Approximating the Solution of Surface Wave Propagation Using Deep Neural Networks*

Wilhelm E. Sorteberg[1,2], Stef Garasto[1,2,*][0000−0001−5742−8073], Chris C. Cantwell[1][0000−0002−2448−3540], and Anil A. Bharath[1][0000−0001−8808−2714]

[1] Dept. of Bioengineering, Imperial College London, South Kensington Campus, SW7 2AZ, London, United kingdom
[*] stef.grs@gmail.com
[2] These authors are joint first authors.

Abstract. Partial differential equations formalise the understanding of the behaviour of the physical world that humans acquire through experience and observation. Through their numerical solution, such equations are used to model and predict the evolution of dynamical systems. However, such techniques require extensive computational resources and assume the physics are prescribed *a priori*. Here, we propose a neural network capable of predicting the evolution of a specific physical phenomenon: propagation of surface waves enclosed in a tank, which, mathematically, can be described by the Saint-Venant equations. The existence of reflections and interference makes this problem non-trivial. Forecasting of future states (i.e. spatial patterns of rendered wave amplitude) is achieved from a relatively small set of initial observations. Using a network to make approximate but rapid predictions would enable the active, real-time control of physical systems, often required for engineering design. We used a deep neural network comprising of three main blocks: an encoder, a propagator with three parallel Long Short-Term Memory layers, and a decoder. Results on a novel, custom dataset of simulated sequences produced by a numerical solver show reasonable predictions for as long as 80 time steps into the future on a hold-out dataset. Furthermore, we show that the network is capable of generalising to two other initial conditions that are qualitatively different from those seen at training time.

Keywords: Deep Learning for Physical Systems · Recurrent Neural Networks · Representation Learning.

1 Introduction

Partial differential equations (PDEs) are used to model physical systems in many different fields, including aeronautical, mechanical, electrical and electromagnetic systems design. The algorithms traditionally used for their numerical solution can achieve high accuracy, but can be computationally very expensive. Solvers may also require extensive parameter tuning and the dynamics are governed by the

* Supported by the Rosetrees trust.

PDEs and therefore prescribed *a priori*. There has been recent interest in using deep networks and machine learning to learn the behaviour of physical systems directly from data and to complement the role of traditional solvers [2, 3, 9–11]. For example, using approximate, but computationally low-cost, solutions could reduce the time needed for design iteration or provide real-time active control in complex systems. To achieve this, it is critical for the deep neural network to have the ability to generalise to many different initial conditions, even those that differ substantially from the data used during training, and to sustain long-term predictions without resorting back to numerical simulation software.

The main contribution of the present paper is the investigation of the use of deep learning to perform state-evolution prediction on the rendered amplitudes of surface waves enclosed by solid wall boundary conditions, using a novel modification of an existing architecture. This physical system is described by the Saint-Venant equations and used in many important applications, such as avalanche [7] and urban flood modelling [8]. It is non-trivial to solve, given the complexity of the equations, and the presence of reflections and interference phenomena. To the best of our knowledge, state evolution of surface wave phenomena using deep networks has not been previously attempted. The network acts solely on visual input, without knowing the parameters of the operating environment. From snapshots of rendered wave-amplitude observed at 5 instants in time (at the constant but arbitrary sampling rate of 100 Hz), we forecast the wave patterns over the next 10 time steps (i.e. 100 ms) using an encoder-propagator-decoder architecture [2]. Then, the last 5 predictions are repeatedly used as new inputs to enable long term predictions. The test of prediction quality is ambitious: after training on predictions of the next 20 frames, we propagate for 80 time steps (i.e. 800 ms) into the future at test time. Finally, we test the generalisation ability of the network on qualitatively new initial conditions.

2 Related Work

Previous uses of deep neural networks to predict the evolution of states of a system can be found within a variety of fields. Examples include predictions of multi-body dynamics [10], trajectories [2], a robot end-effector's interaction with its surroundings [3, 4], and the evolution of chaotic systems [9]. In fluid dynamics, two pivotal studies that showed how neural networks can help to speed-up numerical simulations for divergence-free fluids governed by the incompressible Navier-Stokes equations are those by Yang et al. [15] and Tompson et al. [12]. In particular, the latter focused on accelerating the pressure step of the numerical solver [12]. More recently, Wiewel et al. used an LSTM-based architecture (Long Short-Term Memory), alternated with numerical simulations for stability, to predict the evolution of the 3-dimensional pressure field of a fluid [14]. Moreover, Kim et al. presented a generative neural network that synthesises parametrisable, but not fully arbitrary, velocity fields for fluids from a low-dimensional parameters set [5]. It is worth noting that most of the neural networks applied to the Navier-Stokes equations are trained to predict either the pressure or the velocity fields,

