Multiresolution Mesh Networks For Learning Dynamical Fluid Simulations

Viet Bach Nguyen *
FPT AI Center
Hanoi, Vietnam
bachnv4@fsoft.com.vn

Truong Son Hy *
University of California San Diego
La Jolla, CA 92093, US
tshy@ucsd.edu

Long Tran-Thanh
University of Warwick
Coventry CV4 7AL, UK
Long.Tran-Thanh@warwick.ac.uk

Risi Kondor
University of Chicago
Chicago, IL 60637, US
risi@uchicago.edu

Abstract

In this paper, we introduce Multiresolution Mesh Networks-enhanced MeshGraphNets (MGN-MeshGraphNet) for learning mesh-based dynamical fluid simulations. The novelty of our proposal comes from the ability to capture multiscale structures of fluid dynamics via a learnable coarse-graining mechanism on meshes (i.e. mesh multiresolution), along with long-range dependencies between multiple timesteps and resolutions for robust prediction. Our proposed method has shown competitive numerical results in comparison with other machine learning approaches based on graph neural networks. Given the flexibility of our data-driven approach for building mesh multiresolution, our method has better generalizability for new fluid dynamical simulations outside of the training data while attaining high accuracies on multiple resolutions and computational speedup compared to the existing PDE numerical solvers of Navier–Stokes equation.

1 Introduction

Fluid dynamic simulation is a significant component of numerous fields of science and engineering, including but not limited to biology, astrophysics, geophysics, environmental science, and aviation engineering (Panthi et al. [2007], Economon et al. [2016], Yazid et al. [2009], Ramamurti and Sandberg [2001], Schwarzbach et al. [2011]). Accurate and efficient modeling of fluid dynamics is essential for gaining insight into research and engineering problems in these fields.

There exists solid approaches to the problem of fluid dynamic simulation, both from the field of physics and computer science or machine learning. Numerical solvers model the underlying differential equations of the simulated fluid dynamics (Kim and Boysan [1999]), while partial differential equations (PDE)-based deep learning models and other geometric model (Groueix et al. [2018], Tathawadekar et al. [2021], de Avila Belbute-Peres et al. [2020], Shoaib et al. [2022]) efficiently model the dynamics. However, each of these respective approaches has their downsides. Numerical solvers and fluid function models are precise and robust, yet struggle to quickly produce simulations and cannot scale to large fluid dynamic environments (Karniadakis and Sherwin [2005]). On the other hand, geometric deep learning models are faster and effient, but bottlenecked by approximation errors and inconsistency problems; more importantly, many models cannot effectively generalize to environments that can be modeled by the PDEs but are not within the training data distribution.

*Co-first authors.

NeurIPS 2021 AI for Science Workshop.

The fluid dynamic environment in question can be modeled and vectorized in several ways, to which each kinds of approximation models seek to exploit. Captured visual snapshots of the dynamics can be feed through a Convolutional Neural Network (CNN)-based model with an autoregressive generator component, iteratively generating predictions for the next time step in the simulation (Guo et al. [2016], Tompson et al. [2017], Lee and You [2019]). A more geometric approach to modeling the parameters of the environment is on the rise, notably the representation of fluid environments as meshes. Meshes are dynamic non-Euclidean data format that are a specific variation of the general graph data, enabling physics simulation with graph neural networks (Pfaff et al. [2021]). Mesh-based graph neural network models can both underpin the underlying dynamics of internal systems, while also capable of estimating external dynamics such as contact and collision (Pfaff et al. [2021]).

With the motivation of using graph neural networks to simulate fluid dynamics in mind, we propose an improved and more robust multiresolution version of the original meshgraphnets (Battaglia). Our architecture models the fluid environments on multiple resolutions and granularity, effectively capturing both local dynamics and global interactions of the simulated environments. Our contributions are summarized as follow:

- Utilization of Multiple-resolution Hierarchical Graph Neural Networks (MGNN) in the meshgraphnet framework to increase simulation robustness and expose dynamical properties on multiple simulation scale.
- A more generalized remeshing procedure based on clustering of mesh nodes (i.e., non-Euclidean simulation subunit) that allows dynamical fluid environment details.
- Experiments with competitive results on basic fluid dynamics datasets including cylindral interactions and flow visualizations.

