Original Research Article

Analysis of the Accuracy and Efficiency of Neural Networks to Simulate Navier-Stokes Fluid Flows with Obstacles

Elliot McGuire[1*], Rui Hespanha[1*], João Hespanha[2]

[1]*Dos Pueblos High School, 7266 Alameda Ave, Goleta, CA 93117, United States;*
[2]*Department of Electrical Engineering, University of California at Santa Barbara, United States*

ABSTRACT

Conventional fluid simulations can be time consuming and energy intensive. We researched the viability of a neural network for simulating incompressible fluids in a randomized obstacle-heavy environment, as an alternative to the numerical simulation of the Navier-Stokes equation. We hypothesized that the neural network predictions would have a relatively low error for simulations over a small number of time steps, but errors would eventually accumulate to the point that the output would become very noisy. Over a rich set of obstacle configurations, we achieved a root-mean-square-error of 0.32% on our training dataset and 0.36% on a testing dataset. These errors only grew to 1.45% and 2.34% at time step $t = 10$ and, 2.11% and 4.16% at time step $t = 20$. We also found that an accurate neural network can be approximately 8,800 times faster at predicting the flow than a conventional simulation. These findings indicate that neural networks can be extremely useful at simulating fluids in obstacle-heavy environments. Useful applications include modeling forest fire smoke, pipe fluid flow, and underwater/flood currents.

Keywords: Fluid mechanics; simulation; Navier-Stokes; A.I.

INTRODUCTION

The Navier-Stokes partial differential equation has long been used to model fluid dynamics. Although accurate, numerical simulations of this equation are computationally intensive and consume much energy. We propose to train a neural network on data collected from Navier-Stokes-based numerical simulations to predict the fluid velocity profile and investigate the accuracy and speed of the neural network predictions of the fluid flow, when compared to a conventional numerical simulation. We consider environments with a large number of obstacles that interact with the fluid, and we specifically want to study the neural network's ability to predict fluid velocities for obstacle configurations that are not in the set of data used to train the neural network.

There has been a growing interest in using neural networks in solving fluid dynamics problems (1, 2). Preeminent research papers in the field investigate the introduction of physical laws into the training process (an approach commonly known as physics-informed neural networks or PINNs) (3), which integrate novel techniques for training networks to improve accuracy (4, 5), using machine learning to improve turbulence models (6), create reduced order models for fluid dynamics simulations in the aerospace domain (7), solving Navier-Stokes equations in structural wind engineering (8),

Corresponding author: Elliot McGuire, E-mail: elliotrmcguire@gmail.com.

Accepted March 3, 2026
*These authors contributed equally to this work.
https://doi.org/10.70251/HYJR2348.422634

predicting instantaneous turbulent flow fields based on wall measurements (9). We chose to research the efficiency and accuracy of a neural network to predict fluid flows in an obstacle-filled environment, which is a problem that has not been extensively studied using classical neural network training (non-PINN). We were motivated by (10), which states that machine learning has challenges in the generalization of trained neural networks to new configurations not present in the training set. Our research aims to investigate whether neural networks can generalize predictions of fluid flows across configurations of obstacles. Our approach utilizes 1000 distinct, randomized obstacle configurations of which 100 are separated for training the network. We hypothesized a classical, non-PINN approach would prove more effective at predicting fluid flow in our environments and tested our approach against a PINN. We hypothesize that our non-PINN neural network will have similar accuracy but perform much faster than PINNs at predicting fluid flow over our obstacle configurations, overcoming aforementioned challenges of generalization without utilizing the more computationally intensive approach of a physics informed network.

We hypothesized that our network would be relatively accurate at making predictions over short time horizons, but performance would degrade over time, leading to "noisy" predictions of the speed profile. In reality, our neural networks achieved 99.68-99.64% RMSE accuracy and this accuracy degraded to just 97.89-95.84% after 20-time steps, with a speed increase 8,800x over the Navier-Stokes fluid simulation in 2 dimensions. The measurement takes into account total experiment runtime and includes data loading, preprocessing, forward loading and backpropagation. Note that the computational fluid solver PhiFlow we are measuring it against is heavily optimized for efficiency but not to the extent of other models such as OpenFOAM or Nek5000. Rather than generating "noisy" predictions, the neural network actually produced velocity profiles that were overly smooth. This indicates that our method of fluid simulation is reliable and fast for complex environments with incompressible fluid flow and could be expanded upon to provide approximations for turbulent flow patterns or more general physical environments.

