HYBRID LAGRANGIAN-EULERIAN MODEL FOR LAGRANGIAN FLUID SIMULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Fluid dynamics are typically discretized using either an Eulerian formulation, which resolves physical fields on fixed spatial nodes, or a Lagrangian formulation, which tracks discrete moving particles carrying physical quantities along their trajectories. Building on the latter, Lagrangian neural simulators learn physical dynamics by predicting particle state transitions over time, offering exceptional geometric flexibility and exact advection for modeling free-surface fluids and moving boundaries. However, the absence of a fixed global reference frame as used in Eulerian formulation introduces two severe limitations: (1) spatial redundancy, as accurate gradients require densely packed particles, many of which move nearly identically and waste model capacity; and (2) temporal error accumulation, as small trajectory errors corrupt the local neighborhoods used for information exchange and compound over rollouts. Inspired by classical hybrid numerical solvers, we propose a **Hybrid Lagrangian–Eulerian** neural solver that augments Lagrangian dynamics with an Eulerian representation. To reduce spatial redundancy, an adaptive downsampler learns to prune particles that move nearly identically to their neighbors, focusing model capacity on regions with complex dynamics; the retained particles are then aggregated onto Eulerian nodes to resolve large-scale dynamics. To counter temporal error accumulation, cross-attention lets the aggregated particle features query an Eulerian representation on fixed nodes, which serves as a stable spatial anchor to correct trajectory deviations at every timestep. We conducted comprehensive experiments across 16 datasets and 16 baselines, establishing the most extensive benchmark in the Lagrangian neural simulation literature. Our results demonstrate that this approach establishes a new state-of-the-art.

1 INTRODUCTION

Neural network–based physical simulators have recently achieved remarkable progress in fluid dynamics (Wang et al., 2024; 2025). These models are typically built on either an Eulerian formulation, where dynamics are resolved on fixed spatial domains, or a Lagrangian formulation, which models the fluid as a set of discrete particles that carry and update physical quantities along their moving trajectories. Lagrangian representations are highly favored for their physical intuition and geometric flexibility. By tracking discrete particles, these methods naturally handle complex phenomena like moving domains and free surfaces, which are notoriously difficult to capture on fixed meshes and require costly re-meshing (Monaghan, 1992; Gingold & Monaghan, 1977). Furthermore, since particles carry their properties as they move, advection is treated exactly and computational effort is concentrated where mass exists, avoiding the numerical diffusion and wasted resolution that plague Eulerian schemes. For example, the dam break dataset in Section 5.1 and the stretching droplet dataset in Section 5.4 are particularly well-suited for a Lagrangian representation. An Eulerian approach would be highly inefficient here, as tracking the water splashes would require a prohibitively large and dense mesh, and the stretching droplet would demand costly re-meshing at every time step (Monaghan, 1994).

However, the pure Lagrangian neural simulator introduces fundamental modeling challenges, primarily stemming from the lack of a fixed global reference frame. This absence cascades into two intrinsic challenges. First, without a background grid, Lagrangian methods suffer from spatial redundancy. While an Eulerian mesh can calculate stable finite differences with as few as two neighbors, particle-based methods rely on kernel averaging that often requires a dense neighborhood of 30–50 particles to

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.

yield stable gradients (Gingold & Monaghan, 1977). Consequently, maintaining Eulerian-equivalent fidelity requires a drastically higher particle count (Agertz et al., 2007; Price, 2012; Dehnen & Aly, 2012). Because of this dense packing, many neighboring particles move nearly identically, forcing the neural simulator to spend much of its capacity on redundant, nearly uniform local dynamics rather than on the complex interactions that matter. Second, the lack of a global anchor causes rapid temporal error accumulation. Because particles update their states based mainly on local message passing, any slight deviation in a trajectory causes incorrect neighborhood connections in the next timestep. Whereas errors on a fixed mesh perturb only field values, here they corrupt the graph structure itself; without fixed Eulerian nodes to anchor and correct them, these topological errors compound geometrically, causing the simulation to drift significantly.

In classical computational physics, this precise dilemma led to the development of hybrid methods, most notably the Particle-in-Cell (PIC) (Francis Harvey Harlow, 1955; Dawson, 1983; Tskhakaya, 2008) method. PIC fundamentally improves Lagrangian simulation by leveraging an Eulerian grid as a global computational anchor. Instead of resolving complex PDE dynamics strictly between unstructured particle neighbors, PIC transfers particle properties onto a fixed background grid. The governing equations are then solved stably and efficiently on this Eulerian grid, and the resulting physical fields are interpolated back to update the particles. This approach anchors the drifting particle trajectories to a robust global structure while preserving the Lagrangian advantages. Inspired by this classical synergy, we investigate **how Eulerian representations can be similarly leveraged to enhance the accuracy and robustness of Lagrangian neural solvers.**

We propose a **Hybrid Lagrangian–Eulerian** neural model that downsamples particles and aggregates their information onto a set of Eulerian nodes, where the governing equations are solved efficiently. We further employ a cross-attention mechanism that facilitates information exchange between the Lagrangian and Eulerian representations. By pruning particles that move nearly identically to their neighbors, the downsampler removes spatial redundancy and concentrates model capacity on regions with complex dynamics, thereby improving prediction accuracy. This architecture also eases modeling difficulty through a hierarchical decomposition (Gao & Ji, 2019; Cao et al., 2023). Instead of resolving all particle interactions at a single scale, message passing in the downsampling and upsampling stages captures micro-scale details, while the processor resolves the underlying PDE on the coarse Eulerian nodes. Additionally, because Eulerian nodes never move and their features are tied to fixed spatial locations, the Eulerian branch acts as a spatial anchor. Cross-attention re-grounds the Lagrangian update at every step against this stable reference, suppressing the error accumulation that causes pure-Lagrangian surrogates to drift into unphysical particle distributions. Ultimately, our design exploits the Eulerian representation to enhance the accuracy of Lagrangian neural simulators.

Our contributions are as follows: **(i) Problem Identification:** We identify the fundamental limitations of pure Lagrangian neural simulators, specifically, spatial redundancy and temporal error accumulation, and explain the benefits of a global Eulerian reference frame; **(ii) Practical Solution:** Inspired by classical hybrid methods like PIC, we present a **Hybrid Lagrangian–Eulerian** model that unifies the strengths of both representations within a single neural simulator; **(iii) Experimental Validation:** Through comprehensive experiments across 16 datasets, we demonstrate that our model achieves significant improvements compared to 16 baseline methods.

2 Related Work

Several prior works have explored Lagrangian fluid simulations. A prominent line of research focuses on purely graph-based methods to model particle interactions. GNS (Sanchez-Gonzalez et al., 2020) applies message-passing graph neural networks to learn the dynamics of fluids, granular media, and deformable bodies. Building on this foundation, several works have introduced physical constraints and solver mechanics to improve these networks. Toshev et al. (2024b) improves upon GNS by incorporating various components from standard SPH solvers. Furthermore, Toshev et al. (2023) introduces an architecture that enforces Euclidean symmetries, including translation, rotation, and reflection invariance, resulting in improved generalization, while Prantl et al. (2022b) presents a novel method for guaranteeing linear momentum in the neural simulator. Subsequent works have sought to enhance efficiency and scalability by incorporating particle-grid transfers. Rochman-Sharabi et al. (2025) integrates the material point method into a neural framework, enabling learned particle–grid transfers and updates for efficient emulation of high-fidelity simulations. Xu et al. (2025) integrates

a neural simulator with the classical material-point method (Johnson, 1996; Brackbill & Ruppel, 1986; Sulsky et al., 1994; Nairn, 2003) specifically to achieve real-time simulation. To standardize the evaluation of this growing body of literature, LagrangeBench (Toshev et al., 2024a) provides a comprehensive benchmark for particle-based fluid simulation learning, systematically comparing neural models across multiple physical configurations. Finally, Alkin et al. (2024) proposes a framework that generalizes across various discretizations. However, this approach falls under the domain of field learning as it requires ground-truth particle coordinates to query velocity and cannot directly predict future particle positions. Due to this fundamental difference in problem formulation, it is not included in our comparative analysis.

In the domain of Eulerian simulation, significant progress has been made using deep learning to accelerate solvers on fixed spatial discretizations. This includes (1) grid-based methods (Li et al., 2021; 2023b; Brandstetter et al., 2022; Williams et al., 2023; Taccari et al., 2022; Tran et al., 2023; Huang et al., 2024; Cao, 2021; Lu et al., 2021), and (2) mesh-based techniques designed to generalize across irregular geometries (Pfaff et al., 2021; Wu et al., 2024; Luo et al., 2025; Cao et al., 2023; Li et al., 2023a; Wu et al., 2023; Wang & Wang, 2024; Hao et al., 2023). While these approaches offer efficiency for problems with static topology, they do not address the specific challenges of Lagrangian tracking and dynamic neighborhood configurations central to the particle-based methods discussed in this work.

Beyond forward simulation, neural models are widely utilized for foundational models (Hao et al., 2024; McCabe et al., 2024; Ye et al., 2024; Herde et al., 2024; Shen et al., 2024; Zhou et al.), fluid field reconstruction (Li et al., 2025; Zhong et al., 2023; Mo & Magri, 2024; Yadav et al., 2025; Jing et al., 2024; He et al., 2022), ML-assisted classical solver (List et al., 2022; Sun et al., 2023; Greenfeld et al., 2019; Sappl et al., 2019), aerodynamic shape optimization (Elrefaie et al., 2025; 2024), and inverse design (Behrmann et al., 2019; Teng & Choromanska, 2019; Kruse et al., 2021). While these works fall within the broader area of data-driven neural PDE modeling, they address computational tasks distinct from the sequential forward time-stepping of Lagrangian particles and therefore lie outside the primary scope of our work.

3 Problem Statement

We follow the setup in LagrangeBench (Toshev et al., 2024a) and FD-Bench (Wang et al., 2025) to focus on the weakly-compressible Navier–Stokes equation.

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} = -\rho\left(\nabla\cdot\dot{\boldsymbol{x}}\right), \quad \frac{\mathrm{d}\dot{\boldsymbol{x}}}{\mathrm{d}t} = -\frac{1}{\rho}\nabla p + \frac{1}{\mathrm{Re}}\nabla^2\dot{\boldsymbol{x}} + \frac{1}{\rho}\boldsymbol{F}, \tag{1}$$

where ρ is density, $\dot{\boldsymbol{x}}$ is velocity, p is pressure, Re is the Reynolds number, and $\boldsymbol{F}$ is an external force field. This choice is made solely due to the availability of extensive benchmark datasets. Our design is not tied to this particular equation, and we expect the model to generalize to other fluid scenarios. We adopt a particle-based representation, where the solution is discretized into a set of particles carrying physical properties, denoted by $\boldsymbol{x} \in \mathbb{R}^{N_{\mathrm{lag}}\times d}$, where N_{lag} represents the number of Lagrangian particles. Here, d denotes the spatial dimension, e.g., 2D or 3D. Our objective is to learn a neural surrogate model Ψ that, given τ consecutive frames of particle positions, predicts the positions in the next frame: $\boldsymbol{x}_{t+1} = \Psi\left(\boldsymbol{x}_{(t-\tau+1):t}\right)$.

4 Methodology

In this section, we present our **Hybrid Lagrangian–Eulerian** model, which adopts an Encoder – Downsampler – Processor – Upsampler – Decoder architecture. We describe the role of each component in the following subsections and Figure 1.

Preprocess Given the initial particle positions $\boldsymbol{x}_{(t-\tau+1):t}$, we compute the velocities $\boldsymbol{v}_{(t-\tau+2):t}$ using the backward finite difference method. We then construct the Lagrangian graph $\mathcal{G}_{\mathrm{lag}} = (\mathcal{V}_{\mathrm{lag}}, E_{\mathrm{lag}})$, whose feature vector is defined as $\mathcal{V}^i_{\mathrm{lag}} = \left[\boldsymbol{x}^i_t \,\|\, \boldsymbol{v}^i_{(t-\tau+2):t} \,\|\, \mathrm{type}^i \,\|\, \Theta\right]$, where type^i denotes the type of particle i, and Θ represents global variables (both are optional). The edge set E_{lag} contains directed edge indices and edge attributes, where the attributes include the relative position vector and its norm. The edge indices are determined via a radius graph.

