PEGNet: A Physics-Embedded Graph Network for Long-Term Stable Multiphysics Simulation

Can Yang, Zhenzhong Wang, Junyuan Liu, Yunpeng Gong, Min Jiang*

School of Informatics, Xiamen University
Key Laboratory of Digital Protection and Intelligent Processing of Intangible Cultural Heritage of Fujian and Taiwan, Ministry of Culture and Tourism, Xiamen University
minjiang@xmu.edu.cn

Abstract

Accurate and efficient simulations of physical phenomena governed by partial differential equations (PDEs) are important for scientific and engineering progress. While traditional numerical solvers are powerful, they are often computationally expensive. Recently, data-driven methods have emerged as alternatives, but they frequently suffer from error accumulation and limited physical consistency, especially in multiphysics and complex geometries. To address these challenges, we propose PEGNet, a Physics-Embedded Graph Network that incorporates PDE-guided message passing to redesign the graph neural network architecture. By embedding key PDE dynamics like convection, viscosity, and diffusion into distinct message functions, the model naturally integrates physical constraints into its forward propagation, producing more stable and physically consistent solutions. Additionally, a hierarchical architecture is employed to capture multi-scale features, and physical regularization is integrated into the loss function to further enforce adherence to governing physics. We evaluated PEGNet on benchmarks, including custom datasets for respiratory airflow and drug delivery, showing significant improvements in long-term prediction accuracy and physical consistency over existing methods.

Code — https://github.com/Yanghuoshan/PEGNet
Datasets —
https://huggingface.co/datasets/PEGNet/PEGNet
Extended version — https://arxiv.org/abs/2511.08697

Introduction

Multiphysics simulations often use PDEs to model the spatio-temporal evolution of physical quantities, generating time series from numerical solutions at specific spatial points to reveal system dynamics. Increasingly, data-driven methods are emerging as alternatives to traditional numerical solvers (Gong et al. 2024). Once trained, these methods can offer rapid inference with modest computational cost, overcoming the high resource demands and long runtimes of conventional approaches.

Notable advances such as Physics-Informed Neural Networks (PINNs) (Raissi, Perdikaris, and Karniadakis 2019), Fourier Neural Operators (FNOs) (Li et al. 2021), Graph Neural Networks (GNNs) (Battaglia et al. 2018), Neural ODEs (Chen et al. 2018), Transformer-based models (Han et al. 2022; Li, Meidani, and Farimani 2023; Wang et al. 2025b), and Diffusion-based solvers (Zhou et al. 2025; Li et al. 2024) have demonstrated impressive results in modeling physical processes, offering both accuracy and efficiency. Among these, GNNs, particularly the Graph Network-based Simulator (GNS) framework (Sanchez-Gonzalez et al. 2020), have shown great promise in simulating a wide range of physical systems due to their ability to handle irregular structures. The advantages of the GNS framework inspired us to apply it in the field of medical physics simulations, as this domain typically involves complex human geometries and multiphysics. Furthermore, this domain still requires differentiable models that enable rapid simulations and support optimization frameworks to optimize therapeutic strategies (Wang et al. 2024; Salarpour, Salarpour, and Dogaheh 2025).

However, deep learning models for long-term prediction in time-dependent physical systems primarily face two challenges. First, autoregressive strategies, used by many models (Sanchez-Gonzalez et al. 2020; Li et al. 2021), accumulate small errors over time, which can cause predictions to drift from physical reality (Yang, Vinuesa, and Kang 2025). Second, they often neglect underlying physics. For instance, when numerically solving incompressible Navier-Stokes equations, velocity requires time integration, while pressure is directly solved and has no time evolution (Patankar and Spalding 1983). Many models overlook this, treating all physical quantities uniformly and attempting to learn a time evolution for pressure, which is physically unsound. This limits physical consistency and worsens error accumulation, particularly in multiphysics systems.

Main Idea. To address these issues, we introduce the PDE-guided message passing approach, which directly embeds the mathematical and physical semantics of governing equations into the GNN's architecture. This approach imposes a strong inductive bias by incorporating physical priors into the model's forward propagation, guiding it to naturally follow physical constraints and produce more physically plausible solutions. This ensures that every prediction step is guided by physical priors, thereby reducing the ac-

*Corresponding author.