METHODS AND MATERIALS

To train the neural network, 1,000 fluid simulations with different obstacle configurations were generated, each with 30-time steps (Figure 1), randomly split into 900 training and 100 testing simulations using PhiFlow version 3.20's pixel-based Navier-Stokes fluid modeling. Our code is available in (11). The initial learning rate was set to 1×10^{-5} and halved after loss reached certain thresholds. The threshold was also set as 1×10^{-5} and this threshold followed an inverse square root decay pattern as the network matured. Each simulation used a 12 by 24-pixel domain with 2×2 obstacles (Figure 3). To test how generalizable the network was, different sets of randomly located obstacles were used for each of the 1000 simulations.

In every simulation, each one of the 30-time steps or frames was paired with the next one, totaling 26,100 data inputs (with 2,900 data inputs for testing). The neural networks were implemented using PyTorch version 2.20 and take the first-time step in each pair as input and predict the *difference* between the two consecutive paired time steps. The network was initialized through PyTorch's standard initialization method, setting all weights to with small, random floating-point numbers. The neural networks were trained using a mean-square error loss, for a total of 2,000 training epochs with randomized batches containing 300-time step pairs each, leading to 87 batches that are randomized for each epoch to sufficiently mix the data. 2000 Epochs was near-optimal in a ratio of processing speed to loss while being a comparable number to other studies. Multiple depths (i.e., number of layers) and widths (i.e., number of nodes per layer) were tested. To make predictions over $n > 1$ time-steps, the neural networks are called n times, each time predicting the next step (or more precisely the increment with respect to the previous step) based on the previously predicted step.

In all simulations, fluid at the bottom of the domain is subject to a set "inflow" velocity. This fluid then exits the opposite side of the domain. The fluid entering the simulation has no difference in properties from the fluid originally inside the simulation at time $t = 0$. The inflow velocity is 1 pixel per time step, or 20 pixels per second. Fluid cannot flow out through the side "walls" of the domain, creating areas with high velocity between the obstacles and "walls". Velocity is represented as a staggered grid in PhiFlow while the input fluid is represented by a centered grid such that velocity components are stored in the vertical and horizontal faces of individual pixels for ease of enforcing an incompressible state through ensuring divergence is always equal to 0.

Viscosity was negligible in our numerical simulations, placing the flow in the Euler (inviscid) regime. Unlike