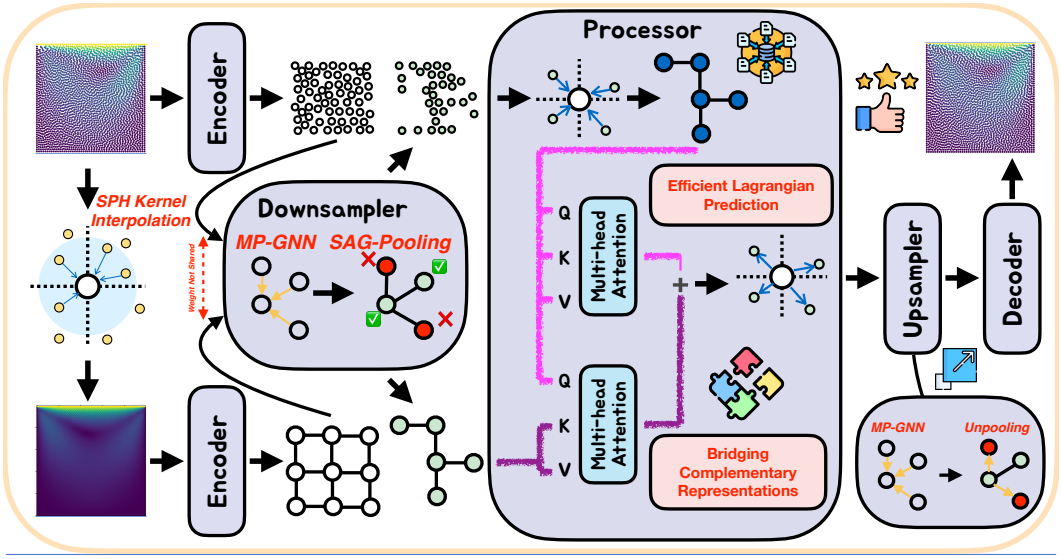


Figure 1: Overall framework of the proposed model. Lagrangian inputs are converted to Eulerian graphs, and the node and edge features are encoded via MLPs and downsampled into sparse representations. The processor solves coarse PDE dynamics by particle aggregation and cross-attention, after which the upsampler and decoder predict the final kinematics.

We also define a corresponding Eulerian graph. Let $\tilde{\boldsymbol{x}} \in \mathbb{R}^{N_{\text{eu}} \times d}$ denote the positions of grid or mesh points in the spatial domain Ω, where N_{eu} represents the number of Eulerian nodes. Unlike the Lagrangian particle positions, which evolve over time, the Eulerian nodes remain fixed. We choose N_{eu} such that $N_{\text{eu}} \ll N_{\text{lag}}$. To obtain Eulerian velocities, we follow Ramachandran et al. (2021) and aggregate the particle velocities using a quintic kernel $W : \mathbb{R}_{\geq 0} \times \mathbb{R}_{>0} \to \mathbb{R}_{\geq 0}$, defined by $W(r,h) = \frac{\sigma_2}{h^2}\left[(3-q)^5 - 6(2-q)^5 + 15(1-q)^5\right]_+$, where $q = \frac{r}{h}$, $\sigma_2 = \frac{7}{478\pi}$, and $[\cdot]_+ := \max(\cdot, 0)$. The Eulerian velocity aggregation is given by $\tilde{\boldsymbol{v}}_j = \frac{\sum_{i=1}^{N} \boldsymbol{v}_i\, W(\|\tilde{\boldsymbol{x}}_j - \boldsymbol{x}_i\|, h)}{\sum_{i=1}^{N} W(\|\tilde{\boldsymbol{x}}_j - \boldsymbol{x}_i\|, h) + \varepsilon}$, where $\tilde{\boldsymbol{v}}_j$ denotes the velocity at Eulerian node j. The Eulerian interaction graph is defined as $\mathcal{G}_{\text{eu}} = (\mathcal{V}_{\text{eu}}, E_{\text{eu}})$, where $\mathcal{V}^i_{\text{eu}} = \left[\tilde{\boldsymbol{x}}^i_t \,\|\, \tilde{\boldsymbol{v}}^i_{(t-\tau+2):t}\right]$, and E_{eu} contains directed edge indices that connect each node to its immediate horizontal and vertical neighbors, with edge features encoding the relative position vector and its norm. For regularly spaced Eulerian grids, omitting edge attributes and node positions does not negatively impact the model.

We analyze Eulerian grid configurations directly in Appendix E, yielding a practical selection rule. We find that both extremely coarse and overly fine resolutions yield the worst results. Instead, optimal performance corresponds to the resolution that a conventional neural PDE solver would typically use for the same domain. Furthermore, performance remains stable across a broad range of intermediate resolutions, demonstrating our model's robustness to this hyperparameter.

Encoder: In the encoding stage, we employ four distinct MLPs denoted by $\mathcal{E}_{\text{lag},\mathcal{V}}$, $\mathcal{E}_{\text{lag},E}$, $\mathcal{E}_{\text{eu},\mathcal{V}}$, and $\mathcal{E}_{\text{eu},E}$ to embed node and edge features into latent space. Formally,

$$\boldsymbol{h}_{\text{lag}} = \mathcal{E}_{\text{lag},\mathcal{V}}(\mathcal{V}_{\text{lag}}), \qquad \boldsymbol{e}_{\text{lag}} = \mathcal{E}_{\text{lag},E}(E_{\text{lag}}), \tag{2}$$

$$\boldsymbol{h}_{\text{eu}} = \mathcal{E}_{\text{eu},\mathcal{V}}(\mathcal{V}_{\text{eu}}), \qquad \boldsymbol{e}_{\text{eu}} = \mathcal{E}_{\text{eu},E}(E_{\text{eu}}). \tag{3}$$

Here, $\boldsymbol{h}_{\text{lag}} \in \mathbb{R}^{N_{\text{lag}} \times d_{\text{latent}}}$ and $\boldsymbol{h}_{\text{eu}} \in \mathbb{R}^{N_{\text{eu}} \times d_{\text{latent}}}$.

Downsampler: Our downsampler consists of M successive blocks, each comprising a SAGPooling (Lee et al., 2019) operation followed by message passing. Given a node feature $\boldsymbol{h} \in \mathbb{R}^{N \times d_{\text{latent}}}$ as well as edge index E with latent features $\boldsymbol{e}$. SAGPooling first computes a scalar importance score for each node via a self-attention GNN: $\boldsymbol{s} = \text{GNN}(\boldsymbol{h}, \boldsymbol{e})$. Then, it selects the top-k nodes with the highest importance scores: $\text{TopK}(\mathbf{s}, k)$, where $k = \lceil \lambda N \rceil$ and λ is the downsampling rate, and retains only the corresponding node features and the induced subgraph. Next, we apply a message

passing GNN with the update rule $\mathbf{m}_i = \sum_{(j\to i)\in E} \phi_m(\mathbf{h}_i, \mathbf{h}_j, \mathbf{e}_{i,j})$ and $\mathbf{h}_i \leftarrow \phi_h(\mathbf{h}_i, \mathbf{m}_i)$, where ϕ_m and ϕ_h are parametrized by MLPs.

We apply this downsampling procedure independently to both the Lagrangian and Eulerian graphs, yielding latent node embeddings $\mathbf{q}_{\text{lag}} \in \mathbb{R}^{\tilde{N}_{\text{lag}}\times d_{\text{latent}}}$ and $\mathbf{q}_{\text{eu}} \in \mathbb{R}^{\tilde{N}_{\text{eu}}\times d_{\text{latent}}}$, with associated particle or node coordinates $\mathbf{y}$ and $\tilde{\mathbf{y}}$, respectively. By construction, $\tilde{N}_{\text{eu}} \ll \tilde{N}_{\text{lag}}$. To reduce the number of hyperparameters, we use a shared pooling rate λ for both the Lagrangian and Eulerian graphs. Empirically, this choice is sufficient to achieve strong performance. Nevertheless, if computational resources permit, we recommend assigning separate rates to the two graphs and performing hyperparameter tuning.

Our downsampler fundamentally differs from those used in existing works. While prior methods (Lino et al., 2021; HAN et al., 2022) also leverage downsampling to propagate information between distant nodes and overcome the limited receptive field of local message passing, our approach is tailored specifically to Lagrangian particle simulation. In this setting, high fidelity requires massive particle counts, resulting in many particles being kinematically indistinguishable from their neighbors. Specifically, in regions where the flow is smooth or nearly uniform, such as laminar zones, these redundant particles move together or remain nearly static, offering little new information if treated separately. To address this, our downsampler prunes these particles into a smaller set of representative super-nodes, capturing their shared properties without expending computation on unnecessary features. This reduction not only improves efficiency but also ensures that the model allocates its capacity to regions with complex, highly variable dynamics where detailed interactions matter. The processor, introduced in the next section, then operates on this compressed representation to solve the governing equations at a global level. Without this downsampling step, the processor would be forced to carry the full burden of redundant particle interactions, wasting computation while diluting attention away from the regions that matter most.

Processor: We define a learnable aggregation kernel $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_{>0}$, parametrized by MLPs, that maps a Lagrangian particle position and an Eulerian node position to a positive weight. For each particle i, $\mathcal{K}$ assigns a weight to each Eulerian node j, satisfying the normalization condition $\sum_{j=1}^{\tilde{N}_{\text{eu}}} \mathcal{K}(\boldsymbol{y}_i, \tilde{\boldsymbol{y}}_j) = 1$. Let $\boldsymbol{W} \in \mathbb{R}^{\tilde{N}_{\text{lag}}\times\tilde{N}_{\text{eu}}}$ denote the weight matrix produced by this kernel. Using $\boldsymbol{W}$, we aggregate the downsampled Lagrangian particles onto the downsampled Eulerian nodes:

$$\mathbf{Z}_{\text{lag}} = \boldsymbol{W}^\top \boldsymbol{q}_{\text{lag}} \in \mathbb{R}^{\tilde{N}_{\text{eu}}\times d_{\text{latent}}}.$$

We then apply multi-head self-attention to the aggregated Lagrangian feature,

$$\boldsymbol{A} = \text{MHA}(\text{Query} = \mathbf{Z}_{\text{lag}},\ \text{Key} = \mathbf{Z}_{\text{lag}},\ \text{Value} = \mathbf{Z}_{\text{lag}}), \tag{4}$$

producing coarse predictions based solely on the aggregated Lagrangian representation. Next, we perform cross-attention between the aggregated Lagrangian feature and the downsampled Eulerian feature,

$$\boldsymbol{C} = \text{MHA}(\text{Query} = \boldsymbol{Z}_{\text{lag}},\ \text{Key} = \boldsymbol{q}_{\text{eu}},\ \text{Value} = \boldsymbol{q}_{\text{eu}}), \tag{5}$$

allowing the model to jointly consider Lagrangian and Eulerian features. The outputs are then combined and scattered back to the downsampled Lagrangian nodes:

$$\boldsymbol{F} = \boldsymbol{A} + \alpha\,\boldsymbol{C}, \quad \hat{\boldsymbol{q}}_{\text{lag}} = \boldsymbol{W}\,\boldsymbol{F} \in \mathbb{R}^{\tilde{N}_{\text{lag}}\times d_{\text{latent}}}, \tag{6}$$

where α is a learnable parameter.

The processor is inspired by PIC (Francis Harvey Harlow, 1955; Dawson, 1983; Tskhakaya, 2008), adapted to the hierarchical latent space. In classical PIC, particles are projected onto a grid using voxelization to solve field equations. The learnable aggregation kernel $\mathcal{K}$ plays the role of this projection operator, while the self-attention on aggregated Lagrangian feature functions as a latent field solver on a coarse level.

