LEARNING VORTEX DYNAMICS FOR FLUID INFERENCE AND PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a novel machine learning method based on differentiable vortex particles to infer and predict fluid dynamics from a single video. The key design of our system is a particle-based latent space to encapsulate the *hidden*, Lagrangian vortical evolution underpinning the *observable*, Eulerian flow phenomena. We devise a novel differentiable vortex particle system in conjunction with their learnable, vortex-to-velocity dynamics mapping to effectively capture and represent the complex flow features in a reduced space. We further design an end-to-end training pipeline to directly learn and synthesize simulators from data, that can reliably deliver future video rollouts based on limited observation. The value of our method is twofold: first, our learned simulator enables the inference of hidden physics quantities (*e.g.* velocity field) purely from visual observation, to be used for motion analysis; secondly, it also supports future prediction, constructing the input video's *sequel* along with its future dynamics evolution. We demonstrate our method's efficacy by comparing quantitatively and qualitatively with a range of existing methods on both synthetic and real-world videos, displaying improved data correspondence, visual plausibility, and physical integrity.[1]

1 INTRODUCTION

As small as thin soap films, and as large as atmospheric eddies observable from outer space, fluid systems can exhibit intricate dynamic features on different mediums and scales. However, despite the inspiring recent progress, to effectively represent these flow features, identify the underlying dynamics system, and predict the future evolution, remains an open problem for scientific machine learning, due to the noisy data, imperfect modeling, and unavailable, *hidden* physics quantities.

Here, we identify three fundamental challenges that currently hinder the success of such endeavors. First, *flow features are difficult to represent*. Traditional methods learn the fluid dynamics by storing velocity fields either using regularly-spaced grids or smooth neural networks. These approaches have demonstrated promising results for fluid phenomena that are relatively damped and laminar (*e.g.* Chu et al., 2022), but for fluid systems that can exhibit turbulent features on varying scales, these methods fall short due to the problem's curse of dimensionality (high-resolution space and time), local non-smoothness, and hidden constraints. As a result, more compact and structured representation spaces and data structures are called for.

Secondly, *hidden flow dynamics is hard to learn.*. Fluid systems as prescribed by the Navier-Stokes equations tightly couple multiple physical quantities (*i.e.* velocity, pressure, and density), and yet, only the density information can be accessibly measured. Due to the system's complexity, ambiguity, and non-linearity, directly learning the underlying dynamics from the observable density space is infeasible; and successful learning is usually contingent on velocity or pressure supervision, a requirement that distances these methods from deployment in real-world scenarios.

Exciting recent progress has been made in hidden dynamics inference by PDE-based frameworks such as (Raissi et al., 2020), which uncover the underlying physics variables solely from density observations. However, this type of methods encounter the third fundamental challenge, which is that *performing future prediction is difficult.* As we will demonstrate, although strong results are ob-

[1]We invite the readers to view the video results via an anonymized link: `learning-vortex-dynamics.github.io`

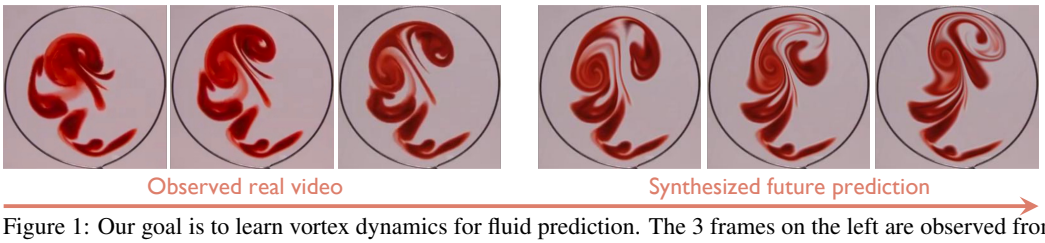

Figure 1: Our goal is to learn vortex dynamics for fluid prediction. The 3 frames on the left are observed from a real video recording a soap film on a circular metal rim, where the red ink is spreading. The 3 frames on the right are future prediction results produced by our method.

tained for interpolating inside the observation window provided by the training data, these methods render themselves unsuited for extrapolating into the future, profoundly limiting their usage.

In this paper, we propose a novel fluid learning method to tackle the three aforementioned challenges in a unified framework. In particular, harnessing the physics insights developed for the *vortex methods* in the computational fluid dynamics (CFD) literature, we design a novel, data-driven Lagrangian vortex system to serve as a compact and structured *latent representation* of the flow dynamics. We learn the complex dynamic system in the high-dimensional image space by learning a surrogate, low-dimensional model on the latent vortex space, and use a physics-based, learnable mechanism: the vortex-to-velocity dynamics module, to *decode* the latent space dynamics back to the image space. Leveraging this advantageous representation, we design an end-to-end training pipeline that learns from a single video containing only density information, to jointly perform accurate inference of hidden physics quantities and robust long-term future predictions, as shown in Figure 1.