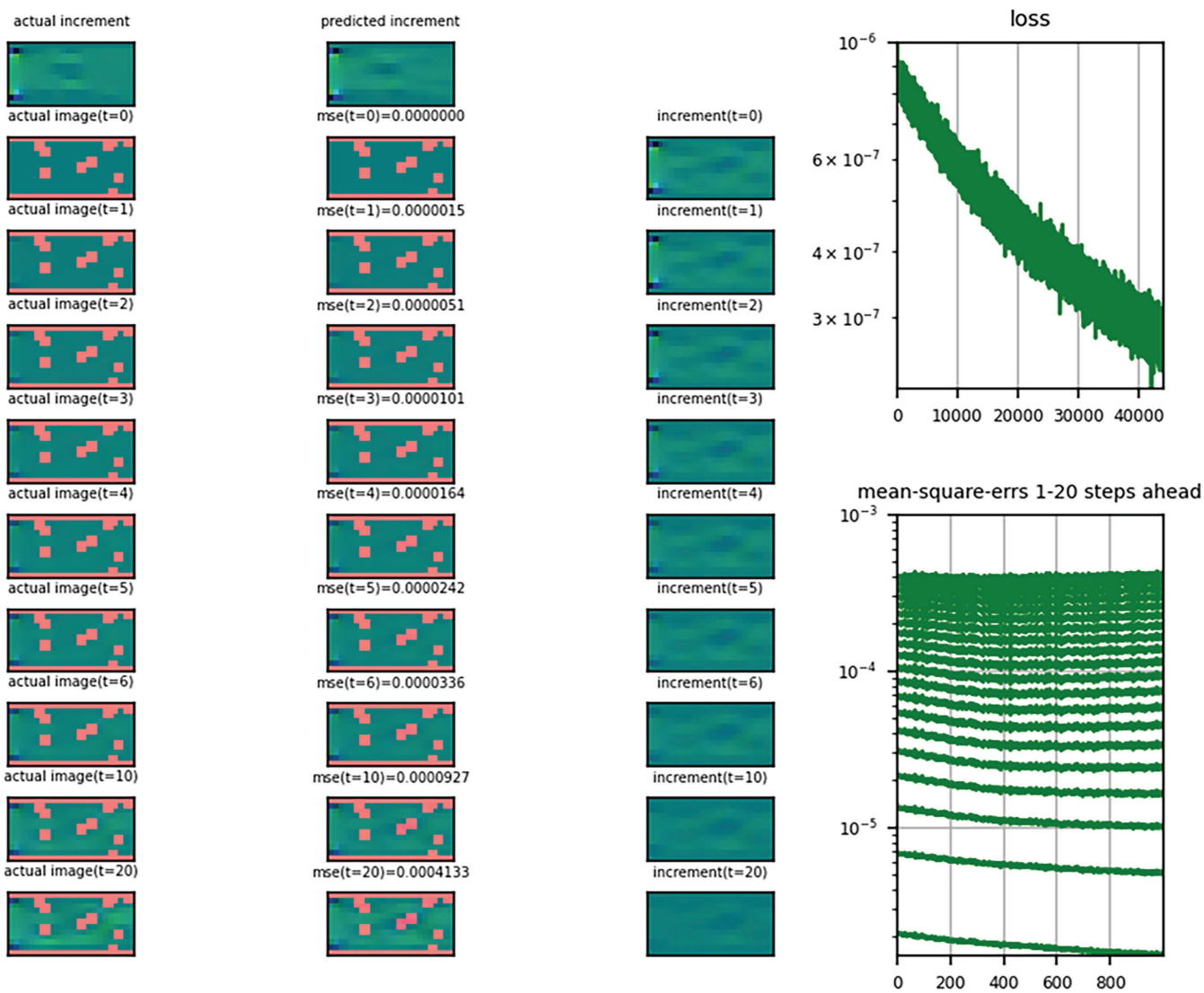

Figure 1. ***Visualization of Navier-Stokes-simulated and neural network-simulated velocity graphs.*** *Column 1 illustrates fluid flow frames obtained through a numerical simulation of the Navier-Stokes equation. Column 2 shows the same frames as predicted by the neural network, with a mean square error measure displayed above each frame. Column 3 shows the increment (change) between each frame and the next one. Column 4 shows the evolution of the neural network loss function (mean square error in the y-axis of the top plot) and the mean-square error for 1 through 20 steps-ahead predictions (y-axis of the bottom plot) during training vs. the number training samples (x-axis).*

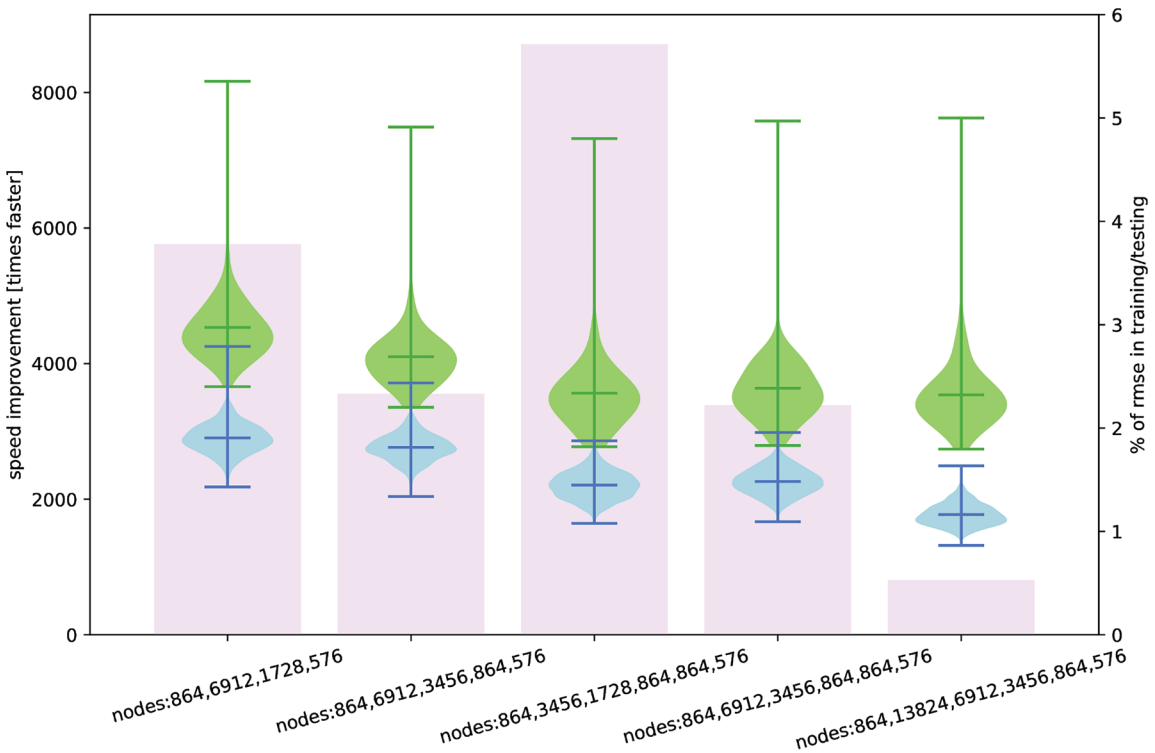

Figure 2. ***Accuracy and efficiency of different neural networks.*** *The x-axis displays the number of nodes on each layer of 5 distinct neural networks. The left y-axis, corresponding to the pink bars, displays how many times the neural network is faster than a numerical simulation of the Navier-Stokes equation. The right y-axis, corresponding to the colored violin plots, displays the distribution of the root mean square error (RMSE), in percentage, for the different neural networks. The blue violin plots show the RMSE for the training data, while the green violin plots show the RMSE for the testing data.*