The Lagrangian and Eulerian representations serve distinct but highly complementary roles. The particle-based representation retains the capacity to capture complex micro-scale dynamics, while the fixed Eulerian grid provides the global structural anchor that pure particle methods lack. If the model relied solely on self-attention within the Lagrangian representation, updates would be strictly constrained to moving neighborhoods and any slight trajectory deviation would lead to incorrect

	Metric	GNS	SEGNN	GraphUnet	GraphTransformer	AdvDIFFormer	NeuralMPM	PhysicsNFP	**Ours**
2D TGV	MSE	7.920×10^{-3}	1.260×10^{-2}	5.356×10^{-3}	2.963×10^{-2}	1.586×10^{-2}	3.051×10^{-2}	7.140×10^{-3}	$\mathbf{4.551 \times 10^{-3}}_{\uparrow 15.02\%}$
	KE	1.795×10^{-3}	2.854×10^{-3}	1.246×10^{-3}	4.729×10^{-3}	2.339×10^{-3}	6.402×10^{-3}	1.652×10^{-3}	$\mathbf{1.082 \times 10^{-3}}_{\uparrow 13.20\%}$
	Sinkhorn	4.150×10^{-5}	3.722×10^{-4}	3.767×10^{-5}	3.099×10^{-3}	1.744×10^{-4}	9.607×10^{-4}	3.323×10^{-5}	$\mathbf{1.422 \times 10^{-5}}_{\uparrow 57.22\%}$
2D LDC	MSE	3.542×10^{-5}	6.293×10^{-6}	4.413×10^{-6}	6.490×10^{-4}	3.955×10^{-5}	2.308×10^{-3}	2.629×10^{-6}	$\mathbf{1.868 \times 10^{-6}}_{\uparrow 28.94\%}$
	KE	7.880×10^{-6}	1.043×10^{-5}	1.163×10^{-7}	1.635×10^{-4}	1.316×10^{-5}	5.724×10^{-2}	1.068×10^{-7}	$\mathbf{7.240 \times 10^{-8}}_{\uparrow 32.23\%}$
	Sinkhorn	8.239×10^{-6}	1.388×10^{-5}	2.227×10^{-6}	1.803×10^{-5}	2.214×10^{-5}	3.172×10^{-4}	8.768×10^{-7}	$\mathbf{4.675 \times 10^{-7}}_{\uparrow 46.68\%}$
2D RPF	MSE	2.629×10^{-1}	5.779×10^{-2}	4.067×10^{-2}	8.835×10^{-1}	2.842×10^{-1}	1.044×10^{-1}	3.862×10^{-2}	$\mathbf{2.426 \times 10^{-2}}_{\uparrow 37.19\%}$
	KE	7.961×10^{-2}	6.374×10^{-2}	2.471×10^{-2}	4.104×10^{0}	5.848×10^{-2}	2.694×10^{-2}	3.181×10^{-2}	$\mathbf{1.595 \times 10^{-2}}_{\uparrow 35.42\%}$
	Sinkhorn	3.627×10^{-2}	2.134×10^{-2}	7.909×10^{-3}	4.120×10^{-1}	4.812×10^{-2}	2.163×10^{-2}	1.684×10^{-2}	$\mathbf{6.140 \times 10^{-3}}_{\uparrow 22.37\%}$
2D DAM	MSE	2.363×10^{-4}	9.245×10^{-4}	1.505×10^{-4}	4.552×10^{-3}	6.360×10^{-3}	2.010×10^{-4}	1.458×10^{-4}	$\mathbf{9.171 \times 10^{-5}}_{\uparrow 37.10\%}$
	KE	9.461×10^{-7}	1.817×10^{-6}	9.235×10^{-7}	3.783×10^{-4}	6.238×10^{-4}	6.080×10^{-6}	7.856×10^{-7}	$\mathbf{4.709 \times 10^{-7}}_{\uparrow 40.06\%}$
	Sinkhorn	2.131×10^{-4}	1.935×10^{-4}	3.710×10^{-4}	7.751×10^{-3}	8.843×10^{-3}	2.196×10^{-4}	1.886×10^{-4}	$\mathbf{1.216 \times 10^{-4}}_{\uparrow 35.55\%}$

Table 1: Comparison of MSE, Kinetic Energy (KE) error, and Sinkhorn divergence on **2D datasets**. The relative performance gain against the strongest baseline is computed as $\frac{\text{Baseline}-\text{Ours}}{\text{Baseline}} \times 100\%$.

connections, causing local topological errors to compound. Cross-attention directly addresses this by allowing each Lagrangian aggregation token to query and retrieve coarse-grained, stabilized features from the Eulerian background grid. Because Eulerian nodes never move and their features are tied to fixed spatial locations, the Eulerian branch acts as a persistent spatial anchor. By re-grounding the Lagrangian updates against this stable reference at every step, cross-attention breaks the cycle of local error accumulation, preventing unphysical particle drift and ensuring robust simulation stability over rollouts.

We repeat this layer for L steps. Note that the downsampled Lagrangian particles and Eulerian nodes are fairly small. The calculation of the weight matrix and aggregation does not incur significant computational overhead.

Upsampler: With $\hat{\boldsymbol{q}}_{\text{lag}}$, we reconstruct the full particle latent representation $\hat{\boldsymbol{h}}_{\text{lag}} \in \mathbb{R}^{N_{\text{lag}} \times d_{\text{latent}}}$ by executing M successive upsampling operations, each coupled with message-passing steps, while integrating a residual link from the downsampling stage.

Decoder: The latent state $\hat{\boldsymbol{h}}_{\text{lag}}$ is passed through a learnable MLP decoder to produce the predicted position, velocity, or acceleration. If the network output is velocity or acceleration, position is computed with Euler integration.

Remark: While NeuralMPM (Rochman-Sharabi et al., 2025) shares a conceptual link with our work in that it also aggregates particles to Eulerian nodes, our work is not simply an extension of this method with additional components. The voxelization approach used in this prior work is limited exclusively to regular grids, and they only perform a single aggregation step at the beginning of the process before scattering back at the end. In contrast, our method is adaptive by utilizing a completely different aggregation strategy that performs distinct particle aggregations at each layer, which we believe significantly enhances the model's expressivity. Furthermore, because our Eulerian representation is defined on a general mesh graph rather than a fixed grid, it naturally extends to arbitrary domains and adapts to complex geometries and deformable domains, a capability we demonstrate in subsequent experiments. By combining the hierarchical, layer-wise aggregation with a cross-attention mechanism, our framework achieves geometric generality and fidelity in a manner entirely separate from existing methods.

5 Experiment

Dataset We use Lagrangian particle-based datasets (Toshev et al., 2024a) solved based on the weakly-compressible Navier–Stokes equations using SPH (Desbrun & Cani, 1996; Hoover, 2006; Monaghan, 1992) with the quintic kernel. The *Taylor–Green Vortex (TGV)* contains a periodic-domain simulation initialized from an analytical solution without any external forcing. The *Lid-Driven Cavity Flow (LDC)* introduces no-slip boundaries, including a moving top lid that drives the internal circulation, creating sharp shear layers and corner vortices. *Reverse Poiseuille Flow (RPF)* features opposing constant body forces in the upper and lower halves of the domain. *Dam Break (DAM)* captures the transient free-surface flow following the sudden release of a water column into an open downstream region, resulting in splashing and wave breaking. The datasets are sampled every 100 steps of the

	Metric	GraphUnet	PhysicsNFP	**Ours**
3D TGV	MSE	4.512×10^{-1}	3.034×10^{0}	$\mathbf{3.649 \times 10^{-1}}_{\uparrow 19.14\%}$
	KE	3.719×10^{0}	1.176×10^{2}	$\mathbf{3.215 \times 10^{0}}_{\uparrow 13.56\%}$
	Sinkhorn	8.799×10^{-2}	7.560×10^{-1}	$\mathbf{1.094 \times 10^{-2}}_{\uparrow 87.56\%}$
3D LDC	MSE	4.810×10^{-3}	2.455×10^{-2}	$\mathbf{3.059 \times 10^{-3}}_{\uparrow 36.40\%}$
	KE	1.464×10^{-4}	1.807×10^{-2}	$\mathbf{1.271 \times 10^{-4}}_{\uparrow 13.17\%}$
	Sinkhorn	4.647×10^{-4}	3.452×10^{-2}	$\mathbf{2.436 \times 10^{-4}}_{\uparrow 47.59\%}$
3D RPF	MSE	2.161×10^{-2}	6.386×10^{-2}	$\mathbf{1.810 \times 10^{-2}}_{\uparrow 16.25\%}$
	KE	1.167×10^{-2}	7.377×10^{-2}	$\mathbf{1.022 \times 10^{-2}}_{\uparrow 12.38\%}$
	Sinkhorn	3.721×10^{-4}	8.551×10^{-3}	$\mathbf{3.018 \times 10^{-4}}_{\uparrow 18.89\%}$

Table 2: Comparison of MSE, Kinetic Energy (KE) error, and sinkhorn divergence across five rollout steps on **3D datasets**.

	Metric	GraphUnet	PhysicsNFP	**Ours**
3600	MSE	5.356×10^{-3}	7.140×10^{-3}	$\mathbf{4.551 \times 10^{-3}}_{\uparrow 15.02\%}$
	KE	1.246×10^{-3}	1.652×10^{-3}	$\mathbf{1.082 \times 10^{-3}}_{\uparrow 13.20\%}$
	Sinkhorn	3.767×10^{-5}	3.323×10^{-5}	$\mathbf{1.422 \times 10^{-5}}_{\uparrow 57.22\%}$
4489	MSE	6.396×10^{-3}	1.623×10^{-2}	$\mathbf{4.572 \times 10^{-3}}_{\uparrow 28.52\%}$
	KE	1.530×10^{-3}	4.895×10^{-3}	$\mathbf{1.087 \times 10^{-3}}_{\uparrow 28.97\%}$
	Sinkhorn	2.229×10^{-5}	1.397×10^{-3}	$\mathbf{1.171 \times 10^{-5}}_{\uparrow 47.47\%}$
10000	MSE	6.798×10^{-3}	1.796×10^{-2}	$\mathbf{5.999 \times 10^{-3}}_{\uparrow 11.75\%}$
	KE	1.697×10^{-3}	6.252×10^{-3}	$\mathbf{1.468 \times 10^{-3}}_{\uparrow 13.47\%}$
	Sinkhorn	2.444×10^{-5}	1.540×10^{-3}	$\mathbf{9.640 \times 10^{-6}}_{\uparrow 60.55\%}$
40000	MSE	6.923×10^{-3}	1.669×10^{-2}	$\mathbf{5.063 \times 10^{-3}}_{\uparrow 26.87\%}$
	KE	1.403×10^{-3}	6.631×10^{-3}	$\mathbf{1.205 \times 10^{-3}}_{\uparrow 14.10\%}$
	Sinkhorn	5.086×10^{-6}	2.183×10^{-3}	$\mathbf{4.295 \times 10^{-6}}_{\uparrow 15.55\%}$

Table 3: Comparison of MSE, Kinetic Energy (KE) error, and Sinkhorn divergence on **higher resolution simulations**.

ground truth solver. Thus, a five-step rollout corresponds to 500 simulation steps. We refer readers to Appendix C for additional details on the datasets.

Task Setup and Baselines To evaluate our model, we benchmark it against a range of strong baselines. We include GNS (Sanchez-Gonzalez et al., 2020) and SEGNN (Brandstetter et al., 2021), the top-performing models reported in LagrangeBench (Toshev et al., 2024a). Graph-Unet (Gao & Ji, 2019) and GraphTransformer (Dwivedi & Bresson, 2021) are also considered due to their wide adoption and proven effectiveness in graph-based learning tasks. In addition, we evaluate against AdvDIFFormer (Wu et al., 2025) and PhysicsNFP (Jiang et al., 2025), which are recently proposed architectures that have achieved competitive results in diverse physical simulation settings. We also include NeuralMPM (Rochman-Sharabi et al., 2025), as it represents the work most closely related to our approach. We refer to Appendix D for detailed training and implementation of all models.

Model performance is evaluated using three metrics. The mean squared error (MSE) measures the accuracy of trajectory predictions. The kinetic energy error (KE) captures global discrepancies in the fluid's energy. Finally, the Sinkhorn divergence quantifies the distance between predicted and reference particle distributions using optimal transport.

All models are configured to directly predict particle positions from unperturbed inputs. We deliberately exclude auxiliary training strategies found in prior literature (Pfaff et al., 2021; Sanchez-Gonzalez et al., 2020) to isolate the specific contributions of our proposed modules.

