FULLY CONVOLUTIONAL APPROACH FOR SIMULATING WAVE DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

In this work, we investigate the performance of fully convolutional networks to predict the motion and interaction of surface waves in open and closed complex geometries. We focus on a U-Net architecture and assess its ability to capture and extrapolate wave propagation in time as well as the reflection, interference and diffraction of waves. We investigate how well the network generalises both to long-time predictions and to geometric configurations not seen during training. We demonstrate that our U-Net architecture is capable of accurately predicting the height distribution of waves on a liquid surface within curved and multi-faceted open and closed geometries, when only simple box and right-angled corner geometries were seen during training. We found the RMSE of the predictions remained of order 1×10^{-4} times the characteristic length of the domain for at least 20 time-steps.

1 INTRODUCTION

Predicting the spatio-temporal dynamics of physical systems is a recurrent problem in many areas of sciences and engineering. A well-established process consists of describing the physical phenomena by human-engineered mathematical models, whose complexity may prevent finding analytical solutions. Thus, scientists turn to numerical solvers to simulate such mathematical models and generate accurate approximations to their solution.

On the other hand, the huge progress in machine learning (ML) during the last decade has motivated an increased popularity of data-driven physics. In this field, the interpolation capabilities of neural networks (NNs) have been mostly used in two ways: first, to solve partial differential equations (PDEs) in an unsupervised manner (Dissanayake & Phan-Thien, 1994; Lagaris et al., 1998; 2000; Raissi et al., 2019) and second, to predict the physical dynamics from previous observations without knowledge of the underlying equations (Guo et al., 2016; Farimani et al., 2017; Thuerey et al., 2018; Lee & You, 2018). As opposed to the first approach, the second approach does not allow for an analytical expression of the physical dynamics, however, it makes possible to produce simulations for a diversity of physical domains, boundary conditions and initial conditions without having to re-train the network, provided that the physical laws are unaltered. Compared to numerical solvers, recent research on applying convolutional neural networks (CNNs) to simulate fluid dynamics has reported a speed-up of four orders of magnitude while keeping a similar accuracy (Guo et al., 2016). The major shortcoming of NNs are their poor generalization to unseen configurations and poor long-time predictions in unsteady simulations.

In this work, we investigate the application of fully convolutional neural networks (CNNs) to forecasting surface wave dynamics, the motion of which is described by the shallow water equations. Computational modelling of surface waves is widely used in seismology, computer animation and flood modelling (Ersoy et al., 2017; García-Navarro et al., 2019). Our network learnt to simulate a range of physical phenomena including wave propagation, reflection, interference, superposition, refraction and diffraction at sharp corners. This kind of NN could replace numerical algorithms used to solve the shallow water PDEs, reducing the inference time by several orders of magnitude and allowing for real-time solutions, what has a big relevance in iterative design and applications such as tsunami prediction.

Contribution. We demonstrate that our U-Net architecture is able to predict surface wave dynamics in straight-sided and curved geometries, even when trained only on data sets with simple straight-

sided boundaries. The generalisation to different initial conditions and longer-time predictions are also evaluated. Additionally, we also show how including the MSE of the wave gradient into the loss function significantly reduces spurious oscillations in predicted solutions and could also help in predicting the position of the wavefronts. Our network is able to simulate wave dynamics four orders of magnitude faster than a hp/spectral element solver (Karniadakis & Sherwin, 2013), so it could be a perfect replacement of numerical solvers for real-time applications.

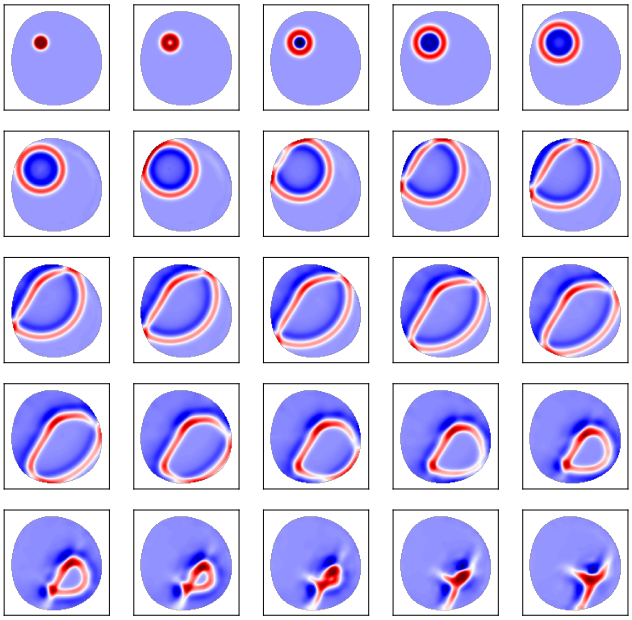


Figure 1: Rollouts of our U-Net. It simulates wave motion on a fluid surface with the possible existence of solid walls [video].

