PREDICTING FLUID DYNAMICS IN PHYSICAL-INFORMED MESH-REDUCED SPACE

Yeping Hu[*]
Lawrence Livermore National Laboratory
Livermore, CA 94550, USA
hu25@llnl.gov

Bo Lei[*]
Lawrence Livermore National Laboratory
Livermore, CA 94550, USA
lei4@llnl.gov

Victor M. Castillo
Lawrence Livermore National Laboratory
Livermore, CA 94550, USA
castillo3@llnl.gov

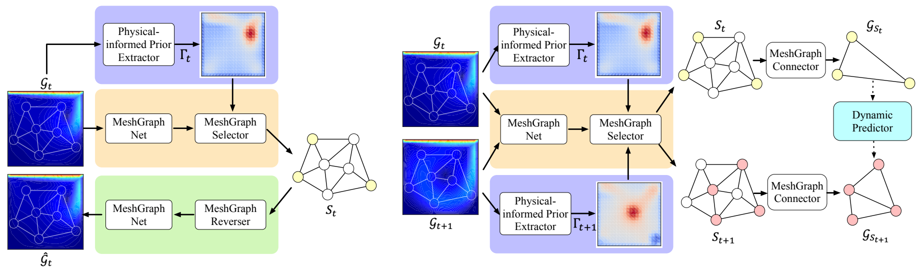
ABSTRACT

For computational fluid dynamics, there is a considerable interest in using neural networks for accelerating simulations. However, these learning-based models suffer from scalability issues when training on high-dimensional and high-resolution simulation data generated for real-world applications. In this work, we study the problem of improving accuracy of desired physical properties using graph learning models for learning complex fluid dynamics, while operating on mesh-reduced space. We design several tailored modules to incorporate physical-informed knowledge into a two-stage prediction model, which directs the learning process to focus more on the region of interest (ROI). Prediction will then be made in a mesh-reduced space, which helps reduce computational costs while preserving important physical properties. Results on simulated unsteady fluid flow data show that even under reduced operational space, our method still achieves desirable performance on accuracy and generalizability of both prediction and physical consistency over region of interests.

1 INTRODUCTION

In industrial applications such as furnace and aerodynamic simulation, predicting physical dynamics with Graph Neural Network (GNN) models has become an active and growing area of research in machine learning due to their flexibility and power. However, the dynamic systems in these applications are often high-dimensional and require very fine meshes to resolve, resulting in significantly larger computational costs. Moreover, as the mesh becomes finer, GNNs have to perform more message passing steps to pass information along the same physical distance, which may result in accuracy reduction caused by over-smoothing (Li et al., 2018) and over-squashing (Alon & Yahav, 2020). In fact, in many cases, simulating the entire dynamic system accurately may not be necessary, as certain regions can be more important than others. For instance, in the steel manufacturing process, the von Kármán flow (Zandbergen & Dijkstra, 1987) that occurs during steel casting operations can cause many defects in the final product. Vortex formation is one of the fluid flow phenomena that occur in the casting mold and is believed to contribute significantly to mold powder entrapment. The entrapped flux may be carried deeply down into the mold by downward flows and results in uncleanness of steel (He, 1993). Therefore, accurate simulation of the areas where vortex formation occurs is more important than that of other regions. By focusing the learning process on the region of interest (ROI), we can achieve better accuracy on important regions while reducing computational costs by paying less attention to other areas. Our contributions of this paper are in three folds:

[*]Equal Contribution. This work was performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL release number: LLNL-CONF-845077.



(a) Mesh Graph Autoencoder (b) Graph Predictor in Mesh-reduced Space

Figure 1: The proposed two-stage training framework.

- We study the problem of improving accuracy of desired physical properties using graph learning models for learning complex fluid dynamics (e.g. turbulence), while operating on mesh-reduced space.
- We design tailored modules to incorporate physical-informed knowledge into a two-stage training model for fluid dynamic forecasting.
- When evaluate on unsteady fluid flow prediction, our method achieves desired performance on accuracy and generalizability of both prediction and physical consistency over ROI.

2 METHODS

Our proposed prediction system contains two training stages. A Mesh Graph Autoencoder will first learned to incorporate physical-informed priors into the full mesh graph and summarize the information into subgraph, forming representations of each time frame into a mesh-reduced space. Then, a Graph Predictor will learn to predict subgraph evolution in the mesh-reduced space. We denote an attributed graph with N nodes as a tuple $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. All fluid simulations in this work use *Eulerian* systems to model the evolution of continuous fields, such as velocity, over a fixed mesh. As a result, the graph remains fixed for each data across all time steps