To examine the efficacy of our method, we compare our method performance on both *motion inference* and *future prediction* against various state-of-the-art methods along with their extensions. We conduct benchmark testing on synthetic videos generated using high-order numeric simulation schemes as well as real-world videos in the wild. Evaluation is carried out both quantitatively through exhaustive numerical analysis, and qualitatively by generating a range of realistic visual effects. We compare the uncovered velocities both in terms of data correspondence and physical integrity, and the predicted visual results in terms of both pixel-level and perceptual proximity. Results indicate that our proposed method provides enhanced abilities on both fronts, inferring hidden quantities at higher accuracy, and predicting future evolution with higher plausibility.

In summary, the main technical contributions of our framework align with the three challenges we have addressed regarding flow representation, dynamics learning, and simulator synthesis. (1) We devise a novel fluid dynamics representation with *differentiable vortex particles*, to drastically reduce the learning problem's dimensionality on complex flow fields. Motivated by the vortex methods in CFD, we establish the vorticity-carrying fluid particles as a new type of learning primitive to tackle the existing PDE-constrained optimization problem to a particle ODE trajectory learning problem. (2) We design a novel particle-to-field paradigm for learning the Lagrangian vortex dynamics. Instead of learning the interaction among particles (*e.g.* Sanchez-Gonzalez et al., 2020), our model learns the continuous vortex-to-velocity induction mapping to naturally connect the vortex particle dynamics in the latent space and the fluid phenomena captured in the image space. (3) We develop an *end-to-end differentiable pipeline* composed of two network models to synthesize data-driven simulators based on single, short RGB videos.

2 Related Work

Hidden Dynamics Inference. The problem of inferring dynamics systems based on noisy or incomplete observations has been addressed using a variety of techniques, including symbolic regression (Bongard & Lipson, 2007; Schmidt & Lipson, 2009), dynamic mode decomposition (Schmid, 2010; Kutz et al., 2016), sparse regression (Brunton et al., 2016; Rudy et al., 2017), Gaussian process regression (Raissi et al., 2017; Raissi & Karniadakis, 2018), and neural networks (Raissi et al., 2019; Yang et al., 2020; Jin et al., 2021; Chu et al., 2022). Among these inspiring advancements, the "hidden fluid mechanics" (HFM) method proposed in Raissi et al. (2020) is particularly noteworthy, as it uncovers the continuous solutions of fluid flow using only images (the transport of smoke or ink).

Data-driven Simulation. Recently, growing interests are cast on learning numerical simulators according to data supervision, which has shown promise to reduce computation time (Ladický et al.,

2015; Guo et al., 2016; Wiewel et al., 2019; Pfaff et al., 2020; Sanchez-Gonzalez et al., 2020; Tompson et al., 2017), increase simulation realism (Chu & Thuerey, 2017; Xie et al., 2018), enable stylized control (Kim et al., 2020), estimate dynamic quantities such as viscosity and energy (Chang et al., 2016; Battaglia et al., 2016; Ummenhofer et al., 2019), and facilitate the training of control policies (Sanchez-Gonzalez et al., 2018; Li et al., 2018). Akin to Watters et al. (2017), our system takes images as inputs and performs dynamics simulation on a low-dimensional latent space; but our method learns purely from the input video and performs future rollout in the image space.

Vortex Methods. The underlying physical prior incorporated in our machine learning system is rooted in the family of vortex methods that are rigorously derived, analyzed, and tested in the computational fluid dynamics (CFD) (Leonard, 1980; Perlman, 1985; Beale & Majda, 1985; Winckelmans & Leonard, 1993; Mimeau & Mortazavi, 2021) and computer graphics (CG) community (Selle et al., 2005; Park & Kim, 2005; Weißmann & Pinkall, 2010; Brochu et al., 2012). Xiong et al. (2020) is pioneering for combining the Discrete Vortex Method with neural networks, but its proposed method relies on a large set of ground truth velocity sequences, whereas our method learns from single videos without needing the ground truth velocity.

3 Physical Model

We consider the velocity-vorticity formulation of the Navier–Stokes equations (by taking the curl operator on both sides of the momentum equation, see Cottet et al. (2000) for details):

$$\frac{D\boldsymbol{\omega}}{Dt} = \frac{\partial \boldsymbol{\omega}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{\omega} = (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} + \nu \nabla^2 \boldsymbol{\omega} + \nabla \times \boldsymbol{b}, \tag{1}$$

$$\boldsymbol{u} = \nabla \times \boldsymbol{\phi}, \quad \nabla^2 \boldsymbol{\phi} = -\boldsymbol{\omega}, \tag{2}$$

where $\boldsymbol{\omega}$ denotes the vorticity, $\boldsymbol{u}$ the velocity, $\boldsymbol{b}$ the body force, ν the kinematic viscosity, and ϕ the streamfunction. If we ignore the viscosity and stretching terms (inviscid 2D flow), we obtain $D\boldsymbol{\omega}/Dt = \mathbf{0}$, which directly conveys the Largangian conservative nature of vorticity (*i.e.* a particle's vorticity will not change during its advection).