5.1 Main Results

Table 1 summarizes the quantitative results for the 2D experiments, where our proposed model demonstrates consistently superior predictive accuracy. In particular, our method significantly outperforms NeuralMPM, validating the efficacy of our architectural design choices. The most substantial gains are observed in the Sinkhorn divergence metric, with improvements of 57.22% and 46.68% on the TGV and LDC datasets, respectively. This indicates that our model is far more effective at capturing the underlying distributional geometry and global structure of the data than the baseline methods, rather than merely fitting local features. This success is driven by the downsampler's ability to prioritize high-importance areas, while the Eulerian cross-attention ensures the predictions remain grounded in a consistent global structure.

3D Datasets To further assess the effectiveness of our approach, we compare the proposed model against the two strongest baselines, GraphUnet and PhysicsNFP, on challenging 3D datasets. The results, summarized in Table 2, demonstrate that our model consistently surpasses both baselines across evaluation metrics. This indicates that the advantages of our design generalize beyond 2D settings and remain robust even in the more complex 3D scenarios.

5.2 Higher Resolution and Scalability Simulation

High-resolution simulations are essential for providing a detailed and accurate description of fluid flow, enabling more precise queries of velocity and pressure at arbitrary locations. However, they introduce

Metric	GraphUnet	PhysicsNFP	**Ours**
MSE	5.572×10^{-5}	1.150×10^{-2}	$\mathbf{4.955 \times 10^{-5}}_{\uparrow 11.07\%}$
KE	1.207×10^{-7}	9.560×10^{-3}	$\mathbf{9.980 \times 10^{-8}}_{\uparrow 17.31\%}$
Sinkhorn	1.949×10^{-6}	2.130×10^{-2}	$\mathbf{1.682 \times 10^{-6}}_{\uparrow 13.70\%}$

Table 4: Comparison of MSE, Kinetic Energy (KE) error, and Sinkhorn divergence on **fluid-solid interaction over an irregular domain**.

Metric	GraphUnet	PhysicsNFP	**Ours**
MSE	3.602×10^{-5}	1.379×10^{-4}	$\mathbf{2.421 \times 10^{-5}}_{\uparrow 32.79\%}$
KE	1.469×10^{-5}	7.203×10^{-6}	$\mathbf{4.229 \times 10^{-6}}_{\uparrow 41.29\%}$
Sinkhorn	5.742×10^{-5}	1.646×10^{-4}	$\mathbf{3.107 \times 10^{-5}}_{\uparrow 45.89\%}$

Table 5: Comparison of MSE, Kinetic Energy (KE) error, and Sinkhorn divergence on **deforming domain**.

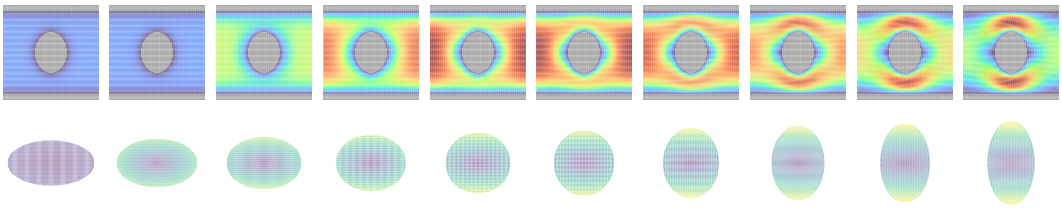


Figure 2: Visualization of datasets with irregular and deforming domains. **Top row**: Periodic flow around a fixed cylinder centered in the domain, creating an irregular flow field. **Bottom row**: A fluid droplet subjected to a velocity field, demonstrating continuous stretching and domain deformation.

two compounding challenges: the inherent difficulty of modeling complex micro-scale details and the computational burden of processing massive particle counts. Our approach is specifically designed to address these distinct hurdles. To manage the large particle counts, our downsampler condenses indistinguishable particles, effectively reducing computational overhead. Simultaneously, our hierarchical framework allows for the separate modeling of micro-scale details and coarse-level large structures. We validate this capability using the *Taylor Green Vortex*, varying particle spacing from 0.02 to 0.005, resulting in 3,600 to 40,000 particles. This experiment serves as a scalability test for both the model's scalability and its ability to resolve intricate flow features. We refer readers to Appendix C.1 for dataset details.

The results are summarized in Table 3, where we compare our proposed model against the two strongest baselines, GraphUnet and PhysicsNFP, introduced in Section 5.1. Our model consistently outperforms both baselines. In particular, PhysicsNFP shows a noticeable drop in performance as the resolution increases, indicating that hierarchical architectures such as GraphUnet and our model are better suited for simulations with larger particle counts.

5.3 Fluid-Solid Interaction Simulation

The voxelization strategy adopted in Rochman-Sharabi et al. (2025) is restricted to fixed, rectangular grids, which prohibits generalization to irregular domains that frequently arise in fluid–solid interaction problems. We visualize the data in Figure 2. To evaluate our model's ability to handle such scenarios, we conduct experiments on the *Flow Around Cylinder* task within an irregular domain, comparing against the two strongest baselines, GraphUnet and PhysicsNFP. We refer readers to Appendix C.2 for dataset details. The results, reported in Table 4, demonstrate that our proposed model consistently outperforms both baselines. These findings highlight not only the effectiveness of our design but also its broader applicability to complex geometries where voxelization-based approaches face fundamental limitations.

5.4 Deformable Domain Simulation

Lagrangian simulations excel at modeling deformations and free-surface boundaries. In addition to the dam break scenario evaluated in Section 5.1, we also test our model on a dynamically deforming domain. Specifically, we simulate a 2D circular drop of fluid that is subjected to a velocity field, stretching it into an ellipse (Monaghan, 1994). Because the fluid is incompressible, its total area remains strictly conserved during this deformation. We provide a visualization in Figure 2.

The experimental results are presented in Table 5. As in previous experiments, we compare our approach against GraphUnet and PhysicsNFP. The results show that our proposed model achieves

superior performance. This is expected, as our architectural design is inherently well-suited for continuously deforming domains.

5.5 Extended Rollout

We investigate our model's performance on extended rollouts using the Water dataset (Sanchez-Gonzalez et al., 2020). This dataset features a volume of water dropped into a box, involving complex splashing dynamics. We compare our model against GraphUnet, as PhysicsNFP is unable to handle the varying particle counts inherent in this dataset. We apply all auxiliary training strategies to both the baseline and our proposed models. These models predict higher-order temporal derivatives and utilize Euler integration to calculate subsequent positions. Additionally, we incorporate noise into the training data to enhance rollout stability. As shown in Table 6, our model maintains superior performance over extended horizons, surpassing the strongest baseline at every step. Notably, the kinetic energy error at the 200-step rollout is orders of magnitude smaller than that of the baseline. We attribute this to the cross-attention with Eulerian features, which helps the model correct particle drift. We present results on three additional materials in Appendix G.

Model	Metric	Steps					
		25	50	75	100	150	200
GraphUnet	MSE	1.801×10^{-4}	1.080×10^{-3}	2.810×10^{-3}	6.234×10^{-3}	2.111×10^{-2}	4.201×10^{-2}
	KE	3.252×10^{-2}	8.987×10^{-1}	6.726×10^{0}	3.114×10^{1}	3.652×10^{2}	1.545×10^{3}
	Sinkhorn	1.225×10^{-4}	4.905×10^{-4}	1.398×10^{-3}	3.966×10^{-3}	1.967×10^{-2}	3.235×10^{-2}
Ours	MSE	$\mathbf{1.612 \times 10^{-5}}$	$\mathbf{8.718 \times 10^{-5}}$	$\mathbf{5.244 \times 10^{-5}}$	$\mathbf{1.895 \times 10^{-3}}$	$\mathbf{9.250 \times 10^{-3}}$	$\mathbf{1.681 \times 10^{-2}}$
	KE	$\mathbf{1.015 \times 10^{-5}}$	$\mathbf{1.237 \times 10^{-4}}$	$\mathbf{3.230 \times 10^{-3}}$	$\mathbf{4.894 \times 10^{-2}}$	$\mathbf{1.212 \times 10^{0}}$	$\mathbf{3.735 \times 10^{0}}$
	Sinkhorn	$\mathbf{1.397 \times 10^{-5}}$	$\mathbf{1.314 \times 10^{-4}}$	$\mathbf{9.414 \times 10^{-4}}$	$\mathbf{3.680 \times 10^{-3}}$	$\mathbf{1.716 \times 10^{-2}}$	$\mathbf{3.042 \times 10^{-2}}$

Table 6: Comparison of MSE, Kinetic Energy (KE) error, and Sinkhorn divergence on the **Water** dataset over 200 rollout steps. We compare against the strongest baseline, GraphUNet. PhysicsNFP is excluded, as it cannot handle a varying number of particles.

6 Additional Experiment

Ablation Study and Model Analysis: We conduct extensive ablation studies to evaluate each component of our model, alongside an analysis of key parameters, in Appendix E.

Additional Baselines on Eulerian Mesh Simulator: Because Eulerian mesh-based neural simulators operate on graph structures analogous to our Lagrangian framework, they are inherently compatible with our setting. Consequently, we compare our approach against a comprehensive suite of state-of-the-art Eulerian mesh models in Appendix F.

Additional Lagrangian Baselines: Appendix I provides a comparison and discussion of additional Lagrangian baselines, demonstrating that our model consistently achieves superior performance.

Additional Experiments on Diverse Materials and Long Rollout: In Appendix G, we demonstrate that our model generalizes well to diverse material domains and achieves strong performance in long rollout.

Computational Complexity: We report the runtime, memory consumption, and model size of the evaluated models in Appendix H. Despite processing both Lagrangian and Eulerian representations, our approach leverages parallelization to maintain highly competitive inference speeds.

Visualizations: Appendix **??** provides visualizations of the model predictions and the downsampled particle graphs.

7 Conclusion

In this work, we propose a **Hybrid Lagrangian–Eulerian** model that unifies the complementary strengths of both representations through particle downsampling, particle aggregation, and cross-attention. Our experiments demonstrate that this hybrid framework achieves higher accuracy.

8 AI USE STATEMENT

In this work, we used generative AI tools for debugging code and for improving the clarity, flow, and grammar of our writing. We have reviewed all AI-assisted work. Specifically, all AI-suggested text revisions were manually reviewed by the authors to ensure they accurately preserved our original scientific meaning and intent, and all AI-assisted code debugging was rigorously tested and verified for correctness in our execution environment. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Oscar Agertz, Ben Moore, Joachim Stadel, Doug Potter, Francesco Miniati, Justin Read, Lucio Mayer, Artur Gawryszczak, Andrey Kravtsov, Ake Nordlund, Frazer Pearce, Vicent Quilis, Douglas Rudd, Volker Springel, James Stone, Elizabeth Tasker, Romain Teyssier, James Wadsley, and Rolf Walder. Fundamental differences between sph and grid methods: Simulating fluids using sph and grid techniques. *Monthly Notices of the Royal Astronomical Society*, 380(3):963–978, August 2007. ISSN 0035-8711. doi: 10.1111/j.1365-2966.2007.12183.x. URL http://dx.doi.org/10.1111/j.1365-2966.2007.12183.x.

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers. *arXiv preprint arXiv:2402.12365*, 2024.

J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, and J.-H. Jacobsen. Invertible residual networks. In *International Conference on Machine Learning*, pp. 573–582. PMLR, 2019.

Jeremiah U. Brackbill and Hans M. Ruppel. FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational Physics*, 65(2):314–343, 1986. doi: 10.1016/0021-9991(86)90211-1.

Johannes Brandstetter, Rob Hesselink, Elise van der Pol, Erik Bekkers, and Max Welling. Geometric and physical quantities improve e(3) equivariant message passing. 2021.

Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural PDE solvers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=vSix3HPYKSU.

Shuhao Cao. Choose a transformer: Fourier or galerkin, 2021. URL https://arxiv.org/abs/2105.14995.

Yadi Cao, Menglei Chai, Minchen Li, and Chenfanfu Jiang. Efficient learning of mesh-based physical simulation with bi-stride multi-scale graph neural network. In *International Conference on Machine Learning*, 2023. URL https://openreview.net/forum?id=2Mbo7IEtZW.

John M. Dawson. Particle simulation of plasmas. *Rev. Mod. Phys.*, 55:403–447, Apr 1983. doi: 10.1103/RevModPhys.55.403. URL https://link.aps.org/doi/10.1103/RevModPhys.55.403.