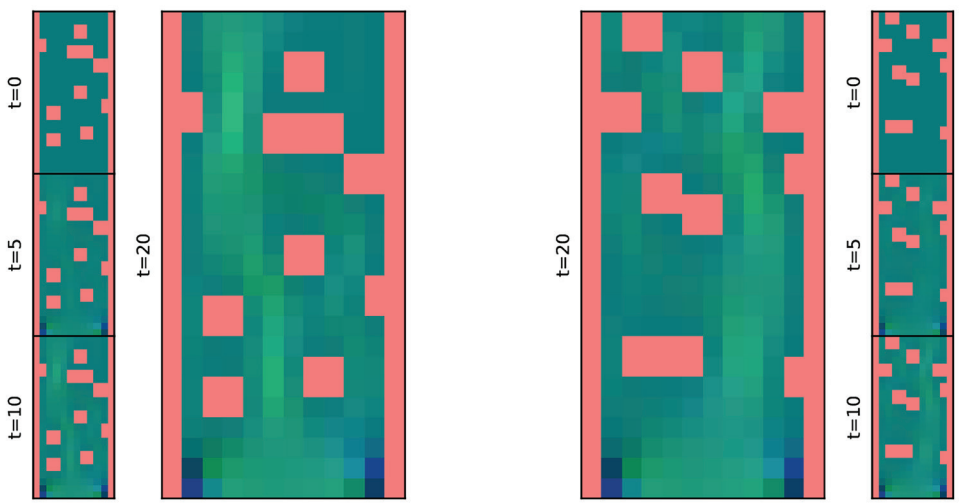

Figure 3. ***Two PhiFlow simulations (one at left and one at right) showing a velocity graph and obstacles.*** *Red indicates an obstacle (where fluid cannot enter) and the red-blue shading the fluid velocity. Intense blue indicates a high velocity towards the left, and intense shades of green indicate a high velocity towards the upward direction. The velocity graphs are shown at different points in time, with the labels to the left of each figure showing the time step t. The left simulation was taken from the training data while the right simulation was taken from the testing data set.*

real high-Reynolds-number flow, there is no viscous dissipation. However, vortices naturally form due to inertial forces. Since viscosity was negligible, the Reynolds number approximates infinity.

RESULTS

The neural networks' ability to predict the fluid flow profile was evaluated over a large number of obstacle configurations and, for each configuration, we compared the difference between the fluid velocities predicted by the neural network with those computed by a numerical simulation of the Navier-Stokes equation. The percentage errors reported here refer to the root-mean-square-error (RMSE) averaged over the whole spatial simulation domain, as a fraction of the largest fluid speed over the same domain. We consider multiple neural networks with different numbers of layers and nodes per layer (see x-axis labels in Figure 2). For most neural networks, we observed errors in the 2-3% range (see y-axis labels in Figure 2). Smaller networks, such as the first three shown in Figure 2, computed prediction of the fluid velocities 3,500 to above 8,000 times faster than the conventional simulation. The main limiting factor on network computation speed was width (i.e., number of nodes per layer) despite this parameter actually showing negligible performance improvements. Larger networks (4 & 5 in Figure 2) required larger computation times, with both being less than 3,500 times faster, though total network size was not the main factor contributing to speed as the fastest network had a medium size and low width. The difference in error between training and testing data is relatively low: at the end of 19 step predictions, the network corresponding to the fourth column of Figure 2 achieves an error of 1.48% error on the training data, which only grows to 2.39% on the testing data. We recall that the testing data set was entirely composed of obstacle configurations that were not present in the training data.

Comparing our results with Figures 6 & 7, which characterize the entirely randomized obstacle data set, we see similar results. The errors are both within standard deviation of each other, and there is never more than 0.2% more error in the randomized obstacle set than the obstacle set with randomized location but consistent obstacle shape and size, indicating that in both setups, the network generalizes similarly effectively. More research is needed to determine if this relationship holds true over longer time horizons.