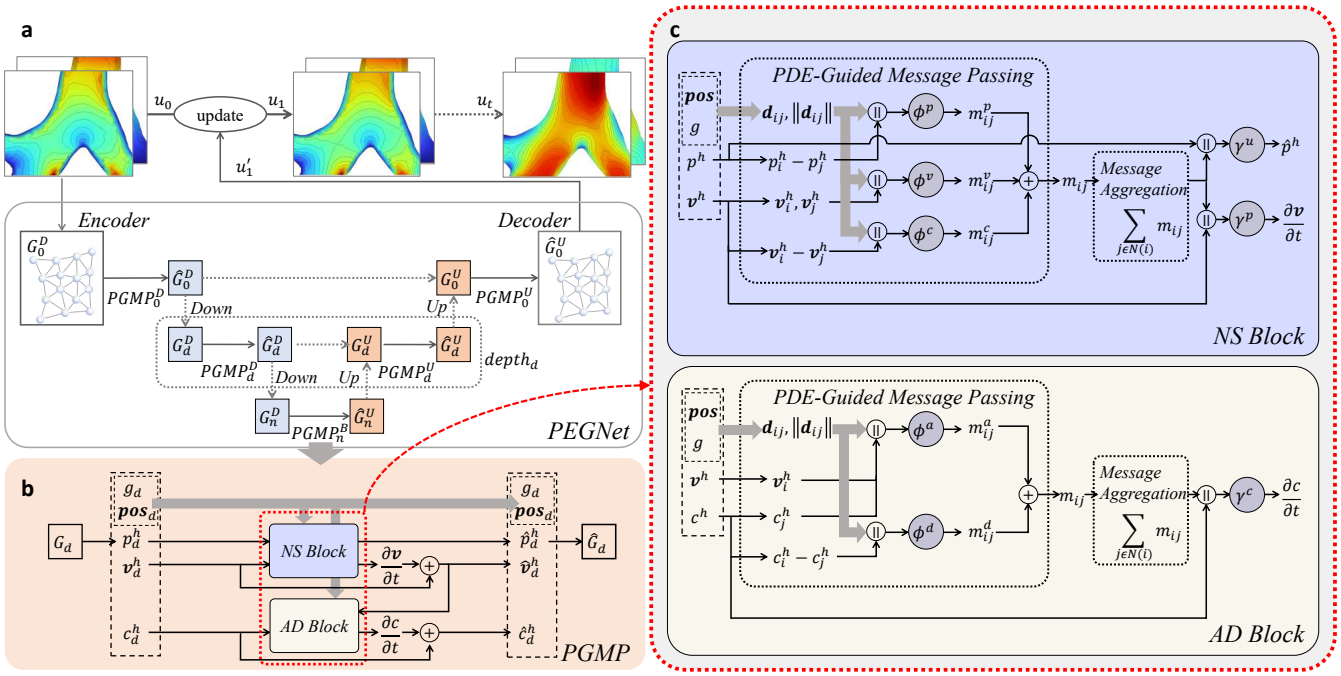

Figure 1: (a) The overall framework of PEGNet. (b) The internal structure of the PGMP module, which consists of a one-way coupled NS Block and AD Block. (c) A detailed view of the internal structures of the NS Block and AD Block, showing how PDE-guided message passing is implemented.

cumulation of errors caused by physical inconsistencies in long-term prediction. Furthermore, our design features separate forward propagation pathways for different types of physical variables, ensuring each quantity is handled according to its true physical nature. This architecture significantly enhances both prediction accuracy and physical consistency.

Contributions. We summarize the main contributions of this work as follows:

- To address the issues of error accumulation and physical inconsistency, we propose PEGNet, a Physics-Embedded Graph Network with PDE-guided message passing, which naturally integrates physical constraints into the model architecture.
- We've built respiratory simulation datasets with two components for benchmarking single-phase flow and multiphysics drug delivery prediction: one for airflow dynamics and another for drug delivery.
- Our evaluation, based on long-term errors and proposed physical consistency metrics, demonstrates that PEGNet not only reduces error accumulation in long-term rollouts but also exhibits superior physical consistency compared to purely data-driven models.

Related Work

GNNs for Physical Simulation. Pioneering works from Interaction Networks (Battaglia et al. 2016) to GNS (Sanchez-Gonzalez et al. 2020) and MGN (Pfaff et al. 2020) have shown that GNNs provide a natural framework for learning physical dynamics on irregular domains such as particle systems or unstructured meshes. These methods have been successfully applied in rigid-body physics (Yang et al. 2025), fluid dynamics (Gao and Jaiman 2024), and elasticity (Deshpande, Bordas, and Lengiewicz 2024). However, most GNN-based methods adopt purely data-driven message passing, neglecting physical constraints. This limitation results in error accumulation and violations of physical consistency over long-term predictions.

Hierarchical GNN Architectures. Hierarchical GNNs address long-range dependencies, resolution heterogeneity, and computational cost by constructing multi-scale graph representations via pooling or coarsening. Methods such as DiffPool (Ying et al. 2018) and TopK pooling (Cangea et al. 2018) have proven effective in graph compression. In physical modeling, approaches like SGUNET (Shen et al. 2025) and BSMS-GNN (Cao et al. 2023) employ non-learnable pooling to improve generalization. These designs are especially useful for complex domains like anatomical airways, where the meshes are numerous and multi-scale.

Physics-Inspired Learning. PINNs (Raissi, Perdikaris, and Karniadakis 2019) incorporate PDE residuals into loss functions, while operator learning methods such as DeepONets (Lu et al. 2021) and FNOs (Li et al. 2021) aim to directly learn mappings between function spaces. Physics-informed operator networks (Goswami et al. 2023) and physics-encoded or physics-embedded approaches (Rao et al. 2023; Guo et al. 2021) further embed physical constraints into model design, with ongoing work exploring higher-order integration schemes (Wang et al. 2025a). To

handle irregular geometries and large-scale data, several studies combine these physics-based approaches with GNNs (Horie and Mitsume 2022; Zeng et al. 2025), achieving promising results in single-phase flow modeling but requiring further development for multiphysics scenarios.

Proposed Method

Overview of PEGNet

PEGNet is a data-driven simulator designed to model multiphysics systems with explicit incorporation of physical priors from governing partial differential equations (PDEs). As illustrated in Figure 1(a), the architecture follows a general Encode-Process-Decode (Battaglia et al. 2018) paradigm, widely adopted in learned physical simulators (Sanchez-Gonzalez et al. 2020; Pfaff et al. 2020), while incorporating physics-structured message passing mechanisms and a multi-scale structure. Given an initial state graph with node features u_0 (e.g., velocity, pressure, position), PEGNet performs autoregressive rollout over n time steps. At each step t, the model predicts the output node features u'_t, and recursively feeds them back to update the input state u_{t-1} for the next step u_t. This enables long-term simulation over the full time window $(t_0, t_0+n\Delta t]$. The model architecture consists of three main modules:

Encoder. A multi-layer perceptron (MLP) maps the input graph with physical node features u_t into a latent graph representation G_d^D in the feature space. The encoded features include latent physical quantities along with positional and topological information. For more detailed input feature information, please refer to Appendix A.