Walter Dehnen and Hossam Aly. Improving convergence in smoothed particle hydrodynamics simulations without pairing instability: Sph without pairing instability. *Monthly Notices of the Royal Astronomical Society*, 425(2):1068–1082, August 2012. ISSN 0035-8711. doi: 10.1111/j.1365-2966.2012.21439.x. URL http://dx.doi.org/10.1111/j.1365-2966.2012.21439.x.

Mathieu Desbrun and Marie-Paule Cani. Smoothed particles: a new paradigm for animating highly deformable bodies. In *Proceedings of Eurographics Workshop on Computer Animation and Simulation*, Poitiers, France, August 1996.

Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs, 2021. URL https://arxiv.org/abs/2012.09699.

Mohamed Elrefaie, Florin Morar, Angela Dai, and Faez Ahmed. Drivaernet++: A large-scale multimodal car dataset with computational fluid dynamics simulations and deep learning benchmarks. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 499–536. Curran Associates, Inc., 2024.

Mohamed Elrefaie, Angela Dai, and Faez Ahmed. Drivaernet: A parametric car dataset for data-driven aerodynamic design and prediction. *Journal of Mechanical Design*, 147(4), March 2025. ISSN 1528-9001. doi: 10.1115/1.4068104. URL http://dx.doi.org/10.1115/1.4068104.

Robert D. Richtmyer Francis Harvey Harlow, Martha Evans. A machine calculation method for hydrodynamic problems. Report, Los Alamos Scientific Laboratory of the University of California, 1955.

Hongyang Gao and Shuiwang Ji. Graph u-nets. In *International Conference on Machine Learning*, pp. 2083–2092, 2019.

R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics: Theory and application to non-spherical stars. *Mon. Not. Roy. Astron. Soc.*, 181:375, 1977.

Daniel Greenfeld, Meirav Galun, Ronen Basri, Irad Yavneh, and Ron Kimmel. Learning to optimize multigrid PDE solvers. In *International Conference on Machine Learning*, pp. 2415–2423. PMLR, 2019.

XU HAN, Han Gao, Tobias Pfaff, Jian-Xun Wang, and Liping Liu. Predicting physics in mesh-reduced space with temporal attention. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=XctLdNfCmP.

Zhongkai Hao, Chengyang Ying, Zhengyi Wang, Hang Su, Yinpeng Dong, Songming Liu, Ze Cheng, Jun Zhu, and Jian Song. Gnot: A general neural operator transformer for operator learning. *arXiv preprint arXiv:2302.14376*, 2023.

Zhongkai Hao, Chang Su, Songming Liu, Julius Berner, Chengyang Ying, Hang Su, Anima Anandkumar, Jian Song, and Jun Zhu. Dpot: Auto-regressive denoising operator transformer for large-scale pde pre-training. *arXiv preprint arXiv:2403.03542*, 2024.

Xiaodong He, Yinan Wang, and Juan Li. Flow completion network: Inferring the fluid dynamics from incomplete flow information using graph neural networks. *Physics of Fluids*, 34(8), August 2022. ISSN 1089-7666. doi: 10.1063/5.0097688. URL http://dx.doi.org/10.1063/5.0097688.

Maximilian Herde, Bogdan Raonic, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bezenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for PDEs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=JC1VKK3UXk.

William Graham Hoover. *Smooth Particle Applied Mechanics: The State of the Art*. World Scientific, Singapore, 2006.

Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. Diffusionpde: Generative pde-solving under partial observation, 2024. URL https://arxiv.org/abs/2406.17763.

Ethan Ji, Yuanzhou Chen, Arush Ramteke, Fang Sun, Tianrun Yu, Jai Parera, Wei Wang, and Yizhou Sun. Corgi: Gnns with convolutional residual global interactions for lagrangian simulation, 2025. URL https://arxiv.org/abs/2511.22938.

Haoyang Jiang, Jindong Wang, Xingquan Zhu, and Yi He. Topology-aware neural flux prediction guided by physics. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=8Um2YotdbD.

Gang Jing, Huan Wang, Xianting Li, Guijin Wang, and Yingying Yang. An airflow velocity field reconstruction method with sparse or incomplete data using physics-informed neural network. *Journal of Building Engineering*, 88:109231, July 2024. Published: 1 July 2024.

Norman L. Johnson. The legacy and future of CFD at Los Alamos. In *Proceedings of the 1996 Canadian CFD Conference*. Office of Scientific and Technical Information (OSTI), 1996. OSTI 244662.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

J. Kruse, L. Ardizzone, C. Rother, and U. Köthe. Benchmarking invertible architectures on inverse problems. *arXiv preprint*, 2021.

Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *Proceedings of the 36th International Conference on Machine Learning*, 09–15 Jun 2019.

Ruoyan Li, Guancheng Wan, Zijie Huang, Zixiao Liu, Haixin Wang, Xiao Luo, Wei Wang, and Yizhou Sun. Flow field reconstruction with sensor placement policy learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=1esFEjMUBS.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations, 2020. URL https://arxiv.org/abs/2003.03485.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Zongyi Li, Nikola Borislavov Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Prakash Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3d PDEs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=86dXbqT5Ua.

Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations, 2023b. URL https://arxiv.org/abs/2111.03794.

Mario Lino, Chris D. Cantwell, Anil Anthony Bharath, and Stathi Fotiadis. Simulating continuum mechanics with multi-scale graph neural networks. *ArXiv*, abs/2106.04900, 2021. URL https://api.semanticscholar.org/CorpusID:235376839.

B. List, L.-W. Chen, and N. Thuerey. Learned turbulence modelling with differentiable fluid solvers: physics-based loss functions and optimisation horizons. *Journal of Fluid Mechanics*, 949:A25, 2022.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for PDEs on million-scale geometries. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=AM7iAh0krx.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Holden Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=DKSI3bULiZ.

Yaxin Mo and Luca Magri. Reconstructing unsteady flows from sparse, noisy measurements with a physics-constrained convolutional neural network, 2024. URL https://arxiv.org/abs/2409.00260.

J.J. Monaghan. Simulating free surface flows with sph. *Journal of Computational Physics*, 110(2): 399–406, 1994. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.1994.1034. URL https://www.sciencedirect.com/science/article/pii/S0021999184710345.

Joseph J. Monaghan. Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 30:543–574, 1992. doi: 10.1146/annurev.aa.30.090192.002551.

John A. Nairn. Material point method calculations with explicit cracks. *Computer Modeling in Engineering & Sciences*, 4(6):649–664, 2003. doi: 10.3970/cmes.2003.004.649.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. URL https://arxiv.org/abs/1912.01703.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. 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 *Conference on Neural Information Processing Systems*, 2022a.

Lukas Prantl, Benjamin Ummenhofer, Vladlen Koltun, and Nils Thuerey. Guaranteed conservation of momentum for learning particle-based fluid dynamics. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022b. URL https://openreview.net/forum?id=6niwHlzh10U.

Daniel J. Price. Smoothed particle hydrodynamics and magnetohydrodynamics. *Journal of Computational Physics*, 231(3):759–794, February 2012. ISSN 0021-9991. doi: 10.1016/j.jcp.2010.12.011. URL http://dx.doi.org/10.1016/j.jcp.2010.12.011.

Prabhu Ramachandran, Aditya Bhosale, Kunal Puri, Pawan Negi, Abhinav Muta, A. Dinesh, Dileep Menon, Rahul Govind, Suraj Sanka, Amal S. Sebastian, Ananyo Sen, Rohan Kaushik, Anshuman Kumar, Vikas Kurapati, Mrinalgouda Patil, Deep Tavker, Pankaj Pandey, Chandrashekhar Kaushik, Arkopal Dutt, and Arpit Agarwal. PySPH: A Python-based Framework for Smoothed Particle Hydrodynamics. *ACM Transactions on Mathematical Software*, 47(4):1–38, December 2021. ISSN 0098-3500, 1557-7295. doi: 10.1145/3460773.

Omer Rochman-Sharabi, Sacha Lewin, and Gilles Louppe. A neural material point method for particle-based emulation. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=zSK81A2hxQ.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks, 2020. URL https://arxiv.org/abs/2002.09405.

J. Sappl, L. Seiler, M. Harders, and W. Rauch. Deep learning of preconditioners for conjugate gradient solvers in urban water related problems. *arXiv preprint*, 2019.

Junhong Shen, Tanya Marwah, and Ameet Talwalkar. UPS: Efficiently building foundation models for PDE solving via cross-modal adaptation. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=0r9mhjRv1E.

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

Z. Sun, Y. Yang, and S. Yoo. A neural PDE solver with temporal stencil modeling. *arXiv preprint*, 2023.

Maria Luisa Taccari, Jonathan Nuttall, Xiaohui Chen, He Wang, Bennie Minnema, and Peter K. Jimack. Attention u-net as a surrogate model for groundwater prediction. *Advances in Water Resources*, 163:104169, 2022. ISSN 0309-1708. doi: https://doi.org/10.1016/j.advwatres.2022.104169. URL https://www.sciencedirect.com/science/article/pii/S0309170822000458.

Y. Teng and A. Choromanska. Invertible autoencoder for domain adaptation. *Computation*, 7(2):20, 2019.

Artur Toshev, Gianluca Galletti, Fabian Fritz, Stefan Adami, and Nikolaus Adams. Lagrangebench: A lagrangian fluid mechanics benchmarking suite. *Advances in Neural Information Processing Systems*, 36, 2024a.

Artur P. Toshev, Gianluca Galletti, Johannes Brandstetter, Stefan Adami, and Nikolaus A. Adams. Learning lagrangian fluid mechanics with e(3)-equivariant graph neural networks, 2023.

Artur P. Toshev, Jonas A. Erbesdobler, Nikolaus A. Adams, and Johannes Brandstetter. Neural sph: Improved neural modeling of lagrangian fluid dynamics, 2024b. URL https://arxiv.org/abs/2402.06275.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tmIiMPl4IPa.

David Tskhakaya. Chapter 6: The particle-in-cell method. In Holger Fehske, Ralf Schneider, and Alexander Weiße (eds.), *Computational Many-Particle Physics*, volume 739 of *Lecture Notes in Physics*, pp. 271–?,. Springer, Berlin, Heidelberg, 2008. ISBN 978-3-540-74685-0. doi: 10.1007/978-3-540-74686-7.

Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=B1lDoJSYDH.

Hrishikesh Viswanath, Yue Chang, Julius Berner, Peter Yichen Chen, and Aniket Bera. Reduced-order neural operators: Learning lagrangian dynamics on highly sparse graphs. *arXiv preprint arXiv:2407.03925*, 2024.

Haixin Wang, Yadi Cao, Zijie Huang, Yuxuan Liu, Peiyan Hu, Xiao Luo, Zezheng Song, Wanjia Zhao, Jilin Liu, Jinan Sun, Shikun Zhang, Long Wei, Yue Wang, Tailin Wu, Zhi-Ming Ma, and Yizhou Sun. Recent advances on machine learning for computational fluid dynamics: A survey, 2024. URL https://arxiv.org/abs/2408.12171.

Haixin Wang, Ruoyan Li, Fred Xu, Fang Sun, Kaiqiao Han, Zijie Huang, Guancheng Wan, Ching Chang, Xiao Luo, Wei Wang, and Yizhou Sun. Fd-bench: A modular and fair benchmark for data-driven fluid simulation, 2025. URL https://arxiv.org/abs/2505.20349.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems, 2024. URL https://arxiv.org/abs/2406.03923.

Christopher Williams, Fabian Falck, George Deligiannidis, Christopher C. Holmes, Arnaud Doucet, and Saifuddin Syed. A unified framework for u-net design and analysis. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=43ruO2fMjq.

Haixu Wu, Tengge Hu, Huakun Luo, Jianmin Wang, and Mingsheng Long. Solving high-dimensional pdes with latent spectral models. In *International Conference on Machine Learning*, 2023.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. In *International Conference on Machine Learning*, 2024.

Qitian Wu, Chenxiao Yang, Kaipeng Zeng, and Michael M. Bronstein. Supercharging graph transformers with advective diffusion. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=MaOYl3P84E.

Jingxuan Xu, Hong Huang, Chuhang Zou, Manolis Savva, Yunchao Wei, and Wuyang Chen. Hybrid neural-mpm for interactive fluid simulations in real-time, 2025. URL https://arxiv.org/abs/2505.18926.