If we assume the fluid domain has an open boundary, we can further obtain the vorticity-to-velocity induction formula, which is derived by solving the the Poisson equation on ϕ using Green's method (also known as Biot-Savart Law in fluid mechanics):

$$\boldsymbol{u}(\boldsymbol{x}) = \int K(\boldsymbol{x} - \boldsymbol{x}')\boldsymbol{\omega}(\boldsymbol{x}')d\boldsymbol{x}', \tag{3}$$

The kernel K exhibits a type-II singularity at 0 and causes numerical instabilities, therefore in CFD practices, K is replaced by various mollified versions K_δ to improve the simulation accuracy (Beale & Majda, 1985). We note that the mollified version K_δ is not unique, and can be customized and tuned in different numerical schemes per human heuristics. Different types and parameters for the mollification bring about vastly different simulation results.

Takeaways. The mathematical models above provide two central physical insights guiding the design of our vortex-based learning framework. (1) The Lagrangian conservation of vorticity $\boldsymbol{\omega}$ suggests the suitablity of adopting Lagrangian data structures (*e.g.* particles as opposed to grids) to capture the dynamics. Since the tracked variable $\boldsymbol{\omega}$ remains temporally-invariant for each Lagrangian vortex, the evolution of the continuous flow field is embodied fully by the movement of these vortices, which significantly alleviates the difficulty in learning. (2) Equation 3 presents an induction mapping from the vorticity $\boldsymbol{\omega}$, a Lagrangian quantity carried by particles, to the velocity $\boldsymbol{u}$, an Eulerian variable that can be queried continuously at arbitrary location $\boldsymbol{x}$. This lends the possibility for the Lagrangian method to be used in conjunction with Eulerian data structures (*e.g.* a grid) for learning from the widely available video data. Furthermore, such a mapping can benefit from data-driven learning, as we can replace human heuristics by learning the mollified kernel K_δ (which is shared among all vortex particles) to minimize the discrepancy between the induced and observed flow phenomena.

4 Method

System Overview. Following the physics insight conveyed in Section 3, we design a learning system whose workflow is illustrated in Figure 2. As shown on the top row, our system takes as input a single RGB video that captures the vortical flow phenomena. As shown on the bottom row, our

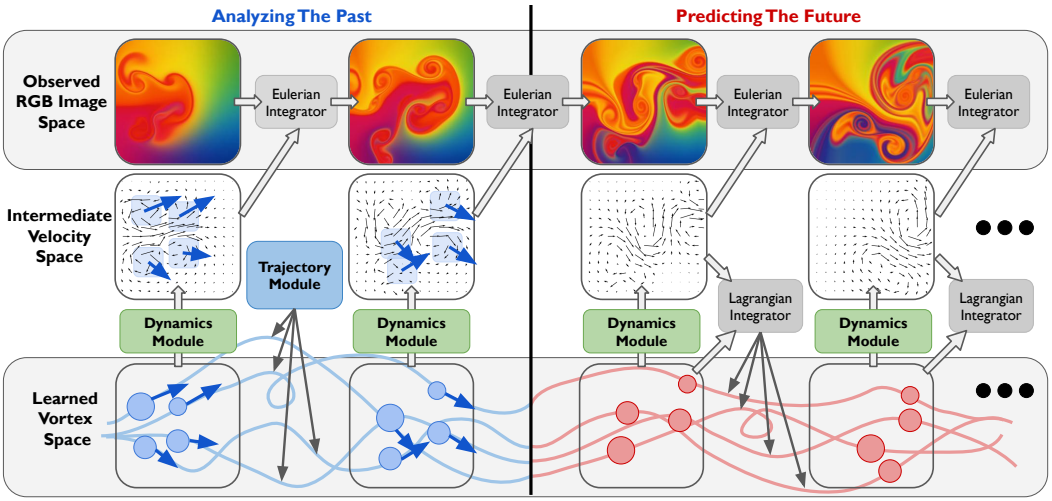

Figure 2: Given an input RGB image sequence (top row), we learn the dynamic system of a low-dimensional vortex space (bottom row), whose motion is decoded into the motion of the high-dimensional image space to explain the observed phenomena.