Processor. Instead of using generic message passing layers (Gilmer et al. 2017), we propose the PDE-guided message passing approach. To instantiate this approach in our model, we design a Physics-Guided Message Passing (PGMP) module that consists of two components: the Navier-Stokes (NS) Block and the Advection-Diffusion (AD) Block as shown in Figure 1(b). Each block follows a specific PDE and is then coupled within the PGMP module. Their detailed structures are presented in the following subsections. These blocks act on the hierarchical graph G_d and output updated latent graph $\hat{G}_d$, indicating that PGMP module operates on a particular resolution level d. Multiple modules are stacked with downsampling and upsampling layers, enabling the effective fusion of local and global physical dynamics.

Decoder. A final MLP decodes the processed latent graph into target node outputs u'_t, depending on the specific modeling task. Detailed information on the output node features is included in Appendix A.

PDE-Guided Message Passing

The neural network in Neural ODEs (Chen et al. 2018) can be specifically represented as follows:

$$\frac{du(t)}{dt} = f_\theta(t, u(t)). \tag{1}$$

Inspired by this, we can solve the time partial derivative part in the PDE according to this formula, while using the neural network to map the remaining parts. Consider the following PDE:

$$\frac{\partial u}{\partial t} = F\left(t, \boldsymbol{x}, u, \nabla u, \nabla^2 u, \dots\right), \tag{2}$$

where $u(\boldsymbol{x}, t) \in R^n$ is the state tensor representing n physical quantities, and $\nabla^i u$ denotes the i-th order derivative of u. The PDE-guided message passing approach establishes a structural mapping from the right-hand side of Equation 2 to the message functions of a GNN.

We discretize the continuous PDE system onto a set of interacting nodes, forming a graph $G = (V, E)$, where each node $i \in V$ represents a fluid or material element with features u_i and $\mathcal{N}(i)$ denotes the set of neighbor nodes of node i. The message m_{ij} passing from nodes j to i is formulated to approximate the PDE terms that govern their interaction. Accordingly, the PDE-guided message passing approach can be specified as follows:

$$\begin{aligned} m_{ij} &= \sum_{r=1}^{R} \phi_r\left(z_{ij}^{(r)}\right), \\ \frac{\partial u_i}{\partial t} &= \gamma(u_i, \text{AGG}(\{m_{ij} : j \in \mathcal{N}(i)\})), \end{aligned} \tag{3}$$

where each $z_{ij}^{(r)} \in \{u_i, u_j, \nabla u_i, \nabla^2 u_i, \dots\}$ denotes the r-th component of the input features (e.g., raw values, gradients, or higher-order derivatives), and each ϕ_r is an independent mapping function (e.g., an MLP). The aggregation (AGG) is performed via summation, and γ is another mapping function responsible for calculating the time partial derivative.

Instead of learning generic, black-box interactions, we design a specific message passing scheme to explicitly compute each term of the PDE in the latent space. This approach introduces a strong inductive bias that guides the GNN to capture physical dynamics consistent with the underlying governing equations. As illustrated in Figure 1(b), this is realized through two specialized blocks within the PGMP Module: the NS Block and the AD Block.

NS Block for Fluid Dynamics. The NS Block is designed to model fluid flow dynamics, governed by the incompressible Navier-Stokes equations:

$$\begin{aligned} \frac{\partial \boldsymbol{v}}{\partial t} + (\boldsymbol{v} \cdot \nabla)\boldsymbol{v} &= -\frac{1}{\rho}\nabla p + \nu \nabla^2 \boldsymbol{v}, \\ \nabla \cdot \boldsymbol{v} &= 0, \end{aligned} \tag{4}$$

where $\boldsymbol{v}$ denotes the fluid velocity field, ρ is the fluid density and ν is the kinematic viscosity. The first line of Equation 4 presents the momentum equation, which, from left to right, consists of the time derivative term, the advection term, the pressure gradient term, and the viscous term. The second line imposes the divergence-free constraint. Following Equation 3, we map each PDE term in Equation 4, excluding the partial time derivative, to a specific message function computed by a dedicated MLP. Consequently, message m_{ij} is decomposed into the pressure gradient message m_{ij}^p, the viscosity message m_{ij}^v, and the advection message

m_{ij}^c. The specific PDE-guided message passing process of the NS Block is thus formulated as follows:

$$
\begin{aligned}
m_{ij}^p &= \phi^p([p_i^h - p_j^h, \boldsymbol{d}_{ij}, \|\boldsymbol{d}_{ij}\|]), \\
m_{ij}^v &= \phi^v([\boldsymbol{v}_i^h - \boldsymbol{v}_j^h, \|\boldsymbol{d}_{ij}\|]), \\
m_{ij}^c &= \phi^c([\boldsymbol{v}_i^h, \boldsymbol{v}_j^h, \boldsymbol{d}_{ij}, \|\boldsymbol{d}_{ij}\|]), \\
\frac{\partial \boldsymbol{v}_i}{\partial t} &= \gamma^u([\boldsymbol{v}_i^h, \sum_{j \in \mathcal{N}(i)} (m_{ij}^p + m_{ij}^v + m_{ij}^c)]), \\
\hat{p}_i^h &= \gamma^p([p_i^h, \sum_{j \in \mathcal{N}(i)} (m_{ij}^p + m_{ij}^v + m_{ij}^c)]).
\end{aligned}
\tag{5}
$$