Vikas Yadav, Mario Casel, and Abdulla Ghani. Rf-pinns: Reactive flow physics-informed neural networks for field reconstruction of laminar and turbulent flames using sparse data. *Journal of Computational Physics*, 524:113698, March 2025. Published: 1 March 2025.

Zhanhong Ye, Xiang Huang, Leheng Chen, Hongsheng Liu, Zidong Wang, and Bin Dong. PDEformer: Towards a foundation model for one-dimensional partial differential equations. In *ICLR 2024 Workshop on AI4DifferentialEquations In Science*, 2024. URL https://openreview.net/forum?id=GLDMCwdhTK.

Y. Zhong, K. Fukami, B. An, et al. Sparse sensor reconstruction of vortex-impinged airfoil wake with machine learning. *Theor. Comput. Fluid Dyn.*, 37:269–287, 2023. doi: 10.1007/s00162-023-00657-y.

Hang Zhou, Yuezhou Ma, Haixu Wu, Haowen Wang, and Mingsheng Long. Unisolver: Pde-conditional transformers towards universal neural pde solvers. In *Forty-second International Conference on Machine Learning*.

A Impact Statement

Our neural simulator has the potential to transform computational fluid dynamics (CFD) and the broader sciences by enabling faster, more scalable, and more adaptable modeling tools. The development of approaches that combine data-driven learning with physically grounded representations opens new opportunities for fast simulations in areas such as engineering design, climate modeling, and multiphase flow analysis. By improving the efficiency and robustness of CFD, these methods can accelerate scientific discovery and expand access to large-scale simulations that were previously computationally prohibitive.

We do not identify any adverse societal or ethical impacts associated with this paper.

B Limitations and Future Work

While this framework significantly enhances the rollout performance of Lagrangian neural simulators, a primary limitation remains the divergence of trajectories over extreme temporal horizons, such as those spanning 100 physical hours. This drift is not unique to our architecture. Rather, it stems from the irreducible local truncation errors inherent in both neural simulators and numerical solvers. As these microscopic errors compound over millions of integration steps, the global truncation error eventually accumulates, leading to a potential breakdown in physical fidelity.

This phenomenon represents a fundamental, systemic challenge in numerical analysis and the simulation of dynamical systems. Because error accumulation is an intrinsic property of discrete solvers, achieving stability over infinite horizons remains an open problem for the community. Future research could investigate specialized neural architectures that explicitly regularize long-term stability or incorporate error-correction manifolds. By moving beyond the imitation of classical solvers, there is a distinct opportunity to develop neural simulators that eventually surpass the stability and accuracy of traditional numerical methods. Resolving these foundational issues regarding long-term numerical stability will require a sustained, collective effort from the broader physics-ML community. Fully addressing these systemic error accumulation patterns is beyond the scope of the current work.

C Datasets

We provide a brief description of the main benchmark dataset used in this work and refer the readers to Toshev et al. (2024a) for additional details.

Taylor Green Vortex The dataset represents a decaying flow system characterized by periodic boundary conditions and a specific initial velocity field that leads to kinetic energy decay through

viscous interactions. Dataset generation employs a Lagrangian SPH scheme, solving the weakly-compressible Navier-Stokes equations coupled with a barotropic equation of state. The solver setup involves randomly drawing particle positions, which are then relaxed under periodic boundary conditions via 1000 steps of SPH relaxation to create the initial state. The 2D TGV case is defined on a 1×1 spatial domain with 2,500 particles at a Reynolds number of 100, using a particle spacing (Δx) of 20×10^{-3} and a time step (Δt) of 40×10^{-3}. Conversely, the 3D TGV case utilizes a $2\pi \times 2\pi \times 2\pi$ domain populated by 8,000 particles at a Reynolds number of 50, with a Δx of 314.16×10^{-3} and a Δt of 500×10^{-3}. The 2D version follows an analytical solution of exponentially decaying velocity.

Lid-Driven Cavity Flow The dataset simulates a flow within a confined domain driven by a moving top wall. This dataset is generated using the Lagrangian SPH solver, which solves the weakly-compressible Navier-Stokes equations. The ground truth setup initializes the fluid velocity to zero and runs the simulation until the system reaches a statistically stationary equilibrium state, at which point data collection begins. While the solver uses a generalized wall boundary condition with multiple layers of dummy particles to enforce impermeability and no-slip conditions, the dataset retains only the innermost layer. In the 2D case, the simulation comprises 2,708 particles within a 1.12×1.12 spatial domain at a Reynolds number of 100, utilizing a particle spacing (Δx) of 20×10^{-3} and a time step (Δt) of 40×10^{-3}55. The 3D case involves 8,160 particles in a $1.25 \times 1.25 \times 0.5$ domain, also at a Reynolds number of 100, with a Δx of 41.667×10^{-3} and a Δt of 90×10^{-3}. Notably, the 3D version applies periodic boundary conditions in the z-direction to recover the dynamics of the 2D solution.

Reverse Poiseuille Flow The dataset simulates a fully periodic flow driven by a spatially varying external force field. This dataset is generated using the Lagrangian SPH solver, which solves the weakly-compressible Navier-Stokes equations. The ground truth setup initializes with zero fluid velocity and applies a force of magnitude 1 in the lower half of the domain and -1 in the upper half, running the simulation until it reaches a statistically stationary equilibrium state before data collection begins. In the 2D case, the system consists of 3,200 particles within a 1×2 spatial domain at a Reynolds number of 10, utilizing a particle spacing (Δx) of 25×10^{-3} and a time step (Δt) of 40×10^{-3}. The 3D counterpart involves 8,000 particles in a $1 \times 2 \times 0.5$ domain, also at a Reynolds number of 10, with a Δx of 50×10^{-3} and a Δt of 100×10^{-3}5. Although theoretically similar to laminar channel flow, this specific setup at $Re = 10$ is not fully laminar and exhibits mixing, making the dynamics more diverse for learning tasks.

Dam Break The dataset is designed to benchmark the simulation of free-surface flows involving significant topological changes. Generated using the Lagrangian SPH solver to solve the weakly-compressible Navier-Stokes equations, this dataset distinctly calculates density via evolution equations rather than summation to accurately account for the lack of full kernel support at the free surface. The ground truth setup initializes the system by randomly drawing particle positions and letting them relax under non-periodic boundary conditions for 1000 steps. To manage the solid boundaries, the solver employs generalized wall boundary conditions with dummy particles to enforce no-slip and impermeability, though only the innermost particle layer is retained in the final dataset. The 2D simulation is defined on a spatial domain of 5.486×2.12 and contains 5,740 particles. While this problem is typically modeled as inviscid, this specific implementation includes a small amount of physical viscosity to minimize wall artifacts, resulting in a Reynolds number of 40,000, with a particle spacing (Δx) of 20×10^{-3} and a time step (Δt) of 30×10^{-3}.

C.1 Higher Resolution Simulation Data

To generate higher-resolution simulation data for the Taylor-Green Vortex, we systematically reduce the initial particle spacing, Δx, starting from 20×10^{-3} and progressively decreasing it to 15×10^{-3}, 10×10^{-3}, and 5×10^{-3}. These configurations yield simulations with increasing particle counts of 3,600, 4,489, 10,000, and 40,000, respectively. Reducing the inter-particle spacing enhances the spatial resolution of the Lagrangian discretization, which is critical for numerical accuracy. A denser particle distribution enables more precise approximation of spatial derivatives, resulting in more accurate gradient computations and improved fidelity when querying field variables such as velocity and pressure. Furthermore, this setup serves as a scalability experiment for the neural surrogate models. The significant increase in particle number challenges the model to generalize to larger, more computationally intensive graph structures while maintaining predictive accuracy.

C.2 Fluid-Solid Interaction Data

The simulation takes place in a rectangular fluid domain with physical dimensions of 18.0×18.0 units, extending from $x = 0$ to 18.0 and $y = -9.0$ to 9.0. A cylinder with a diameter of 1.2 units is positioned at coordinates $(6.0, 0.0)$ to obstruct the flow. The simulation uses a fixed time step (dt) of approximately 0.0027 seconds, calculated initially based on the minimum of CFL and viscous stability criteria, without utilizing adaptive timestepping. The system saves the simulation state every 10 time steps, effectively capturing the dynamics of the flow at a Reynolds number of 200. The dataset is generated using Ramachandran et al. (2021).

C.3 Deformable Domain Data

The data is generated by simulating the Navier-Stokes equations for an incompressible fluid, modeling the evolution of a circular patch of fluid deformed into an ellipse by an initial linear velocity field. This process perfectly conserves the area of the fluid. The simulation utilizes approximately 5,026 particles at standard resolution and dynamically calculates an adaptive time step of roughly $\Delta t = 5.27 \times 10^{-6}$ seconds using a CFL constraint. We generate 10000 frames for training, 1000 for validation, and 1000 for testing. The dataset is generated using Ramachandran et al. (2021).

D Model Training and Implementation Details

All models are implemented using the PyTorch framework (Paszke et al., 2019) and trained with the Adam optimizer (Kingma & Ba, 2017). All models are configured to directly predict particle positions from unperturbed inputs. We deliberately exclude these auxiliary training strategies found in prior literature (Pfaff et al., 2021; Sanchez-Gonzalez et al., 2020) to isolate the specific contributions of our proposed modules.

GNS The model employs a message-passing GNN architecture (Sanchez-Gonzalez et al., 2020), adhering to the hyperparameter configuration detailed in Toshev et al. (2024a). Both the encoder and decoder are composed of 4-layer MLPs with ReLU activations, while the processor utilizes 10 message-passing layers with a latent dimension of 128.

SEGNN We implement the geometric architecture proposed by Brandstetter et al. (2021), adopting the hyperparameter guidelines from Toshev et al. (2024a). The network comprises 10 message-passing layers with a latent dimension of 64, using 2-layer MLP blocks throughout. We restrict the maximum spherical harmonic degree (L_{max}) to 1 for both hidden representations and attributes. The input features are treated as isotropic.

GraphUnet We adopt the hierarchical architecture proposed by Gao & Ji (2019), which features three downsampling and upsampling stages with a pooling ratio of 0.6. The encoder and decoder modules are composed of 4-layer MLPs with ReLU activations, mapping to a latent dimension of 128. At the coarsest resolution (the bottom level), the processor employs a stack of 4 message-passing GNN layers.

GraphTransformer We adopt the model architecture proposed in Dwivedi & Bresson (2021). We use 6 layers with a latent dimension of 128. The encoder and decoder modules are composed of 4-layer MLPs with ReLU activations.

AdvDIFFormer "We adopt the model architecture proposed in Wu et al. (2025), configured with a hidden channel size of 128. We set the approximation order K to 1 and utilize 2 attention heads.

NeuralMPM We adopt the architecture proposed by Rochman-Sharabi et al. (2025), which operates on a fixed 32×32 Eulerian grid. The central processor employs a U-Net structure with a base channel dimension of 64.

PhysicsNFP We adopt the topology-aware architecture proposed by Jiang et al. (2025). The model comprises 10 layers with a hidden channel dimension of 128.

Ours Our architecture incorporates $M = 2$ downsampling and upsampling stages with a ratio of 0.8. The model operates with a latent dimension of 128 and utilizes a central processor comprising $L = 4$ steps. Both the encoder and decoder modules are constructed as 4-layer MLPs with ReLU activations. The model is trained using a learning rate schedule that decays from 10^{-4} to 10^{-6}.

E Model Analysis

We conduct extensive analysis on the *Lid-Driven Cavity Flow* dataset to isolate the contribution of each module.

Ablation Study on Downsampler: We perform an ablation study to evaluate the effect of the downsampler and upsampler. Results are reported in Figure 3. Across all metrics, the model equipped with these components consistently outperforms the variant without them over the five-step rollout. This observation aligns with our design rationale. In the Lid-Driven Cavity flow dataset, many particles in the lower-left region remain nearly stationary, while significant dynamics occur elsewhere. The downsampler and upsampler encourage the model to focus its capacity on regions with strong flow variation. As a result, the model with these modules achieves notably lower MSE and kinetic energy error. We attribute this to the difference in the processor, as the processor of the ablated model must explicitly model interactions among all particles, whereas our hierarchical framework delegates the coarse forward PDE dynamics to the processor and leaves the fine-scale refinements to the downsampler and upsampler.