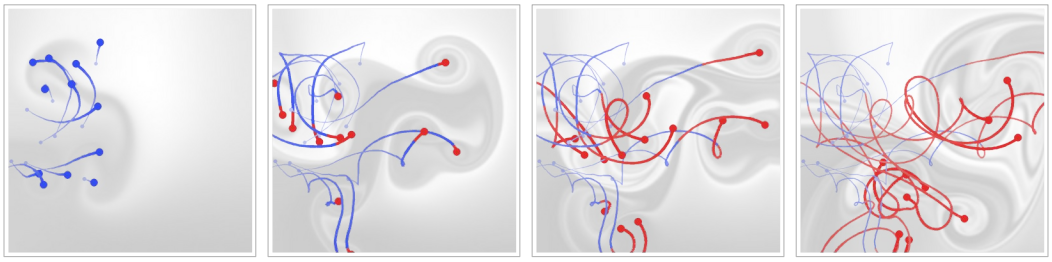

Figure 3: We encapsulate the motion of a continuous field by the motion of discrete particles. The blue trajectory is encoded by a neural network N_1, corresponding to the input video; while the red trajectory is unrolled using a numeric integrator, corresponding to the future prediction.

method learns and outputs a dynamic simulator — not on the image space itself, but on a latent space consisting of discrete vortices. Learning the latent dynamics in the vortex space would only be useful and feasible if we can tie it back to the image space, because it is the image space that we want to perform future prediction on, and we have no ground truth values for the vortex particles to begin with. The bridge to tie the vortex space with the image space is derived from Equation 3, which supplies the core insight that there exists a learnable mapping from vortex particles to the continuous velocity field at arbitrary position. This mapping is modelled by our learned dynamics module $\mathcal{D}$, which gives rise to the intermediate velocity space, as shown in the middle row of Figure 2.

4.1 Differentiable Vortex Particles

We initialize a group of n vortex particles in the fluid domain and optimize their conserved or time-varying properties to match the observed fluid phenomena. Each vortex V_i is the 3-tuple $(\boldsymbol{x}_i, \omega_i, \delta_i)$, where $\boldsymbol{x}$ represents the position, ω the strength, and δ the size. Let $\mathcal{V}$ denote the collection of vortices, *i.e.* $\mathcal{V} := [V_1, \ldots, V_n]$ where n is a hyperparamter, our learned method should be able to numerically solve for $\mathcal{V}_t$ for an arbitrary time t for future prediction.

Learning Particle Trajectory. As shown in Figure 3, the particle trajectory module represents a query function $\mathcal{T}$ such that $\mathcal{V}_t = \mathcal{T}(t)$, which outputs the configuration of all the vortices at some time $t \in [t_S, t_E]$. Since we assume that the dynamics system is inviscid and 2-dimensional, the vorticity amount and distribution carried by each vortex is time-invariant, hence we will treat δ and ω as a learnable, time-invariant quantity for each vortex. As a result, we use two trainable $n \times 1$ vectors Δ and Ω to represent the collection of δ and ω respectively, such that $[\omega_1, \ldots, \omega_n] = \sin(\Omega)$ and $[\delta_1, \ldots, \delta_n] = \text{sigmoid}(\Delta) + \epsilon$. We use a network $N_1(t)$ to encode $N_1(t) = [(\boldsymbol{x}_1)_t, \ldots, (\boldsymbol{x}_n)_t]$. We note that the particle velocities can be extracted as $\frac{dN_1}{dt}$ using automatic differentiation.

Learning the Vorticity-to-Velocity Mapping. The vorticity-to-velocity mapping is performed by our dynamics module, which predicts the velocity $\boldsymbol{u}$ given arbitrary query point $\boldsymbol{x}$ and the collection of vortices $\mathcal{V} = [(\boldsymbol{x}_1, \omega_1, \delta_1), \ldots, (\boldsymbol{x}_n, \omega_n, \delta_n)]$. Following the physics insight conveyed in Section 3, $\mathcal{D}$ embodies the integration:

$$\boldsymbol{u}(\boldsymbol{x}) = \int K_\delta(\boldsymbol{x} - \boldsymbol{x}')\boldsymbol{\omega}(\boldsymbol{x}')d\boldsymbol{x}', \tag{4}$$

which replace the kernel K by a learnable $K_\delta : \mathbb{R}^d \to \mathbb{R}^d$ mapping, with d representing the spatial dimension. Rather than directly using a neural network to model this $\mathbb{R}^d \to \mathbb{R}^d$ mapping, we further incorporate physical insights by analyzing the structure of K_δ. As derived in Beale & Majda (1985), the kernel K_δ for 2-dimensional flow exhibits the following form:

$$K_\delta(\boldsymbol{z}) = \frac{1}{2\pi r} M(\delta) R_{\frac{2}{\pi}}(\boldsymbol{z}), \tag{5}$$