2 Related Work

Physics-informed NNs for solving PDEs. The application of NNs for the solution of differential equations has drawn attention since the early 1990s. Most of the relevant research in that decade built on the idea of exploiting the universal function approximator property of multi-layer perceptrons (MLPs) (Dissanayake & Phan-Thien, 1994; Dissanayake & Phan-Thien, 1994; Lagaris et al., 1998). In such an approach, the solutions to the PDEs are approximated as MLPs whose only inputs are the spatio-temporal coordinates. These MLPs are trained in an unsupervised way to satisfy the governing PDEs as well as the initial and boundary conditions. The main advantage of this paradigm is that the solution is obtained in a differentiable, closed analytic form, easily used in any subsequent calculation. Nevertheless, these networks cannot extrapolate to different domains, boundary conditions or initial conditions; making it necessary to re-train the network for every slight modification of the problem. These techniques inspired the more modern Physics-informed neural networks (PINNs) (Raissi et al., 2017; 2018; 2019; Lu et al., 2019), which include deeper MLPs and random collocations points; and proved to extrapolate outside of the time interval used for tuning the learnable parameters. PINNs have been used to solve the one-dimensional Burger's equation and the two dimensional Schrodinger equation (Raissi et al., 2017), as well as the two-dimensional Navier-Stokes equations (Raissi et al., 2018). Although PINNs are successful at solving PDEs with high accuracy, they have to be retrained for every variation of the boundary conditions or initial conditions. For this reasons, PINNs are not an ideal alternative to numerical solvers when short computational time is a priority.

Computer vision for simulating steady physics. During the last five years, most of the networks used to predict continuous physics have included convolution layers. For instance, CNNs have been used to solve the Poisson's equation in Tang et al. (2018) and Özbay et al. (2019), and to solve

the steady Navier-Stokes equations in Guo et al. (2016); Miyanawala & Jaiman (2017); Yilmaz & German (2017); Farimani et al. (2017); Thuerey et al. (2018) and Zhang et al. (2018). The use of CNNs allows for visual inputs representing certain physical information, such as the domain geometry or the initial condition, and for visual outputs representing the solution fields. In contrast to PINNs, the network predictions are purely based on observation, without knowledge of the underlying governing equations. This paradigm has proved to well extrapolate to domain geometries, boundary conditions and initial conditions not seen during training (Thuerey et al., 2018). Besides, these networks evaluation is considerably faster than traditional PDE solvers, allowing to achieve relatively accurate solutions between one and four orders of magnitude faster (Guo et al., 2016; Farimani et al., 2017). These reasons make CNNs perfect for developing surrogate models replacing expensive numerical solvers (Guo et al., 2016; Miyanawala & Jaiman, 2017) or for real-time animations (Kim et al., 2019). Our work takes from Guo et al. (2016) the use of a binary geometry field to represent the physical domain. Although Guo et al. (2016); Farimani et al. (2017) and Thuerey et al. (2018) proved the generalisation of their predictions to domain geometries not seen during training, these unseen domains contain elementary geometrical entities included within the training data. We go one step further by training the network with exclusively straight boundaries and proving the generalisation on boundaries with varying radius of curvature.

Computer vision for simulating unsteady physics. Although in a smaller proportion compared to steady physics, unsteady physics have also been explored from the computer vision perspective (Fotiadis et al., 2020; Lee & You, 2018; Sorteberg et al., 2018; Wiewel et al., 2019; Kim et al., 2019). Here, a sequence of past solution fields is used as input to the network, whereas the output is a sequence of solution fields at a future time. When predicting unsteady phenomena there is a new challenge: keeping the predictions accurate along time. To address this, Sorteberg et al. (2018); Wiewel et al. (2019) and Kim et al. (2019) proposed to use encoder-propagator-decoder architectures, whereas Fotiadis et al. (2020) and Lee & You (2018) kept using encoder-decoder architectures similar to the ones used for steady problems. Inspired by Fotiadis et al. (2020), which showed that feed-forward networks perform at least as well as recurrent networks in wave forecasting, we opt to use a U-Net architecture (Ronneberger et al., 2015) to perform each time-step of the simulations.

3 METHOD

3.1 WAVE DYNAMICS DATA SETS

The data sets used during training and testing were generated by solving the inviscid, two-dimensional shallow water equations with Nektar++, a high-order finite element solver (Cantwell et al., 2015). In conservative form, these equations are given by

$$\frac{\partial}{\partial t}\begin{pmatrix} h \\ hu \\ hv \end{pmatrix} + \nabla \cdot \begin{pmatrix} hu & hv \\ hu^2 + gh^2/2 & hvu \\ huv & hv^2 + gh^2/2 \end{pmatrix} = \mathbf{0}, \quad (x, y) \in \mathcal{D} \tag{1}$$

where $g = 9.80665$ m/s^2 is the acceleration due to gravity and $\mathcal{D} \in \mathbb{R}^2$ denotes the domain under consideration. Unknown variables in this system are the water depth $h(x, y, t)$ and the components of the two-dimensional velocity vector $u(x, y, t)$ and $v(x, y, t)$.