2 Related work

In the field of computational fluid dynamics, highh resolution simulations are often very slow while learned model can provide faster approximations, resulting in lower turnaround time in applications in science and engineering (Albergo et al. [2019], Afshar et al. [2019], Abbott et al. [2022]). Traditionally, most deep neural networks architectures used for predicting fluid dynamics have utilized convolutional layers; CNN-based solvers are from one to two times faster than numerical solvers (Guo et al. [2016], Tompson et al. [2017], Lee and You [2019], Kim et al. [2019], Wiewel et al. [2019], Fotiadis et al. [2020]), these models also have shown good extrapolation to unseen domain geometry and initial conditions (Thuerey et al. [2018], Valencia et al. [2020]).

More recently, Graph Neural Networks (GNNs) have been used to simulate the motion of discrete systems of solid particles, deformeable solids, and fluids discretised into Lagrangian particles (Battaglia et al. [2016], Chang et al. [2017], Mrowca et al. [2018], Sanchez-Gonzalez et al. [2020]). Our work is heavily based upon the basic MeshGraphNets model from Pfaff et al. [2021], where the model operates on a mesh-based domain for physical fluids dynamic simulation. However efficient and accurate, the original MeshGraphNets does not make use of hierarchical structures to increase generalisation. Other works closely related to ours include Li et al. [2020] and Liu et al. [2021], where multiresolution GNNs are used to infer steady solutions, albeit with a rather simple pooling solution and absence of extrapolation.

Our model is a combination of the hierarchical multiresolution model used for molecule generation and generalized task from Hy and Kondor [2021]. The two frameworks are combined to create a mixed, hierarchy- and multiresolution-aware MeshGraphNets model.

3 Method

We describe the state of the fluid dynamical system at time t using a simulation mesh $M^t = (V, E^M)$ with nodes V connected by mesh edges E^M (Pfaff et al. [2021], Sanchez-Gonzalez et al. [2020]). The features of each node $i \in V$ in the mesh include a reference mesh-space coordinate $\mathbf{u}_i$ which spans the simulation mesh, and additional dynamical quantities $\mathbf{q}_i$ that we want to model. In particular, we experimented mostly with Eulerian system CylinderFlow where $\mathbf{q}_i$ sample continuous fields such as velocity over a fixed mesh.

The task is to learn a multiresolution forward model, predicting the aforementioned dynamical quantities $\mathbf{u}_i$ at time $t+1$ given the current mesh M^t and optionally a history of previous meshes. The historical information can be directly encoded within the feature vector of each node in the mesh. We use the same mesh notation and conventions as from Battaglia's, adding a multiresolution mechanism on top of the existing MeshGraphNet framework.

3.1 Multiresolution Graph Networks

The Multiresolution Graph Networks (MGN) and its generative cousin have been proposed by Hy and Kondor [2021]. Here, we reintroduce the general construction of MGN.

An L-level Multiresolution Graph Network (MGN) of a graph G consists of $L-1$ tuples of three network components $\{(\boldsymbol{c}^{(\ell)}, \boldsymbol{e}^{(\ell)}, \boldsymbol{p}^{(\ell)})\}_{\ell=2}^{L}$. The ℓ-th tuple encodes $G^{(\ell)}$ and transforms it into a lower resolution graph $G^{(\ell-1)}$ in the higher level. Each of these network components has a separate set of learnable parameters $(\boldsymbol{\theta}_c^{(\ell)}, \boldsymbol{\theta}_e^{(\ell)}, \boldsymbol{\theta}_p^{(\ell)})$. For simplicity, we collectively denote the learnable parameters as $\boldsymbol{\theta}$ and drop the superscript. The network components are defined as follows:

1. Clustering procedure $\boldsymbol{c}(G^{(\ell)}; \boldsymbol{\theta})$, which partitions graph $G^{(\ell)}$ into K clusters $V_1^{(\ell)}, ..., V_K^{(\ell)}$. Each cluster is an induced subgraph $G_k^{(\ell)}$ of $G^{(\ell)}$ with adjacency matrix $A_k^{(\ell)}$.
2. Encoder $\boldsymbol{e}(G_k^{(\ell)}; \boldsymbol{\theta})$, which is a permutation-equivariant graph neural network that takes as input the subgraph $G_k^{(\ell)}$ and outputs a set of node latents $Z_k^{(\ell)}$ represented as a matrix $|V_k^{(\ell)}| \times d_z$.
3. Pooling network $\boldsymbol{p}(Z_k^{(\ell)}; \boldsymbol{\theta})$, which is a permutation-invariant neural network that takes the set of nodes latents $Z_k^{(\ell)}$ and outputs a single cluster latent $\tilde{z}_k^{(\ell)}$.