where $R_{\frac{2}{\pi}}$ is the rotation matrix, applying which to $\boldsymbol{z}$ computes the unit direction of the cross product of $\boldsymbol{z}$ and the out-of-plane vorticity vector $\boldsymbol{\omega}$; and $M(\delta)$ is the human heuristic term that varies by choice. Hence, we opt to replace $\frac{1}{2\pi r} M(\delta)$ by a $\mathbb{R} \to \mathbb{R}$ neural network $N_2(r, \delta)$ so that eventually:

$$\begin{aligned}
\boldsymbol{u}(\boldsymbol{x}) &= \int N_2(|\boldsymbol{x} - \boldsymbol{x}'|, \delta_i) R_{\frac{2}{\pi}}(\boldsymbol{x} - \boldsymbol{x}')\boldsymbol{\omega}(\boldsymbol{x}')d\boldsymbol{x}' && (6)\\
&= \sum_{i=1}^{n} N_2(|\boldsymbol{x} - \boldsymbol{x}_i|, \delta_i) R_{\frac{2}{\pi}}(\boldsymbol{x} - \boldsymbol{x}_i)\boldsymbol{\omega}_i && (7)\\
&= \mathcal{D}\big([(\boldsymbol{x}_1, \omega_1, \delta_1), \ldots, (\boldsymbol{x}_n, \omega_n, \delta_n)]\big) = \mathcal{D}(\mathcal{V}) && (8)
\end{aligned}$$

4.2 End-to-end training

As previously mentioned, the dynamics on the latent vortex space is bridged to the evolution of the image space through the differentiable, dynamic module $\mathcal{D}$. Hence, we can optimize the vortex representation $\mathcal{V}_t = \mathcal{T}(t)$ at time t using images as supervision. First, we select m frames: $[I_t, \ldots, I_{t+m}]$ from the video. Then, we compute $\boldsymbol{u}_t = \mathcal{D}(\mathcal{V}_t)$. After that, $(\boldsymbol{u}_t, I_t)$ is fed into a integrator on the Eulerian grid to predict $\tilde{I}_{t+1}$. Simultaneously, $(\boldsymbol{u}_t, \mathcal{V}_t)$ is fed into an integrator on Lagrangian particles to predict $\tilde{\mathcal{V}}_{t+1}$. The process is then repeated, using $\tilde{I}_{t+1}$ in place of I_t and $\tilde{\mathcal{V}}_{t+1}$ in place of $\mathcal{V}_t$, to generate $\tilde{I}_{t+2}$ and $\tilde{\mathcal{V}}_{t+2}$, and so on. Eventually, we would obtain $[\tilde{I}_{t+1}, \ldots, \tilde{I}_{t+m}]$, which are the predicted future outcome starting at time t. We optimize $\mathcal{T}$ and $\mathcal{D}$ jointly by minimizing its difference between $[\tilde{I}_{t+1}, \ldots, \tilde{I}_{t+m}]$ and $[I_{t+1}, \ldots, I_{t+m}]$ in and end-to-end fashion.

By picking different values of t to cover the entire duration of the input video, and optimizing $\mathcal{V}_t = \mathcal{T}(t)$ in the abovementioned manner, we essentially optimize the vortex trajectories to better explain the observed image data. There remains one more caveat — which is that these trajectories are not enforced to be consistent with the dynamics we uncovered in $\mathcal{D}$, because each frame of $\mathcal{V}_t$ is optimized individually. In other words, if we evaluate the particle velocities $[\dot{\boldsymbol{x}_1}, \ldots, \dot{\boldsymbol{x}_n}] = \frac{d\mathcal{N}_1}{dt}$ as prescribed by $\mathcal{T}$, it should coincide with $[\mathcal{D}(\mathcal{V})(\boldsymbol{x_1}), \ldots, \mathcal{D}(\mathcal{V})(\boldsymbol{x_n})]$, which as prescribed by $\mathcal{D}$. Hence, in training, another loss is computed between $\frac{d\mathcal{T}}{dt}$ and $[\mathcal{D}(\mathcal{V})(\boldsymbol{x_1}), \ldots, \mathcal{D}(\mathcal{V})(\boldsymbol{x_n})]$ to align the vortex trajectory and the predicted velocity.

Deployment. After successful training, the learned system allows us to perform two important tasks. First, using our continuous query function $\mathcal{T}(t)$, we are able to interpolate for $\mathcal{V}(t)$, which then uncovers the hidden velocity field $\boldsymbol{u} = \mathcal{D}(\mathcal{V}_t)$ at arbitrary precision, which provides the same functionality as Raissi & Karniadakis (2018), but using vorticity instead of pressure as the secondary variable. Moreover, with the dynamics module $\mathcal{D}$, we can perform future prediction to unroll the input video, a feature unsupported by previous methods. As shown in Figure 4, since our method is forward-simulating by nature, it can provide more realistic and robust future prediction than existing methods and their extensions.