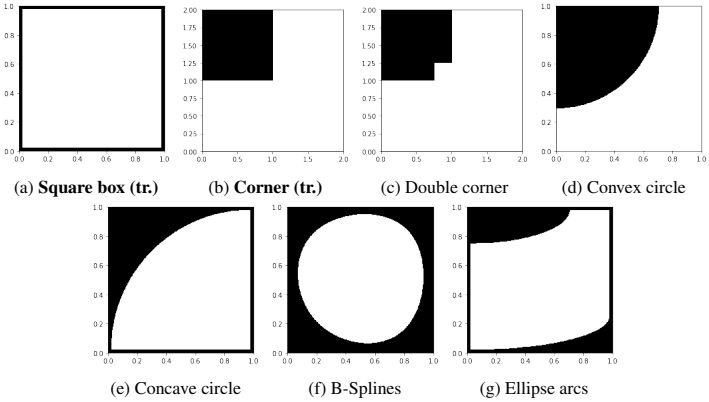
We imposed two forms of boundary conditions: solid wall boundaries, which result in wave reflection, refraction and deflection; and open boundaries, which allow waves to exit the domain. As initial conditions we considered a *droplet*, represented mathematically by a localized two-dimensional Gaussian:

$$h_0^1 = 1 + I \exp\Big(- C((x - x_c)^2 + (y - y_c)^2)\Big) \tag{2}$$

where I is set to 0.1 m, C is randomly sampled from a uniform distribution between 400 and 1000 m^{-2} and (x_c, y_c) are randomly sampled from a uniform distribution in $\mathcal{D}$. Additionally, we generated two data sets with two droplets as initial condition:

$$\begin{aligned} h_0^2(x, y) = 1 &+ I_1 \exp\left(-C_1\left((x - x_{c,1})^2 + (y - y_{c,1})^2\right)\right) \\ &+ I_2 \exp\left(-C_2\left((x - x_{c,2})^2 + (y - y_{c,2})^2\right)\right) \end{aligned} \tag{3}$$

Here C_i, $x_{c,i}$ and $y_{c,i}$ follow the same uniform distributions as in equation (2), $I_1 = 0.1$ m and I_2 is randomly sample from a uniform distribution between 0.05 and 0.1 m.



(a) **Square box (tr.)** (b) **Corner (tr.)** (c) Double corner (d) Convex circle

(e) Concave circle (f) B-Splines (g) Ellipse arcs

Figure 2: Geometry field, Ω, for the seven categories of flow domains. White: $\Omega = 0$, Black: $\Omega = 1$. The dimensions are given in metres.

Each simulation in the data sets is associated with a binary geometry field, $\Omega(x, y)$, which satisfies

$$\Omega(x, y) = \begin{cases} 0, & \text{if } (x, y) \in \mathcal{D} \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

Therefore, $\Omega = 0$ inside the fluid domain and $\Omega = 1$ in the solid boundaries (Guo et al., 2016). This geometry will be an additional input to the network, required to provide information about the walls location. Figure 2 shows the geometry field for the seven categories of fluid domains included in the data sets. For full details of the data sets, see Appendix A. [1]

3.2 Neural Network as Simulation Engine

Our neural network model is based on the U-Net architecture (Ronneberger et al., 2015), which has been extensively used for image-to-image translation tasks (Farimani et al., 2017; Fotiadis et al., 2020; Isola et al., 2016; Thuerey et al., 2018). This architecture consists of a fully convolutional feed-forward encoder-decoder network with skip connections between the encoder and the decoder. In wave dynamics forecasting, the input sequence and the target share an important amount of information at different length scales, this makes the U-Net a particularly appropriate architecture for our problem. Our U-Net receives six fields as input: the geometry field, Ω, and a sequence of five consecutive height fields, $\{h_s, h_{s+1}, h_{s+2}, h_{s+3}, h_{s+4}\}$; and outputs a prediction of the subsequent height field $\hat{h}_{s+5}$. Hence, each evaluation of the network corresponds to performing a single time-step, and the network has to be re-fed to make further predictions. See Appendix B for more details about our U-Net architecture.

3.3 Gradient Loss

The loss function used in the present work is given by

$$\mathcal{L} = (1 - \lambda) MSE(\hat{h}_i, h_i) + \lambda \left[MSE\left(\frac{\partial \hat{h}_i}{\partial x}, \frac{\partial h_i}{\partial x} \right) + MSE\left(\frac{\partial \hat{h}_i}{\partial y}, \frac{\partial h_i}{\partial y} \right) \right] \tag{5}$$

where λ is a hyper-parameter. The derivatives in equation (5) were computed using second order finite differences. This loss function penalises oscillations in the predicted fields and could favour smooth solutions. This is especially important for temporal forecasting by consecutively re-feeding the network, since the spurious oscillations are amplified in each new prediction. In the case of wave dynamics, the gradients have a large magnitude on the leading and trailing edges of the wavefront. Capturing such small-scale information enables the network to accurately learn and reproduce the wave speed and width of wavefront.

[1] All data set are availble on to be revealed.

