MegaFlow2D: A Parametric Dataset for Machine Learning Super-resolution in Computational Fluid Dynamics Simulations

Wenzhuo Xu
wzxu@cmu.edu
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA

Noelia Grande Gutiérrez
ngrandeg@andrew.cmu.edu
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA

Christopher McComb
ccm@cmu.edu
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA

ABSTRACT

This paper introduces MegaFlow2D, a dataset of over 2 million snapshots of parameterized 2D fluid dynamics simulations of 3000 different external flow and internal flow configurations. It's worth noting that, simulation results on both low and high mesh resolutions are provided to facilitate the training of machine learning (ML) models for super-resolution purposes. This is the first large-scale multi-fidelity fluid dynamics dataset ever provided. We build the entire data generation and simulation workflow on open-source and efficient interfaces that can be utilized for a variety of data samples according to the user's specific needs. Finally, we provide a use case to demonstrate the potential value of the MegaFlow2D dataset in applications related to error correction.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Neural networks**; • **Computer systems organization** → *Embedded and cyber-physical systems*; • **Applied computing** → **Engineering**.

KEYWORDS

datasets, neural networks, computational fluid dynamics, discretization error

ACM Reference Format:
Wenzhuo Xu, Noelia Grande Gutiérrez, and Christopher McComb. 2023. MegaFlow2D: A Parametric Dataset for Machine Learning Super-resolution in Computational Fluid Dynamics Simulations. In *Cyber-Physical Systems and Internet of Things Week 2023 (CPS-IoT Week Workshops '23), May 9–12, 2023, San Antonio, TX, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3576914.3587552

1 INTRODUCTION

In recent years, the empirical success of machine learning in various fields has motivated researchers to apply such methods to fluid dynamics simulation [14]. The complexity of fluid dynamics induces a large variety of approaches in designing ML models to emulate computational fluid dynamics (CFD), often to serve as surrogate models. The most common approach is called the generative method, which directly solves the original mesh and boundary conditions with machine learning models [5, 10, 15, 28]. This often involves training a generative adversarial network to generate flow features from scratch. However, such methods are highly restricted in that the predictions learned by the network apply exclusively to the boundary condition and physical properties of the training data and thus generalize poorly to different geometry and boundary conditions [17]. More advanced techniques include neural operators [19, 20, 30], which learn a mapping between mesh and solution spaces via a partial differential equation (PDE)-like operator encoded in neural network structure. These neural operators solve the generalization issue at the cost of computational complexity, as it grows in $O(n^2)$ with respect to domain size.

Another approach that has recently drawn significant attention is the incorporation of numerical solvers directly into machine learning models. This consists of performing numerical simulations on a coarse mesh and then upsampling the coarse simulation into a highly-accurate dense solution via machine learning models, a process known as super-resolution [7, 18, 21, 27]. A significant advantage of this approach is that numerical solvers are used to handling complicated geometry and boundary conditions, which prove to be extremely difficult for machine learning models to capture accurately. An additional benefit of incorporating a numerical solver is that it regularizes the solution results and generates significantly lower errors compared to pure machine learning models [27].

Despite all the progress in developing innovative machine-learning models for fluid dynamics purposes, very little attention has been put towards providing a systematic and truly large-scale dataset for training and benchmarking neural network models. Several airfoil-based simulation datasets have been published[2, 3], but with only hundreds of samples, such datasets are still limited by size and variety of simulation scenarios. We also note that, for the specific task of CFD super-resolution, there is scarcely any public record of a dataset for which labels on both low and high-resolution simulations are provided for the same geometry. Datasets are often not prioritized as a research product [8], but very large datasets, such as IMAGENET[6] and WILDS[16], stand as one of the most important reasons behind the huge success of machine learning in tackling real-world problems.

