame neighborhoods for transient interactions.
- We propose a modified attention framework that incorporates explicit damage modeling of bonds between points.
- We construct a **benchmark dataset** for learned 3D fracture dynamics spanning different geometries, loading conditions, and single- and multiple-crack scenarios.

2 Related Work

Machine learning for peridynamics. Machine learning has been combined with PD mainly for constitutive modeling, including learned influence functions and nonlocal interaction kernels (Xu et al., 2021; You et al., 2024; Yin et al., 2024), for parameter calibration (Lin et al., 2025), and for accelerating conventional PD solvers (Nguyen et al., 2021; Ramesh Babu & Gopalakrishanan, 2024; Ramesh Babu & Gopalakrishnan, 2024; Nie et al., 2025). Neural non-ordinary state-based models relax the requirement that bond forces align with the deformed bond and predict vector force states, including for rubber-like materials and for hyperelastic membranes undergoing large deformation and fracture (Chen et al., 2024; Yang et al., 2024). In computer graphics, PD has been used to simulate complex dynamic scenes (Xu et al., 2018; Guo et al., 2025; He et al., 2018; Lu et al., 2024; Wang et al., 2025; Qin et al.), but with hand-designed interaction models. Overall, prior work either

improves individual components of the PD pipeline, such as the constitutive law, its parameters, or the solver, or retains hand-designed interaction models. In contrast, Neural-PD learns the coupled material and contact dynamics of multi-object 3D scenes with fracture directly from trajectories.

Neural operators and PNO. Neural operators learn mappings between function spaces through learnable kernel integrals (Kovachki et al., 2023). The Peridynamic Neural Operator (PNO) brings this idea to peridynamics by learning the force state inside the PD neighborhood integral (Jafarzadeh et al., 2024), and its heterogeneous extension adds spatially varying material structure (Jafarzadeh et al., 2025). Both learn an ordinary state-based constitutive law from displacement and loading measurements of individual specimens, without contact between bodies or bond failure. Neural-PD instead learns the dynamics of multi-body scenes from trajectories. Besides, Neural-PD embeds its interaction blocks in a U-Net that propagates force information across spatial scales, supporting dynamics prediction at large time steps, whereas PNO cannot deal with such long-range interactions.

Learned physical simulators. Learned simulators based on particle, mesh, and graph representations have been widely explored (Li et al., 2019; Sanchez-Gonzalez et al., 2020; Pfaff et al., 2021; Han et al., 2022; Allen et al., 2023b;a; Ummenhofer et al., 2020; Prantl et al., 2022; Kim & Fuxin, 2024; Wang et al., 2026a). Other designs use hybrid particle–grid representations (Zhang et al., 2025b), object-centric Transformers for rigid-body contact (Dou et al., 2026), or Point Transformer V3 (Zhobro et al., 2026; Wu et al., 2024), on which we build the Dynamics-PTv3 baseline. Several methods separate internal and contact interactions, e.g., through mesh- and world-space edges (Pfaff et al., 2021) or within- and cross-object convolutions (Kim & Fuxin, 2024). Their neighborhoods, however, are either rebuilt from current positions or fixed by a mesh or object labels, so they cannot record lost material connectivity: a radius graph cannot distinguish a closed crack from intact material, and a fixed partition cannot form new fracture surfaces. Outside the machine learning literature, computational mechanics has explored learning-based fracture models for prescribed configurations, including crack propagation and phase-field fracture dynamics (Perera et al., 2022; Perera & Agrawal, 2024; Hamdi & Lejeune, 2026); Neural-PD instead targets 3D multi-object impact and drop scenes, where contact drives deformation and fracture and the model must jointly capture evolving inter-object contact and irreversible loss of intra-object material connectivity.

3 NEURAL-PD

We first briefly review Peridynamics and then explain how its physical structure helps to determine the model components and then describe the learned operators used to predict the dynamics.

3.1 PERIDYNAMICS

Classical numerical methods such as FEM and MPM model deformation through continuum formulations in which internal forces depend on spatial derivatives of stress and displacement fields (Moës et al., 1999; Sulsky et al., 1994). This derivative-based formulation becomes difficult to apply directly in the presence of discontinuities such as cracks and fractures. Peridynamics (PD) (Silling & Askari, 2005; Silling et al., 2007), on the other hand, avoids spatial derivatives in its governing equation. Let us denote each material point by its reference position as $\mathbf{x}$ and its current position at time t as

$$\mathbf{y}(\mathbf{x}, t) = \mathbf{x} + \mathbf{u}(\mathbf{x}, t),$$

where $\mathbf{u}(\mathbf{x}, t)$ is its displacement. The reference position refers to an undisturbed, resting position of the material point in which it is not stretched by outside forces. The equation of motion is

$$\rho(\mathbf{x})\ddot{\mathbf{u}}(\mathbf{x}, t) = \int_{\mathcal{H}_\delta(\mathbf{x})} \mathbf{f}_{\text{mat}}(\mathbf{q}, \mathbf{x}, t)\, d\mathbf{q} + \mathbf{b}(\mathbf{x}, t), \tag{1}$$

where $\mathcal{H}_\delta(\mathbf{x}) = \{\mathbf{q} : \|\mathbf{q} - \mathbf{x}\| \leq \delta\}$ denotes a δ-ball neighborhood of $\mathbf{x}$, $\mathbf{f}_{\text{mat}}(\mathbf{q}, \mathbf{x}, t)$ is the force density exerted by a neighboring point $\mathbf{q}$ on $\mathbf{x}$, and $\mathbf{b}(\mathbf{x}, t)$ refers to external body forces such as gravity. Thus, internal forces are obtained by integrating interactions over a local neighborhood rather than through spatial derivatives. The interaction between $\mathbf{x}$ and $\mathbf{q} \in \mathcal{H}_\delta(\mathbf{x})$ is referred to as a *bond*. Fracture can be modeled by breaking bonds, i.e., by setting the corresponding force $\mathbf{f}_{\text{mat}}(\mathbf{q}, \mathbf{x}, t)$ to zero, allowing cracks to emerge without explicitly representing the crack surface.

For scenes containing multiple objects, PD can additionally incorporate short-range contact interactions between distinct bodies (Littlewood et al., 2023). The dynamics can then be written as

$$\rho(\mathbf{x})\ddot{\mathbf{u}}(\mathbf{x},t) = \int_{\mathcal{H}_\delta(\mathbf{x})} \mathbf{f}_{\text{mat}}(\mathbf{q},\mathbf{x},t)\, d\mathbf{q} + \int_{\mathcal{H}_c(\mathbf{y}(\mathbf{x},t))} \mathbf{f}_{\text{cnt}}(\mathbf{q},\mathbf{x},t)\, d\mathbf{q} + \mathbf{b}(\mathbf{x},t), \tag{2}$$

where $\mathbf{f}_{\text{mat}}$ governs interactions between materially connected points, while $\mathbf{f}_{\text{cnt}}$ captures contact between distinct bodies. The material neighborhood $\mathcal{H}_\delta(\mathbf{x})$ is defined in the reference configuration, whereas the contact neighborhood $\mathcal{H}_c(\mathbf{y})$ depends on the current geometry.