or both, rather than the motion of particles within the fluid. However, it is harder to measure these fields experimentally as opposed to particle tracking. Thus, other works have applied deep learning to model particle motion [11]. Finally, Long et al. [6] use a hybrid algorithm combining a numerical solver based on Cellular Neural Networks with a LSTM that forecasts the evolution of external forces applied to the fluid. To the best of our knowledge, we are the first to predict the evolution of a system described by the Saint-Venant equations.

A strategy that is shared across many neural networks for dynamical predictions consists of first allowing the network to build its own representation of the spatio-temporal data and then propagating this latent space in time[2, 3, 9]. This procedure is based on the assumption that each individual propagation in time is governed by the same function. However, the propagator is either a convolutional layer or a single LSTM. Furthermore, there are differences in the way the system of interest is given as input to the network. For instance, Sanchez-Gonzales et al. use multiple graph networks that contain information about static or dynamic properties of the system [10], while Finn et al. employ a guided procedure where the network is presented with sensory data on actuators' live position and forces [3], and Ehrhardt et al. use a neural network which solely relies on visual input [2]. We followed the latter strategy, with the aim of extending the present work from simulations to experimental data.

3 The Saint-Venant equations

(a) $T = 0$. (b) $T = 25$. (c) $T = 50$. (d) $T = 75$. (e) $T = 100$.

Fig. 1: Example of surface wave propagation, generated using TriFlow [1]. T represents the number of time steps since the initial perturbation.

We aim to solve the 2-dimensional Saint-Venant equations in non-conservative form with no coriolis or viscous forces (a non-linear coupled system). They successfully model wave propagation phenomena and are used in many fields, including avalanche [7] and urban flood modelling [8]. They are given by:

$$\begin{aligned} h_t + ((H+h)u)_x + ((H+h)v)_y &= 0 \\ u_t + uu_x + vu_y + gh_x - \nu(u_{xx} + u_{yy}) &= 0 \\ v_t + uv_x + vv_y + gh_y - \nu(v_{xx} + v_{yy}) &= 0 \end{aligned}$$

Here, H is the reference water height and h is the deviation from this reference, u and v are the velocities in the x and y direction, respectively, g is the gravitational

acceleration, ν is the kinematic viscosity (set to 10^{-6} $\mathrm{m^2 s^{-1}}$), and u_x is the partial derivative of u with respect to x. We used solid wall boundary conditions along the whole perimeter of the environment. The initial condition was given by a droplet (a localised 2-dimensional Gaussian), and we varied its location, as well as the propagation speed, within the dataset. The simulations were run using TriFlow[1], with the generation of each simulated frame taking, on average, 0.44 seconds on a CPU. The final images (Figure 1) are given by a rendering model with azimuth of 45 degrees and altitude of 20 degrees. We ran 3,000 simulations, each with 100 images at a resolution of 128×128 and a sampling frequency of 100 Hz (300,000 images in total). During training, random data augmentation was implemented by random flips in both the horizontal and the vertical direction. For each network trained, we randomly split the 3,000 simulations into training (70%), validation (15%), and test (15%) sets. To assess the network's generalisation performance onto more challenging scenarios, we also simulated two new initial conditions: two droplets with random locations and a linear wave front.

4 Long term predictor network architecture

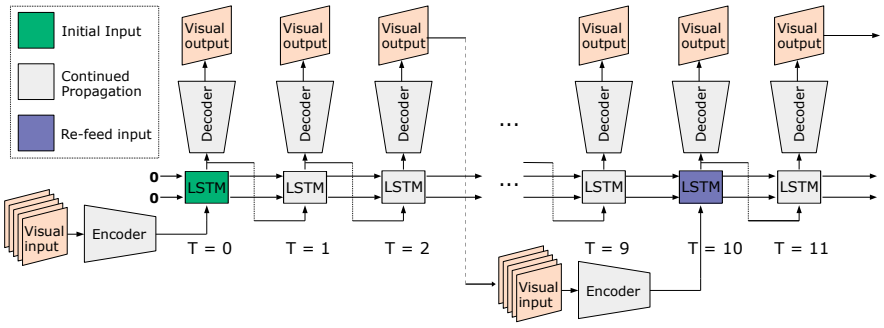


Fig. 2: Depiction of the internal structure and data flow of the prediction network with three LSTMs. Only one of the three LSTMs is used at any time.