Producing a truly large-scale CFD dataset is a challenging task as fluid dynamics simulations are computationally costly and highly sensitive to different boundary conditions. It is also difficult to provide a dataset with sufficient geometric variability to avoid overfitting for subsequent applications. To address this issue, we introduce MegaFlow2D, a dataset of 2,000 parametrically generated external flow configurations and 1,000 internal flow configurations. For each case, we discretize the domain into a high- and low-resolution mesh and perform a simulation. The time step for each simulation

CPS-IoT Week Workshops '23, May 9–12, 2023, San Antonio, TX, USA

ACM ISBN 979-8-4007-0049-1/23/05.
https://doi.org/10.1145/3576914.3587552

varies depending on the complexity of the flow field. With a total of 1,806,000 external flow snapshots and 1,250,000 internal flow snapshots, our dataset is over 100 times larger than any of its known counterparts [2, 3, 27], and more data can be easily filled into the dataset via a parametric simulation workflow that we distribute alongside the dataset.

The key contributions of this paper are:

1. We construct a parallel simulation pipeline for simulating and storing CFD case scenarios. Such a pipeline is entirely open-source and can be easily incorporated into machine learning modules in future work.
2. We release MegaFlow2D, with 3,000 distinctive geometries and a total of 3,056,000 solution snapshots. For each geometry simulation data on both low and high resolutions are provided, giving it a unique multi-fidelity feature for wider applications.
3. We demonstrate a use case for MegaFlow2D by training a brief error estimation network on the dataset and showing that the network produces accurate and diverse results without overfitting, illustrating the efficacy of the dataset.

2 DATASET & METHODOLOGY

We modify the open-source numerical solver Oasis [25], based on the FEniCS [1, 22–24] computing platform to run our simulation pipeline and geometry and mesh is generated via Gmsh [9]. We choose flow around circles and ellipses as the major content of our dataset as they are readily parameterizable but at the same time induce almost every external flow characteristic of interest, such as laminar sections around the obstacle, vortex behind the obstacle, and boundary layer flow. Metadata for every node within every simulation snapshot includes velocity in every geometry dimension, pressure value, and coordinates wrapped in a Pytorch Geometric (PyG) interface. The overall dataset characteristics are reported in Table. 1.

			Node statistics				Edge statistics			
Geometry	Num of sample	Mesh size	Max	Min	Average	std	Max	Min	Average	std
Circle	823999	2	116	74	98.76	10.18	622	362	518.22	61.60
		0.3	2799	2222	2671.3	98.4	8173	6384	7777.3	307.6
Ellipse	982000	2	128	79	103.98	11.33	694	396	549.76	68.21
		0.3	2847	2453	2728.9	57.8	8320	7092	7955.8	182.0
Nozzle	1190000	2	1072	948	1000.85	27.68	3022	2665	2815.09	78.48
		0.3	23481	21108	22136.2	613.4	69484	62436	65488.7	1816.6

Table 1: Dataset general statistics

2.1 Physical Properties

The physical laws underlying our dataset are represented by the incompressible Navier-Stokes equations, which can be expressed as:

$$\left\{ \begin{array}{l} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \nu \nabla^2 \mathbf{u} - \nabla p + \mathbf{f}, \\ \nabla \cdot \mathbf{u} = 0, \end{array} \right. \tag{1}$$

where $\mathbf{u}$ is the velocity, p is the pressure, $\mathbf{f}$ denotes the body force vector (which is not considered in the current simulation), and ν is the kinematic viscosity. A wide range of numerical methods has been proposed for solving these equations while balancing computational cost and accuracy. For constructing the dataset presented here, we adopt one of the most commonly used numerical methods, the Incremental Pressure Correction Scheme (IPCS) [4]. We adopt a high-performance implementation of the IPCS scheme as described in [25]. For time discretizations, this implementation uses semi-implicit Crank-Nicolson for the viscous term as well as for the convected velocity. The convecting velocity is discretized with an Adam-Bashforth scheme. The computation is divided into three steps, in the first step a tentative velocity vector is computed via Adam-Bashforth projected convecting velocity vector, as in Eq. 2a, the second step corrects the velocity as in Eq. 2b, and the third step corrects the pressure as in Eq 2c.