3.4 TRAINING

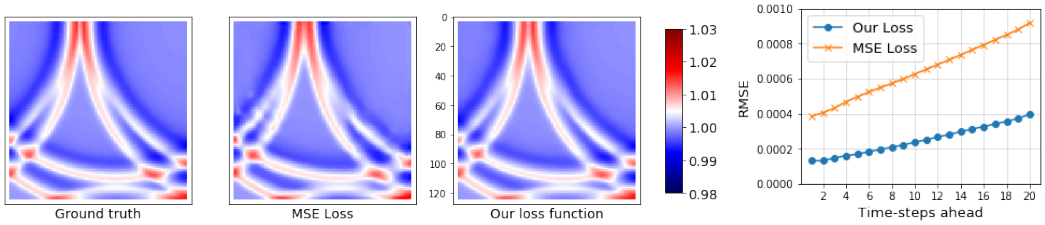
Our U-Net was trained against simple and rectilinear geometries (Figures 2a and 2b) and with a single droplet as initial condition. The time step was set to $\Delta t = 0.12$ seconds and the spatial resolution was set to 128 pix/m. Besides, a series of transformations were applied to perform data augmentation and normalisation (see Appendix C). We trained for 500 epochs with Adam optimiser and its standard parameters (Kingma & Ba, 2014) and the loss function given by (5) with $\lambda = 0.05$. The learning rate was set to 10^{-4} and decreased by a factor of 10 every 100 epochs. The time origin for the input sequences is not at $t = 0$, instead it is randomly selected, and five time-steps are performed before updating the network weights. [2]

4 RESULTS AND DISCUSSION

In this section we evaluate the ability of the trained model to extrapolate to different domain geometries and initial conditions. We also assess the quality of long-time predictions.

4.1 GRADLOSS VS MSE

Figure 3a compares the ground truth and two predictions obtained five time steps ahead of the last temporal height frame used as input. The height field in the central image was predicted by the U-Net trained with the plain MSE, whereas the prediction on the right was obtained after training with the loss function given by (5). Here, the suppression of the oscillations on the wavefront when using the GradLoss instead of the MSE is clearly apparent. To quantify this difference, Figure 3b shows the RMSE (averaged over 500 simulations) during 20 time-steps for both loss functions. From here we can conclude that even for the first predicted time-step, the network trained with the MSE is considerably less accurate. This difference is amplified linearly every time the network is re-fed (i.e. every time-step).



(a) (b)

Figure 3: (a) Predictions obtained after five time-steps (values in metres). (b) RMSE between targets and predictions vs. time-steps into the future.

4.2 GENERALISATION TO DIFFERENT DOMAIN GEOMETRIES

Our network proved to generalise to complex domain geometries not seen during training, some including curved walls, when only simple and straight-sided domains where used for training (Figures 2a and 2b). The three columns of Figure 4b depict the ground truth, prediction and the absolute value of their difference every five time-steps (the input height fields are depicted in Figure 4a). This Figure shows that our network is able to accurately predict the wave speed and correctly infer the appearance of reflections at the solid boundaries. This is evidenced by the predicted height field at frame 25 (after 20 time-steps), where we observe that the position and wavelength of the predicted and true waves are coincident. The main discrepancy between ground truth and predictions is the height of the wavefronts refracted on the edges of the steps, which are of lower magnitude in the predicted fields. The reason for this may be the fact that the network is less exposed to wave refraction than to wave propagation and reflection during training.

[2] GitHub repository availble on to be revealed.