Our neural network consists of an autoencoder structure with a propagator in its centre [2]. The data work-flow is as follows. Five states representing rendered wave patterns at 5 consecutive instants in time are given as the initial input. This is passed through the encoder, then repeatedly propagated in time by selecting one of three fully connected LSTMs (1000 units each), instead of just a single LSTM as applied in [2]. Finally, a decoder transforms the propagated latent vectors into 10 predicted future states. From these, the last 5 fields were reinserted again as input for further propagation. To handle this protocol seamlessly, we used three LSTMs in the propagator, with only one being active at a time, according to the type of prediction. One LSTM dealt with the initialisation of the network, one performed the self-propagation, and the last one was active during the reinsertion

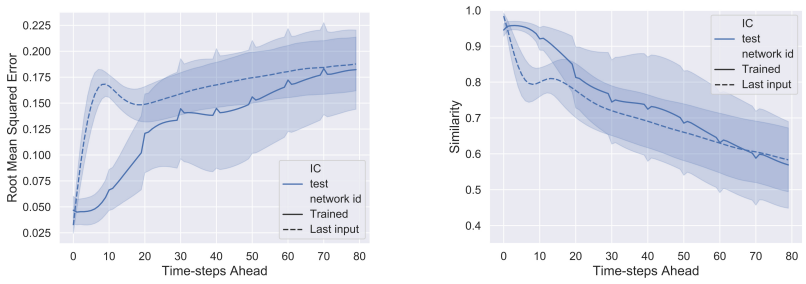
step. Preliminary results (in the Supplementary Material) appear to indicate that this modification is better than a single LSTM. Specifically, using only a single LSTM produced cyclical artefacts in time, with spikes appearing when predictions where reinserted as inputs. Such an effect was reduced when using the three LSTM structure. All three LSTMs shared the same hidden and cell states, ensuring complete information preservation (visualised in Figure 2). The encoder has 4 convolutional layers (60, 120, 240 and 480 units, using a stride of 2, 7×7 and 3×3 kernels for the first and the following layers, respectively), and a fully connected layer to a latent space of 1000 nodes. The *tanh* non-linearity was used throughout. Regularisation was induced with dropout and batch normalisation layers. The architecture of the decoder mirrors that of the encoder, using deconvolutional layers. We used the Mean Square Error (MSE) between targets and predictions as the loss function, and ADAM as the optimization scheme. At training and test time, 20 and 80 frames were predicted, respectively, to test the network's extrapolation abilities. Using Pytorch and a NVIDIA GTX 1070, training each network for 50 epochs took approximately 7 hours.

5 Results

We trained 10 predictive networks, of which 6 converged successfully. The failure of the remaining 4 is likely due to the lack of sufficient data to train three large LSTMs (longer training did not help). By combining LSTM propagation and reinsertion of the resultant outputs, we predict 80 time steps into the future from an initial input of 5 images. We report not only the outcome of the most successful network, but also the average results across the entire ensemble of converged networks, to show the variability intrinsic in training for such a complex task.

5.1 Performance Assessment

A quantitative test of the network's predictive quality as a function of time steps from the last initial input is shown in Figure 3. Specifically, we evaluated the image quality of the predictions using the Root Mean Squared Error (RMSE, Figure 3a) and the Structural Similarity index (SSIM, Figure 3b) between targets and predictions. The former aims to quantify the pixelwise spatio-temporal accuracy of the predictions, however it does not always correlate with perceptual similarity, which is instead captured by the latter [13]. Results are presented by the average (solid lines) and the standard deviation (shaded area) across all the networks that converged. Our best network achieved an average SSIM (RMSE) of 0.96 (0.04), 0.89 (0.09), 0.80 (0.11), 0.66 (0.16) and 0.56 (0.19) at time-steps 1, 20, 40, 60, and 80, respectively. One of the main challenges for the network is to extrapolate predictions to 80 frames after only being trained on 20: as expected, the SSIM decreases and the RMSE increases for longer term predictions. However, the lack of a sharp decrease in performance after $T = 20$ is promising. To contextualise the absolute accuracy values, we also plot RMSE and SSIM between the targets and the last frame of the input sequence ("last



(a) (b)

Fig. 3: RMSE (a) and SSIM (b) between targets and predictions (solid lines). For comparison, we show the same measures computed between the targets and the last input frames (dashed line). Lines and shadowed areas are the mean value and the standard deviation across the whole test dataset, respectively. Our best network achieved an average SSIM (RMSE) of 0.96 (0.04), 0.89 (0.09), 0.80 (0.11), 0.66 (0.16) and 0.56 (0.19) at time-steps 1, 20, 40, 60, and 80, respectively.

input" baseline). The purpose of such a measure is to estimate how well the network would perform without any physical knowledge, that is where its best guess at predicting the future is the last input it received. From $T = 60$, the SSIM suggests modest improvement with respect to this "last input" baseline, while visual inspection (Figure 4) suggests that some predictions are closer to the target patterns than the last input frame.