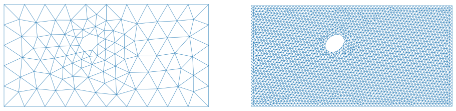
$$\frac{1}{\Delta t}(\mathbf{u}^* - \mathbf{u}^k) + ((1.5\mathbf{u}^k - 0.5\mathbf{u}^{k-1}) \cdot \nabla) \frac{\mathbf{u}^* + \mathbf{u}^k}{2} = \nu \nabla^2 \frac{\mathbf{u}^* + \mathbf{u}^k}{2} - \nabla p^*, \tag{2a}$$

$$\left\{ \begin{array}{l} \frac{1}{\Delta t}(\mathbf{u}^{k+1} - \mathbf{u}^*) = -\nabla \phi^{k+1}, \\ \nabla \cdot \mathbf{u}^{k+1} = 0, \end{array} \right. \tag{2b}$$

$$p^{k+1} = p^* + \phi^{k+1}. \tag{2c}$$

Our test cases consist of external and internal flow configurations. In the external flow scenario, we simulate flow around a two-dimensional body normal to a free-flow stream. We choose circles and ellipses as our primary geometry. This configuration represents a classical fluid dynamics problem relevant to engineering applications. At a certain Reynolds number (Re), the cylinder or ellipse would typically induce a Von Kármán vortex street, a well-studied yet complex fluid dynamics behavior that can be used for evaluating neural network performances. A rectangular domain length of at least three times the size of the obstacle diameter is constructed around the obstacle to allow for complete flow development. For simplicity, the length and width of the domain are fixed to 20 meters and 10 meters for all configurations. For generating a large amount of flow simulation data, the domain needs to be parameterized so that the parameters that define domain geometry (location and size) can be sampled from given probability distributions. The parameter space is specifically designed for each configuration so that variations can fully represent the geometry characteristics. Namely, for circles, we prescribe center location x_i and radius r. For ellipses, more parameters are required to describe the geometry fully, and we designate the center location x_i, long axis length a, short axis length b, and more importantly, angle of attack α. We limit the location of the obstacles so they are not too close to the domain boundary, and boundary conditions do not interfere with the flow field around the obstacle. Also, we assign more weight to locations near the inflow boundary so that more room is left for observing flow disturbance behind the obstacle. An example of one configuration with low and high-resolution mesh is shown in Fig 1. Through a mesh independence study, we discovered that simulation results demonstrated good convergence below a mesh scale of 0.7 meters, and gradually start to disperse on a mesh scale larger than 1.5 meters. Therefore we choose mesh sizes of 0.3 and 2 meters as the scaling factors for low- and high-resolution meshes respectively.

For internal flow, the geometry is specified based on a model of the FDA benchmark medical device for CFD validation [13], which provides detailed analysis for parameterizing a nozzle-shaped geometry. We present an example of the domain in Fig. 2.



(a) Low resolution mesh (b) High resolution mesh

Figure 1: Dataset mesh representataion

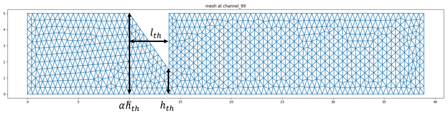
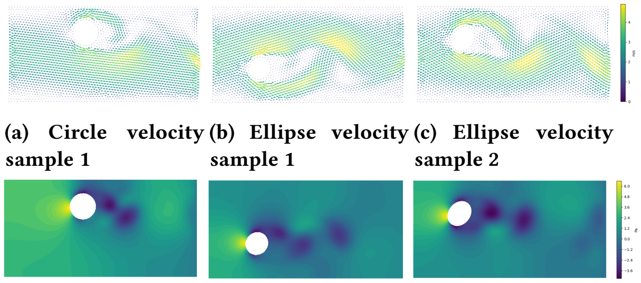


Figure 2: Nozzle shaped geometry, in which h_{th} describes throttle width, and l_{th} describes throttle length. Nozzle shape is specified via a fraction of nozzle width

We performed CFD simulations of the dataset configurations at Reynolds number $Re = 300$ with 0.01s as the time step for external flow and 0.002s as the time step for internal flow. Fluid properties were set to density $\rho = 1 \times 10^3 kg/m^3$ and dynamic viscosity $\nu = 1 \times 10^{-3} m^2/s$. We present a sequence of simulation results in Fig 3.