Equation 5 provides a detailed illustration of the mapping between physical terms and neural components, while the structural diagram is shown in Figure 1(c). In this block, the pressure gradient term, which drives fluid flow from regions of high to low pressure, is approximated using pressure differences between neighboring nodes. For incompressible flows, the fluid density ρ remains constant. Thus, the coefficient term $-\frac{1}{\rho}$ can be omitted for simplicity. The displacement vector $\boldsymbol{d}_{ij} = \boldsymbol{pos}_j - \boldsymbol{pos}_i$ captures the relative position between nodes and enables the network to learn a discretized gradient operator. The MLP learns to scale the pressure difference by this displacement, effectively approximating the gradient term $-\nabla p_i$. Similarly, the viscous and convective terms are modeled by separate MLPs, ϕ^v and ϕ^c, which generate the viscosity and convection messages, respectively, using the same principle. Then, the pressure, viscosity, and convection messages from neighboring nodes are aggregated by summation. Finally, another MLP, γ^u, processes the aggregated messages along with the original velocity to predict the latent time derivative, completing the modeling of the momentum equation.

Subsequently, a separate MLP, γ^p, is used to directly predict the updated pressure field $\hat{p}_i^h$, which is crucial for enforcing the incompressibility constraint. As discussed in Section , traditional numerical solvers do not evolve pressure via time integration. Instead, they determine it by solving the pressure Poisson equation, which enforces incompressibility, ensures mass conservation, and stabilizes local flow behavior. This approach contrasts with the velocity update, which typically involves explicit time integration.

AD Block for Scalar Transport. The AD Block models the transport of a scalar quantity c, within the velocity field $\boldsymbol{v}$. This process is governed by the Advection-Diffusion equation:

$$
\frac{\partial c}{\partial t} + \nabla \cdot (c\boldsymbol{v}) = \nabla \cdot (D \nabla c), \tag{6}
$$

where D is the diffusion coefficient. Assuming an incompressible flow ($\nabla \cdot \boldsymbol{v} = 0$), the equation simplifies to:

$$
\frac{\partial c}{\partial t} + \boldsymbol{v} \cdot \nabla c = D \nabla^2 c. \tag{7}
$$

In Equation 7, from left to right, the second term is the advection term, and the third term is the diffusion term. Similar to the NS Block, we present the structural diagram of the

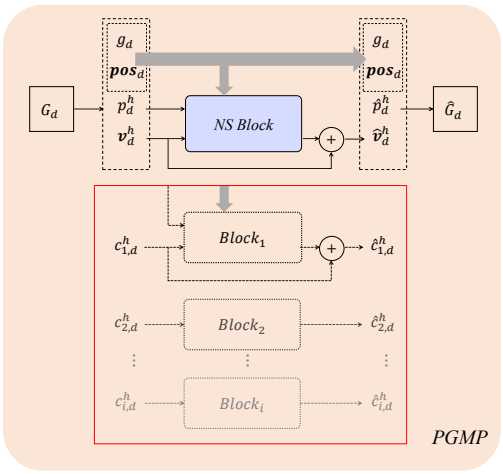

Figure 2: Multi-Task Adaptability of PGMP. The component within the red box can be omitted for single-phase flow, or extended to predict multiple scalar fields.

AD Block in Figure 1(c) and its detailed message passing process is as follows:

$$
\begin{aligned}
m_{ij}^a &= \phi^a([\boldsymbol{v}_i^h, c_j^h - c_i^h, \boldsymbol{d}_{ij}]), \\
m_{ij}^d &= \phi^d([c_j^h - c_i^h, \|\boldsymbol{d}_{ij}\|]), \\
\frac{\partial c_i}{\partial t} &= \gamma^c([c_i^h, \sum_{j \in \mathcal{N}(i)} (m_{ij}^a + m_{ij}^d)]).
\end{aligned}
\tag{8}
$$

In Equation 8, the advection term and the diffusion term are computed by ϕ^a and ϕ^d, respectively, to generate the corresponding advection and diffusion messages. The advection and diffusion messages are summed over the neighborhood of node i. The update MLP γ^c, then predicts the time derivative of the scalar quantity.

Multiphysics Coupling. As shown in Figure 1(b), the PGMP module incorporates a one-way coupling mechanism. Specifically, the NS Block directly predicts the updated pressure, while the time derivative it produces is integrated with the original velocity to obtain the updated velocity. This updated velocity serves not only as the output of the PGMP module but also as the input to the AD Block. The AD Block subsequently computes the evolution of the scalar field, which is also updated through time integration. Therefore, the update of the fluid field is independent of the scalar field, relying solely on the fluid's internal dynamics. On the other hand, the evolution of the scalar field depends on the updated velocity field, directly reflecting the flow's driving influence on the transport process. This coupling sequence adheres to the physical evolution logic of convection-diffusion problems in fluid mechanics and, through clearly defined interfaces between the blocks, offers the flexibility to extend to more complex multiphysics systems. For instance, the extensions could include multi-component transport and adding or ignoring scalar fields. As shown in Figure 2, the PGMP module can be simplified by ignoring the scalar field c, enabling focused simulations