An example sequence of initial inputs, as well as target and predicted rendered wave patterns can be seen in Figure 4. Five frames, taken at regular intervals, are shown to illustrate both short- and long-term prediction. Despite performance degradation over time, the network seems to sustain the relevant features of the propagating waves for long periods of time, almost until the end of the simulation for some features. It is worth remarking that the whole sequence is predicted by the neural network only, without any call to the numerical simulation software. Furthermore, to make correct predictions, the network not only needs to infer how to update the current state, but is also required to perform an implicit visual estimate of relevant physical parameters, such as the viscosity and density, directly from the input data. Figure 5 shows another prediction example (from a different trained network), together with the intensity profile of the ground truth and predicted images, along a row situated near the boundary surface. Such a line was chosen to track the wave front during reflection, which is harder to model. It can be seen that the network is able to replicate the spatial pattern of the rendered wave amplitude (top row), albeit with lower accuracy around the reflection at the border (bottom row). Finally, we report that predicting and saving the entirety of the 80-frames time sequence took less than half a second. More example results can be found in the Supplementary Material (http://bit.ly/2P2Vf6g).

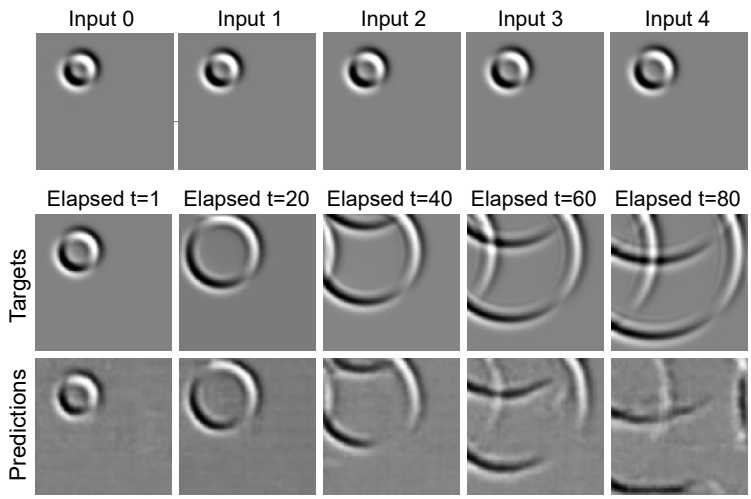


Fig. 4: Example initial inputs (top row), targets from numerical solver (middle row) and spatio-temporal pattern predicted by the trained network number 4 (bottom row): t represents the time-steps elapsed after the last initial input frame.

5.2 Generalisation Performance

On top of assessing the networks on left-out test data, we also tested them on two qualitatively different types of initial conditions: specifically, i) two droplets and ii) a linear propagating wave front. This allowed us to evaluate how well the trained networks captured the physics of the wave propagation. Results are shown in Figure 6 and Figure 7. The former shows the RMSE and the SSIM between targets and predictions for the two new types of initial conditions, averaged across all trained networks. For comparison, we included the same measures computed between the last original input and the current target frame. Figure 7

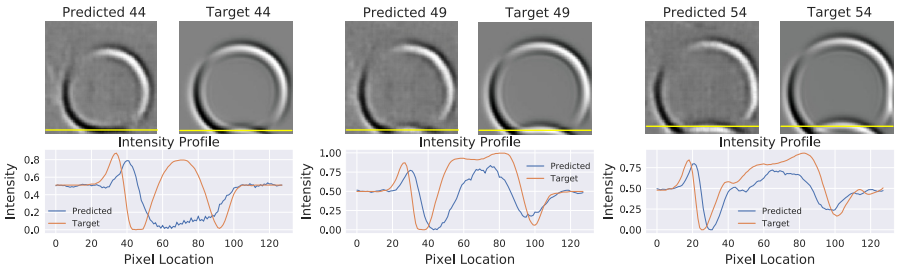
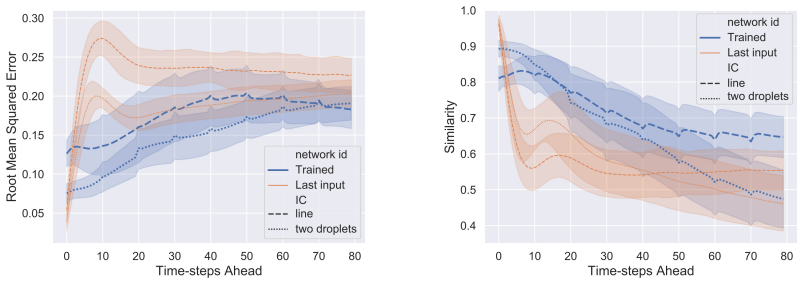