Peridynamics is usually solved numerically with spatial discretization: discretizing the integral into a summation on material points sampled from the neighborhoods.

$$\rho(\mathbf{x})\ddot{\mathbf{u}}(\mathbf{x},t) = \sum_{\mathbf{q}\in\mathcal{H}_\delta(\mathbf{x})} \mathbf{f}_{\text{mat}}(\mathbf{q},\mathbf{x},t)V_{\mathbf{q}} + \sum_{\mathbf{q}\in\mathcal{H}_c(\mathbf{y})} \mathbf{f}_{\text{cnt}}(\mathbf{q},\mathbf{x},t)V_{\mathbf{q}} + \mathbf{b}(\mathbf{x},t), \tag{3}$$

where $V_{\mathbf{q}}$ is the volume each material point $\mathbf{q}$ represents. We note that this local neighborhood-based summation approach with a nonlinear function is strongly connected with point cloud deep learning (Wu et al., 2019; 2023; 2024) where information from a local point-based neighborhood is used to make a nonlinear aggregation. In practice, the material forces are usually explicitly modeling bond damage by comparing the stretch value $\frac{\|\mathbf{y}(\mathbf{q},t)-\mathbf{y}(\mathbf{x},t)\|}{\|\mathbf{x}-\mathbf{q}\|}$ with a threshold s_c decided by a damage model. A high stretch value denotes that the material has been strongly pulled away from the reference position with the $\mathbf{f}_{mat}$ set to 0 if the stretch of the material is over the damage threshold s_c. This explicit bond damage modeling is considered one of the most notable features of peridynamics, and it also shows the importance of considering both the reference position and the current position when modeling fracture and damage.

3.2 From Peridynamics to Neural-PD

In practice, peridynamics uses heuristic constitutive and contact laws to model the material and contact forces, and our goal is to avoid those heuristics and use learnable nonlinear layers to model similar effects, so that we can learn more complex effects directly from data. Equation 3 suggests three computation stages for Neural-PD: a material-interaction module operating on fixed within-body neighborhoods, a contact module operating on evolving cross-body neighborhoods, and a force-prediction and integration stage that updates the particle state. Neural-PD retains this physically meaningful decomposition but learns the material and contact responses from trajectories instead of prescribing constitutive and contact laws.

The basic building block in Neural-PD is an interaction block (Fig. 1 Left) that consists of an intra-body material forces branch (details in Sec. 3.3) and an inter-body contact forces branch (Sec. 3.4). At each interaction block ℓ, both branches receive the same incoming point features $\mathbf{h}_i^{(\ell),t}$ and produce a material message $\mathbf{m}_{i,\text{mat}}^{(\ell),t}$ and a contact message $\mathbf{m}_{i,\text{cnt}}^{(\ell),t}$, respectively. We concatenate the two messages, fuse them with a learned linear layer, and apply layer normalization. We refer to this layer-wise fusion as an *interaction block*:

$$\mathbf{m}_i^{(\ell),t} = \text{LN}^{(\ell)}\left(\mathbf{W}_{\text{fuse}}^{(\ell)}\left[\mathbf{m}_{i,\text{mat}}^{(\ell),t} \,\|\, \mathbf{m}_{i,\text{cnt}}^{(\ell),t}\right] + \mathbf{b}_{\text{fuse}}^{(\ell)}\right), \tag{4}$$

where $\|$ denotes feature concatenation. Material and contact interactions are evaluated in parallel and merged only after both messages have been computed. The fused message updates the point representation passed to the next layer.

The interaction block reflects the inspiration from the PD equation (3) for a single infinitesimal time step. However, conventional simulators integrate over many small time steps to satisfy stability constraints (Littlewood et al., 2023). For neural dynamics modeling, this time integration could equate to having thousands of consecutive interaction blocks. This is not likely to be sustainable in practice, both because of the slow training and evaluation speed, as well as the GPU memory issues if these many layers need to be stored.

On the other hand, we would like to fully utilize the capability of neural networks on learning complex nonlinear functions. Hence, we utilize a U-Net approach that can directly predict the motion over comparatively large time intervals. The U-Net, common in point cloud deep learning models,

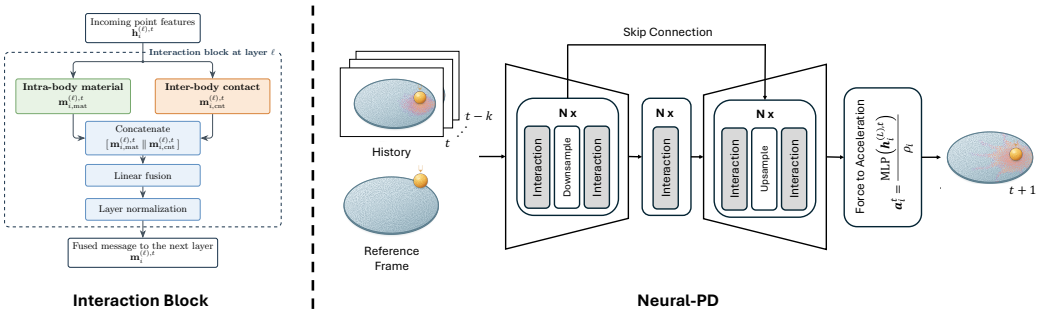

Figure 1: Left: Interaction block used at each U-Net layer. The material and contact branches process the same incoming point features in parallel. Their messages are concatenated, linearly fused, and normalized before being passed to the next layer. Right: the multiscale U-Net, in which the interaction block is repeated at each resolution.

downsamples the points in several stages to obtain coarser point clouds, after bottleneck layers at the coarsest level, it then upsamples back to the full point resolution. Whereas information exchanged only between immediate neighbors requires many layers to travel across the full-resolution point cloud, the coarser U-Net levels provide shorter paths for long-range force propagation. Fig. 1 Right shows the full architecture of the U-Net. All the blocks in the U-Net are interaction blocks, and they can be thought of as resembling many PD time-integration steps. At the end of the U-Net, an MLP predicts the force density, which is integrated to obtain the next state. In the rest of this section we discuss details of the model.

Input representation. Because Neural-PD operates on a discrete point cloud, we represent the continuous material coordinates introduced above by a set of points indexed by i. At time t, each point i is represented by its current position $\mathbf{y}_i^t$, its reference position $\mathbf{x}_i$, its recent velocity history $\{\mathbf{v}_i^{t-k}, \ldots, \mathbf{v}_i^t\}$, and its material properties such as density and volume. We apply a Fourier feature encoding (Rahimi & Recht, 2007) to each velocity vector before the sequence is processed by a History Transformer as in (Zhang et al., 2025a). The resulting history representation is combined with the encoded material properties to form the initial point feature $\mathbf{h}_i^t$. The current and reference positions are used separately to construct the material and contact neighborhoods and to compute their relative geometry. The shared point feature is supplied to both modules, while the geometric information specific to each neighborhood is introduced within the corresponding module.