4.3 Implementation

Integrators. As described above and illustrated in Figure 2, our system embeds two differentiable integrators in the loop. The Eulerian integrator is implemented using the Back and Forth Error

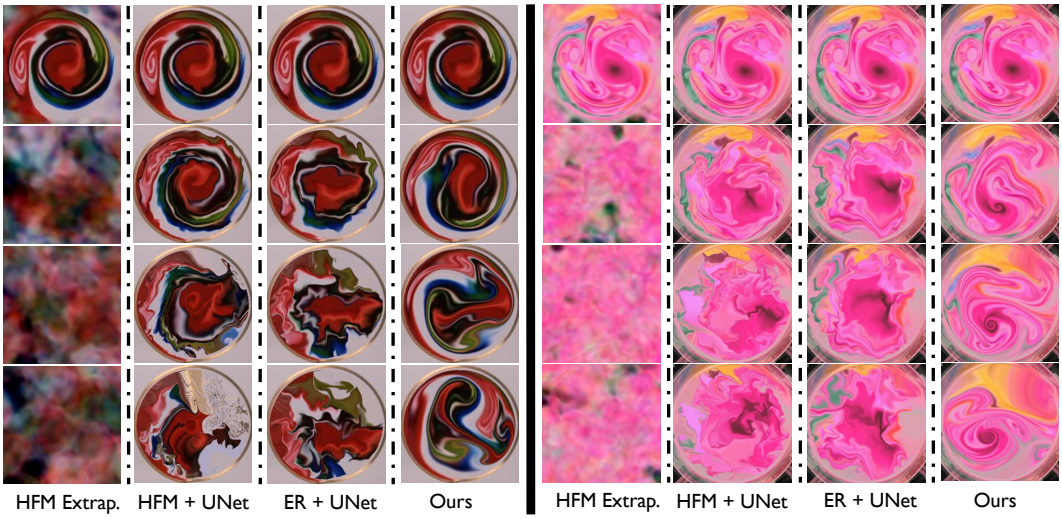

Figure 4: Applied to real-world videos, our Lagrangian based method can create realistic future predictions over long periods of time compared to existing methods (and their extensions).

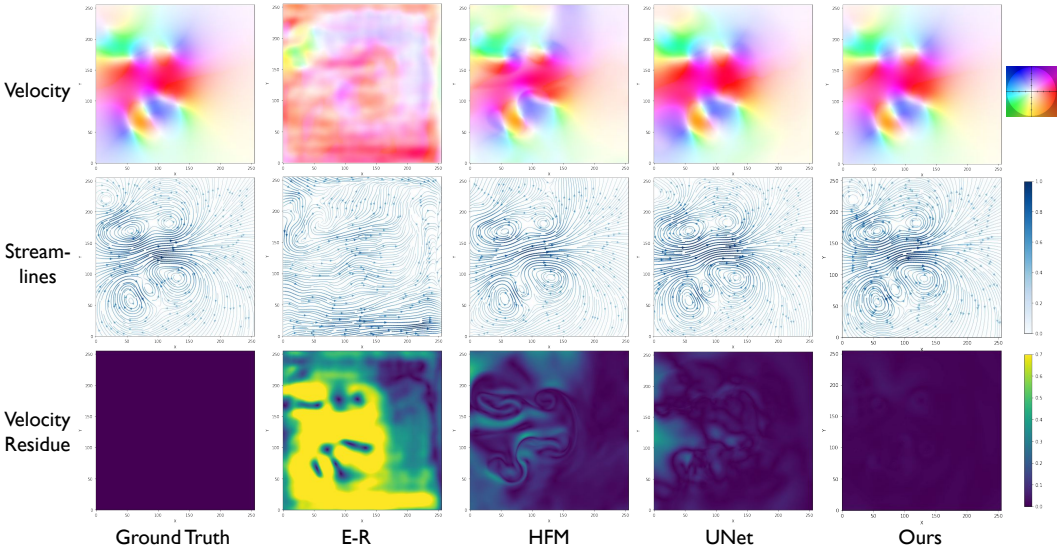

Figure 5: Hidden motion inference compared with existing methods on a synthetic video. Our method uncovers the underlying velocity field with higher accuracy.

Compensation and Correction (BFECC) (Kim et al., 2005) method for backtracking, and the 3rd order Runge-Kutta method for time-stepping. The Lagrangian integrator is implemented using the Forward Euler method.

Learning Details. The network N_1 adopts a series of 3 residue blocks with increasing width $[64, 128, 256]$, whose architecture is similar to He et al. (2016) but with convolution layers replaced by sine layers. The network N_2 shares the same architecture with N_1 but with 4 layers of a increasing width 40. Both the image loss and the alignment loss are MSE, and the velocity loss has an extra scaling factor of 0.001. We use Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and learning rate $= [0.001, 0.0003, 0.005, 0.005]$ for N_1, N_2, Ω and Δ respectively. We schedule for the learning rate to decay to 0.1 of the original value at 20% of the total training iterations.