The traditional non-PINN networks we tested, compared to the PINNs, have greater computation speed by 2.5-25% depending on network size or an average close to 10% as seen by Figures 2 & 8. The PINN

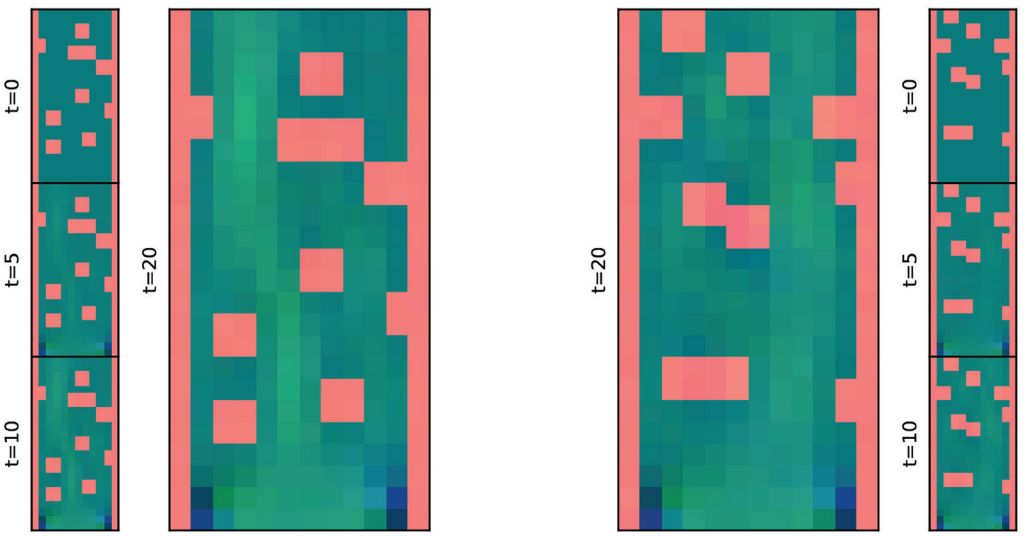

Figure 4. ***Two neural network simulations (one at left and one at right) showing a velocity graph and obstacles.*** *The red color corresponds to obstacles (where fluid cannot enter) and the red-blue shading the fluid velocity. Intense blue indicates a high velocity towards the left, and intense shades of green indicate a high velocity upward. The velocity graphs are shown at different points in time, with the labels to the left of each figure showing the time step t. Each frame in both simulations was generated sequentially by the neural network. The left simulation uses a training data obstacle configuration while the right simulation uses an obstacle configuration not present in the training data.*

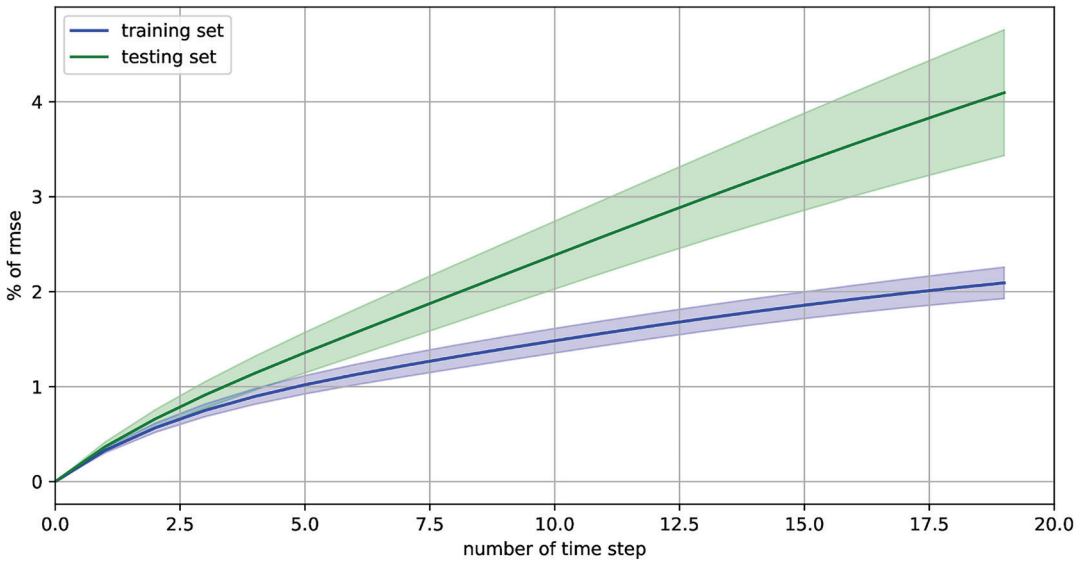

Figure 5. ***Graph of (percentage) RMSE in the neural network.*** *Line graph showing mean (solid lines) with standard deviations (shaded area) of the RMSE (in percentage, in the y-axis) for the obstacles in the training (blue) and testing (green) data sets vs. the number of time steps over which the neural network is doing the prediction (in the x-axis).*