3.3 Intra-Body Material Forces

The first component of Neural-PD models the intra-body material interactions corresponding to the first summation in Eq. 3. Rather than prescribing the peridynamic constitutive law analytically, we approximate this neighborhood interaction with a learned point transformer block.

Following the peridynamic formulation, the material neighborhood is defined in the reference configuration and remains fixed as the object deforms. We further augment each material interaction with a learned history-dependent connectivity to represent irreversible bond failure. We describe the material neighborhood, bond geometry, and connectivity below.

Material neighborhood. We replace the radius-based neighborhood with K-nearest-neighbors, which is commonly used in point-cloud backbones because it enables efficient batched tensor operations. Specifically, the material neighborhood $\mathcal{N}_i^{\text{mat}}$ contains the K nearest points to point i from the same object in the reference configuration. Each pair (i, j) with $j \in \mathcal{N}_i^{\text{mat}}$ defines a material bond. The graph remains fixed as the body deforms, thereby preserving material identity and preventing the material operator from exchanging information between different objects.

Bond geometry. For each material bond (i, j), we describe its relative geometry in both the reference and current configurations:

$$\boldsymbol{\xi}_{ij} = \mathbf{x}_j - \mathbf{x}_i, \qquad \boldsymbol{\eta}_{ij}^t = \mathbf{y}_j^t - \mathbf{y}_i^t. \tag{5}$$

Here, $\boldsymbol{\xi}_{ij}$ denotes the reference bond vector and $\boldsymbol{\eta}_{ij}^t$ its current counterpart. Together, these vectors provide an explicit geometric description of each bond before and after deformation.

Material connectivity. To model irreversible bond failure, we associate each material bond with a learned connectivity $\omega_{ij}^t \in [0, 1]$. A value of one denotes an intact bond, whereas zero denotes a fully broken bond. All bonds begin intact. At each simulation step, a small network at the beginning of the encoder predicts a candidate connectivity $\widehat{\omega}_{ij}^t$ from the previous connectivity, the bond geometry and relative motion, and the features of its two endpoints. The pair encoding is symmetric, so the prediction does not depend on the order of the endpoints. We then apply the recurrent update

$$\omega_{ij}^t = \min\left(\omega_{ij}^{t-1}, \widehat{\omega}_{ij}^t\right), \qquad \omega_{ij}^0 = 1. \tag{6}$$

This update makes connectivity non-increasing, so a broken bond cannot heal. Connectivity updates begin after the corresponding object first experiences contact and remain active thereafter. This prevents the model from predicting damage before any contact event.

Attention-based material interaction. Standard discrete attention can be interpreted as a Monte Carlo or finite-difference approximation of a continuum attention operator (Calvello et al., 2025). Motivated by this connection, we utilize a modified version of multi-head attention (Vaswani et al., 2017) over the discrete material neighborhood. For each attention head, the bond geometry enters as relative positional encoding: a bias term learned with an MLP $b_{ij}^t = \psi_\theta(\boldsymbol{\xi}_{ij}, \boldsymbol{\eta}_{ij}^t)$ formed from the reference and current bond geometry. The material message $\mathbf{m}_{i,mat}^t$ for point i is then

$$\alpha_{ij}^t = \omega_{ij}^t \frac{\exp\left(\langle \mathbf{W}_Q \mathbf{h}_i^t,\, \mathbf{W}_K \mathbf{h}_j^t \rangle / \sqrt{d} \;+\; b_{ij}^t\right)}{\sum_{l \in \mathcal{N}_i^{\mathrm{mat}}} \exp\left(\langle \mathbf{W}_Q \mathbf{h}_i^t,\, \mathbf{W}_K \mathbf{h}_l^t \rangle / \sqrt{d} \;+\; b_{il}^t\right)}, \qquad \mathbf{m}_{i,\mathrm{mat}}^t = \sum_{j \in \mathcal{N}_i^{\mathrm{mat}}} \alpha_{ij}^t \, \mathbf{W}_V \mathbf{h}_j^t. \tag{7}$$

where $\mathbf{W}_Q, \mathbf{W}_K$ and $\mathbf{W}_V$ are standard QKV transform weight matrices, d is the dimensionality of the QK heads. There are 2 main differences from regular attention. First, the relative position encoding is computed from both the reference and current configurations. Second, the bond connectivity ω_{ij}^t is explicitly multiplied to the α_{ij}^t weights after the softmax, so the material message decays as damage accumulates, and a fully broken bond between i and j would lead to 0 contribution from the the neighbor point j.

3.4 Inter-Body Contact Forces

The second component of Neural-PD models the inter-body contact interactions corresponding to the second integral in Eq.(3). Unlike the material neighborhood, which is fixed in the reference configuration, the contact neighborhood is constructed from the current geometry and recomputed at every simulation step. Connectivity gating is also not applied for inter-body forces.

Contact neighborhood. For each point i, $\mathcal{N}_i^{\mathrm{cnt},t}$ contains up to the K nearest points within radius c that belong to a different object. We found filtering with radius beneficial to performance in this case, and the neighborhood is padded with zeros if it ends up with less than K points.

For each contact pair (i, j), we compute the current relative position

$$\mathbf{d}_{ij}^t = \mathbf{y}_j^t - \mathbf{y}_i^t, \tag{8}$$

which describes the local geometry between points on different objects. A learned relative-position embedding maps $\mathbf{d}_{ij}^t$ into the contact operator.

Attention-based contact interaction. We instantiate the contact backbone $\mathcal{P}_{\theta_{\mathrm{cnt}}}$ using a separate multi-head attention operator over the cross-object neighborhood. Here we use the regular multi-head attention, with the relative position embedding coming from an MLP over $\mathbf{d}_{ij}^t$.

3.5 Force Prediction, Time Integration and Training Objectives

The material and contact operators introduced above are applied repeatedly across the resolutions of a hierarchical point-cloud U-Net, allowing the two types of interactions to be jointly propagated and refined before predicting the resulting force density. At the end of the decoder of the U-Net, a multilayer perceptron maps the final full-resolution representation $\mathbf{h}_i^{(L),t}$ to a predicted force density. Following the equation of motion in Eq. (2), we obtain the acceleration through

$$\rho_i \, \widehat{\mathbf{a}}_i^t = \mathrm{MLP}\left(\mathbf{h}_i^{(L),t}\right). \tag{9}$$

We then advance the point state using symplectic Euler integration:

$$\widehat{\mathbf{v}}_i^{t+1} = \mathbf{v}_i^t + \Delta t\, \widehat{\mathbf{a}}_i^t, \qquad \widehat{\mathbf{y}}_i^{t+1} = \mathbf{y}_i^t + \Delta t\, \widehat{\mathbf{v}}_i^{t+1}. \tag{10}$$

We supervise Neural-PD with a position Huber loss over a multi-step rollout, $\mathcal{L}_{\text{train}} = \sum_{r=1}^{R} \mathcal{L}_{\text{pos}}^r$, where $\mathcal{L}_{\text{pos}}$ measures the Huber error between predicted and ground-truth positions.