5 Experiments

We evaluate our method's ability to perform motion inference and future prediction on both synthetic and real videos, comparing against existing methods.

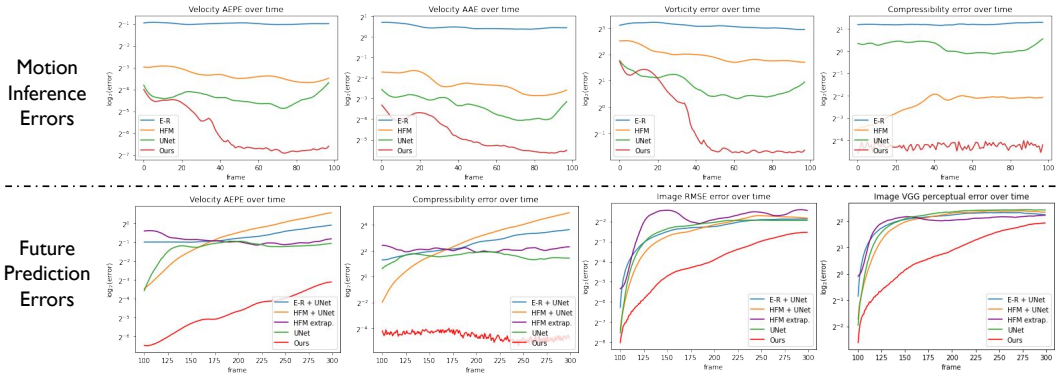

Figure 6: Error analysis on a synthetic dataset. The top row plots the inference errors of velocity, vorticity, and compressibility. The bottom row plots the future prediction errors, which consider both the dynamic error in the velocity and the perceptual error of the generated image sequence.

Baselines. For motion inference, we compare our method against Raissi & Karniadakis (2018) (HFM) and Zhang et al. (2022) (ER). We reimplement the HFM method as prescribed, making only the modification that instead of using only a single concentration variable c and its inverse d as specified by (Raissi & Karniadakis, 2018), we create three (c, d) pairs for each of the RGB channel for the support of colored videos. The E-R method is evaluated using the published pretrained models. We further compare against an ablated version of our proposed method, termed "UNet", which essentially replaces the Lagrangian components of the system with a UNet architecture (Ronneberger et al., 2015), a classic method for conducting field-to-field mapping. The UNet baseline takes two images I_t and I_{t+1} and predicts a velocity field $\boldsymbol{u}_{t+1}$ to predict I_{t+2} using the same Eulerian integrator as our method. For future prediction, there do not exist previous methods that perform in the same setting, so we extend the inference methods in a few ways to support future prediction in a logical and straightforward manner. First, since HFM offers a query function parameterized by t, we test its future prediction behavior by simply extrapolating with $t > t_E$; this is referred to as "HFM extp.". Since both Raissi & Karniadakis (2018) and Zhang et al. (2022) uncovers the time-varying velocity field, we use a UNet to learn the evolution from $\boldsymbol{u}_t$ to $\boldsymbol{u}_{t+1}$, and use this velocity update mechanism to perform future prediction, the two baselines thus obtained are referred to as "HFM+UNet" and "ER+UNet" respectively. The ablated version "UNet" does support future prediction intrinsically.

5.1 Synthetic Video

The synthetic video for vortical flow is generated using the Discrete Vortex Method with a first-order Gaussian mollifying kernel $M(\delta)$. The high-fidelity BFECC advection scheme with Runge-Kutta-3 time integration is deployed. The simulation advects a background grid of 256×256, with a time step $dt = 0.01$ to create 300 simulation videos. Only the first 100 frames will be seen to train all methods, and future predictions are tested and examined on the latter 200 frames.

Motion Inference. The results for the uncovering of hidden dynamic variables are illustrated in Figure 5 and Figure 6. Shown in Figure 5 are the velocities uncovered by all 4 methods against the ground truth, at frame 55 of a synthetic video with 100 frames. The velocity is visualized in the form of colors (top row) as well as streamlines (middle row), while the velocity residue, measured in end-point error(EPE), is depicted in the bottom row. It can be seen that HFM, UNet, and our method achieve agreeing results, and matches the ground truth values to high accuracy. On the bottom row, it can be seen that while both HFM and UNet provide sensible results, our method generates the inference velocity that best matches the unseen ground truth.

The inference results over the full 100 frames at the top of Figure 6. We evaluate the velocity with four metrics: the average end-point error (AEPE), average angular error (AAE), vorticity RMSE and compressibility RMSE. From all 4 metrics, it can be seen that our method outperforms the baselines consistently. The time-averaged data for all four metrics are shown on the left of Table 1, which deems our method favorable for all metrics used.