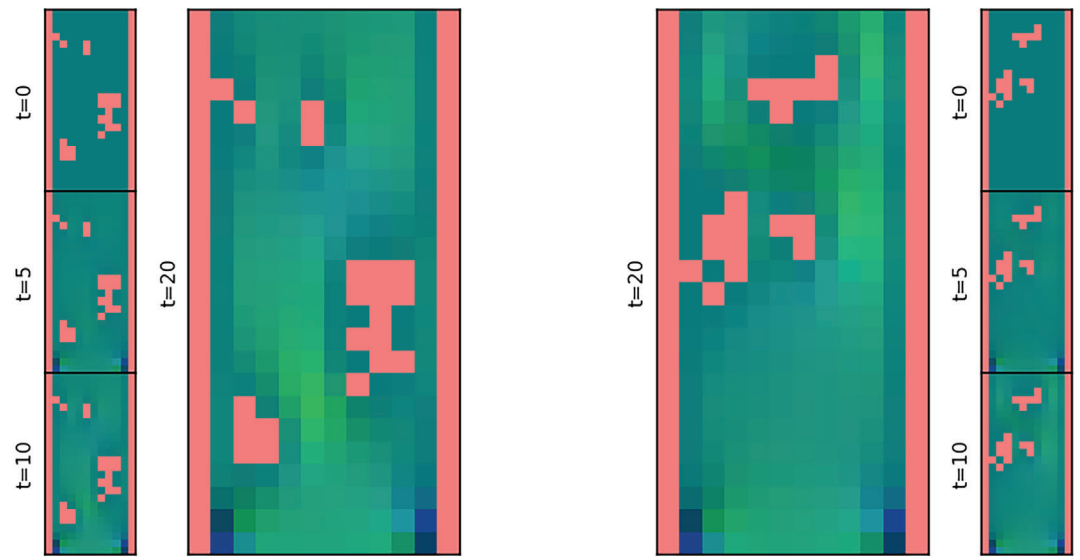

Figure 6. ***Two PhiFlow simulations (one at left and one at right) showing a velocity graph and obstacles from the randomized obstacle data set.*** *Red indicates an obstacle (where fluid cannot enter) and the red-blue shading the fluid velocity. Intense blue indicates a high velocity towards the left, and intense shades of green indicate a high velocity towards the upward direction. The velocity graphs are shown at different points in time, with the labels to the left of each figure showing the time step t. The left simulation was taken from the training data while the right simulation was taken from the testing data set.*

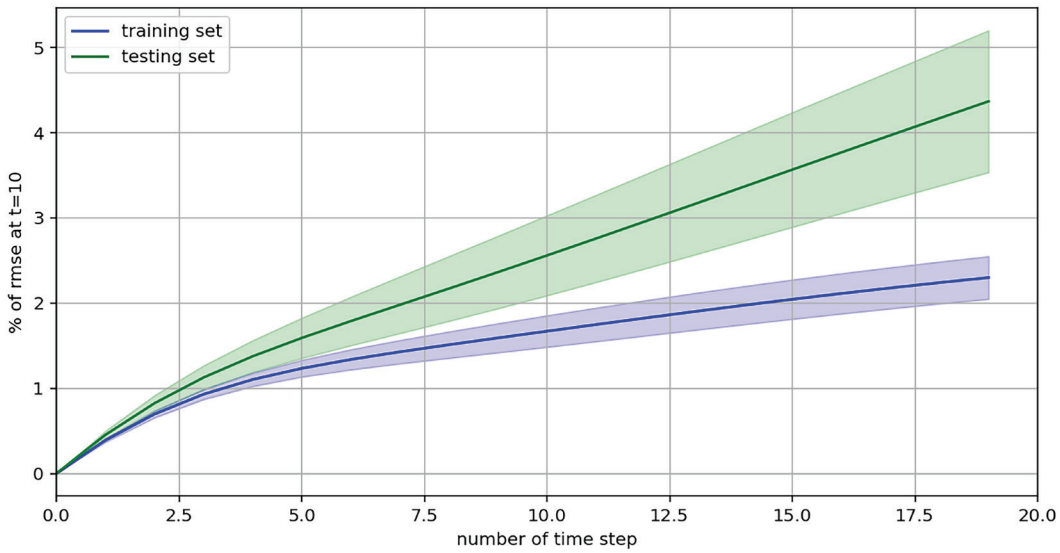

Figure 7. ***Graph of (percentage) RMSE in the neural network for the completely randomized obstacle configuration data set.*** *Line graph showing mean (solid lines) with standard deviations (shaded area) of the RMSE (in percentage, in the y-axis) for the obstacles in the training (blue) and testing (green) data sets vs. the number of time steps over which the neural network is doing the prediction (in the x-axis).*