4 Experiments

4.1 Datasets

Because no existing benchmark quantitatively evaluates damage and fracture in learned physics simulation, we create a benchmark of 3,000 peridynamic trajectories containing 383,000 frames. It consists of four subsets: Ball-Plate, Mug-Drop, Bottle-Bullet, and Cloth-Fall. The first three contain contact-driven damage and fracture, including a ball breaking a glass plate, a dropped mug shattering, and a bullet piercing a bottle. Cloth-Fall contains no fracture and instead evaluates viscoelastic deformation as a thin sheet falls onto fixed obstacles. Across all subsets, we randomize initial conditions such as loading direction, contact location, and object pose. Additional dataset and simulation details are provided in the Appendix.

4.2 Baselines

Learned simulators. We compare against a Transformer based baseline, Dynamics-PTv3, adapted from 3DGSim (Zhobro et al., 2026), Object Dynamics Model (Kim & Fuxin, 2024; Kim et al., 2026), and Subequivariant Graph Neural Networks (SGNN) (Han et al., 2022). Table 1 is reserved for results produced under the matched evaluation protocol.

Particle-based numerical methods. We additionally compare Peridynamics, the standard Material Point Method (MPM), continuum-damage MPM (CD-MPM) (Wolper et al., 2019), and Neural-PD. In this case, it is debatable which simulator generates more realistic trajectories. We refer the readers to Fig. 2 for a visual comparison (where we believe Peridynamics crack scenes look more natural than MPM and CD-MPM), but do not insist on Peridynamics being the absolute ground truth to evaluate other simulators against. Instead, we report qualitative results and the speed of respective methods. All methods are evaluated using the same initial conditions and rollout horizons.

4.3 Experimental Setup

We evaluate Neural-PD on held-out 3D trajectories and compare it with learned particle and graph simulators as well as particle-based numerical methods. All methods use the same train, validation, and test scenes, rollout horizon, and state normalization where applicable. Results are computed over complete autoregressive rollouts rather than teacher-forced one-step predictions.

4.4 Main Results

We report the main quantitative comparisons across learned simulators and physics-based numerical methods. Table 1 compares autoregressive trajectory accuracy against the learned baselines when all methods are rolled out to 30 steps, while Table 3 summarizes computational efficiency of different simulators and Neural-PD. It can be seen that Neural-PD significantly outperforms all other

Table 1: Mean Euclidean position error at rollout step 30 across datasets. Entries are the mean $\pm$ standard deviation across random seeds. The best result in each completed dataset is shown in bold. *NaN (diverged)* denotes numerical rollout instability that prevents a valid aggregate.

Dataset	SGNN	Object Dynamics	Dynamics-PTv3	Neural-PD (ours)
Ball–Plate	$(1.43 \pm 1.6) \times 10^{0}$	*NaN (diverged)*	$(9.5 \pm 1.2) \times 10^{-4}$	$\mathbf{(5.88 \pm 0.27) \times 10^{-4}}$
Mug–Drop	$(1.94 \pm 0.21) \times 10^{-2}$	$(2.59 \pm 0.27) \times 10^{-2}$	$(2.32 \pm 0.19) \times 10^{-2}$	$\mathbf{(1.980 \pm 0.09) \times 10^{-2}}$
Bottle–Bullet	*NaN (diverged)*	$(1.71 \pm 0.17) \times 10^{-3}$	$(6.0 \pm 4.1) \times 10^{-3}$	$\mathbf{(9.1 \pm 1.6) \times 10^{-4}}$
Cloth-Fall	$(2.1 \pm 0.56) \times 10^{-2}$	$(15.39 \pm 12.12) \times 10^{-2}$	$(3.5 \pm 1.6) \times 10^{-2}$	$\mathbf{(7.67 \pm 1.15) \times 10^{-3}}$

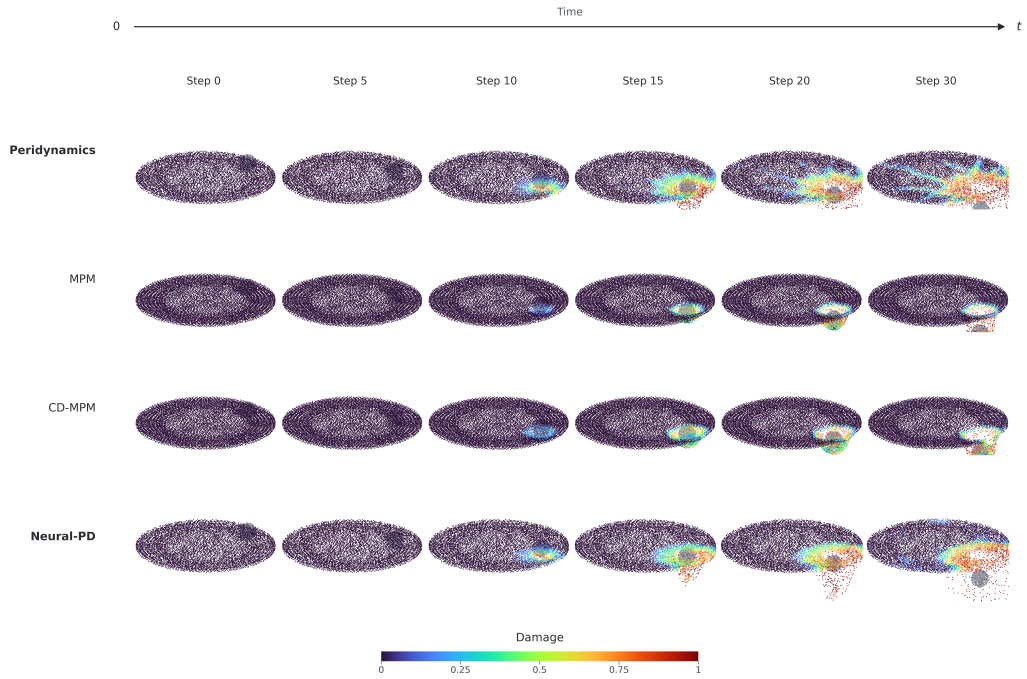

Figure 2: Qualitative comparison on Ball–Plate. Rows show rollouts from Peridynamics, MPM, CD-MPM, and Neural-PD at selected steps. Color indicates damage, from zero (purple) to one (red). Gray points denote the impactor (ball).

neural dynamics simulators, achieves $38\% - 93\%$ lower error than the second best competitor on all scenarios. Besides, Neural-PD is significantly faster than all simulators. Table 2 shows Neural-PD results on the Ball-Plate dataset with different number of rollout steps. It can be seen that from 10 steps on Neural-PD is shown to have significantly less cumulative errors than baselines.