2.1 PHYSICAL-INFORMED PRIOR (PIP) EXTRACTOR

The underlying physical prior incorporated in our graph learning system is rooted in the family of vortex methods that are rigorously derived, analyzed, and tested in the computational fluid dynamics (CFD) (Graftieaux et al. (2001); Leonard (1980)). In this work, we apply the vortex center identification algorithm to characterize the location of the center. Specifically, for any given node i in a mesh graph $\mathcal{G}_t$, we utilized a scalar function Γ to quantify the streamline topology of the flow in the vicinity of node center C_i and the rotation sign of the vortex:

$$\Gamma_{i,t} = \frac{1}{L}\sum_{l=1}^{L} \sin\theta_{M_l} = \frac{1}{L}\sum_{l=1}^{L} \frac{\vec{C_i M_l} \times v^t_{M_l}}{\|\vec{C_i M_l}\| \cdot \|v^t_{M_l}\|}, \tag{1}$$

where M_l is one of the evenly spaced L points on a circle centered at C_i with radius r. θ_{M_l} represents the angle between the velocity vector $v^t_{M_l}$ at time t and the radius vector $\vec{C_i M_l}$. The velocity $v^t_{M_l}$ at M_l is computed by spatial interpolation. The nodes with high $|\Gamma|$ generally locate near vortices. Here, Γ is a dimensionless scalar, with $|\Gamma| \leq 1$. It can be shown that this bound is reached at the vortex center if the vortex is axisymmetrical.

2.2 MESH GRAPH AUTOENCODER

The proposed Mesh Graph Autoencoder model ($\mathcal{G}_t \rightarrow S_t \rightarrow \hat{\mathcal{G}}_t$), shown in Figure 1(a), consists of a Physical-informed Prior (PiP) Extractor, two MeshGraphNet (MGN), a MeshGraphSelector (MGS), and a MeshGraphReverser (MGR). After finishing training this module, a set of pivotal nodes S_t with feature Z_t will be obtained and further used as input for the next-stage Graph Predictor (discussed in 2.3).

The internal structure of MGN is an Encoder-Process-Decoder (EPD) network Pfaff et al. (2020). After the original graph $\mathcal{G}_t$ passing through the first MGN module (denoted as f^1_{MGN}), the updated graph along with the prior information obtained from the PiP extractor is fed into MGS. In the MGS module (denoted as f_{MGS}), it adaptively selects a subset of nodes based on k-largest values in Γ_t to form a new but smaller graph. Eventually, the new graph will contain information from the mesh-reduced space with a set of $S_t \in \mathcal{N}$ pivotal nodes. These pivotal nodes will preserve as much PiP information as possible at each time step while ensuring consistency of the graph across time-steps.

The features of the pivotal nodes need to go through the MGR module (denoted as f_{MGR}) to restore information of full nodes from the original mesh graph. This step is implemented by spatial interpolation of the non-pivotal nodes. A commonly used approach is to use inverse distance weighted (IDW) interpolation Alet et al. (2019); Han et al. (2022). However, as these pivotal nodes are irregularly spaced points and we will later construct triangular mesh graph using these pivotal nodes, we instead use another spatial interpolation technique named by triangular interpolation network (TIN) Wood et al. (1990); Vivoni et al. (2004). After reversed into its original form, the mesh graph will be fed into a second MDN module (denoted as f^2_{MGN}) which will output a reconstructed graph $\hat{\mathcal{G}}_t$. We then minimize the difference between the original graph $\mathcal{G}_t$ and the reconstruction graph $\hat{\mathcal{G}}_t$ at each time step through backpropagation.

2.3 Graph Predictor in Mesh-reduced Space

The proposed Graph Predictor, shown in Figure 1(b), operates in mesh-reduced space, which consists of a MeshGraphConnector (MGC) and a Dynamic Predictor. The MeshGraphConnector is necessary to construct valid subgraphs with proper connectivity, while the Dynamic Predictor operates on these subgraphs to make time-series predictions.

As related edges are removed when removing nodes in MGS, the nodes in the selected graph might become isolated. Therefore, we need to increase connectivity among nodes in the selected graph through MGC. The connector function determines, for each pair of supernodes S_a, S_b, the presence or absence of an edge between the corresponding nodes a and b in the pooled graph. The function also computes the attributes to be assigned to new edges. There are many different connection functions to choose from Grattarola et al. (2022) and we use Delaunay triangulation for its nearly unique and optimal triangulation Watson & Philip (1984). For a set of points, the Delaunay criterion ensures that a circle that passes through three points on any triangle contains no additional points.