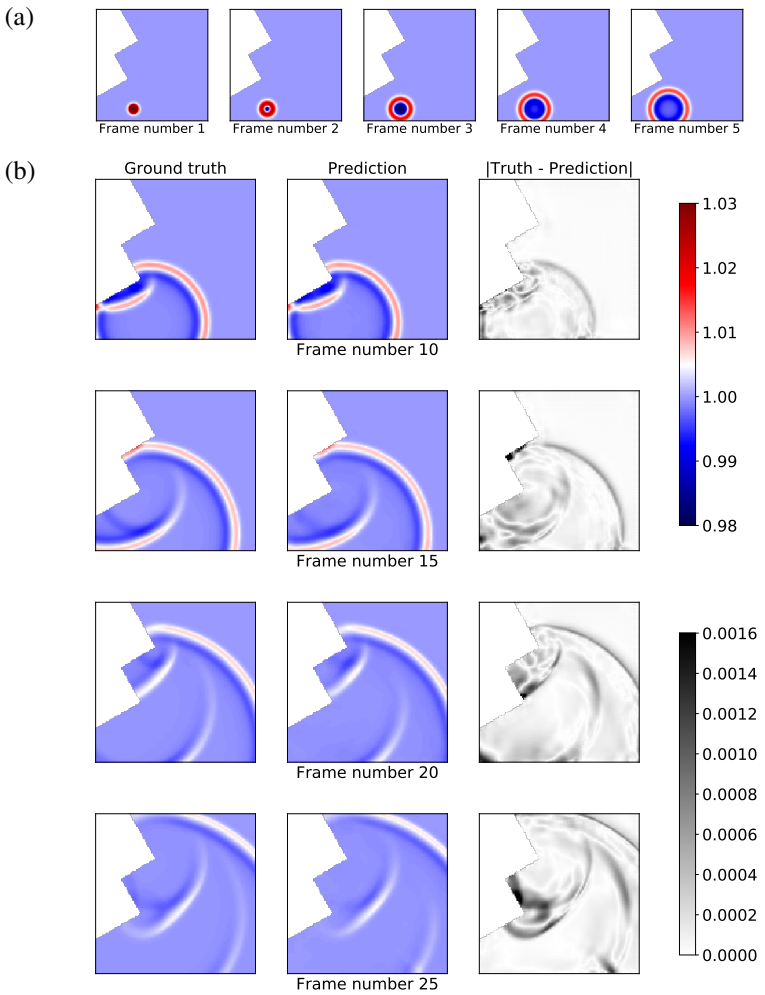


Figure 4: (a) Input sequence to the network. (b) Predictions (left column), ground truth (centre) and absolute difference (right) for such inputs. Values in metres [video].

The previous domain can be understood as an extension of the domain in Figure 2b, which was used during training, so the ability of the network to extrapolate to this configuration is not unexpected. We now explore the generalisation of the network to curved boundaries, which is potentially more challenging. Figures 5a and 5b depict the inputs and outputs for a simulation in the domain depicted in Figure 2d. By visual inspection, these results suggest that the network is able to accurately predict wave reflections on circular convex walls, and errors are of the magnitude with those in Figure 4b. The same conclusion was obtained for wave reflections on circular concave walls, as illustrated by Figure 6a.

The network is also able to produce good quality simulations involving reflections on walls whose radius of curvature is not uniform, although in these cases, more substantial differences between targets and predictions can be seen. For instance, the prediction in Figure 6b shows a high resemblance with the ground truth even after 20 time-steps, however, the difference between the ground truth and the predictions is larger than in simpler geometries. The temporal frames depicted in Figure 1 also illustrate the good generalisation to a fluid domain surrounded by spline-sided walls. The successful generalisation to this kind of domains is likely due to the localised support of the convolution kernels and the architecture. When the convolutions are applied, the network may interpret curved walls as polygonal walls made up of lots of many straight walls of size equal to the stride size. In that case, the higher the image resolution, the smaller radius of curvature that could be handled by the network.

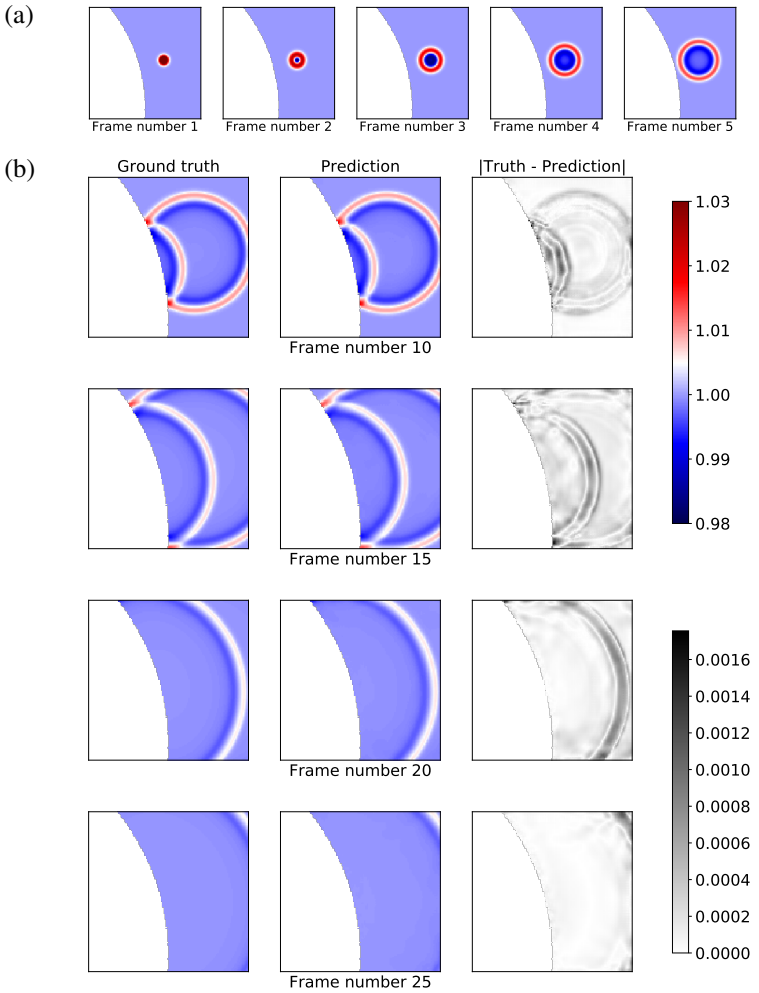


Figure 5: (a) Input to the network. (b) Predictions (left column), ground truth (centre) and absolute difference (right) for such inputs. Values in metres [video].