3.2 Integrating the MGN layer into MeshGraphNet

We preserve the original implementation of the MeshGraphNet model, while adding several multiresolution layers on top of the regular Encoder-Processor-Decoder stack. To ensure that the MGN layer is compatible with the rest of the network and that the output is in the same format as a standard graph latent embedding, we devised a method to augment the original input graph. Given original latent graph $G^{(1)} = (V, E, A)$ with node set V of size $|V| \times d_z$, edge set E, and adjacency matrix $A \in \mathbb{R}^{|V| \times d_z}$, we generate graphs $G^{(2)}, ..., G^{(L)}$ where $L-1$ is the number of coarsened graph levels. The multiresolution-augmented graph G_a is generated as follows:

1. At each coarsening step, we save the assignment matrix $\pi^{(\ell)} \in R^{d^{(\ell-1)} \times d^{(\ell)}}$.
2. Given coarsened node feature matrix $Z^{(\ell)}$ at level ℓ, we uniformly duplicate these features to previous level $\ell - 1$ by calculating the matrix product:
$$A_\ell^{(\ell-1)} = \pi^{(\ell)} \times Z^{(\ell)}$$
3. The matrix A_ℓ is iteratively augmented by multiplying with the assignment matrix at each resolution level until reaching the initial graph's node count:
$$A_\ell = \pi^{(2)} \times ... \times \pi^{(\ell-1)} \times \pi^{(\ell)} \times Z^{(\ell)}$$
4. The augmented coarsening matrices are finally concatenated to the original graph latent along the embedding dimension to generate a final augmented graph latent of size $|V| \times d_a$, where $d_a = L \times d_z$:
$$A = V \oplus A_2 \oplus ... \oplus A_\ell$$

The augmented matrix $A \in R^{|V| \times d_a}$ of graph G serves as the multiresolution-aware input latent to the decoder of the original MeshGraphNet architecture.

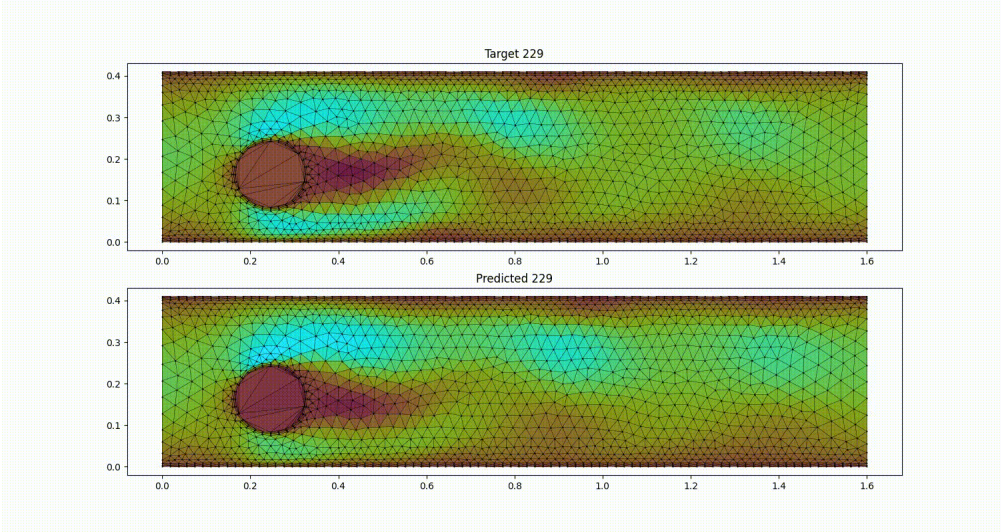

Figure 1: Rollout visual simulation of the MGN-MeshGraphNets model on one sample of the CylinderFlow dataset. Figure is one frame cut from the rollout video generated by the model.

4 Experiments

We tested our MGN-MeshGraphNets model on one experimental domain and compare the results to the baseline MeshGraphNets model. Our model ran slightly faster than groud truth simulator while being able to produce accurate rollouts on the tested domain.