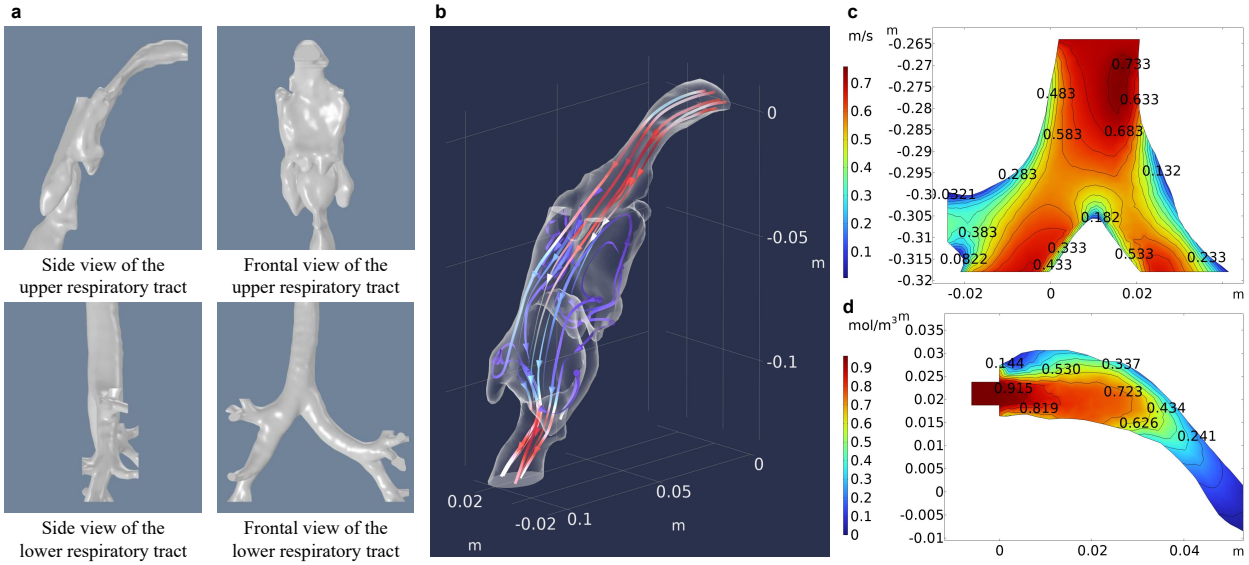

Figure 3: Overview of respiratory simulation datasets generated by traditional numerical methods: (a) Partial view of the human airway. (b) Streamlines of inhaled airflow in the upper respiratory tract, showing complex flow with backflow and vortices. (c) Cross-section at the lower tract bifurcation, colored by speed magnitude with contour lines. (d) Cross-section of the upper tract with a mouthpiece, colored by concentration magnitude with contour lines.

on the fluid dynamics alone, while additional blocks can be added to predict more scalar fields. The ability to decouple the concentration and fluid fields within the PGMP framework allows for a targeted approach to different simulation needs, making the system adaptable to varying degrees of complexity and different physical setups.

Model Enhancements: Multi-Scale Design

While PEGNet is built around the PDE-guided message passing approach, we integrate bi-stride multi-scale pooling (Cao et al. 2023) to efficiently capture global and local dynamics. This non-learnable pooling reduces mesh complexity with increasing network depth, enabling fast computation on large geometries like human airways, and mitigates oversmoothing by employing skip connections, similar to U-Net (Ronneberger, Fischer, and Brox 2015). However, unlike the one-MP (message passing) used in the Bi-Stride Multi-Scale Graph Neural Network (Cao et al. 2023), the PGMP module at each scale better preserves physical semantics.

Experiment

Experiment Setup

Datasets. From MeshGraphNets (MGN) (Pfaff et al. 2020), we selected the publicly released dataset: cylinder flow, which is a fluid simulation dataset based on the incompressible Navier-Stokes equations. Following this, we established respiratory simulation datasets, which comprise both airflow simulation and drug delivery simulation components. To validate the PDE-guided message passing approach's applicability to other PDEs, we also constructed the Gray-Scott reaction-diffusion dataset. The details of these datasets are included in Appendix A.

Model Training. In this work, following the principles of PINNs (Raissi, Perdikaris, and Karniadakis 2019), we incorporate physical constraints into the training process, directly guiding the model output to adhere to the underlying physical laws while retaining the efficiency and flexibility of data-driven learning. The loss function is designed to include the following terms: (1) *Node Prediction Error* (L_{pred}): This term is computed as the mean squared error (MSE) between the predicted node feature $\hat{p}_i$ and the true features p_i at each node, which is expressed as:

$$L_{\text{pred}} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{p}_i - p_i\right)^2. \tag{9}$$

(2) *Divergence Regularization Loss* (L_{div}): This term enforces the divergence-free condition on the velocity field, which is defined as:

$$L_{\text{div}} = \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j\in\mathcal{N}(i)}\frac{1}{|\mathcal{N}(i)|}\left(\boldsymbol{v}_j - \boldsymbol{v}_i\right)\boldsymbol{d}_{ij}\right)^2. \tag{10}$$

(3) *Mass Conservation Regularization Loss* (L_{mass}): This term ensures compliance with mass conservation principles, which is defined as:

$$L_{\text{mass}} = \frac{1}{N}\sum_{i=1}^{N}\Bigg(\left(c_i^{t+1} - c_i^t\right) + \sum_{j\in\mathcal{N}(i)}\left((\boldsymbol{v}_i^{t+1}\cdot\boldsymbol{d}_{ij})c_i^t - (\boldsymbol{v}_j^{t+1}\cdot\boldsymbol{d}_{ij})c_j^t\right)\Bigg)^2. \tag{11}$$

Case	Measurements	PEGNet(ours)	BSMS-GNN	MS-GNN-GRID	MGN	GraphUNets
Cylinder Flow	RMSE-1 [1e-2]	0.216	**0.204**	0.220	0.226	0.809
	RMSE-50 [1e-2]	**1.55**	2.42	2.74	4.39	18.7
	RMSE-last [1e-2]	**2.56**	8.37	8.49	9.06	165
Airflow	RMSE-1 [1e-2]	**0.781**	0.797	0.790	0.883	12.1
	RMSE-50 [1e-2]	**1.09**	2.40	1.93	8.35	26.7
	RMSE-last [1e-2]	**8.30**	44.1	59.2	44.8	911
Drug Delivery	RMSE-1 [1e-3]	1.37	**0.914**	1.21	2.05	28.2
	RMSE-50 [1e-3]	**2.15**	2.47	5.32	6.08	30.6
	RMSE-last [1e-3]	**5.19**	5.68	15.6	19.2	601

Table 1: Comparison of different models in long-term prediction.