After the training of Mesh Graph Autoencoder, we are able to obtain representative subgraphs for every full graphs in the data. The Dynamic Predictor will then be trained to learn a function that can predict $\mathcal{G}_{t+1,S_{t+1}}$ from $\mathcal{G}_{t,S_t}$, where $\mathcal{G}_{t,S_t}$ denotes a subgraph of $\mathcal{G}_t$ with a set of $S_t \in \mathcal{N}$ nodes at time step t. The challenge here is that for unsteady or turbulent flow, pivotal nodes from PiP extractor will vary across time and thus $S_t \neq S_{t+1}$. Therefore, we need to find a way to make the prediction. The first step within the Graph Predictor is to feed $\mathcal{G}_{t,S_t}$ into another MGN module (denoted as f^3_{MGN}) to predict a subgraph at time $t+1$, denoted as $\hat{\mathcal{G}}_{t+1,S_t}$ which has a feature matrix Z_{t+1,S_t} with pivotal node set S_t at time step $t+1$. Then, $\hat{\mathcal{G}}_{t+1,S_t}$ is consecutively fed into f_{MGR} and f^2_{MGN} to obtain a full graph $\hat{\mathcal{G}}_{t+1}$. Finally, to obtain the subgraph $\hat{\mathcal{G}}_{t+1,S_{t+1}}$ (with feature matrix $Z_{t+1,S_{t+1}}$) that has pivotal node set S_{t+1} at time step $t+1$, $\hat{\mathcal{G}}_{t+1}$ is sequentially passed into f^1_{MGN} and f_{MGS}. To train the dynamic predictor module, we minimize the difference between $\hat{\mathcal{G}}_{t+1}$ and the $\mathcal{G}_{t+1}$ at each time step through backpropagation.

3 Experiment

To generate our simulation data, we use a numerical solver with Navier-Stokes equations on fixed 2D Eulerian meshes. Specifically, two dataset are generated: *Lid-driven Cavity Flow* dataset simulates the flow inside a cavity with a moving top wall; *Cylinder Flow* dataset simulates the flow around a cylinder inside a tunnel. To make the dataset more challenging and to futher test generalizability of our model, we generate data with irregular cavity shape and with more than one cylinder in the tunnel. Additional dataset details and visualizations are available in Appendix B.

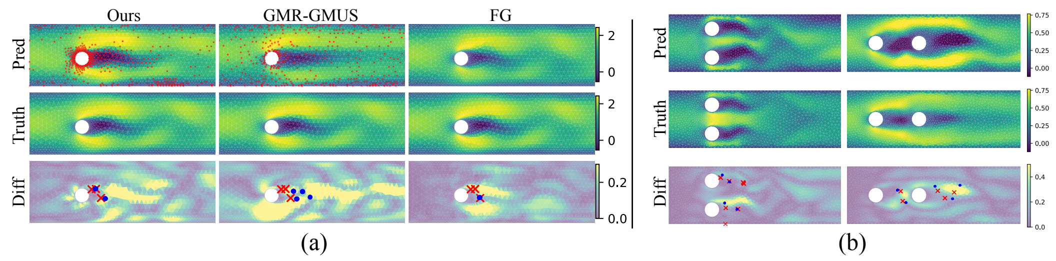


Figure 2: (a) Simulation results comparison of different methods tested on a cylinder flow data at t = 100. In the first row, pivotal nodes used in the mesh-reduced space are colored in red. (b) Generalization results of our method on unseen two-cylinder test cases at t = 200. For both plots, on their third row, red 'X' marks denote vortices detected from the actual data, while blue dots denote vortices detected from predicted velocity fields.

We compare the performance of our approach with a baseline method GMR_GMUS Han et al. (2022) which also aims to reduce computational cost by learning a low-dimensional mesh graph. Different from our approach, they used random sampling to construct subgraphs and did not take physical-informed priors into account. In this section, we visually illustrate performance of our method on accuracy and generalizability of both prediction and physical consistency over ROI. Details on model parameter selection, in-depth quantitative analysis, and other experiment results are available in Appendix C.

The prediction results for a selected case are visualized for our model, GMR_GMUS, and a full graph (FG) predictor in Figure 2(a). The first and second row of the figure show the mesh color representing the magnitude of V_x, while the third row shows the magnitude of the difference between the predicted and ground truth values. According to the results, the PiP-selected nodes of our model are located near the cylinder and the area to the right, where vortices commonly appear, while randomly selected nodes in GMR_GMUS do not attend to this region. The prediction results of our model are visually close to the ground-truth and the FG prediction, showing a clear sign of unsteady flow past the cylinder. On the other hand, GMR_GMUS's prediction shows a steady flow, which is incorrect. In the third row, we can observe that the detected vortices from the ground truth and our model's prediction are located close to each other, while the vortex detection from GMR_GMUS's prediction is a failure.

We further investigate the generalizability of our model using unseen test cases with two cylinders. Figure 2(b) shows two different placements of the cylinders, one vertical and the other horizontal. As expected, the overall performance of our model decreases on these new cases, but it is still able to produce reasonable velocity predictions. Most importantly, the detected vortices from the ground-truth and our model predictions still appear close to each other. If we take a closer look at the third row in Figure 2(b), the absolute prediction errors of our model are small around ROI.