Dataset We tested the model on the CylinderFlow dataset from Battaglia's original MeshGraphNets model. The parsing procedure and node parameterization method for modeling the dynamical quantities $\mathbf{q}_i$ of the Eulerian system are identical to the original paper (Pfaff et al. [2021]), where continuous fields such as velocity and direction are sampled at the fixed mesh nodes.

Model configuration Our adaptation of the MGN layer into the MeshGraphNets framework reimplemented (`https://github.com/echowve/meshGraphNets_pytorch`) means that the model is identical to the original MeshGraphNets, given that no clustering parameter is passed. The original MeshGraphNets model is trained and compared with the multiresolution-aware MGN-MeshGraphNets model. Both models have identical number of messing passing layers and model parameters (including embedding size of 128), processing the same dataset, and trained using the same optimizer and learning rate.

Both models are trained using the Adam optimizer, $lr = 10^{-4}$. Two configurations of the MGN-based model are tested, one with a single coarsening of the graph to a constant number of nodes (64 nodes, one level of coarsening), and another with two layers of coarsening at 64 and 32 nodes, respectively.

Results Testing on the reimplemented MeshGraphNets framework in PyTorch, under the same training time and computational power (single GPU, P100), we obtained better results with the multiresolution-based model compared to the baseline model. Videos of rollouts indicate that the predicted dynamics are plausible and almost identical to the ground truth (Figure 1). Table 1 shows 1-step prediction and rollout errors in the CylinderFlow dataset. The experiments and results double as an ablation study of the multiresolution coarsening layers as the zero-th resolution model is identical to the original MeshGraphNet model (Pfaff et al. [2021]).

5 Conclusion

MGN-MeshGraphNets is a novel multiresolution-aware enhancement to mesh-based graph neural networks for approximating fluid dynamics models. The clustering procedure from the MGN model allows the MeshGraphNets-based model to dynamically learn clusters that best describe the original physical meshes. The hierarchical multiresolution modality, coupled with mechanisms to

Table 1: Root mean square error (RMSE, $\times 10^{-2}$) error of our methods on CylinderFlow, compared to the original MeshGraphNets, for a single prediction step (1-step), 50-step rollouts, and rollout of the whole trajectory.

Simulation Models				
Model Type	Coarsening	1-step	rollout-50	rollout-all
MeshGraphNets	0-layer	1.93	5.48	14.5
MGN-MeshGraphNets	1-layer, 64 nodes	**0.90**	**1.72**	6.75
MGN-MeshGraphNets	2-layer, 64-32 nodes	1.21	2.35	**6.43**

incorporate coarsened graph information into the original graph. The enriched graph latent enhances the representation of dynamic physical meshes, as the model predicts visually sensible and more loss-accurate dynamics. This work is a potentially important iterative improvement to mesh-based models for physical simulation, demonstrating the advantage of building graph and mesh networks with hierarchical structures.

References

Sanjay Kumar Panthi, N. R. Ramakrishnan, K. K. Pathak, and J. S. Chouhan. An analysis of springback in sheet metal bending using finite element method (fem). *Journal of Materials Processing Technology*, 186:120–124, 2007.

Thomas D. Economon, Francisco Palacios, Sean R. Copeland, Trent Lukaczyk, and Juan J. Alonso. Su2: An open-source suite for multiphysics simulation and design. *AIAA Journal*, 54:828–846, 2016.

Abdelaziz Yazid, Nabbou Abdelkader, and Hamouine Abdelmadjid Hamouine Abdelmadjid. A state-of-the-art review of the x-fem for computational fracture mechanics. *Applied Mathematical Modelling*, 33:4269–4282, 2009.

Ravi Ramamurti and William Sandberg. Simulation of flow about flapping airfoils using finite element incompressible flow solver. *AIAA Journal*, 39:253–260, 2001.

Christoph Schwarzbach, Ralph-Uwe Börner, and Klaus Spitzer. Three-dimensional adaptive higher order finite element simulation for geo-electromagnetics—a marine csem example. *Geophysical Journal International*, 187:63–74, 2011.

Sung-Eun Kim and F. E. Boysan. Application of cfd to environmental flows. *Journal of Wind Engineering and Industrial Aerodynamics*, 81:145–158, 1999.

Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation. *arXiv: Computer Vision and Pattern Recognition*, 2018.