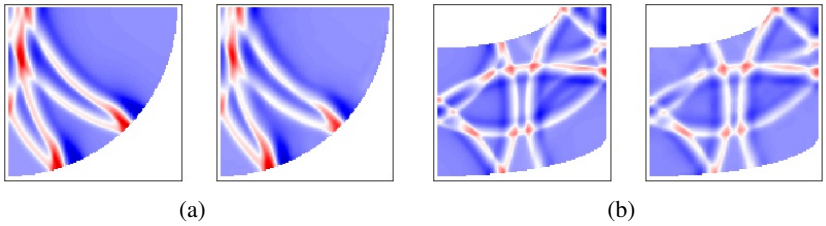


Figure 6: Subfigures (a) [video] and (b) [video] show predictions (left) and targets (right) after 20 time-steps.

4.3 Two Droplets as Initial Condition

The main challenge in using our model to extrapolate to the initial condition in equation (3) is the dissipation of the wavefronts after the interference between the two initial excitations. We observed that this effect is higher the greater number of interacting wavefronts. Simulations in the domains from Figures 2a and 2b can be accessed on https://ibb.co/SVFsDJq and https://ibb.co/NCw5gSg respectively.

4.4 LONG-TIME PREDICTIONS

We assessed the ability of the network to make accurate predictions over long-time periods by simulating in a larger domain. Figure 7 shows predictions and ground truth after 30, 55 and 80 time-steps. The initial condition is a droplet (equation 2) placed at the middle of the narrower channel. The network is able to predict accurately the speed of the leading wavefront, however, the wave magnitude is not retained and it is not able to maintain the propagation of the wavefronts originating from reflections within the narrower channel, which can be seen to quickly dissipate. This problem could be addressed by training the U-Net for longer output sequences and possibly increasing the depth of the network.

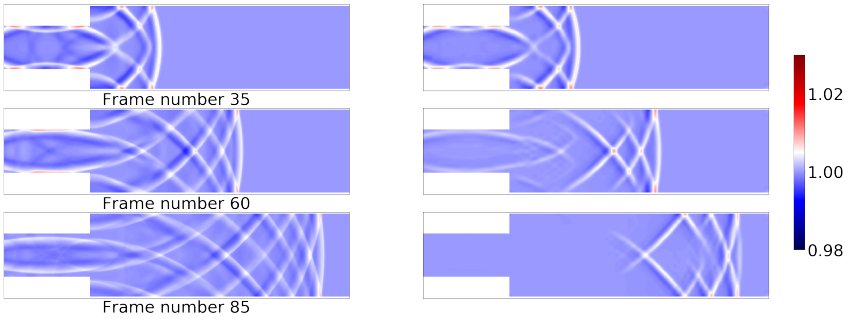


Figure 7: Predictions (left) and ground truth (right). The initial condition is a droplet placed at the middle of the narrower segment. Values in metres.

5 CONCLUSION

In this work we investigated the application of fully convolutional deep neural networks for forecasting wave dynamics on fluid surfaces. In particular, we focused on a U-Net architecture with two skip connections and we trained the network to predict the spatio-temporal evolution of wave dynamics, including: wave propagation, wave interference, wave reflection and wave refraction. We demonstrated that including into the loss function the MSE between the gradients of the predicted height and the truth height significantly reduces spurious oscillations in the solution and helped predict more accurately the position of the wavefronts. This suggests that loss functions which capture discrepancies in the predicted spatial gradients provide valuable information when training networks to forecast wave dynamics, especially over long-time intervals. The flow domains considered during training only included polygonal walls and right-angled corners. However, our U-Net was able to extrapolate to curved walls with varying radii of curvature. The RMSE was of order 0.0001 times the characteristic length for at least 20 time steps for both the training and testing datasets. Besides, when ran on a GPU these simulations are around 10^4 times faster than the ones ran for generating our data sets. These findings highlight the potential for neural networks to well approximate the evolution of wave dynamics with computational times several orders of magnitude smaller.

REFERENCES

Chris D Cantwell, David Moxey, Andrew Comerford, Alessandro Bolis, Gabriele Rocco, Gianmarco Mengaldo, Daniele De Grazia, Sergey Yakovlev, J-E Lombard, Dirk Ekelschot, et al. Nektar++: An open-source spectral/hp element framework. *Computer physics communications*, 192:205–219, 2015.

M. W. M. G. Dissanayake and N. Phan-Thien. Neural-network-based approximations for solving partial differential equations. *Communications in Numerical Methods in Engineering*, 10(3):195–201, 1994. doi: 10.1002/cnm.1640100303. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.1640100303.

M. W.M.G. Dissanayake and N. Phan-Thien. Neural-network-based approximations for solving partial differential equations. *Communications in Numerical Methods in Engineering*, 10(3): 195–201, 1994. ISSN 10990887. doi: 10.1002/cnm.1640100303.

Mehmet Ersoy, Omar Lakkis, and Philip Townsend. A saint-venant shallow water model for overland flows with precipitation and recharge, 2017.

Amir Barati Farimani, Joseph Gomes, and Vijay S. Pande. Deep Learning the Physics of Transport Phenomena, 2017. URL http://arxiv.org/abs/1709.02432.