4.5 Qualitative Results

Figure 2 and 3 compare Neural-PD predictions with the reference trajectories on the Mug–Drop and Ball–Plate datasets. The selected frames show how the deformation and damage evolve over an autoregressive rollout. It can be seen that neural-PD results are closer to the peridynamics baseline than other competitors.

4.6 Ablation Studies

We compare the full Neural-PD model with two variants of its Peridynamic-inspired material interactions on the ball-plate task. The current-frame kNN variant recomputes the material neighborhood from predicted positions at every rollout step and does not use the material-connectivity gate. The no-gate variant retains the reference-frame material neighborhood but does not use material-connectivity gating. Comparing the two no-gate variants isolates the effect of the neighborhood

Table 2: Mean Euclidean position error on the Ball–Plate dataset for learned simulators.Each entry is the mean $\pm$ sample standard deviation across random seeds. Lower is better; the lowest mean at each rollout step is shown in bold.

Rollout step	SGNN	Object Dynamics	Dynamics-PTv3	Neural-PD (ours)
1	$(7.8 \pm 7.4) \times 10^{-4}$	$\mathbf{(4.01 \pm 0.13) \times 10^{-7}}$	$(1.85 \pm 0.27) \times 10^{-6}$	$(6.7 \pm 5.8) \times 10^{-7}$
5	$(2.2 \pm 2.8) \times 10^{-2}$	$\mathbf{(5.96 \pm 0.34) \times 10^{-6}}$	$(2.68 \pm 0.42) \times 10^{-5}$	$(7.3 \pm 1.5) \times 10^{-6}$
10	$(4.5 \pm 4.4) \times 10^{-2}$	$(6.04 \pm 0.33) \times 10^{-5}$	$(1.27 \pm 0.25) \times 10^{-4}$	$\mathbf{(3.80 \pm 0.36) \times 10^{-5}}$
15	$(1.10 \pm 0.90) \times 10^{-1}$	$(2.85 \pm 0.09) \times 10^{-4}$	$(3.03 \pm 0.49) \times 10^{-4}$	$\mathbf{(1.47 \pm 0.07) \times 10^{-4}}$
20	$(2.6 \pm 2.4) \times 10^{-1}$	$(5.32 \pm 0.14) \times 10^{-4}$	$(5.03 \pm 0.75) \times 10^{-4}$	$\mathbf{(2.87 \pm 0.12) \times 10^{-4}}$
25	$(6.1 \pm 6.2) \times 10^{-1}$	$(8.7 \pm 1.2) \times 10^{-4}$	$(7.2 \pm 1.0) \times 10^{-4}$	$\mathbf{(4.35 \pm 0.19) \times 10^{-4}}$
30	$(1.4 \pm 1.6) \times 10^{0}$	*NaN (diverged)*	$(9.5 \pm 1.2) \times 10^{-4}$	$\mathbf{(5.88 \pm 0.27) \times 10^{-4}}$

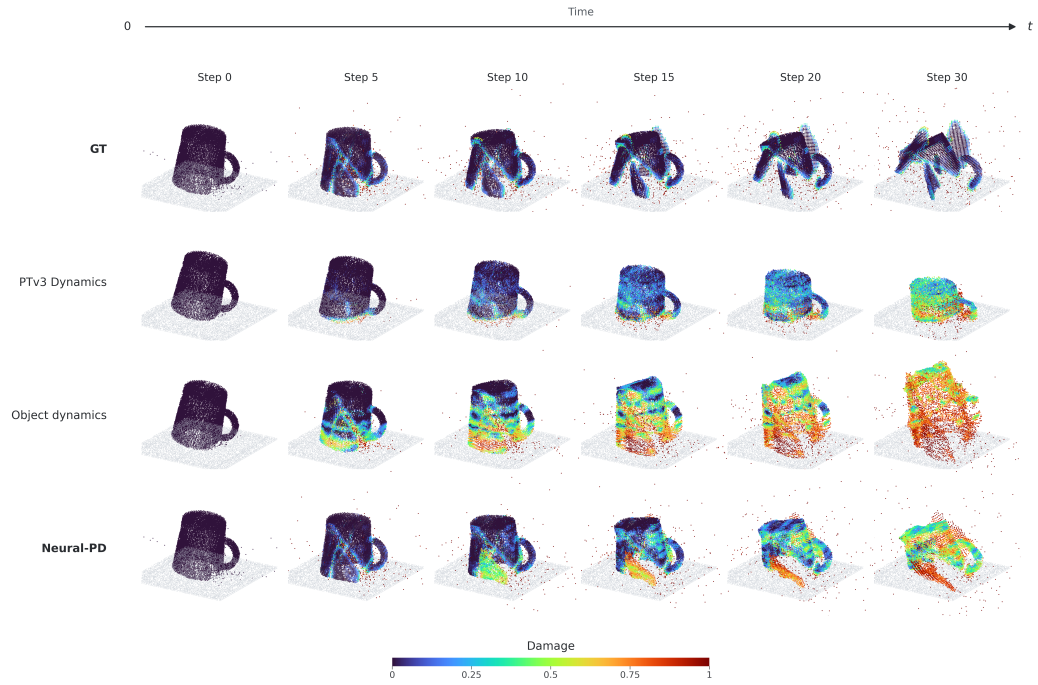

Figure 3: Qualitative results on the Mug-drop dataset, comparing our method against the ground truth and baselines

Table 3: Simulation speed in frames per second (FPS) across datasets under matched simulation conditions and rollout horizons.

Method	Ball–Plate	Cloth–Fall	Bottle–Bullet	Mug–Drop
Peridynamics	0.00433	0.0622	0.703	0.161
MPM	5.61	8.67	4.35	2.98
CD-MPM	5.49	8.57	4.30	2.59
Neural-PD (ours)	**19.73**	**20.42**	**8.34**	**12.71**

definition, while comparing full Neural-PD with the reference-frame no-gate variant measures the contribution of irreversible material connectivity. All other settings are held fixed. Results show that the full Neural-PD approach significantly outperforms those versions, reducing the errors by at least 42%, showcasing the usefulness of these PD-inspired designs.

Table 4: Ablation of the material-neighborhood definition and material- connectivity gate, evaluated using mean Euclidean position error.

Variant	Material kNN	Gate	Euclidean ↓
Neural-PD (full)	**Reference**	**Yes**	5.88×10^{-4}
Neural-PD, current-frame kNN	Current	No	10.52×10^{-4}
Neural-PD, without material gate	Reference	No	12.92×10^{-4}

5 Conclusion

We propose Neural-PD, a Peridynamics-inspired neural dynamics model for fracture and damage. Neural-PD uses reference-configuration neighborhoods for material interactions and current-state neighborhoods for contact, together with an explicit material-connectivity gate and connectivity-aware attention to model bond damage. A U-Net enables dynamics prediction over relatively large time steps. We also introduce a benchmark covering four types of scenarios for fracture, damage, and deformation, on which Neural-PD significantly outperforms other neural dynamics models while running substantially faster than traditional simulators.