Nilam Tathawadekar, Nguyen Anh Khoa Doan, Camilo F. Silva, and Nils Thuerey. Hybrid neural network pde solvers for reacting flows. 2021.

Filipe de Avila Belbute-Peres, Thomas D. Economon, and J. Zico Kolter. Combining differentiable pde solvers and graph neural networks for fluid flow prediction. *ArXiv*, abs/2007.04439, 2020.

Muhammad Shoaib, Marwan Abukhaled, Muhammad Asif Zahoor Raja, Muhammad Abdul Rehman Khan, Muhammad Tauseef Sabir, Kottakkaran Sooppy Nisar, and Iqra Iltaf. Heat and mass transfer analysis for unsteady three-dimensional flow of hybrid nanofluid over a stretching surface using supervised neural networks. In *Frontiers in Physics*, 2022.

George Em Karniadakis and Spencer J. Sherwin. Spectral/hp element methods for computational fluid dynamics. 2005.

Xiaoxiao Guo, Wei Li, and Francesco Iorio. Convolutional neural networks for steady flow approximation. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin. Accelerating eulerian fluid simulation with convolutional networks. *ArXiv*, abs/1607.03597, 2017.

Sangseung Lee and Donghyun You. Data-driven prediction of unsteady flow over a circular cylinder using deep learning. *Journal of Fluid Mechanics*, 879:217 – 254, 2019.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. *ArXiv*, abs/2010.03409, 2021.

M. S. Albergo, G. Kanwar, and P. E. Shanahan. Flow-based generative models for markov chain monte carlo in lattice field theory. *Physical Review D*, 100(3), 8 2019. doi: 10.1103/PhysRevD.100.034515.

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

Ryan Abbott, M. S. Albergo, Denis Boyda, Kyle Cranmer, Daniel C. Hackett, Gurtej Kanwar, Sébastien Racanière, Danilo Jimenez Rezende, Fernando Romero-L'opez, Phiala E. Shanahan, Betsy Tian, and Julian M Urban. Gauge-equivariant flow models for sampling in lattice field theories with pseudofermions. *ArXiv*, abs/2207.08945, 2022.

Byungsoo Kim, Vinicius C. Azevedo, Nils Thürey, Theodore Kim, Markus H. Gross, and Barbara Solenthaler. Deep fluids: A generative network for parameterized fluid simulations. *Computer Graphics Forum*, 38, 2019.

Steffen Wiewel, Moritz Becher, and Nils Thürey. Latent space physics: Towards learning the temporal evolution of fluid flow. *Computer Graphics Forum*, 38, 2019.

Stathi Fotiadis, Eduardo Pignatelli, Mario Lino Valencia, Chris D. Cantwell, Amos J. Storkey, and Anil Anthony Bharath. Comparing recurrent and convolutional neural networks for predicting wave propagation. *ArXiv*, abs/2002.08981, 2020.

Nils Thuerey, Konstantin Weissenow, Lukas Prantl, and Xiangyu Y. Hu. Deep learning methods for reynolds-averaged navier–stokes simulations of airfoil flows. *AIAA Journal*, 58:15–26, 2018.

Mario Lino Valencia, Chris D. Cantwell, Stathi Fotiadis, Eduardo Pignatelli, and Anil Anthony Bharath. Simulating surface wave dynamics with convolutional networks. *ArXiv*, abs/2012.00718, 2020.

Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. *ArXiv*, abs/1612.00222, 2016.

Michael Chang, Tomer David Ullman, Antonio Torralba, and Joshua B. Tenenbaum. A compositional object-based approach to learning physical dynamics. *ArXiv*, abs/1612.00341, 2017.

Damian Mrowca, Chengxu Zhuang, Elias Wang, Nick Haber, Li Fei-Fei, Joshua B. Tenenbaum, and Daniel Yamins. Flexible neural representation for physics prediction. *ArXiv*, abs/1806.08047, 2018.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8459–8468. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html`.

Zong-Yi Li, Nikola B. Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations. *ArXiv*, abs/2006.09535, 2020.

Wenzhuo Liu, Mouadh Yagoubi, and Marc Schoenauer. Multi-resolution graph neural networks for pde approximation. In *ICANN*, 2021.

Truong Son Hy and Risi Kondor. Multiresolution equivariant graph variational autoencoder. 2021.