4 Conclusion

In this work, we study the problem of improving accuracy of desired physical properties using graph learning models for learning complex fluid dynamics, while operating on mesh-reduced space. A two-stage graph-based model is proposed, which utilizes physical-informed prior information to learn and predict physical dynamics in a reduced mesh space. We generate challenging unsteady fluid flow data to test and evaluate the performance of our model. Results show that the proposed model is able to accurately detect important physical properties, such as vortex locations, more effectively than other baseline methods, even with reduced mesh space. Additionally, the model demonstrates great generalizability, as it is able to accurately predict flow in out-of-distribution cases. While the overall prediction accuracy of the proposed model may not be as high as models trained using the entire mesh, the trade-off in computational cost reduction and the improvement of prediction accuracy around regions of interest makes the proposed method a useful tool in real-world applications where high-dimensional data cannot be trained using full domain information.

References

Ferran Alet, Adarsh Keshav Jeewajee, Maria Bauza Villalonga, Alberto Rodriguez, Tomas Lozano-Perez, and Leslie Kaelbling. Graph element networks: adaptive, structured computation and memory. In *International Conference on Machine Learning*, pp. 212–222. PMLR, 2019.

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

Emily Alsentzer, Samuel Finlayson, Michelle Li, and Marinka Zitnik. Subgraph neural networks. *Advances in Neural Information Processing Systems*, 33:8017–8029, 2020.

Saakaar Bhatnagar, Yaser Afshar, Shaowu Pan, Karthik Duraisamy, and Shailendra Kaushik. Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64:525–545, 2019.

Steven L Brunton, Bernd R Noack, and Petros Koumoutsakos. Machine learning for fluid mechanics. *Annual review of fluid mechanics*, 52:477–508, 2020.

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

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

Hongyang Gao and Shuiwang Ji. Graph u-nets. In *international conference on machine learning*, pp. 2083–2092. PMLR, 2019.

Zhenglin Geng, Daniel Johnson, and Ronald Fedkiw. Coercing machine learning to output physically accurate results. *Journal of Computational Physics*, 406:109099, 2020.

Laurent Graftieaux, Marc Michard, and Nathalie Grosjean. Combining piv, pod and vortex identification algorithms for the study of unsteady turbulent swirling flows. *Measurement Science and technology*, 12(9):1422, 2001.

Daniele Grattarola, Daniele Zambon, Filippo Maria Bianchi, and Cesare Alippi. Understanding pooling in graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

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

Xu Han, Han Gao, Tobias Pffaf, Jian-Xun Wang, and Li-Ping Liu. Predicting physics in mesh-reduced space with temporal attention. *arXiv preprint arXiv:2201.09113*, 2022.

Qinglin He. Observations of vortex formation in the mould of a continuous slab caster. *ISIJ international*, 33(2):343–345, 1993.

Quercus Hernández, Alberto Badías, David González, Francisco Chinesta, and Elías Cueto. Structure-preserving neural networks. *Journal of Computational Physics*, 426:109950, 2021.

Philipp Holl, Vladlen Koltun, and Nils Thuerey. Learning to control pdes with differentiable physics. *arXiv preprint arXiv:2001.07457*, 2020.

Vivianne Holmén. Methods for vortex identification. *Master's Theses in Mathematical Sciences*, 2012.

Tyler W Hughes, Ian AD Williamson, Momchil Minkov, and Shanhui Fan. Wave physics as an analog recurrent neural network. *Science advances*, 5(12):eaay6946, 2019.

Dmitrii Kochkov, Jamie A Smith, Ayya Alieva, Qing Wang, Michael P Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021.

Anthony Leonard. Vortex methods for flow simulation. *Journal of Computational Physics*, 37(3): 289–335, 1980.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on artificial intelligence*, 2018.

Arvind T Mohan, Nicholas Lubbers, Daniel Livescu, and Michael Chertkov. Embedding hard physical constraints in neural network coarse-graining of 3d turbulence. *arXiv preprint arXiv:2002.00021*, 2020.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W Battaglia. Learning mesh-based simulation with graph networks. *arXiv preprint arXiv:2010.03409*, 2020.

Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

Stephan Rasp and Nils Thuerey. Data-driven medium-range weather prediction with a resnet pre-trained on climate simulations: A new model for weatherbench. *Journal of Advances in Modeling Earth Systems*, 13(2):e2020MS002405, 2021.

Franz M Rohrhofer, Stefan Posch, Clemens Gößnitzer, and Bernhard C Geiger. Understanding the difficulty of training physics-informed neural networks on dynamical systems. *arXiv preprint arXiv:2203.13648*, 2022.

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

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pp. 8459–8468. PMLR, 2020.

Jean Michel Sellier, Gaétan Marceau Caron, and Jacob Leygonie. Signed particles and neural networks, towards efficient simulations of quantum systems. *Journal of Computational Physics*, 387: 154–162, 2019.

Gregory H Teichert, Anirudh R Natarajan, Anton Van der Ven, and Krishna Garikipati. Machine learning materials physics: Integrable deep neural networks enable scale bridging by learning free energy functions. *Computer Methods in Applied Mechanics and Engineering*, 353:201–216, 2019.