Fig. 5: Top: predicted two-dimensional spatial patterns from trained network number 3 vs targets from numerical solver. Bottom: spatial intensity profiles for predictions and targets along the horizontal line in the frames on top. This line approximately follows the wave front as it goes through reflection.



(a) (b)

Fig. 6: RMSE (a) and SSIM (b) between targets and predictions (thick blue lines) for the two new initial conditions (each represented by a different dashed pattern). For comparison, we show the same measures computed between the targets and the last initial input frame (thin orange lines). Lines and shadowed areas are the mean values and the standard deviation, respectively, across all trained networks. For the linear wave front, our best network achieved an average SSIM (RMSE) of 0.83 (0.12), 0.79 (0.15), 0.69 (0.19), 0.65 (0.20) and 0.64 (0.18) at time-steps 1, 20, 40, 60, and 80, respectively. For the two droplets, the same values are 0.91 (0.07), 0.80 (0.11), 0.69 (0.14), 0.51 (0.19), 0.46 (0.20).

displays some example predictions. As shown by the error and accuracy curves corresponding to the "last input" baseline, the new scenarios are harder for the network to predict. The linear wave front is a particularly challenging test, since it involves both a new scenario and a new wave shape. Despite this, the results suggest that, although the predictions are not as accurate, the network can adapt to completely unseen data which varies significantly in spatial distribution from the training data. At the same time, the influence of the curved wave front – a prior created by the training data – is also clearly visible, especially for predictions further ahead in time. At the moment, we do not know exactly how strong this influence is.

6 Discussion

The results presented in this work suggest that our trained network can perform state prediction of environments governed by complex physical laws. We believe this to be the first instance where such an analysis pertains to a system described by the Saint-Venant equations. Thus, no alternative state-of-the-art exists, though possible future extensions might involve comparison with optical flow methods. From 5 frames of visual input, the network is able to predict up to 80 frames into the future with reasonable accuracy, although with a lower quality than the numerical solver, especially for later time steps. Nevertheless, given that only the first 20 frames were back-propagated during training, the network exhibits a good extrapolation performance. The prediction of 80 frames using the neural

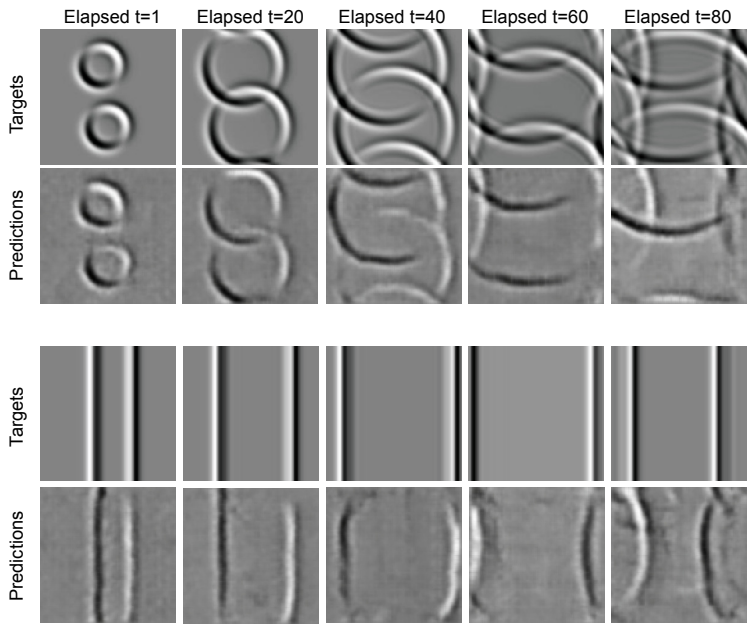


Fig. 7: Example targets from the numerical solver and spatio-temporal patterns predicted by the trained network nb. 3. The first and last two rows show predictions for the initial condition with two droplets and the linear wave front, respectively. The time-steps for each prediction corresponds to time-steps elapsed after the last initial input frame.