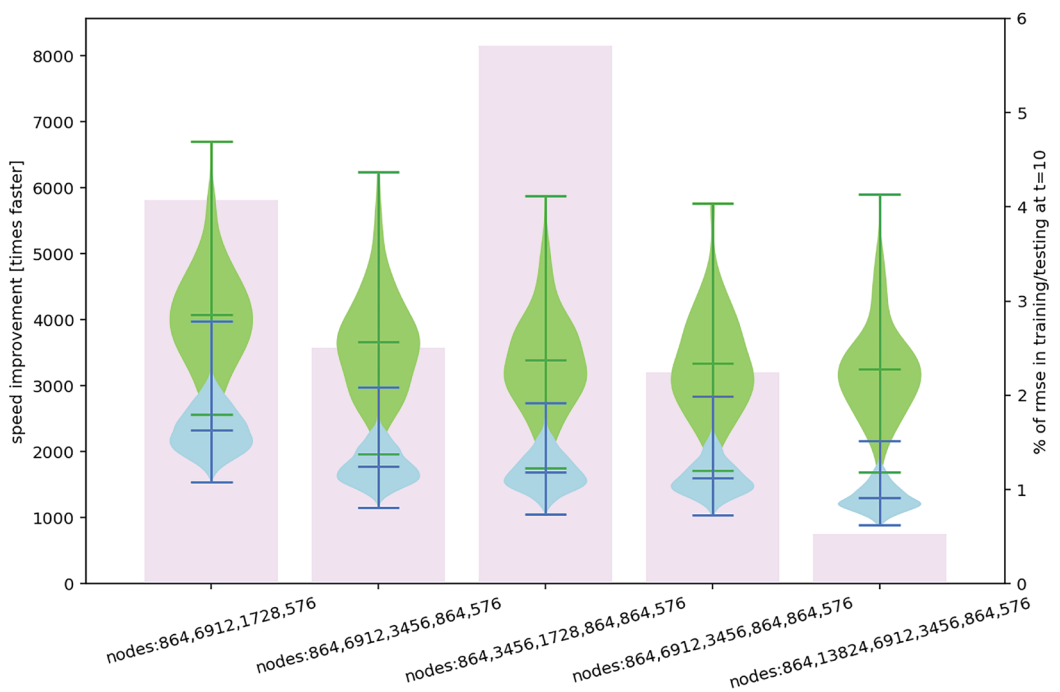

Figure 8. ***Accuracy and efficiency of different PINN models.*** *The x-axis displays the number of nodes on each layer of 5 distinct neural networks. The left y-axis, corresponding to the pink bars, displays how many times the neural network is faster than a numerical simulation of the Navier-Stokes equation. The right y-axis, corresponding to the colored violin plots, displays the distribution of the RMSE, in percentage, for the different neural networks. The blue violin plots show the RMSE for the training data, while the green violin plots show the RMSE for the testing data.*

networks, despite their slower computations, did have any meaningful increases in accuracy compared to the non-PINN networks when measured over 30-time steps. However, the PINNs had greater accuracy than the traditional networks from time steps 0 to 5, indicating the non-PINN networks struggled with divergence near the beginning of the simulation. More research can be done on the temporal rollout stability for different sizes of networks tested, as we don't have explicit comparisons between each neural network's accuracy at each time step. Our hypothesis that the predictions would become noisier over time was incorrect. Instead, the neural network predictions for the fluid velocities were actually overly "smooth." The error was less than we expected.

DISCUSSION

The neural networks generated fluid predictions much faster than numerical simulations with an error that increased over time. Part of this problem could be due to the over-smoothing effect the network has, leading to lower turbulence in the neural network compared to the computation simulation. It's particularly clear that predicting fluid dynamics with obstacles over relatively short time horizons can be done very efficiently by a neural network. Over a longer time frame, error growth magnifies, which effectively limits the use of this approach over very long-time spans. We were limited in terms of the size of the networks as our GPU was a NVIDIA GeForce 3080 with 10 gigabytes of GPU memory. The GPU was used both for numerical simulation and neural network training. Another limitation due to low processing power was the low resolution of our simulation. We did not have the processing power to simulate higher resolutions or 3 spatial dimensions in this study. We conjecture that we would encounter larger errors in 3 dimensions due to flow turbulence.

This approach could be extended to modeling fluids that behave qualitatively similar to incompressible fluids under certain conditions including water and smoke (at 1 atmosphere of pressure and earth-surface temperatures) in obstacle-filled environments such as pipes and forests. It also opens the door to cheaper weather modeling that can be performed using trained networks developed through this process (although more research needs to be done). Since our simulation provides a way to simulate fluid flow around obstacles in short time frames, aerodynamic properties can be calculated by measuring the force exerted from the fluid onto static objects. Similarly, simulating water in a pipe would only require augmenting the spatial dimensional of our simulation and increasing its resolution. These simulations could be utilized to rapidly visualize patterns in fluid flow for given environments.