Nils Thuerey, Konstantin Weißenow, Lukas Prantl, and Xiangyu Hu. Deep learning methods for reynolds-averaged navier–stokes simulations of airfoil flows. *AIAA Journal*, 58(1):25–36, 2020.

Kiwon Um, Xiangyu Hu, and Nils Thuerey. Liquid splash modeling with neural networks. In *Computer Graphics Forum*, volume 37, pp. 171–182. Wiley Online Library, 2018.

Enrique R Vivoni, Valeri Y Ivanov, Rafael L Bras, and Dara Entekhabi. Generation of triangulated irregular networks based on hydrological similarity. *Journal of hydrologic engineering*, 9(4): 288–302, 2004.

DF Watson and GM Philip. Systematic triangulations. *Computer vision, graphics, and image processing*, 26(2):217–223, 1984.

Eric F Wood, Murugesu Sivapalan, and Keith Beven. Similarity and scale in catchment storm response. *Reviews of Geophysics*, 28(1):1–18, 1990.

PJ Zandbergen and D Dijkstra. Von kármán swirling flows. *Annual review of fluid mechanics*, 19 (1):465–491, 1987.

A Related Works

The rapid emergence of machine learning techniques is creating new opportunities for solving identification problems of physical systems by statistically exploring their underlying structures. These techniques have found applications in various areas of physical systems, including quantum physics Sellier et al. (2019), thermodynamics Hernández et al. (2021), material science Teichert et al. (2019), rigid body control Geng et al. (2020), Lagrangian systems Cranmer et al. (2020), and Hamiltonian systems Greydanus et al. (2019). Specifically, in the field of fluid mechanics, machine learning offers a wealth of techniques to extract information from data that could be translated into knowledge about the underlying fluid field, as well as exhibits its ability to augment domain knowledge and automate tasks related to flow control and optimization Brunton et al. (2020); Holl et al. (2020).

Particular for computational fluid dynamics, there is a considerable interest in using neural networks for accelerating simulations of aerodynamics Bhatnagar et al. (2019) or turbulent flows Kochkov et al. (2021). Tasks such as airfoil design Thuerey et al. (2020), weather prediction Rasp & Thuerey (2021), and graphic visualizations for Um et al. (2018) take the advantages of these fast predictions and differentiable learned models. While many of these methods use convolutional networks on 2D grids, recently GNN-based learned simulators, in particular, have shown to be a very flexible approach, which can model a wide range of systems, from articulated dynamics Sanchez-Gonzalez et al. (2018) to dynamics of particle systems Sanchez-Gonzalez et al. (2020). However, as mentioned earlier, the scalability of GNNs to high-dimensional and high-resolution simulations remains a challenge. A recent approach proposed by Fortunato et.al. Fortunato et al. (2022) proposed a possible solution by learned a mapping function from high-dimensional to low-dimensional mesh graph. However, to train the mapping function, it needs to have mesh data in two different dimensionalities, which may not always be available. Another potential approach is to learn a low-dimentional mesh graph that can best represent the full mesh graph Gao & Ji (2019); Alsentzer et al. (2020); Grattarola et al. (2022); Han et al. (2022). However, these methods are either limited to graph classification tasks or lack guarantees that the subgraph captures desirable properties.

Recently, there has been a surge of research aimed at effectively learning fluid dynamics by integrating physical priors into the learning framework. Examples include encoding the Navier-Stokes equations Raissi & Karniadakis (2018), incorporating the concept of an incompressible fluid Mohan et al. (2020), and establishing a correspondence between the dynamics of wave-based physical phenomena and the computation in a recurrent neural network (RNN) Hughes et al. (2019). Most of these approaches fall under the category of Physics-Informed Neural Networks (PINN). However, this type of method often faces convergence issues Rohrhofer et al. (2022) and requires careful design of network architectures.

In this work, we aim to tackle fluid dynamic prediction challenges by integrating two powerful techniques, namely subgraph networks and physics-informed learning. Our proposed graph-based model employs physical-informed prior knowledge to learn subgraphs that exist only in mesh-reduced space. This not only leads to significant reductions in computational expenses but also maintains crucial physical information, thereby achieving the best of both worlds.

B Dataset Details

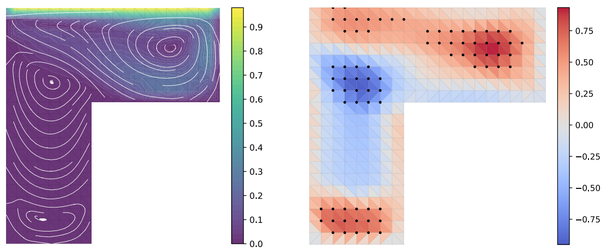
We use OpenFOAM solver to generate our simulation data. The governing equations for incompressible flows, namely the continuity and momentum equations, are numerically solved using an unstructured cell-centered collocated grid finite volume method. Integrated over a small control volume, which in this case is a triangular cell of the mesh, the equations are written as:

$$\nabla \cdot \mathbf{u} = 0, \quad \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u}, \tag{2}$$

where $\mathbf{u}$ is the flow velocity, ρ is the (mass) density, ν is the kinematic viscosity, and p is the pressure.