AI USE STATEMENT

In preparing this work, we used generative AI tools to assist with language editing, code debugging, figure generation and refinement, and improvements to the clarity of the presentation. We did not use these tools to produce experimental results, fabricate or alter data, or determine the final scientific conclusions. The authors reviewed, tested, and verified all AI-assisted code. We checked every AI-assisted figure against the underlying data and reviewed and revised all AI-assisted text for accuracy, clarity, and consistency with the work. The authors take full responsibility for the final content of this paper, including all text, claims, code, figures, and other artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

We provide the model formulation, neighborhood construction, material- connectivity update, multiscale architecture, integration scheme, training objective, and optimization settings in Section 3 and the Appendix. The dataset protocol, baseline comparisons, autoregressive evaluation procedure, rollout horizons, and error metrics are described in Section 4. All learned methods are evaluated using the same training, validation, and test scenes, and results are aggregated across random seeds where available. Upon publication, we will release the benchmark datasets, data-generation and preprocessing scripts, training and evaluation code, configuration files, exact data splits and random seeds, and pretrained checkpoints.

REFERENCES

Kelsey R Allen, Tatiana Lopez Guevara, Yulia Rubanova, Kim Stachenfeld, Alvaro Sanchez-Gonzalez, Peter Battaglia, and Tobias Pfaff. Graph network simulators can learn discontinuous, rigid contact dynamics. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *CORL*, volume 205 of *Proceedings of Machine Learning Research*, pp. 1157–1167. PMLR, 14–18 Dec 2023a. URL https://proceedings.mlr.press/v205/allen23a.html.

Kelsey R Allen, Yulia Rubanova, Tatiana Lopez-Guevara, William F Whitney, Alvaro Sanchez-Gonzalez, Peter Battaglia, and Tobias Pfaff. Learning rigid dynamics with face interaction graph networks. In *ICLR*, 2023b.

Edoardo Calvello, Nikola B. Kovachki, Matthew E. Levine, and Andrew M. Stuart. Continuum attention for neural operators. *Journal of Machine Learning Research*, 26(300):1–52, 2025. URL https://jmlr.org/papers/v26/24-0879.html.

Yujie Chen, Yang Yang, and Yijun Liu. A neural network peridynamic method for modeling rubber-like materials. *International Journal of Mechanical Sciences*, 273:109234, 2024. doi: 10.1016/j.ijmecsci.2024.109234. URL https://www.sciencedirect.com/science/article/pii/S0020740324002765.

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

Dewen Guo, Ran Tian, Sinuo Liu, Guoping Wang, and Sheng Li. Diagonal hessian proxy for efficient elastic simulation using peridynamics. *IEEE Transactions on Visualization and Computer Graphics*, 31(9):6466–6483, 2025. doi: 10.1109/TVCG.2024.3516480.

Erfan Hamdi and Emma Lejeune. Towards robust surrogate models: Benchmarking machine learning approaches to expediting phase field simulations of brittle fracture. *Computer Methods in Applied Mechanics and Engineering*, 449:118526, 2026. doi: 10.1016/j.cma.2025.118526. URL https://doi.org/10.1016/j.cma.2025.118526.

Jiaqi Han, Wenbing Huang, Hengbo Ma, Jiachen Li, Joshua B. Tenenbaum, and Chuang Gan. Learning physical dynamics with subequivariant graph neural networks. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/a845fdc3f87751710218718adb634fe7-Abstract-Conference.html.

Xiaowei He, Huamin Wang, and Enhua Wu. Projective peridynamics for modeling versatile elastoplastic materials. *IEEE Transactions on Visualization and Computer Graphics*, 24(9):2589–2599, 2018. doi: 10.1109/TVCG.2017.2755646.

Bei Huang, Yixin Chen, Ruijie Lu, Gang Zeng, Hongbin Zha, Yuru Pei, and Siyuan Huang. GaussianFluent: Gaussian simulation for dynamic scenes with mixed materials. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21583–21593, 2026.

Siavash Jafarzadeh, Stewart Silling, Ning Liu, Zhongqiang Zhang, and Yue Yu. Peridynamic neural operators: A data-driven nonlocal constitutive model for complex material responses. *Computer Methods in Applied Mechanics and Engineering*, 425:116914, 2024. doi: 10.1016/j.cma.2024.116914.

Siavash Jafarzadeh, Stewart Silling, Lu Zhang, Colton J. Ross, Chung-Hao Lee, S. M. Rakibur Rahman, Shuodao Wang, and Yue Yu. Heterogeneous peridynamic neural operators: Discover biotissue constitutive law and microstructure from digital image correlation measurements. *Foundations of Data Science*, 7(1):226–270, 2025. doi: 10.3934/fods.2024041. URL https://www.aimsciences.org/article/doi/10.3934/fods.2024041.

Chanho Kim and Li Fuxin. Object dynamics modeling with hierarchical point cloud-based representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20977–20986, 2024.

Chanho Kim, Suhas V. Sumukh, and Li Fuxin. Learning a particle dynamics model with real-world videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, 2026.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023. URL https://www.jmlr.org/papers/v24/21-1524.html.

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

Zhe Lin, Eric Gu, Surong Huang, and Lei Wang. An efficient pinn–based calibration method for mesoscale peridynamic concrete models. *Structural Control and Health Monitoring*, 2025(1): 6641629, 2025. doi: https://doi.org/10.1155/stc/6641629.

David J. Littlewood, Michael L. Parks, John T. Foster, John A. Mitchell, and Patrick Diehl. The peridigm meshfree peridynamics code. *Journal of Peridynamics and Nonlocal Modeling*, 2023. doi: 10.1007/s42102-023-00100-0.

Zixuan Lu, Xiaowei He, Yuzhong Guo, Xuehui Liu, and Huamin Wang. Projective peridynamic modeling of hyperelastic membranes with contact. *IEEE Transactions on Visualization and Computer Graphics*, 30(8):4601–4614, 2024. doi: 10.1109/TVCG.2023.3271511.

Christian Miehe, Martina Hofacker, and Fabian Welschinger. A phase field model for rate-independent crack propagation: Robust algorithmic implementation based on operator splits. *Computer Methods in Applied Mechanics and Engineering*, 199(45–48):2765–2778, 2010. doi: 10.1016/j.cma.2010.04.011.

Nicolas Moës, John Dolbow, and Ted Belytschko. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering*, 46(1):131–150, 1999. doi: 10.1002/(SICI)1097-0207(19990910)46:1⟨131::AID-NME726⟩3.0.CO;2-J.

Cong Tien Nguyen, Selda Oterkus, and Erkan Oterkus. A physics-guided machine learning model for two-dimensional structures based on ordinary state-based peridynamics. *Theoretical and Applied Fracture Mechanics*, 112:102872, 2021. ISSN 0167-8442. doi: https://doi.org/10.1016/j.tafmec.2020.102872. URL https://www.sciencedirect.com/science/article/pii/S0167844220304481.