network takes less than half a second, promoting the idea of using these techniques for real-time simulation and control. Given sufficiently general training, similar networks could be employed as an approximate simulation tool to perform initial parameter sweeps, or quickly iterate over designs. Results on the generalisation to two different unseen initial conditions are also promising. However, there are still some limitations. First, a minority of the networks we trained did not reach convergence and there was some variability in performance between networks. Secondly, some portions of the rendered wave amplitude pattern are lost over time. Finally, when tested on new scenarios, the influence of the prior learned during training is evident. For future work, we will seek to: a) increase prediction accuracy (and thus the generalisation performance); b) investigate architectures and benchmarks for generalisation (e.g. multiple sources, wave front shapes and boundary conditions); c) stabilise the training by assessing different cost functions; d) analyse the latent space representation learned by the network; e) extend the approach to data from real-world wave propagation.

Bibliography

[1] Cellier, N.: locie/triflow: Triflow v0.5.0. (May 2018), https://doi.org/10.5281/zenodo.1239703

[2] Ehrhardt, S., Monszpart, A., Mitra, N.J., Vedaldi, A.: Learning A Physical Long-term Predictor. ArXiv e-prints arXiv:1703.00247 (Mar 2017)

[3] Finn, C., Goodfellow, I., Levine, S.: Unsupervised learning for physical interaction through video prediction. In: NeurIPS. pp. 64–72 (2016)

[4] Guevara, T.L., Gutmann, M.U., Taylor, N.K., Ramamoorthy, S., Subr, K.: Adaptable Pouring: Teaching Robots Not to Spill using Fast but Approximate Fluid Simulation. ArXiv e-prints arXiv: 1708.01465v2 (Aug 2017)

[5] Kim, B., Azevedo, V.C., Thuerey, N., Kim, T., Gross, M., Solenthaler, B.: Deep fluids: A generative network for parameterized fluid simulations. ArXiv e-prints arXiv:1806.02071 (Jun 2018)

[6] Long, Y., She, X., Mukhopadhyay, S.: HybridNet: Integrating Model-based and Data-driven Learning to Predict Evolution of Dynamical Systems. ArXiv e-prints arXiv:1806.07439 (Jun 2018)

[7] Mangeney-Castelnau, A., Vilotte, J.P., Bristeau, M.O., Perthame, B., Bouchut, F., Simeoni, C., Yerneni, S.: Numerical modeling of avalanches based on saint venant equations using a kinetic scheme. Journal of Geophysical Research: Solid Earth **108**(B11) (2003)

[8] Oezgen, I., Zhao, J., Liang, D., Hinkelmann, R.: Urban flood modeling using shallow water equations with depth-dependent anisotropic porosity. Journal of Hydrology **541**, 1165–1184 (2016)

[9] Pathak, J., Lu, Z., Hunt, B.R., Girvan, M., Ott, E.: Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data. Chaos: An Interdisciplinary Journal of Nonlinear Science **27**(12), 121102 (Dec 2017)

[10] Sanchez-Gonzalez, A., Heess, N., Springenberg, J.T., Merel, J., Riedmiller, M., Hadsell, R., Battaglia, P.: Graph networks as learnable physics engines for inference and control. ArXiv e-prints arXiv:1806.01242 (Jun 2018)

[11] Schenck, C., Fox, D.: Spnets: Differentiable fluid dynamics for deep neural networks. In: Billard, A., Dragan, A., Peters, J., Morimoto, J. (eds.) Proceedings of The 2nd Conference on Robot Learning. Proceedings of Machine Learning Research, vol. 87, pp. 317–335. PMLR (29–31 Oct 2018)

[12] Tompson, J., Schlachter, K., Sprechmann, P., Perlin, K.: Accelerating eulerian fluid simulation with convolutional networks. ArXiv e-prints arXiv:1607.03597v6 (Jun 2017)

[13] Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)

[14] Wiewel, S., Becher, M., Thuerey, N.: Latent-space Physics: Towards Learning the Temporal Evolution of Fluid Flow. ArXiv e-prints arXiv:1802.10123v2 (Jun 2018)

[15] Yang, C., Yang, X., Xiao, X.: Data-driven projection method in fluid simulation. Computer Animation And Virtual Worlds **27**, 415–424 (2016)