B.1 Lid-driven Cavity Flow

This dataset simulates the flow inside a cavity on a fixed 2D Eulerian mesh, where the top wall moves in the x direction at a constant speed. The dataset consists of three node types: fluid nodes,


(a) Lid-driven cavity flow

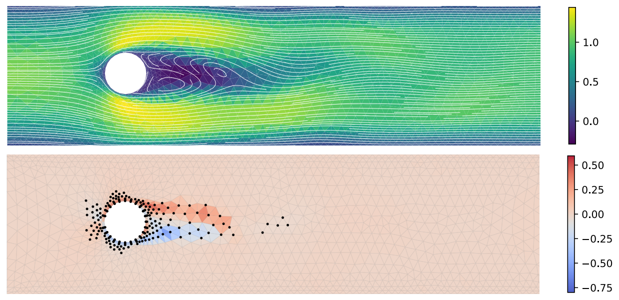

(b) Cylinder flow

Figure 3: Illustration of velocity fields (streamlines are colored in white) of two fluid dataset and corresponding physical-informed priors (selected pivotal nodes are colored in black)

side wall nodes, and top wall nodes. A typical lid-driven cavity problem assumes a rectangular cavity. We let the height of the cavity vary between 0.5 to 4, while the cavity width remain constant at 1. In order to make the dataset more challenging, a notch is added to the bottom right corner of the cavity resulting in an irregular shaped cavity. The dimensions of the indented notch vary between 0 and 0.8 for both width and height. When both the width and height of the notch are 0, the cavity takes on a regular rectangular shape. The Reynolds number varies between 209 to 2000. The dataset includes 1497 data points recorded at steady state. Figure 3(a) illustrates velocity fields of one of the data and the corresponding physical-informed prior values.

B.2 Cylinder Flow

This dataset simulates the flow around a cylinder on a fixed 2D Eulerian mesh. In all fluid domains, the node type distinguishes fluid nodes, wall nodes and inflow/outflow boundary nodes. The inlet boundary conditions are given by a prescribed parabolic profile, $u_{in} = u_0[1 - 4(y/H)]$ where u_0 and H are the centerline velocity and the distance between the sidewalls, respectively. The simulations were generated for various Reynolds numbers (Re), centerline inflow velocity, and diameter (D) of the cylinder. The Re values ranged from 45 to 250, while the H/D values ranged from 2 to 8. The dataset consists of 1000 training data, 100 validation data, and 100 testing data, each containing 60 time steps sampled at $\Delta t = 0.05$ and 300 time steps sampled at $\Delta t = 0.01$. To test the generalizability of the model, 35 additional testing data were generated, each containing two cylinders randomly placed in the domain. Figure 3(b) illustrates velocity fields of one of the data and the corresponding physical-informed prior values.

C Experiment Details

C.1 Evaluation Metrics

To quantitatively evaluate the performance of the autoencoder and predictor models, we compute the root mean squared error (RMSE) of the velocities in two directions. To further assess the performance on vortex detection, we define two metrics: vortex detection rate (VDR) and mean distance to detected vortices (MDDV). To compute these two metrics, we need to first identify the location of the vortices in both the ground truth and the reconstruction/prediction velocity field. We use linear spatial interpolation to compute the velocities in a 1000×1000 grid and find the location of the vortices on the grid (using the method describe in Holmén (2012)). For a vortex p_i in the ground truth, it is detected if there is a vortex q_j found in the predicted velocity field within a distance threshold d. The detection indicator for p_i is:

$$\mathrm{DI}_i = 1\{\min(d_{i0}, ..., d_{iQ}) \le d\}. \tag{3}$$

Denote the number of vortices in ground truth and prediction as P and Q, the distance between vortex p_i and q_j as d_{ij}. The metrics are defined as:

$$\begin{aligned} \mathrm{VDR}@d &= \frac{\sum_{i=1}^{P} \mathrm{DI}_i}{P} \\ \mathrm{MDDV}@d &= \sum_{i=1}^{P} \mathrm{DI}_i \cdot \min(d_{i0}, ..., d_{iQ}) \end{aligned} \tag{4}$$

For cylinder flow data, the region of interest (ROI) is the area surrounding the cylinder and to the right of the cylinder. We define a ROI box that contains this area. Given cylinder center c and radius r, the top left and bottom right corners of the box are $(c_x - r - 0.05, c_y + r + 0.05)$ and $(c_x + r + 0.45, c_y - r - 0.05)$. For both reconstruction and dynamics prediction, we calculate the RMSE, VDR and MDDV within the ROI box as well as the global RMSE.