Feng Nie, Zhengzheng Wang, Linfeng Liu, Huili Wang, and Jiang Lin. 3d cnn-based crack propagation prediction in peridynamic concrete models under freeze-thaw cycles. *Computers Structures*, 318:107959, 2025. ISSN 0045-7949. doi: https://doi.org/10.1016/j.compstruc.2025.107959. URL https://www.sciencedirect.com/science/article/pii/S0045794925003177.

Kai Partmann, Manuel Dienst, and Kerstin Weinberg. Simulation of fracture and damage with Peridynamics.jl. *Proceedings of the JuliaCon Conferences*, 6(64):165, 2024. doi: 10.21105/jcon.00165. URL https://doi.org/10.21105/jcon.00165.

Roberto Perera and Vinamra Agrawal. Multiscale graph neural networks with adaptive mesh refinement for accelerating mesh-based simulations. *Computer Methods in Applied Mechanics and Engineering*, 429:117152, 2024. doi: 10.1016/j.cma.2024.117152. URL https://doi.org/10.1016/j.cma.2024.117152.

Roberto Perera, Davide Guzzetti, and Vinamra Agrawal. Graph neural networks for simulating crack coalescence and propagation in brittle materials. *Computer Methods in Applied Mechanics and Engineering*, 395:115021, 2022. doi: 10.1016/j.cma.2022.115021. URL https://doi.org/10.1016/j.cma.2022.115021.

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

Lukas Prantl, Benjamin Ummenhofer, Vladlen Koltun, and Nils Thuerey. Guaranteed conservation of momentum for learning particle-based fluid dynamics. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 6901–6913. Curran Associates, Inc., 2022. doi: 10.52202/068431-0500. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/2dd7f33ffbb59b4ff987be5442a13016-Paper-Conference.pdf.

Yuxiong Qin, Qingfu Zhang, and Zhongkai Zhang. Fast nodal hessian computation for peridynamic fracture simulation. *Computer Graphics Forum*, n/a(n/a):e70503. doi: https://doi.org/10.1111/cgf.70503. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.70503.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J. Platt, D. Koller, Y. Singer, and S. Roweis (eds.), *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper_files/paper/2007/file/013a006f03dbc5392effeb8f18fda755-Paper.pdf.

J. Ramesh Babu and S. Gopalakrishanan. Thermal diffusion in discontinuous media: A hybrid peridynamics-based machine learning model. *Computers Structures*, 290:107179, 2024. ISSN 0045-7949. doi: https://doi.org/10.1016/j.compstruc.2023.107179. URL https://www.sciencedirect.com/science/article/pii/S0045794923002092.

J. Ramesh Babu and S. Gopalakrishnan. Synergistic approach: Peridynamics and machine learning regression for efficient pitting corrosion simulation. *Computers Structures*, 305:107588, 2024. ISSN 0045-7949. doi: https://doi.org/10.1016/j.compstruc.2024.107588. URL https://www.sciencedirect.com/science/article/pii/S0045794924003171.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8459–8468. PMLR, 2020.

Stewart A. Silling. Reformulation of elasticity theory for discontinuities and long-range forces. *Journal of the Mechanics and Physics of Solids*, 48(1):175–209, 2000. doi: 10.1016/S0022-5096(99)00029-0.

Stewart A. Silling and Ebrahim Askari. A meshfree method based on the peridynamic model of solid mechanics. *Computers & Structures*, 83(17–18):1526–1535, 2005. doi: 10.1016/j.compstruc.2004.11.026.

Stewart A. Silling, Michael Epton, Olaf Weckner, Jifeng Xu, and Ebrahim Askari. Peridynamic states and constitutive modeling. *Journal of Elasticity*, 88(2):151–184, 2007. doi: 10.1007/s10659-007-9125-1.

Deborah Sulsky, Zhen Chen, and Howard L. Schreyer. A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118(1–2):179–196, 1994. doi: 10.1016/0045-7825(94)90112-0.

Fan Sun, Guilin Wang, Liang Zhang, Runqiu Wang, Tianci Cao, and Xiaotian Ouyang. Material point method for the propagation of multiple branched cracks based on classical fracture mechanics. *Computer Methods in Applied Mechanics and Engineering*, 386:114116, 2021. doi: 10.1016/j.cma.2021.114116.

Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. In *International Conference on Learning Representations*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Caoliwen Wang, Minghao Guo, Siyuan Chen, Heng Zhang, Mengdi Wang, Xingyu Ni, Hanson Sun, Kunyi Wang, Zherong Pan, Kui Wu, Lingjie Liu, Yin Yang, Chenfanfu Jiang, Taku Komura, Wojciech Matusik, and Peter Yichen Chen. Worldparticle: Unified world simulation of lagrangian particle dynamics via transformer, 2026a. URL https://arxiv.org/abs/2605.15305.

Jiamin Wang, Haoping Wang, Xiaokun Wang, Yalan Zhang, Jiří Kosinka, Steffen Frey, Alexandru Telea, and Xiaojuan Ban. Peridynamics-based simulation of viscoelastic solids and granular materials. *Computers Graphics*, 133:104463, 2025. ISSN 0097-8493. doi: https://doi.org/10.1016/j.cag.2025.104463. URL https://www.sciencedirect.com/science/article/pii/S0097849325003048.

Xiaogang Wang, Hongyu Wu, Wenfeng Song, and Kai Xu. Fracture-GS: Dynamic fracture simulation with physics-integrated gaussian splatting. In *International Conference on Learning Representations*, 2026b.

Joshuah Wolper, Yu Fang, Minchen Li, Jiecong Lu, Ming Gao, and Chenfanfu Jiang. CD-MPM: Continuum damage material point methods for dynamic fracture animation. *ACM Transactions on Graphics*, 38(4):119:1–119:15, 2019. doi: 10.1145/3306346.3322949.

Wenxuan Wu, Zhongang Qi, and Fuxin Li. PointConv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9621–9630, 2019. doi: 10.1109/CVPR.2019.00985. URL https://openaccess.thecvf.com/content_CVPR_2019/papers/Wu_PointConv_Deep_Convolutional_Networks_on_3D_Point_Clouds_CVPR_2019_paper.pdf.

Wenxuan Wu, Li Fuxin, and Qi Shan. Pointconvformer: Revenge of the point-based convolution. In *CVPR*, pp. 21802–21813, 2023.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer V3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4840–4851, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Wu_Point_Transformer_V3_Simpler_Faster_Stronger_CVPR_2024_paper.html.

Liyou Xu, Xiaowei He, Wei Chen, Sheng Li, and Guoping Wang. Reformulating hyperelastic materials with peridynamic modeling. *Computer Graphics Forum*, 37, 2018. URL https://api.semanticscholar.org/CorpusID:53090823.