(a) Circle velocity sample 1 (b) Ellipse velocity sample 1 (c) Ellipse velocity sample 2

(d) Circle pressure sample 1 (e) Ellipse pressure sample 1 (f) Ellipse pressure sample 2

Figure 3: Examples of external flow dataset snapshots

2.2 Graphical Structure

Neural networks typically require highly structured data. However, the quality of information preserved is highly dependent on the way the CFD mesh is organized. Structured simulations can generally fit into the network directly by locating each node and its neighbors in the data matrix according to their relative position. However, generating a structured mesh for complex geometries is a challenging task even for modern meshing techniques. Instead, unstructured meshes are better able to represent the complex geometries that arise in applications like artery hemodynamics [11] and fuel cell channels [26] simulations. Previous researchers approached this problem by transforming the simulation domain into an image-like representation [21]. This transformation fits the graph into a designated size determined by the model, but it consequently erases all information about simulation resolution and geometric location. Graph structures are introduced into neural network models as a representation of logical connections, localizing object tracking locations, and chemical compounds [29]. The MegaFlow2D dataset uses a graph representation to maximize the information preserved during the transformation process. During the data processing step, we separate and extract geometrical and fluid property information from a single solution snapshot. Mesh topology is represented as an adjacency matrix while the geometric location of the nodes is stored in the nodal attributes. The label for each data point is defined as the solution of the high-resolution mesh interpolated onto the low-resolution mesh, including velocity value in x and y directions along with the pressure. Velocity and pressure are chosen as model labels as almost all other flow properties can be inferred from these two values. The structure of the data is demonstrated in Fig. 4.

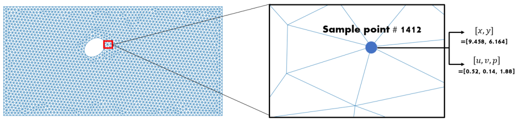


Figure 4: Data transformation of simulation snapshot. The data is split into topology, geometry, and flow features respectively.

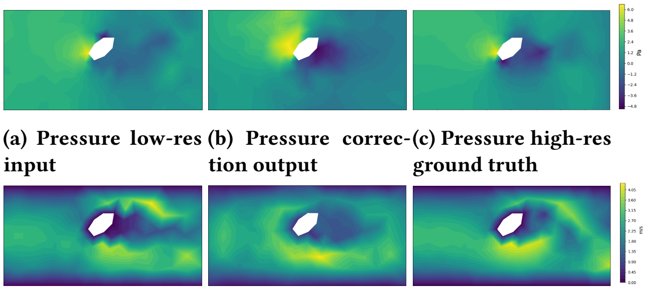
The dataset is distributed with utilities for extracting and operating on the underlying data.

3 POTENTIAL USE CASE DEMONSTRATION

MegaFlow2D is composed of high and low-resolution simulations, and the resolution gap (*i. e.* ratio of node counts) of 10 times makes the task of super-resolution extremely demanding. One of the most important challenges in super-resolution is calibrating the discretization error introduced by the coarse mesh. To briefly demonstrate this effect we construct a GraphSAGE[12] convolution model with four layers, and the layers have kernel depth of 3-64-64-64-3, respectively. Each layer is followed by a *batchnorm* layer and *LeakyReLU* activation function. The results are reported in Fig. 5. It can be seen from Fig. 5 that even a simple four-layer network can achieve moderate success in calibrating discretization error on simulation resolution 5 times lower than the high-resolution mesh. Compared with the original low-resolution input, the network output achieves an overall 58.48% error reduction on maximum and minimum values, and 89.54% less mean square error on all samples.