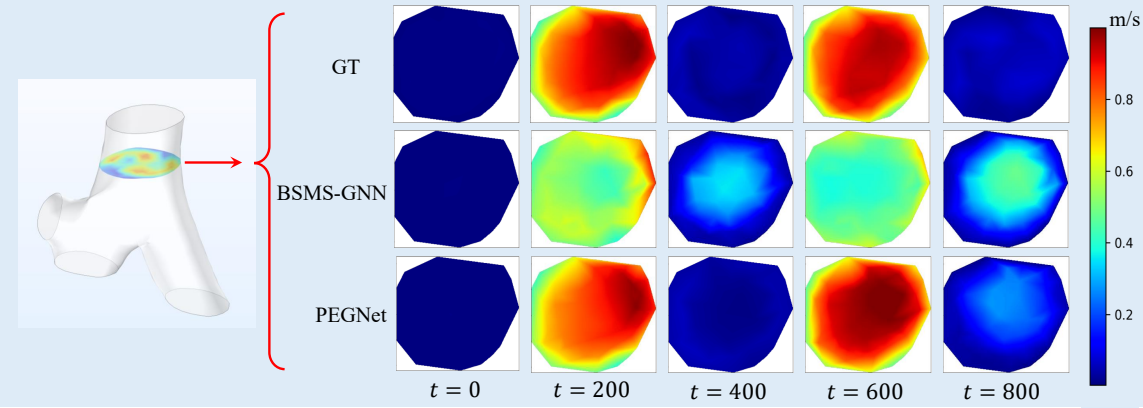

Figure 4: From $t = 0$ to $t = 800$ represents a complete inhalation-exhalation cycle, with key points at $t = 0$ (start of inhalation), $t = 200$ (peak inhalation velocity), $t = 400$ (start of exhalation), $t = 600$ (peak exhalation velocity), and $t = 800$ (end of exhalation). The BSMS-GNN model shows increasing artifacts in the central airway, predicting airflow even after exhalation ends, contrary to reality. In contrast, our model remains more stable with only minor artifacts near the cycle's end.

The specific definitions for these formulas are provided in the Appendix B. Thus, the total loss function can be expressed as:

$$L_{\text{total}} = L_{\text{pred}} + \lambda_{\text{div}} L_{\text{div}} + \lambda_{\text{mass}} L_{\text{mass}}, \tag{12}$$

where λ_{div} and λ_{mass} are hyperparameters controlling the respective regularizations.

For optimization, we used the AdamW optimizer (Loshchilov and Hutter 2019), which incorporates decoupled weight decay and adaptive moment estimation, along with a warmup cosine decay learning rate scheduler (Vaswani et al. 2017).

Baseline. We compare our model with several representative baseline models, including MGN (Pfaff et al. 2020), BSMS-GNN (Cao et al. 2023), MS-GNN-GRID (Lino et al. 2022), and GraphUNets (Gao and Ji 2019). A detailed description of these models is provided in Appendix C.

Evaluation Metrics. The model performance is evaluated using the following metrics: (1) *Root Mean Squared Error (RMSE)*, which measures the accuracy of node predictions and is defined as: $RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|\hat{p}_i - p_i\|^2}$; (2) *Divergence Error (DVE)*, defined as the square root of the divergence regularization loss, which assesses the divergence-free condition of the velocity field: $DVE = \sqrt{L_{\text{div}}}$; (3) *Mass Conservation Error (MCE)*, defined as the square root of the mass conservation regularization loss, which evaluates the consistency of mass transport across nodes: $MCE = \sqrt{L_{\text{mass}}}$. Lower values of these metrics indicate better accuracy and improved physical consistency. Detailed explanations of these metrics are provided in Appendix B.

Benchmarking and Evaluation

Cylinder Flow. In Table 1, for the Case Cylinder Flow, the data for the baseline models comes from BSMS-GNN (Cao et al. 2023). To ensure experimental fairness, PEGNet's training hyperparameters were selected to closely match those of BSMS-GNN (Cao et al. 2023), as detailed in Appendix B. After training the model to convergence on the training set, we evaluated its performance on the test set using an autoregressive rollout to generate trajectories of the same length as the ground truth. We then computed the RMSE at time steps 1, 50, and the last step across all prediction trajectories and averaged the results at each step. The results are shown in the Case Cylinder Flow of Table 1. The bolded values in the table represent the best results within the same measurement. At the first step, most models perform well. However, in the subsequent time steps, our model

Method	Airflow		Drug Delivery		
	Vel	Pre	Vel	Pre	Conc
MGN	20.5	52.3	0.823	2.15	2.32
BSMS-GNN	14.3	68.6	0.146	0.0522	1.28
PEGNet(ours)	**4.67**	**3.87**	**0.145**	**0.0141**	**0.648**

Table 2: Multi-channel evaluation of velocity (Vel), pressure (Pre), and concentration (Conc) using average RMSE[1e-2] over all timesteps.