Stathi Fotiadis, Eduardo Pignatelli, Mario Lino Valencia, Chris Cantwell, Amos Storkey, and Anil A. Bharath. Comparing recurrent and convolutional neural networks for predicting wave propagation, 2020. URL http://arxiv.org/abs/2002.08981.

Pilar García-Navarro, J Murillo, J Fernández-Pato, I Echeverribar, and Mario Morales-Hernández. The shallow water equations and their application to realistic cases. *Environmental Fluid Mechanics*, 19(5):1235–1252, 2019.

Xiaoxiao Guo, Wei Li, and Francesco Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, volume 13-17-Augu, pp. 481–490. Association for Computing Machinery, aug 2016. ISBN 9781450342322. doi: 10.1145/2939672.2939738.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2016.

George Karniadakis and Spencer Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, 2013.

Byungsoo Kim, Vinicius C. Azevedo, Nils Thuerey, Theodore Kim, Markus Gross, and Barbara Solenthaler. Deep Fluids: A Generative Network for Parameterized Fluid Simulations, 2019. ISSN 14678659.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, dec 2014. URL http://arxiv.org/abs/1412.6980.

Isaac E. Lagaris, Aristidis Likas, and Dimitrios I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9 5:987–1000, 1998.

Isaac Elias Lagaris, Aristidis C. Likas, and Dimitrios G. Papageorgiou. Neural-network methods for boundary value problems with irregular boundaries. *IEEE Transactions on Neural Networks*, 11 (5):1041–1049, 2000. ISSN 10459227. doi: 10.1109/72.870037.

Sangseung Lee and Donghyun You. Data-driven prediction of unsteady flow fields over a circular cylinder using deep learning, apr 2018. URL http://arxiv.org/abs/1804.06076http://dx.doi.org/10.1017/jfm.2019.700.

Lu Lu, Xuhui Meng, Zhiping Mao, and George E. Karniadakis. DeepXDE: A deep learning library for solving differential equations, 2019. URL http://arxiv.org/abs/1907.04502.

Tharindu P. Miyanawala and Rajeev K. Jaiman. An Efficient Deep Learning Technique for the Navier-Stokes Equations: Application to Unsteady Wake Flow Dynamics, 2017. URL http://arxiv.org/abs/1710.09099.

Ali Girayhan Özbay, Sylvain Laizet, Panagiotis Tzirakis, Georgios Rizos, and Björn Schuller. Poisson CNN: Convolutional Neural Networks for the Solution of the Poisson Equation with Varying Meshes and Dirichlet Boundary Conditions, 2019. URL http://arxiv.org/abs/1910.08613.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 10902716. doi: 10.1016/j.jcp.2018.10.045. URL https://doi.org/10.1016/j.jcp.2018.10.045.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations, nov 2017. URL http://arxiv.org/abs/1711.10561.

Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden Fluid Mechanics: A Navier-Stokes Informed Deep Learning Framework for Assimilating Flow Visualization Data, aug 2018. URL http://arxiv.org/abs/1808.04327.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi (eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241, Cham, 2015. Springer International Publishing.

Wilhelm E. Sorteberg, Stef Garasto, Alison S. Pouplin, Chris D. Cantwell, and Anil A. Bharath. Approximating the solution to wave propagation using deep neural networks, 2018. URL http://arxiv.org/abs/1812.01609.

Wei Tang, Tao Shan, Xunwang Dang, Maokun Li, Fan Yang, Shenheng Xu, and Ji Wu. Study on a Poisson's equation solver based on deep learning technique. *2017 IEEE Electrical Design of Advanced Packaging and Systems Symposium, EDAPS 2017*, 2018-Janua:1–3, 2018. doi: 10.1109/EDAPS.2017.8277017.

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):15–26, oct 2018. ISSN 0001-1452. doi: 10.2514/1.j058291. URL http://arxiv.org/abs/1810.08217.

S. Wiewel, M. Becher, and N. Thuerey. Latent Space Physics: Towards Learning the Temporal Evolution of Fluid Flow, 2019. ISSN 14678659.

Emre Yilmaz and Brian J. German. A convolutional neural network approach to training predictors for airfoil performance, 2017.

Yao Zhang, Woong Je Sung, and Dimitri Mavris. Application of convolutional neural network to predict airfoil lift coefficient, 2018.

A DATA SETS

The table below summarises the training and tesing sets used in the present work. The sequences in each data set contain 100 snapshots of the height field sampled at intervals of $\Delta t = 0.03$ seconds.