C.2 Reconstruction via Subgraph

Mesh graph autoencoders are trained on both datasets with the aim of summarizing the full mesh graph into a subgraph. The MLPs in the MGNs consisted of two hidden layers and an output size of 128, except for the output layers of the MGNs. The output dimension for the first MGN is 4, indicating that each node in the subgraph is represented by a 4-dimensional vector. For the second MGN, the output dimension corresponds to the prediction dimension.

C.2.1 Lid-driven Cavity Flow

For lid-driven cavity flow, we trained 150K steps with an exponential decay of learning rate from 10^{-4} to 10^{-6} over the last 100K steps. Since the number of nodes varies from case to case, we do not fix the number of subgraph nodes and use half of the total nodes in the original mesh graph. We use a threshold $d = 0.001$ to compute VDR and MVVD.

The reconstruction results are listed in Table 1. Our model outperforms GMR_GMUS in all evaluation metrics when IDW interpolation is used. The use of TIN interpolation improves the results significantly by reducing RMSE by over 30%. Although our model's global_RMSE is marginally worse than GMR_GMUS, it has a notable advantage in vortex-related metrics, with 11% improvement in VDR and 26% reduction in MDDV. This suggests that our model produces better reconstruction results near vortices, an area that a random node selector would neglect. Figure 4 illustrates the reconstruction results of two cases from the lid-driven cavity flow test data. In case (a), the fluid flow in the top right corner is confined by a narrow simulation wall and the change of flow direction is abrupt. Our model is successful in handing this case and identifying the vortex. There is another vortex in the bottom low-velocity region and our model can also accurately detect it. In case (b), although the cavity height is deeper and two more vortices appear in the bottom region, our model can still reconstruct the case accurately and locate all vertices.

C.2.2 Cylinder Flow

For cylinder flow, we trained 600K steps with an exponential decay of learning rate from 10^{-4} to 10^{-6} over the last 400K steps. We fix the number of subgraph nodes to be 512. We report both the global RMSE and the RMSE within the ROI box. We use a threshold $d = 0.001$ to compute VDR. The MDDV is not reported because the value is 0 for all cases listed in the table.

The reconstruction results are listed in Table 2. Our model outperforms GMR_GMUS with a reduction of approximately 26% in global_RMSE, 55% in ROI_RMSE, and an 81% improvement in VDR when IDW interpolation is employed. The model performs better in the vortex regions in comparison to the lid-driven cavity flow cases as cylinder flow involves a much larger simulation system where a random node selector will sample proportionally fewer nodes in vortex regions. Therefore, the results demonstrate that PiP extractor is better suited for large simulation systems with relatively small ROI regions. We also observe an improvement in both global_RMSE and ROI_RMSE when TIN interpolation is applied. The VDR score is less sensitive to the interpolation method. We also test the methods with varying simulation time step Δt and a similar trend is observed. With TIN interpolation and $\Delta t = 0.01$, our model yields a 20% reduction in global_RMSE, 49% reduction in ROI_RMSE, and a 59% increase in VDR. The high performance of our model in ROI demonstrates that PiP extractor is effective in selecting pivotal nodes for summarizing the whole mesh graph with an adaptive concentration on vortex regions.

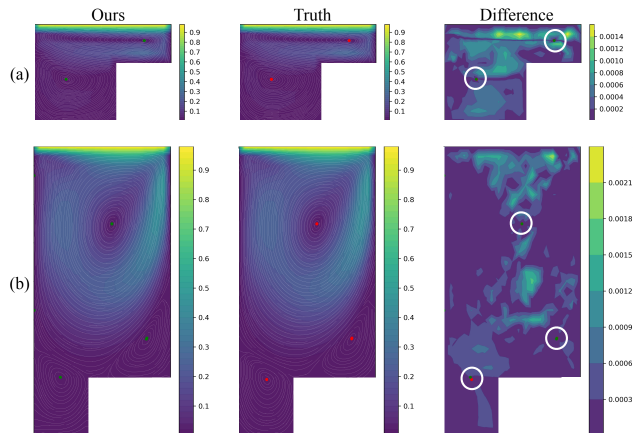


Figure 4: Simulation results of our method tested on lid-driven cavity flow dataset. Vortex centers are extract from velocity fields, which are color in green and red to denote predicted and ground-truth, respectively.

Table 1: Reconstruction results for Lid-driven Cavity Flow dataset.

	global_RMSE (x10^{-3})		VDR @0.001 $\uparrow$	MDDV @0.001 $\downarrow$ (x10^{-3})
	V_x	V_y		
GMR_GMUS(IDW)	2.19	0.96	0.61	0.23
GMR_GMUS(TIN)	1.23	0.68	0.65	0.19
Ours(IDW)	1.82	1.00	0.65	0.18
Ours(TIN)	1.26	0.70	**0.72**	**0.14**

Table 2: Reconstruction results for Cylinder Flow dataset.