Case	Metrics	PEGNet(ours)	BSMS-GNN	MGN
Airflow	DVE-1	**0.647**	0.685	1.25
	DVE-50	**11.1**	11.4	12.7
	DVE-last	**35.1**	250	268
Drug Delivery	DVE-1	**0.218**	0.416	0.921
	DVE-50	**2.39**	2.46	5.37
	DVE-last	**1.13**	1.17	8.80
Drug Delivery	MCE-1	**27.9**	38.5	278
	MCE-50	**5.95**	6.40	17.4
	MCE-last	**7.18**	7.82	18.5

Table 3: Evaluation using DVE[1e-6] and MCE[1e-5].

consistently maintains a significant advantage, demonstrating its stronger long-term predictive stability.

Airflow. After the preliminary evaluation of PEGNet's effectiveness in the 2D fluid prediction task, we further assessed its performance in simulating airflow in the 3D human airway. All models were trained to convergence on the training set, and the same evaluation strategy as in the Cylinder Flow case was adopted. The corresponding results are shown in the Case Airflow of Table 1, which demonstrate that PEGNet outperforms baseline models at time steps 1, 50, and the last step. Figure 4 compares the flow velocities for the ground truth (GT), PEGNet, and BSMS-GNN at the airway cross-section. As shown, PEGNet's predictions align more closely with the ground truth and exhibit greater stability across all time steps. Additional visualizations of the model predictions are included in Appendix E.

Drug Delivery. Since PEGNet demonstrated the ability to simulate airflow in the 3D human airway, we further applied it to the multiphysics drug delivery prediction task. The evaluation procedure remained the same as in the previous two cases. As shown in the Case Drug Delivery of Table 1, the results indicate that although PEGNet does not show a significant advantage in first-step prediction, it exhibits superior stability compared to other models in subsequent time steps.

Physical Consistency. Table 2 presents a multi-channel comparison between PEGNet and two representative models. The results show that, although MGN and BSMS-GNN still exhibit a certain degree of accuracy in overall predictions or velocity field predictions, their performance in predicting coupled physical fields is significantly worse than that of PEGNet, whether coupled with the pressure field or further extended to the concentration field. This indicates that MGN and BSMS-GNN have limitations in learning physical principles governing each quantity and in handling multiphysics coupling. In contrast, PEGNet incorporates physical prior knowledge and is better at capturing the relationships between different physical quantities. Table 3 provides another comparison of these models, using the physical consistency metrics DVE and MCE. We computed DVE and MCE at time steps 1, 50, and the last step across all generated trajectories and averaged the values at each corresponding step, which highlights that our model better adheres to physical constraints over long rollouts than the others.

Measurements	Ours	Model A	Model B	Model C
RMSE-1 [1e-2]	0.781	**0.757**	0.788	0.797
RMSE-50 [1e-2]	**1.09**	1.26	1.54	2.40
RMSE-last [1e-2]	**8.30**	12.7	30.8	41.1

Table 4: Results of the Ablation Study.

Gray-Scott Reaction-Diffusion. To further validate the generality of our PDE-guided message passing approach beyond fluid dynamics, we extended it to the Gray-Scott equation, with details provided in Appendix D. The results demonstrate its potential applicability to other systems governed by PDEs.

Ablation Study

To evaluate the contributions of the PGMP module and the physics-informed loss, we conducted ablation experiments on the airflow dataset by progressively removing these components. Four models were tested: the full PEGNet (Ours); PEGNet without the physics-informed loss (Model A); PEGNet without the PGMP module, using standard message passing instead (Model B); and PEGNet without both modules (Model C). Results in Table 4 show that removing both modules (Model C) yields the worst long-term predictions, while including either significantly improves accuracy and stability. The complete PEGNet achieves the best performance in long-term rollouts, demonstrating the synergistic benefits of both components.

Conclusion

This study proposes PEGNet for stable modeling of airflow and drug delivery in the human airway. Overall, our approach enhances the neural model's long-term predictive accuracy and physical consistency. The results demonstrate that it outperforms other data-driven methods in both aspects. Despite the results, several limitations remain. These include simplified airway models, limited data availability, and superficial integration of physical principles. Future work will focus on incorporating more anatomically realistic and variable airway models, expanding data sources, and strengthening the integration of physical principles with neural networks. Accounting for more complex pulmonary physiological conditions will be key to improving both model performance and clinical applicability.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant No. 52535009 and No. 62276222; in part by the Fundamental Research Funds for the Central Universities under Grant No. 20720250164; in part by the Xiamen Natural Science Foundation under Grant No. 3502Z202571027.

References

Battaglia, P.; Pascanu, R.; Lai, M.; Jimenez Rezende, D.; et al. 2016. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29.

Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; Gulcehre, C.; Song, F.; Ballard, A.; Gilmer, J.; Dahl, G.; Vaswani, A.; Allen, K.; Nash, C.; Langston, V.; Dyer, C.; Heess, N.; Wierstra, D.; Kohli, P.; Botvinick, M.; Vinyals, O.; Li, Y.; and Pascanu, R. 2018. Relational inductive biases, deep learning, and graph networks. arXiv:1806.01261.

Cangea, C.; Veličković, P.; Jovanović, N.; Kipf, T.; and Liò, P. 2018. Towards Sparse Hierarchical Graph Classifiers. arXiv:1811.01287.

Cao, Y.; Chai, M.; Li, M.; and Jiang, C. 2023. Efficient learning of mesh-based physical simulation with bi-stride multi-scale graph neural network. In *International conference on machine learning*, 3541–3558. PMLR.

Chen, R. T.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. K. 2018. Neural ordinary differential equations. *Advances in neural information processing systems*, 31.

Deshpande, S.; Bordas, S. P.; and Lengiewicz, J. 2024. MAgNET: A graph U-Net architecture for mesh-based simulations. *Engineering Applications of Artificial Intelligence*, 133: 108055.

Gao, H.; and Ji, S. 2019. Graph u-nets. In *international conference on machine learning*, 2083–2092. PMLR.