We compared our approach to a PINN approach through implementing divergence ($\nabla \cdot v$) into our loss function, as is commonly done in contemporary PINN studies. The error marginally improved although the speed of simulation dropped. Comparing our study to (5), a benchmark for neural operator neural networks (another family of neural networks) we achieved RMSEs of 2-3% over the course of 30-time steps (1.5 seconds in real time) vs the benchmark's global RMSE (or L2 as referred to in the paper) of between 19-39% for the neural networks tested. over 240-time steps (2.5 seconds). Note that their larger number of time steps per second simplify learning over short time horizons but can lead to overfitting as the training targets are closer to inputs.

Future studies are needed to determine the accuracy of this approach in higher resolutions and in 3-dimensional domains. Additionally, object sizes and shapes could be changed to determine how generalizable the approach is. Further investigation is also needed to determine whether this approach can be extended to fluids subject to external forces (such as buoyancy), varying flow parameters, and obstacle shapes. We are interested in specializing this work to specific application domains, such as real-time graphics, weather prediction and modeling, blood flow modeling, floodwater path modeling and more.

CONCLUSION

The study aimed to determine the viability of a neural network for simulating incompressible fluids in a randomized obstacle-heavy environment. Our simulation is rapid but accumulates error over long periods of time. Our neural networks' unique features to simulate an environment with obstacles at varying locations allows our networks to predict fluid flow patterns in cluttered environments. Our neural networks are useful at modeling any situation with incompressible fluid interacting with an obstacle-filled environment. For air or smoke, modeling fluids in urban environments, forests, or in aerodynamics would work well. For water, modeling its interaction in currents, in pipes, or around ships would work well with our simulation. Our study concludes that using a conventional, non-PINN neural network approach to model fluid simulations in randomized, obstacle-heavy environments is extremely effective under short time horizons.

CONFLICT OF INTEREST

The authors declare that there are no conflicts of interest related to this work.

REFERENCES

1. McCracken MF. Artificial neural networks in fluid dynamics: A novel approach to the Navier-Stokes equations. In: Proceedings of the Practice and Experience on Advanced Research Computing: Seamless Creativity; 2018 Jul 22; Pittsburgh, PA. New York: ACM; 2018; p. 1-4. https://doi.org/10.1145/3219104.3229262
2. Vinuesa R, Brunton SL. Enhancing computational fluid dynamics with machine learning. *Nat Comput Sci*. 2022; 2 (6): 358-66. https://doi.org/10.1038/s43588-022-00264-7
3. Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J Comput Phys*. 2019; 378: 686-707. https://doi.org/10.1016/j.jcp.2018.10.045
4. Kochkov D, Smith JA, Alieva A, Wang Q, *et al*. Machine learning-accelerated computational fluid dynamics. *Proc Natl Acad Sci USA*. 2021; 118 (21): e2101784118. https://doi.org/10.1073/pnas.2101784118
5. Tali R, Rabeh A, Yang CH, Shadkhah M, *et al*. Flowbench: A large scale benchmark for flow simulation over complex geometries. arXiv [Preprint]. 2024. Available from: https://arxiv.org/abs/2409.18032 (accessed on 2025 -12-20).
6. Duraisamy K, Iaccarino G, Xiao H. Turbulence modeling in the age of data. *Annu Rev Fluid Mech*. 2019; 51 (1): 357-77. https://doi.org/10.1146/annurev-fluid-010518-040547
7. Brunton SL, Kutz JN, Manohar K, Aravkin AY, *et al*. Data-driven aerospace engineering: reframing the industry with machine learning. *AIAA J*. 2021; 59 (8): 2820-47. https://doi.org/10.2514/1.J060131
8. Cai K, Wang J. Physics-informed neural networks for solving incompressible Navier-Stokes equations in wind engineering. *Phys Fluids*. 2024; 36 (12): 121303. https://doi.org/10.1063/5.0244094
9. Guastoni L, Encinar MP, Schlatter P, Azizpour H, Vinuesa R. Prediction of wall-bounded turbulence from wall quantities using convolutional neural networks. *J Phys Conf Ser*. 2020; 1522 (1): 012022. https://doi.org/10.1088/1742-6596/1522/1/012022
10. Vinuesa R, Brunton SL, McKeon BJ. The transformative potential of machine learning for experiments in fluid mechanics. *Nat Rev Phys*. 2023; 5 (9): 536-45. https://doi.org/10.1038/s42254-023-00622-y
11. Espanha R. new-physics. GitHub;. Available from: https://github.com/ruihespanha/new-physics (accessed on 2025 -12-20)