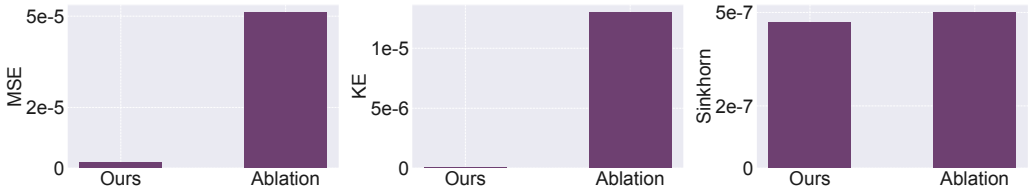


Figure 3: Five-step rollout results of the ablation study on the downsampler and upsampler, reported on the Lid-Driven Cavity flow dataset. Ours denotes the proposed model, while Ablation refers to the variant without the downsampler and upsampler.

Ablation Study on Combined Self–Cross Attention Feature: We conduct an ablation study to examine the role of the combined self–cross attention feature. Specifically, we remove the cross-attention between Lagrangian and Eulerian features and retain only the Lagrangian self-attention, replacing Equation 6 with Equation 7. Results are shown in Figure 4.

$$\boldsymbol{F} = \boldsymbol{A}, \quad \hat{\boldsymbol{q}}_{\text{lag}} = \boldsymbol{W}\,\boldsymbol{F} \in \mathbb{R}^{\tilde{N}_{\text{lag}} \times d_{\text{latent}}}. \tag{7}$$

The model that integrates both Lagrangian self-attention and Lagrangian–Eulerian cross-attention consistently outperforms the ablated variant. This highlights the importance of cross-attention in enhancing overall performance by leveraging information from both representations.

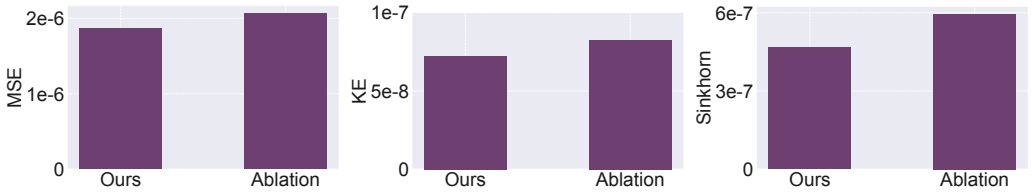


Figure 4: Five-step rollout results of the ablation study on the combined self-cross attention feature, reported on the Lid-Driven Cavity flow dataset. Ours denotes the proposed model, while Ablation refers to the variant without cross-attention.

Ablation Study on Aggregating Lagrangian Nodes to Eulerian Nodes: We also examine the role of Eulerian features in defining aggregation points for Lagrangian features. In the full model, downsampled Eulerian nodes serve as anchors for aggregating Lagrangian representations. To assess their contribution, we design an ablation study where these nodes are replaced by a learnable alternative: an MLP that takes the positions of downsampled Lagrangian particles as input and outputs logits indicating the assignment of each particle to aggregation points. The number of aggregation points is kept equal to the number of downsampled Eulerian nodes, and the logits determine the contribution of each particle to its assigned aggregation node. Results are reported in Figure 5.

The full model that aggregates Lagrangian features onto downsampled Eulerian nodes consistently achieves stronger performance across all metrics. The full model that aggregates Lagrangian features onto downsampled Eulerian nodes consistently achieves stronger performance across all metrics. We attribute these gains to two factors. First, the downsampled Eulerian nodes naturally share a spatial distribution similar to that of the downsampled Lagrangian particles. Aggregating Lagrangian features onto this Eulerian structure produces representations that remain dynamically consistent with the Eulerian features, which is particularly beneficial during cross-attention, where the Eulerian branch provides a global perspective while the Lagrangian branch supplies fine-scale details. Second, the MLP-based aggregation points discard explicit spatial structure, weakening the self-attention's ability to resolve PDE dynamics at a coarse level and diminishing the effectiveness of cross-attention in aligning complementary representations.

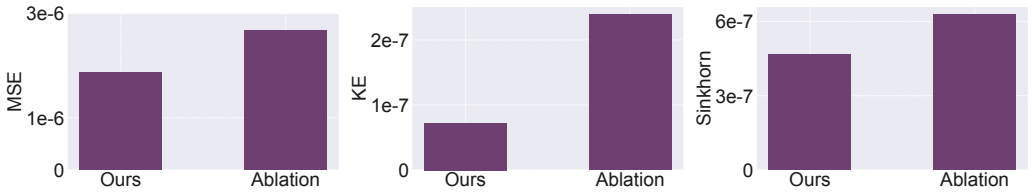


Figure 5: Five-step rollout results of the ablation study on the role of Eulerian feature in Lagrangian aggregation, reported on the Lid-Driven Cavity flow dataset. Ours denotes the proposed model, while Ablation refers to the variant that uses an MLP to determine aggregation points.

Analysis on Eulerian Discretization: The Eulerian features are represented on a discretized grid with N_{eu} nodes. We investigate how the discretization resolution affects model performance using the Lid-Driven Cavity flow dataset, defined on a bounded 1.12×1.12 domain. Each direction is discretized with s nodes, and the results are shown in Figure 6. Among the tested resolutions, $s = 32$ yields the best performance. Extremely coarse grids such as $s = 8$, or overly fine grids such as $s = 64$, perform the worst. With too few Eulerian nodes, the representation lacks sufficient information about the global fluid field. In addition, aggregating Lagrangian features onto such a small set of nodes leads to significant information loss, as particles from diverse locations and flow behaviors are forced to map to the same averaged values. Overly fine grids fail because the Eulerian representation loses the smoothing and global-anchoring role it is meant to play, approaching the particle resolution it was introduced to complement. This yields a practical selection rule rather than a per-problem search. The Eulerian resolution should be set at the scale a conventional neural PDE solver would use for the same domain. $s = 32$ on a 1.12×1.12 domain corresponds to the standard discretization for neural operators on this smooth, low-frequency Navier-Stokes dataset with 100 Reynold number. Additionally, performance is flat across the broad middle of the sweep, so our model is robust in terms of this hyperparameter.

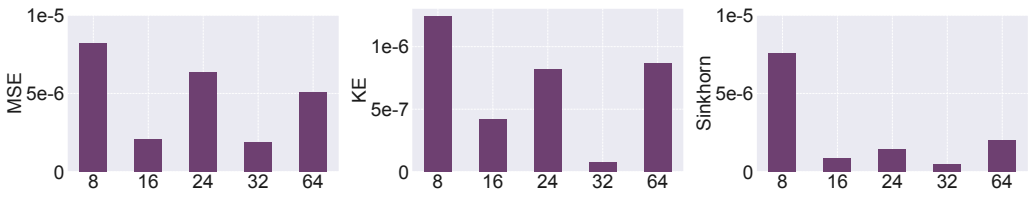


Figure 6: Five-step rollout results of varying the discretization resolution of Eulerian nodes, reported on the Lid-Driven Cavity flow dataset.

Analysis on Downsampling Ratio: We study the impact of the downsampling ratio, λ, on model performance, with results shown in Figure 7. The best accuracy is obtained at $\lambda = 0.8$, while more aggressive choices such as $\lambda = 0.2$ or $\lambda = 0.4$ lead to a sharp decline in performance. This behavior is expected, as the purpose of downsampling is to merge particles that move coherently or remain nearly static, since these contribute little additional information when considered separately. However, if the ratio is too aggressive, the model also discards particles carrying essential information about the fluid dynamics, which in turn degrades the solution.

Selecting the downsampling ratio is not merely a matter of hyperparameter tuning, but is instead guided by physical intuition and dataset characteristics. First, our empirical results suggest that mild retention rates generally work well. The failure mode is one-sided and interpretable. Performance remains remarkably stable under mild pruning and only collapses under aggressive settings. This yields a conservative, practical selection rule. Second, the appropriate ratio is fundamentally governed by how much kinematic redundancy a specific flow contains. Consequently, a prior physical understanding of the dataset can directly inform the optimal ratio.

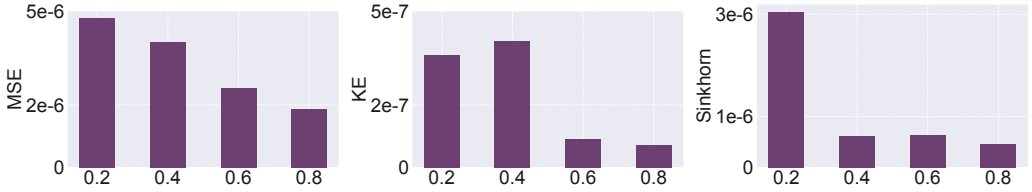


Figure 7: Five-step rollout results of varying the downsampling ratio, reported on the Lid-Driven Cavity flow dataset.

Analysis on Number of Downsampling Layers: We analyze how the number of downsampling layers, M, influences model performance, with results shown in Figure 8. The best results are obtained with $M = 2$, followed by $M = 1$, while deeper configurations with $M = 3$ or $M = 4$ lead to a significant drop in performance. These findings are consistent with the results from the study on the downsampling ratio. Increasing the number of downsampling layers progressively reduces the particle set, which can cause the loss of particles carrying critical information about the fluid dynamics. Interestingly, two layers outperform a single layer. We attribute this to the hierarchical structure. Applying downsampling twice enables the model to better identify and preserve important particles, while additional rounds of message passing improve the transfer of information from distant particles to the aggregation nodes.

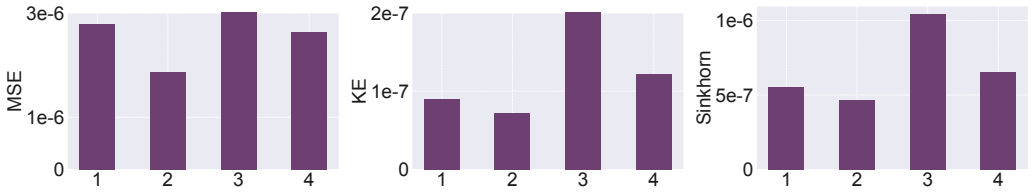


Figure 8: Five-step rollout results of varying the number of downsampling layers, reported on the Lid-Driven Cavity flow dataset.

F Additional Baselines on Eulerian Mesh Simulator

Since Eulerian mesh-based neural simulators operate on graph structures analogous to our Lagrangian framework, they are inherently compatible with our setting. Consequently, we evaluate our approach against a broad suite of state-of-the-art Eulerian mesh models, including the Graph Kernel Operator (GKO) (Li et al., 2020), General Neural Operator Transformer (GNOT) Hao et al. (2023), Directional Transport-Aware Graph Neural Network (DTA-GNN) (Li et al., 2025), Graph Interaction Operator for Reduced-Order Modeling (GIOROM) (Viswanath et al., 2024), Geometry Informed Neural Operator (GINO) (Li et al., 2023a), Transolver (Wu et al., 2024), and Transolver++ (Luo et al., 2025). The experiments are conducted on the *Lid-Driven Cavity Flow* dataset to assess performance.

While several baselines incorporate aggregation or downsampling techniques, our method introduces fundamental improvements in spatial awareness and domain flexibility. For instance, GINO projects particles onto fixed grids to employ Fourier Neural Operators (FNO), a reliance that renders it inapplicable to irregular domains where our method excels. Furthermore, while Transolver and Transolver++ aggregate points into physical tokens based on latent states, our approach leverages physical coordinates to explicitly preserve the underlying spatial structure. Similarly, although GIOROM generates sparse representations of the input graph, it relies on static heuristics such as

Metric	GKO	GNOT	DTA-GNN	GIOROM	GINO	Transolver	Transolver++	**Ours**
MSE	5.205×10^{-4}	1.011×10^{-5}	3.505×10^{-6}	8.335×10^{-6}	3.601×10^{-4}	1.308×10^{-5}	2.665×10^{-5}	$\mathbf{1.868 \times 10^{-6}}$
KE	1.161×10^{-4}	9.113×10^{-7}	5.127×10^{-7}	1.765×10^{-6}	8.656×10^{-5}	1.376×10^{-6}	1.417×10^{-5}	$\mathbf{7.240 \times 10^{-8}}$
Sinkhorn	6.609×10^{-6}	6.470×10^{-7}	1.560×10^{-6}	1.245×10^{-6}	7.321×10^{-6}	2.001×10^{-6}	2.693×10^{-6}	$\mathbf{4.675 \times 10^{-7}}$

Table 7: Comparison of MSE, Kinetic Energy (KE) error, and Sinkhorn divergence on Lid-Driven Cavity flow dataset. The relative performance gain against the strongest baseline is computed as $\frac{\text{Baseline}-\text{Ours}}{\text{Baseline}} \times 100\%$.

random sampling or farthest-point selection. In contrast, our downsampler dynamically identifies and selects salient features based on the data, ensuring the preservation of critical information and the production of more meaningful sparse representations.