	Δt (s)	global_RMSE (x10^{-3})		ROI_RMSE (x10^{-3})		VDR @0.001 $\uparrow$
		V_x	V_y	V_x	V_y	
GMR_GMUS (IDW)	0.05	4.7	2.2	8.5	4.3	0.48
GMR_GMUS (TIN)	0.05	4.3	2.1	7.8	4.0	0.49
	0.01	4.1	1.9	7.5	3.7	0.53
Ours (IDW)	0.05	3.8	1.5	4.4	1.7	**0.87**
Ours (TIN)	0.05	**3.5**	**1.4**	**4.0**	**1.6**	0.86
	0.01	**3.5**	**1.4**	**4.2**	**1.7**	**0.84**

Table 3: Dynamics prediction results for 300-step rollout with $\Delta t = 0.01$ tested in Cylinder Flow dataset.

	N_node	global_RMSE (x10^{-3})		ROI_RMSE (x10^{-3})		VDR @0.02 $\uparrow$	MDDV @0.02 $\downarrow$ (x10^{-3})
		Vx	Vy	Vx	Vy		
FG	$\sim$1885	75.3	42.1	98.1	76.0	0.56	8.7
GMR_GMUS	512	125.9	57.5	216.1	100.6	0.38	10.2
Ours	512	114.9	60.2	183.5	114.4	**0.46**	**9.8**

C.3 Time-series Prediction

To learn the flow dynamic predictor in mesh-reduced space, we use the trained autoencoder to generate ground truth data pairs (Z_{t,S_t}, Z_{t+1,S_t}). The autoencoders we used are the ones with $t = 0.01$ and TIN interpolation. The dynamics predictor is a separate MGN with input and output dimension being 4. The MLPs in the MGN has two hidden layers with output size of 128. Since the subgraph has a reduced size compared to the full mesh graph, we use only 8 message passing steps in the MGN instead of 15. The predictor is trained for 3M steps with the learning rate decaying from 10^{-4} to 10^{-5} in the last 2M steps. Since dynamics prediction is substantially more difficult than reconstruction, we use a larger threshold $d = 0.02$ to compute VDR and MDDV.

The results of the 300-step rollout are presented in Table 3. Our PiP extractor model produces superior performance in comparison to GMR_GMUS with a random extractor, with a reduction of 8.7% in global RMSE and 15.1% in ROI_RMSE for V_x. In spite of the limited improvement in RMSE, our model demonstrates a remarkable 21% enhancement in VDR, indicating that the predicted velocity field by our model is more efficient in identifying vortices. Additionally, we compare our model with a full graph (FG) dynamics predictor, which is a simple MGN. With a 73% decrease in the number of nodes, our predictor only falls 10% behind of the FG model. Although FG achieves a significantly lower velocity RMSE, it is mostly accurate in less important regions and it is not efficient for very large system

Figure 5 depicts the prediction rollouts for two different scenarios of our model in comparison to the ground truth, where our model demonstrates accurate and stable rollout predictions. The color of the mesh indicates the magnitude of V_x, and the red dots signify the selected pivotal nodes. In case (a), the cylinder is large and the flow becomes unsteady, resulting in the vortex shedding phenomenon where alternating vortices are created. Our model successfully learns the dynamics and correctly predicts the beginning and the periodic progression of vortex shedding. In case (b), where the cylinder is small, the flow is laminar and steady, and our model can generate predictions that resemble the ground truth. For both cases, we observe that the pivotal nodes adapt to the change in the velocity field and consistently locate near the cylinder ROI region. This targeted and adaptive design makes our model superior for vortex prediction.

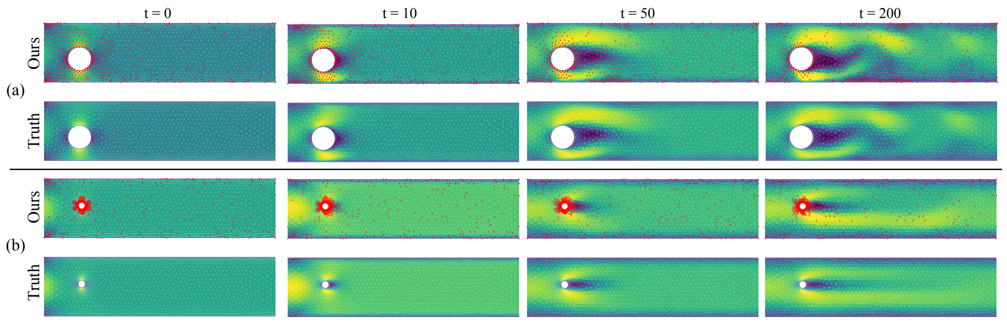


Figure 5: Velocity field simulation results of our method versus ground truth tested on Cylinder Flow dataset. Pivotal nodes used in the mesh-reduced space are colored in red at each time step. Our model extracts informative nodes at each time step and accurately predicts long rollout sequences under different design parameters.