4 CONCLUSION

In the previous sections, we describe a large fluid flow simulation pipeline and dataset, which contains three different types of geometries, more specifically ellipse, cylinder (external flow), and nozzle (internal flow). And for each geometry, two different resolution meshes are generated. Simulations are performed independently on all resolutions, instead of down-sampling the high-resolution results to low resolution, to ensure effective representation of discretization error. All simulation results are extracted via snapshots



(a) Pressure low-res input **(b) Pressure correction output** **(c) Pressure high-res ground truth**

(d) Velocity low-res input **(e) Velocity correction output** **(f) Velocity high-res ground truth**

Figure 5: Examples of applying discretization error correction with the MegaFlow2D dataset

and organized into a graph structure with ready-to-use package-like functions. This work is the first step in constructing a comprehensive flow simulation dataset. We will continue developing the dataset in terms of functionality, dimension, size, and resolution.

This dataset can be used to benchmark the performances of various graph deep learning (GDL) models or to train new ones. Eventually, we hope that this work can help build a new generation of numerical solvers to help engineers optimize their designs much more efficiently. Toward that end, we provide the repository for generating and simulating all the data at https://github.com/cmudrc/FlowDataGeneration and the repository for the MegaFlow2D dataset package at https://github.com/cmudrc/MegaFlow2D.

ACKNOWLEDGMENTS

This material is based upon work supported by the Engineer Research and Development Center (ERDC) under Contract No. W912HZ22C0022. We are grateful to the employees of the Eaton Corporation for their feedback on early versions of this work.

REFERENCES

[1] M. S. Alnaes, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, and G. N. Wells. 2015. The FEniCS Project Version 1.5. *Archive of Numerical Software* 3 (2015). https://doi.org/10.11588/ans.2015.100.20553

[2] Florent Bonnet, Ahmed Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. 2022. AirfRANS: High Fidelity Computational Fluid Dynamics Dataset for Approximating Reynolds-Averaged Navier-Stokes Solutions. *arXiv preprint arXiv:2212.07564* (2022).

[3] Florent Bonnet, Jocelyn Ahmed Mazari, Thibaut Munzer, Pierre Yser, and Patrick Gallinari. 2022. An extensible benchmarking graph-mesh dataset for studying steady-state incompressible Navier-Stokes equations. *arXiv preprint arXiv:2206.14709* (2022).

[4] Alexandre Joel Chorin. 1968. Numerical Solution of the Navier-Stokes Equations. *Math. Comp.* 22, 104 (1968), 745–762. http://www.jstor.org/stable/2004575

[5] Mengyu Chu and Nils Thuerey. 2017. Data-driven synthesis of smoke flows with CNN-based feature descriptors. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–14.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 248–255.

[7] Zhiwen Deng, Chuangxin He, Yingzheng Liu, and Kyung Chun Kim. 2019. Super-resolution reconstruction of turbulent velocity fields using a generative adversarial network-based artificial intelligence framework. *Physics of Fluids* 31, 12 (2019), 125111.

[8] Benedikt Fecher, Sascha Friesike, and Marcel Hebing. 2015. What drives academic data sharing? *PloS one* 10, 2 (2015), e0118053.

[9] Christophe Geuzaine and Jean-François Remacle. 2009. Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. *International journal for numerical methods in engineering* 79, 11 (2009), 1309–1331.

[10] Xiaoxiao Guo, Wei Li, and Francesco Iorio. 2016. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 481–490.

[11] Noelia Grande Gutierrez, Mathew Mathew, Brian W McCrindle, Justin S Tran, Andrew M Kahn, Jane C Burns, and Alison L Marsden. 2019. Hemodynamic variables in aneurysms are associated with thrombotic risk in children with Kawasaki disease. *International journal of cardiology* 281 (2019), 15–21.