The results are summarized in Table 7. We observe that our proposed model achieves significantly superior performance compared to the Eulerian mesh baseline simulators. This highlights the effectiveness of our design choices, particularly regarding sparse representation construction and particle aggregation techniques.

G Additional Experiments on Diverse Materials

Although this work focuses primarily on Lagrangian fluid simulation, we also evaluate our model's performance on diverse materials, including sand, goop, and multi-material interactions. Using datasets from Sanchez-Gonzalez et al. (2020), we report performance across 200 rollout steps. The results, summarized in Table 8, demonstrate that our model generalizes effectively to these alternative materials. We exclude PhysicsNFP from this comparison as it is unable to accommodate varying particle counts.

Data	**Model**	**Metric**	**Steps** 25	50	75	100	150	200
Goop	GraphUnet	MSE	1.668×10^{-4}	1.616×10^{-3}	5.883×10^{-3}	2.173×10^{-2}	2.421×10^{-2}	3.240×10^{-2}
		KE	6.536×10^{-3}	2.414×10^{-1}	8.072×10^{0}	4.850×10^{1}	3.271×10^{1}	7.792×10^{1}
		Sinkhorn	3.095×10^{-4}	3.849×10^{-3}	9.788×10^{-3}	3.947×10^{-2}	3.078×10^{-2}	3.099×10^{-2}
	Ours	MSE	$\mathbf{1.321 \times 10^{-4}}$	$\mathbf{9.191 \times 10^{-4}}$	$\mathbf{3.458 \times 10^{-3}}$	$\mathbf{9.409 \times 10^{-3}}$	$\mathbf{9.422 \times 10^{-3}}$	$\mathbf{1.367 \times 10^{-2}}$
		KE	$\mathbf{3.784 \times 10^{-3}}$	$\mathbf{2.153 \times 10^{-1}}$	$\mathbf{4.183 \times 10^{0}}$	$\mathbf{3.370 \times 10^{1}}$	$\mathbf{2.775 \times 10^{1}}$	$\mathbf{4.773 \times 10^{1}}$
		Sinkhorn	$\mathbf{2.599 \times 10^{-4}}$	$\mathbf{1.847 \times 10^{-3}}$	$\mathbf{6.752 \times 10^{-3}}$	$\mathbf{1.838 \times 10^{-2}}$	$\mathbf{9.632 \times 10^{-3}}$	$\mathbf{1.938 \times 10^{-2}}$
Sand	GraphUnet	MSE	5.578×10^{-5}	3.152×10^{-3}	1.378×10^{-2}	2.729×10^{-2}	4.462×10^{-2}	5.060×10^{-2}
		KE	7.982×10^{-4}	1.145×10^{0}	2.373×10^{1}	9.424×10^{1}	2.675×10^{2}	3.590×10^{2}
		Sinkhorn	9.766×10^{-5}	5.228×10^{-3}	2.567×10^{-2}	5.222×10^{-2}	8.543×10^{-2}	9.783×10^{-2}
	Ours	MSE	$\mathbf{4.545 \times 10^{-5}}$	$\mathbf{2.913 \times 10^{-3}}$	$\mathbf{1.346 \times 10^{-2}}$	$\mathbf{2.429 \times 10^{-2}}$	$\mathbf{2.348 \times 10^{-2}}$	$\mathbf{2.05 \times 10^{-2}}$
		KE	$\mathbf{6.196 \times 10^{-4}}$	$\mathbf{9.243 \times 10^{-1}}$	$\mathbf{2.153 \times 10^{1}}$	$\mathbf{7.226 \times 10^{1}}$	$\mathbf{7.446 \times 10^{1}}$	$\mathbf{5.995 \times 10^{1}}$
		Sinkhorn	$\mathbf{8.544 \times 10^{-5}}$	$\mathbf{4.312 \times 10^{-3}}$	$\mathbf{2.536 \times 10^{-2}}$	$\mathbf{4.572 \times 10^{-2}}$	$\mathbf{4.102 \times 10^{-2}}$	$\mathbf{3.477 \times 10^{-2}}$
Multi Material	GraphUnet	MSE	1.909×10^{-4}	9.311×10^{-4}	2.244×10^{-3}	3.854×10^{-3}	8.534×10^{-3}	1.770×10^{-2}
		KE	2.270×10^{-2}	7.012×10^{-1}	4.394×10^{0}	1.283×10^{1}	3.196×10^{1}	1.369×10^{2}
		Sinkhorn	-	-	-	-	-	-
	Ours	MSE	$\mathbf{1.773 \times 10^{-4}}$	$\mathbf{7.528 \times 10^{-4}}$	$\mathbf{1.571 \times 10^{-3}}$	$\mathbf{2.490 \times 10^{-3}}$	$\mathbf{7.056 \times 10^{-3}}$	$\mathbf{1.259 \times 10^{-2}}$
		KE	$\mathbf{1.923 \times 10^{-2}}$	$\mathbf{4.449 \times 10^{-1}}$	$\mathbf{1.981 \times 10^{0}}$	$\mathbf{4.214 \times 10^{0}}$	$\mathbf{2.013 \times 10^{1}}$	$\mathbf{6.836 \times 10^{1}}$
		Sinkhorn	-	-	-	-	-	-

Table 8: Comparison of MSE, Kinetic Energy (KE) error, and Sinkhorn divergence across datasets containing diverse materials. Sinkhorn divergence is omitted for the multi-material dataset because it does not account for material type.

H Computational Complexity

We evaluate the runtime and memory consumption of different models. Although our approach processes both Lagrangian and Eulerian representations, it is naturally amenable to parallelization at inference time. In particular, the encod-

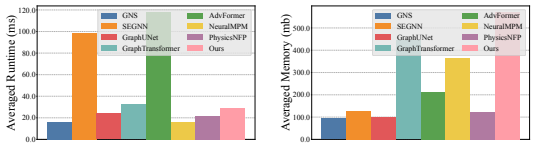


Figure 9: Analysis of the runtime and memory usage of models.

ing and downsampling stages for the two representations can be executed concurrently, which reduces computational time. As illustrated in Figure 9, this design results in only a marginal increase in inference time compared to the baselines. We acknowledge that our model requires additional memory during inference, which could pose a limitation in certain scenarios. Nonetheless, we argue that the overhead remains modest, well within the capacity of widely available commercial GPUs.

We report the model parameters in Table 9. We ensure a fair comparison by evaluating all models at a comparable parameter scale, utilizing standard hyperparameter configurations for each. Because the baselines rely on fundamentally different architectural backbones, aligning their parameter counts exactly is structurally infeasible. For instance, SEGNN operates with significantly fewer parameters (0.36M; 10 layers, 64 latent dimensions) because its prohibitive training and inference times prevent scaling to larger dimensions; however, this setup faithfully reflects the standard configurations established in prior literature. Crucially, our full dual-branch model requires fewer parameters than the best-performing baseline, GraphUnet. Furthermore, when the Eulerian branch is excluded, our model operates with fewer parameters than most baselines. While our complete Lagrangian–Eulerian architecture inherently demands a larger overall parameter budget, we emphasize that inference speed is the most critical practical constraint for neural solvers. Because the encoding and downsampling stages of our two branches execute concurrently, our model maintains an inference time strictly comparable to the baselines.

Model	GNS	SEGNN	GraphUnet	GraphTransformer	AdvDifformer
Params (M)	1.20	0.36	1.97	1.45	1.32

Model	NeuralMPM	PhysicsNFP	Ours	Ours (w/o Eulerian)
Params (M)	1.35	1.36	1.76	1.28

Table 9: Comparison of model sizes for the methods reported in Table 1.

I Additional Lagrangian Baselines

UPT (Alkin et al., 2024): Universal Physics Transformers (UPT) introduces a unified neural simulator designed to handle both Eulerian and Lagrangian dynamics. However, UPT frames Lagrangian simulation as a continuous field learning problem rather than a standard forward-prediction task. Specifically, the model is provided with the ground-truth particle positions at the target time step and tasked with predicting the corresponding particle velocities. Consequently, their evaluation metrics focus strictly on velocity reconstruction. This diverges fundamentally from the standard autoregressive learning paradigm adopted in our work, where the model must infer future kinematic positions directly. Thus, a direct empirical comparison between our approach and UPT would be inequitable, and it is only discussed as related work rather than evaluated as a baseline.

CORGI (Ji et al., 2025): CORGI builds on NeuralMPM by adding message-passing GNNs before and after the particle-grid transfer. However, it inherits a key limitation from NeuralMPM: the particle-grid transfer relies on voxelization, which is restricted to regular grids and cannot map data to irregular meshes. This poses two severe challenges. First, for simulations involving irregular geometries—such as the flow around a cylinder or the deformable domains evaluated in our work—CORGI is inapplicable because it cannot aggregate to an unstructured Eulerian mesh. Second, for free-surface datasets like the dam break, both NeuralMPM and CORGI require Eulerian grids to span the entire computational domain, even in regions devoid of fluid particles. Consequently, many grid nodes remain empty, wasting the processor's computational capacity. Our work addresses this inefficiency by introducing a downsampler that dynamically prunes empty Eulerian nodes, allowing the processor to dedicate its capacity strictly to regions with meaningful dynamics.

Furthermore, we have already compared our approach against GINO (Li et al., 2023a), a model that employs a virtually identical architectural pipeline, and demonstrated significantly better performance. The architectures of these models can be summarized as follows:

- **NeuralMPM:** *No Encoder → Particle-to-Grid → U-Net → Grid-to-Particle → No Decoder*
- **CORGI:** *Message-Passing GNN → Particle-to-Grid → U-Net → Grid-to-Particle → Message-Passing GNN*
- **GINO:** *Graph Neural Operator → Particle-to-Grid → Fourier Neural Operator → Grid-to-Particle → Graph Neural Operator*

Given that our model substantially outperforms GINO, we are confident in its ability to surpass CORGI as well.

Cconv (Ummenhofer et al., 2020) and DMCF (Prantl et al., 2022a): We compare with two additional Lagrangian baselines in Table 10.

	Metric	Cconv	DMCF	**Ours**
LDC	MSE	7.317×10^{-5}	4.871×10^{-6}	$\mathbf{1.868 \times 10^{-6}}$
	KE	3.658×10^{-5}	2.435×10^{-7}	$\mathbf{7.240 \times 10^{-8}}$
	Sinkhorn	6.675×10^{-6}	2.974×10^{-6}	$\mathbf{4.675 \times 10^{-7}}$
TGV	MSE	1.996×10^{-2}	7.218×10^{-3}	$\mathbf{4.551 \times 10^{-3}}$
	KE	9.980×10^{-3}	3.609×10^{-3}	$\mathbf{1.082 \times 10^{-3}}$
	Sinkhorn	4.406×10^{-3}	6.641×10^{-5}	$\mathbf{1.422 \times 10^{-5}}$

Table 10: Comparison of MSE, Kinetic Energy (KE) error, and Sinkhorn divergence for LDC and TGV datasets.

J Hardware Specification

We implement models in PyTorch (Paszke et al., 2019). All experiments can be run on servers/workstations with the following configuration:

- 80 CPUs, 503G Mem, 8 x NVIDIA V100 GPUs.
- 48 CPUs, 220G Mem, 8 x NVIDIA TITAN XP GPUs.
- 96 CPUs, 1.0T Mem, 8 x NVIDIA A100 GPUs.
- 64 CPUs, 1.0T Mem, 8 x NVIDIA RTX A6000 GPUs.
- 224 CPUs, 1.5T Mem, 8 x NVIDIA L40S GPUs.
- 128 CPUs, 480G Mem, 8 × NVIDIA RTX 4090 GPUs.