Table 1: Training and testing data sets

ID	Dataset	Purpose	Geometry	Initial Condition	Sequences
A	Box_Single_Drop	Training	Figure 2a	Single Drop (eq. (2))	500
B	Corner_Single_Drop	Training	Figure 2b	Single Drop (eq. (2))	500
C	Steps_Single_Drop	Testing	Figure 2c	Single Drop (eq. (2))	200
D	Convex_Single_Drop	Testing	Figure 2d	Single Drop (eq. (2))	250
E	Concave_Single_Drop	Testing	Figure 2e	Single Drop (eq. (2))	500
F	Spline_Single_Drop	Testing	Figure 2f	Single Drop (eq. (2))	200
G	Ellipse_Single_Drop	Testing	Figure 2g	Single Drop (eq. (2))	200
H	Box_Double_Drop	Testing	Figure 2a	Double Drop (eq. (3))	100
I	Corner_Double_Drop	Testing	Figure 2b	Double Drop (eq. (3))	200

The data sets A-C contain only straight-sided fluid domains. Data set A only includes wall boundary conditions, whereas B and C include open boundaries. Data sets D-G contain curved-sided domains. The domains in data set F were generated randomly given four random control points and using B-Splines to create a closed domain. The domains in data set G were also generated randomly given a concave quarter of an ellipse and a convex quarter of an ellipse, whose minor axes follow a uniform distribution between 0.25 and 0.5 m. Only the data sets A and B were used during training. The remaining data sets are used to demonstrate the ability of the network to generalise to arbitrary fluid domain geometries and to two droplets as initial condition.

B U-NET ARCHITECTURE

The diagram in Figure 8 depicts the U-Net architecture used in the present work. The network receives six fields as input: the geometry field Ω and a sequence of five consecutive height fields $\{h_s, h_{s+1}, h_{s+2}, h_{s+3}, h_{s+4}\}$. The output is a prediction of the subsequent height field $\hat{h}_{s+5}$. Whereas some recent studies (Fotiadis et al., 2020; Thuerey et al., 2018) have used bi-linear interpolation to perform the up-sampling, we opted to use transpose convolutions with a 2x2 kernel and stride 2. This increases the number of trainable parameters to 1,864,577, but we also noticed a significant improvement in the quality of the predictions.

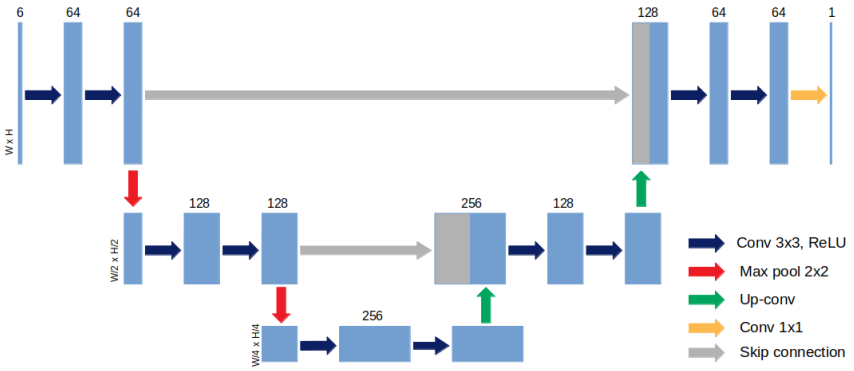


Figure 8: Our U-net architecture with 1,864,577 trainable parameters. It receives six fields as input: the geometry field and a sequence of five consecutive height fields. The output is a prediction of the height field in the subsequent time step.

C DATA AUGMENTATION AND GENERALISATION

To improve generalisation across a range of wave dynamics, the height fields were re-scaled according to $\tilde{h} \leftarrow (h - \bar{h})/(\max(h) - \bar{h})$, where $\bar{h} = 1$ and $\max(h) = 1.1$. This re-scaling was reversed for visualising the network predictions. In order to avoid over-fitting and improve the generalisation capabilities of the network, we apply two sets of transformations to the training data sets. For the data set *A* such transformations consist on random rotations of 90, 180 and 270 deg as well as horizontal and vertical flips. For the dataset *B* random rotations by multiples of 15 deg are applied and the physical size of the frames is reduced to 1 m $\times$ 1 m by cropping the original frames to domains whose center position follows an uniform distribution from 0.9 to 1.1 m in both spatial directions. Finally, the images of all these sequences were linearly interpolated to a 128 $\times$ 128 resolution to satisfy the 128 pix/m requirement. Regarding the testing data sets, sets *C, D* and *H* are augmented in the same manner as the data set *A*, since they contain open boundaries; and data sets *E, F, G* and *I* are augmented like *B*, as they contain only closed boundaries.

D SIMULATION EXAMPLES

Videos comparing Nektar++ simulations and simulations ran with our U-Net can be accessed on the links in Table 2. The examples from datasets *A* and *B* here included were used during training, whereas all the other simulations are extrapolations.

Table 2: Links to simulation examples

ID	Dataset	Link
A	Box_Single_Drop	https://ibb.co/pWhwrbd
B	Corner_Single_Drop	https://ibb.co/BZ0z1QC
C	Steps_Single_Drop	https://ibb.co/jfWqSdW
D	Convex_Single_Drop	https://ibb.co/PYVVp6V
E	Concave_Single_Drop	https://ibb.co/71bQMS5
F	Spline_Single_Drop	https://ibb.co/Cw4m4wt
G	Ellipse_Single_Drop	https://ibb.co/thjW5n4
H	Box_Double_Drop	https://ibb.co/SVFsDJq
I	Corner_Double_Drop	https://ibb.co/NCw5gSg