[12] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *CoRR* abs/1706.02216 (2017). arXiv:1706.02216 http://arxiv.org/abs/1706.02216

[13] Prasanna Hariharan, Matthew Giarra, Varun Reddy, Steven W. Day, Keefe B. Manning, Steven Deutsch, Sandy F. C. Stewart, Matthew R. Myers, Michael R. Berman, Greg W. Burgreen, Eric G. Paterson, and Richard A. Malinauskas. 2011. Multilaboratory Particle Image Velocimetry Analysis of the FDA Benchmark Nozzle Model to Support Validation of Computational Fluid Dynamics Simulations. *Journal of Biomechanical Engineering* 133, 4 (02 2011). https://doi.org/10.1115/1.4003440 arXiv:https://asmedigitalcollection.asme.org/biomechanical/article-pdf/133/4/041002/5772835/041002_1.pdf 041002.

[14] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. 2021. Physics-informed machine learning. *Nature Reviews Physics* 3, 6 (2021), 422–440.

[15] Byungsoo Kim, Vinicius C Azevedo, Nils Thuerey, Theodore Kim, Markus Gross, and Barbara Solenthaler. 2019. Deep fluids: A generative network for parameterized fluid simulations. In *Computer graphics forum*, Vol. 38. Wiley Online Library, 59–70.

[16] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. 2021. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*. PMLR, 5637–5664.

[17] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. 2021. Neural Operator: Learning Maps Between Function Spaces. https://doi.org/10.48550/ARXIV.2108.08481

[18] Matthew Li and Christopher McComb. 2022. Using physics-informed generative adversarial networks to perform super-resolution for multiphase fluid simulations. *Journal of Computing and Information Science in Engineering* 22, 4 (2022), 044501.

[19] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. 2020. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895* (2020).

[20] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. 2020. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485* (2020).

[21] Bo Liu, Jiupeng Tang, Haibo Huang, and Xi-Yun Lu. 2020. Deep learning methods for super-resolution reconstruction of turbulent flows. *Physics of Fluids* 32, 2 (2020), 025105.

[22] A. Logg, K.-A. Mardal, and G. N. Wells et al. 2012. *Automated Solution of Differential Equations by the Finite Element Method*. Springer. https://doi.org/10.1007/978-3-642-23099-8

[23] A. Logg and G. N. Wells. 2010. DOLFIN: Automated Finite Element Computing. *ACM Trans. Math. Software* 37 (2010). https://doi.org/10.1145/1731022.1731030

[24] A. Logg, G. N. Wells, and J. Hake. 2012. DOLFIN: a C++/Python Finite Element Library. In *Automated Solution of Differential Equations by the Finite Element Method*, K.-A. Mardal A. Logg and G. N. Wells (Eds.). Lecture Notes in Computational Science and Engineering, Vol. 84. Springer, Chapter 10.

[25] Mikael Mortensen and Kristian Valen-Sendstad. 2015. Oasis: A high-level/high-performance open source Navier–Stokes solver. *Computer Physics Communications* 188 (2015), 177–188. https://doi.org/10.1016/j.cpc.2014.10.026

[26] Jianhu Nie and Yitung Chen. 2010. Numerical modeling of three-dimensional two-phase gas–liquid flow in the flow field plate of a PEM electrolysis cell. *International Journal of Hydrogen Energy* 35, 8 (2010), 3183–3197.

[27] Octavi Obiols-Sales, Abhinav Vishnu, Nicholas P Malaya, and Aparna Chandramowlishwaran. 2021. SURFNet: Super-resolution of Turbulent Flows with Transfer Learning using Small Datasets. In *2021 30th International Conference on Parallel Architectures and Compilation Techniques (PACT)*. IEEE, 331–344.

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

[29] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks* 20, 1 (2008), 61–80.

[30] Gege Wen, Zongyi Li, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M Benson. 2022. U-FNO—An enhanced Fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources* 163 (2022), 104180.

Received 10 February 2023