Gao, R.; and Jaiman, R. K. 2024. Predicting fluid–structure interaction with graph neural networks. *Physics of Fluids*, 36(1).

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*, 1263–1272. PMLR.

Gong, Y.; Hou, Y.; Wang, Z.; Lin, Z.; and Jiang, M. 2024. Adversarial Learning for Neural PDE Solvers with Sparse Data. arXiv:2409.02431.

Goswami, S.; Bora, A.; Yu, Y.; and Karniadakis, G. E. 2023. Physics-informed deep neural operator networks. In *Machine learning in modeling and simulation: methods and applications*, 219–254. Springer.

Guo, R.; Shan, T.; Song, X.; Li, M.; Yang, F.; Xu, S.; and Abubakar, A. 2021. Physics embedded deep neural network for solving volume integral equation: 2-D case. *IEEE Transactions on Antennas and Propagation*, 70(8): 6135–6147.

Han, X.; Gao, H.; Pfaff, T.; Wang, J.-X.; and Liu, L.-P. 2022. Predicting Physics in Mesh-reduced Space with Temporal Attention. arXiv:2201.09113.

Horie, M.; and Mitsume, N. 2022. Physics-embedded neural networks: Graph neural pde solvers with mixed boundary conditions. *Advances in Neural Information Processing Systems*, 35: 23218–23229.

Li, Z.; Han, W.; Zhang, Y.; Fu, Q.; Li, J.; Qin, L.; Dong, R.; Sun, H.; Deng, Y.; and Yang, L. 2024. Learning spatiotemporal dynamics with a pretrained generative model. *Nature Machine Intelligence*, 6(12): 1566–1579.

Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2021. Fourier Neural Operator for Parametric Partial Differential Equations. arXiv:2010.08895.

Li, Z.; Meidani, K.; and Farimani, A. B. 2023. Transformer for Partial Differential Equations' Operator Learning. arXiv:2205.13671.

Lino, M.; Fotiadis, S.; Bharath, A. A.; and Cantwell, C. D. 2022. Multi-scale rotation-equivariant graph neural networks for unsteady Eulerian fluid dynamics. *Physics of Fluids*, 34(8).

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. arXiv:1711.05101.

Lu, L.; Jin, P.; Pang, G.; Zhang, Z.; and Karniadakis, G. E. 2021. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3): 218–229.

Patankar, S. V.; and Spalding, D. B. 1983. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. In *Numerical prediction of flow, heat transfer, turbulence and combustion*, 54–73. Elsevier.

Pfaff, T.; Fortunato, M.; Sanchez-Gonzalez, A.; and Battaglia, P. 2020. Learning mesh-based simulation with graph networks. In *International conference on learning representations*.

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

Rao, C.; Ren, P.; Wang, Q.; Buyukozturk, O.; Sun, H.; and Liu, Y. 2023. Encoding physics to learn reaction–diffusion processes. *Nature Machine Intelligence*, 5(7): 765–779.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, 234–241. Springer.

Salarpour, S.; Salarpour, S.; and Dogaheh, M. A. 2025. Advancing Pharmaceutical Science with Artificial Neural Networks: A Review on Optimizing Drug Delivery Systems Formulation. *Current Pharmaceutical Design*, 31(7): 507–520.

Sanchez-Gonzalez, A.; Godwin, J.; Pfaff, T.; Ying, R.; Leskovec, J.; and Battaglia, P. 2020. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, 8459–8468. PMLR.

Shen, S.; Liu, Y.; Biggs, D.; Hafez, O.; Yu, J.; Zhang, W.; Cui, B.; and Shan, J. 2025. Transfer learning in Scalable Graph Neural Network for Improved Physical Simulation. arXiv:2502.06848.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, Q.; Mi, Y.; Wang, H.; Zhang, Y.; Chengze, R.; Liu, H.; Wen, J.-R.; and Sun, H. 2025a. MultiPDENet: PDE-embedded Learning with Multi-time-stepping for Accelerated Flow Simulation. arXiv:2501.15987.

Wang, X.; Ma, K.; Zhong, R.; Wang, X.; Fang, Y.; Xiao, Y.; and Xia, T. 2024. Towards dual transparent liquid level estimation in biomedical lab: Dataset, methods and practices. In *European Conference on Computer Vision*, 198–214. Springer.

Wang, Z.; Zhang, X.; Liao, J.; and Jiang, M. 2025b. Cross-Field Interface-Aware Neural Operators for Multiphase Flow Simulation. arXiv:2511.08625.

Yang, S.; Peng, T.; Li, Z.; Wu, F.; and Zhao, J. 2025. MBDS: A MultiBody Dynamics Simulation Dataset for Graph Networks Simulators. In *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, 1–5. IEEE.

Yang, S.; Vinuesa, R.; and Kang, N. 2025. Long-Term Auto-Regressive Prediction using Lightweight AI Models: Adams-Bashforth Time Integration with Adaptive Multi-Step Rollout. arXiv:2412.05657.

Ying, Z.; You, J.; Morris, C.; Ren, X.; Hamilton, W.; and Leskovec, J. 2018. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31.

Zeng, B.; Wang, Q.; Yan, M.; Liu, Y.; Chengze, R.; Zhang, Y.; Liu, H.; Wang, Z.; and Sun, H. 2025. PhyMPGN: Physics-encoded Message Passing Graph Network for spatiotemporal PDE systems. arXiv:2410.01337.

Zhou, A.; Li, Z.; Schneier, M.; Jr, J. R. B.; and Farimani, A. B. 2025. Text2PDE: Latent Diffusion Models for Accessible Physics Simulation. arXiv:2410.01153.