Xiao Xu, Marta D'Elia, and John T. Foster. A machine-learning framework for peridynamic material models with physical constraints. *Computer Methods in Applied Mechanics and Engineering*, 386:114062, 2021. doi: 10.1016/j.cma.2021.114062. URL https://arxiv.org/abs/2101.01095.

Yang Yang, Yujie Chen, and Yijun Liu. A novel neural-network non-ordinary state-based peridynamic method for large deformation and fracture analysis of hyperelastic membrane. *Computer Methods in Applied Mechanics and Engineering*, 431:117239, 2024. doi: 10.1016/j.cma.2024.117239. URL https://www.sciencedirect.com/science/article/pii/S004578252400495X.

Binbin Yin, Jiasheng Huang, and Weikang Sun. Peridynamics-fueled convolutional neural network for predicting mechanical constitutive behaviors of fiber reinforced composites. *Computer Methods in Applied Mechanics and Engineering*, 431:117309, 2024. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2024.117309. URL https://www.sciencedirect.com/science/article/pii/S0045782524005656.

Huaiqian You, Yue Yu, Stewart Silling, and Marta D'Elia. Nonlocal operator learning for homogenized models: From high-fidelity simulations to constitutive laws. *Journal of Peridynamics and Nonlocal Modeling*, 6(4):709–724, 2024. doi: 10.1007/s42102-024-00119-x. URL https://link.springer.com/article/10.1007/s42102-024-00119-x.

Bowei Zhang, Lei Ke, Adam Harley, and Katerina Fragkiadaki. Tapip3d: Tracking any point in persistent 3d geometry. *Advances in Neural Information Processing Systems*, 38:135284–135303, 2025a.

Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Particle-grid neural dynamics for learning deformable object models from RGB-D videos. In *Proceedings of Robotics: Science and Systems*, 2025b.

Mikel Zhobro, Andreas René Geist, and Georg Martius. 3DGSim: Learning 3d-gaussian simulators from RGB videos. In *International Conference on Machine Learning*, 2026.

A Appendix

A.1 Training Details

We use $R = 6$ and a Huber threshold of 10^{-3}. At each step, the predicted state is fed back into the model, allowing the integrated motion to supervise the predicted force indirectly. We train NeuralPD for 100 epochs with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, and a weight decay of 0.001) at a base learning rate of 10^{-4}. The learning-rate schedule uses a five-epoch linear warmup from 10% of the base rate followed by cosine decay toward 10^{-6}, and we clip the gradient norm at 1.0 for stability.

A.2 Dataset and Simulation Details

The benchmark contains 3,000 trajectories and 383,000 frames across four subsets. Each Ball-Plate trajectory contains 100 saved frames, each Mug-Drop and Bottle-Bullet trajectory contains 192 saved frames, and each Cloth-Fall trajectory contains 91 saved frames.

We select simulation backends based on computational efficiency and material-model support. For Mug-Drop and Bottle-Bullet, we use the Julia-based Peridynamics.jl (Partmann et al., 2024) to accelerate data generation, employing bond-based materials with $\nu = 1/4$. Peridigm (Littlewood et al., 2023) is used for Ball-Plate and Cloth-Fall, with state-based elastic and viscoelastic materials, respectively. We use Peridigm for Cloth-Fall because it supports the standard-linear-solid viscoelastic model used to represent the cloth's history-dependent material response. For the fracture scenarios, both backends export the unweighted fraction of failed original bonds as nodal damage.

A.3 Dataset Statistics

Fracture statistics. For the three fracture families, fracture onset is defined as the first saved state at which more than 0.5% of target points satisfy $d > 0.1$. Among trajectories with an observed onset, the onset times (mean $\pm$ standard deviation) are $94.78 \pm 5.53\,\mu$s, $3.211 \pm 1.224\,$ms, and $100.22 \pm 3.72\,\mu$s for the disk, the mug, and the bottle, respectively.

A.4 QUALITATIVE RESULTS

We provide additional qualitative results for the Ball-Plate scenario for comparison with the baselines, where Neural-PD most accurately captures the damage and subsequent object dynamics.

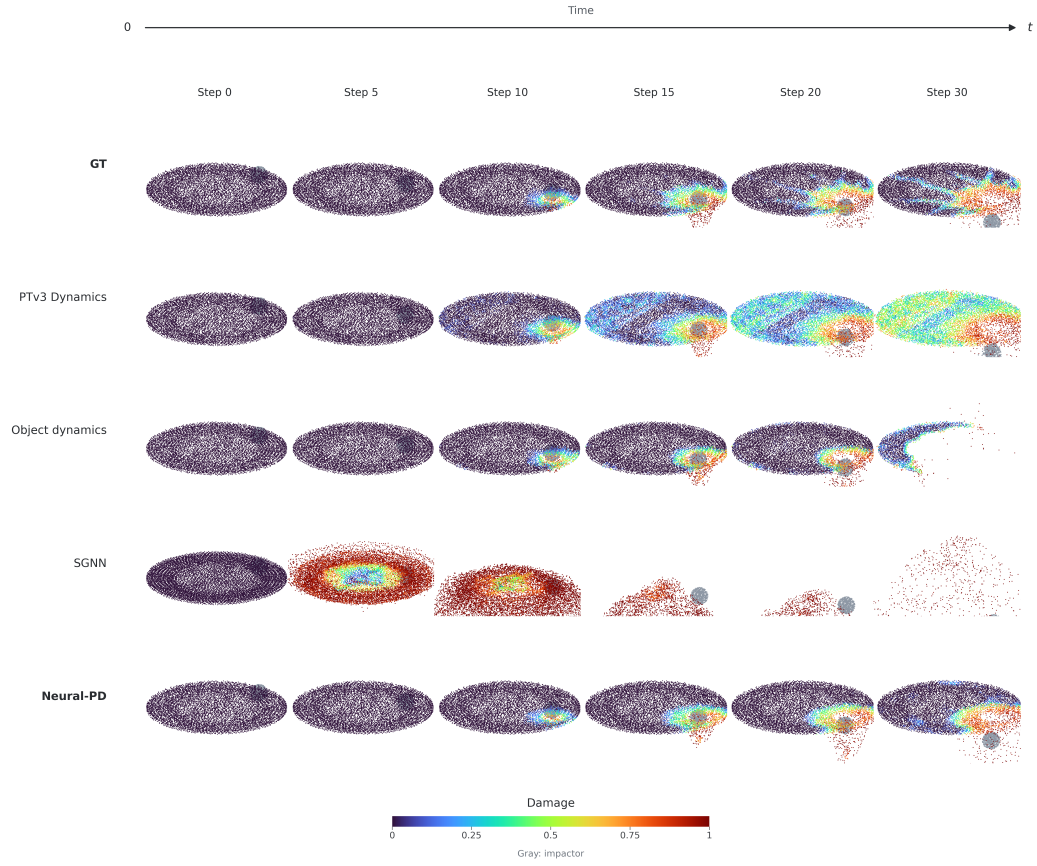

Figure 4: Qualitative results on Ball-Plate dataset, comparing our method against the ground truth and baselines