Future Prediction. In Figure 7, we visually compare the future prediction results from frame 100 to frame 299 done using our method and the 4 benchmarks, against the ground truth. It can be observed

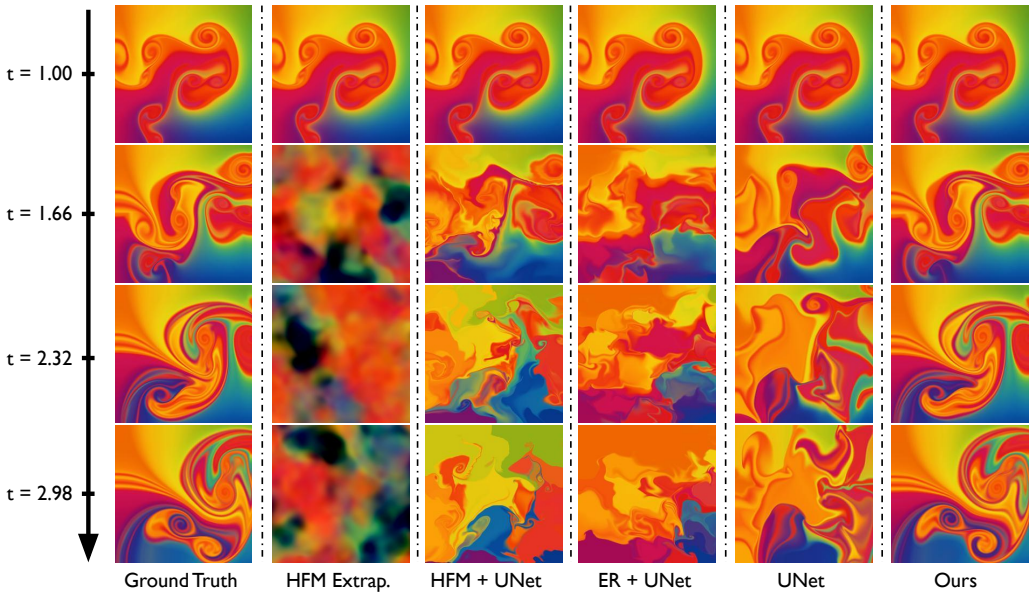

Figure 7: Future predicting capacities of our method compared to benchmarks. Our method accurately predicts the unseen, future sequence that's twice as long as the seen sequence.

Time-averaged Inference Errors					Time-averaged Prediction Errors						
	AEPE	AAE	Vort.	Div.		VGG	RMSE	AEPE	AAE	Vort.	Div.
E-R	0.505	1.393	8.470	2.319	+UNet	4.346	0.205	0.631	1.424	12.84	6.580
HFM	0.100	0.212	3.949	0.202	+UNet	4.258	0.205	0.720	1.062	36.73	10.41
					Extp.	4.080	0.285	0.541	1.464	7.761	4.315
UNet	0.048	0.100	1.799	1.145		4.530	0.211	0.424	1.159	7.334	3.017
Ours	**0.020**	**0.041**	**0.976**	**0.053**		**2.010**	**0.080**	**0.048**	**0.096**	**1.621**	**0.043**

Table 1: Error analysis of benchmark testing on a synthetic dataset.

that the sequence generated by our method best matches with the ground truth video, capturing the vortical flow structures, while the other baselines either quickly diffuse or generates unnatural, hard-edged patterns. Numerical analysis confirms these visual observations. We compare the unrolled 200 frames both in terms of velocity and visual similarity. The velocity analysis inherits the same 4 metrics, and the visual similarity is simultaneously gauged using the pixel-level RMSE and the VGG perceptual loss (Johnson et al.). The time-averaged results of all 6 metrics are documented in the right of Table 1, and 4 are plotted in Figure 6. It can be concluded that our method outperforms the baselines.

5.2 Real Video

A similar numerical analysis is carried out on a real video published on YouTube, as shown in Figure 8. The video has 150 frames: the first 100 frames will be used for training, while the latter 50 will be used for testing. Since the ground truth velocities for the real video are nonexistent, we will only analyze the future predicting performance. For all methods, we perform future prediction for 150 frames, among these, the first 50 frames can be compared with the testing videos, and the latter 100 frames will be compared qualitatively. We note that, since only part of the video is fluid (within the circular rim), we pre-generate a signed distance field for all methods, so that only the fluid regions are considered, and the same no-slip boundary condition will be placed for all unroll methods (except for HFM extp. which requires no advection).

The numerical analysis for the first 50 predicted frames are documented and plotted in Table 2 and Figure 9. We compare all methods against the baseline using the VGG perceptual loss for visual loss, and compare the velocity divergence, which should be close to the theoretical value of 0. It

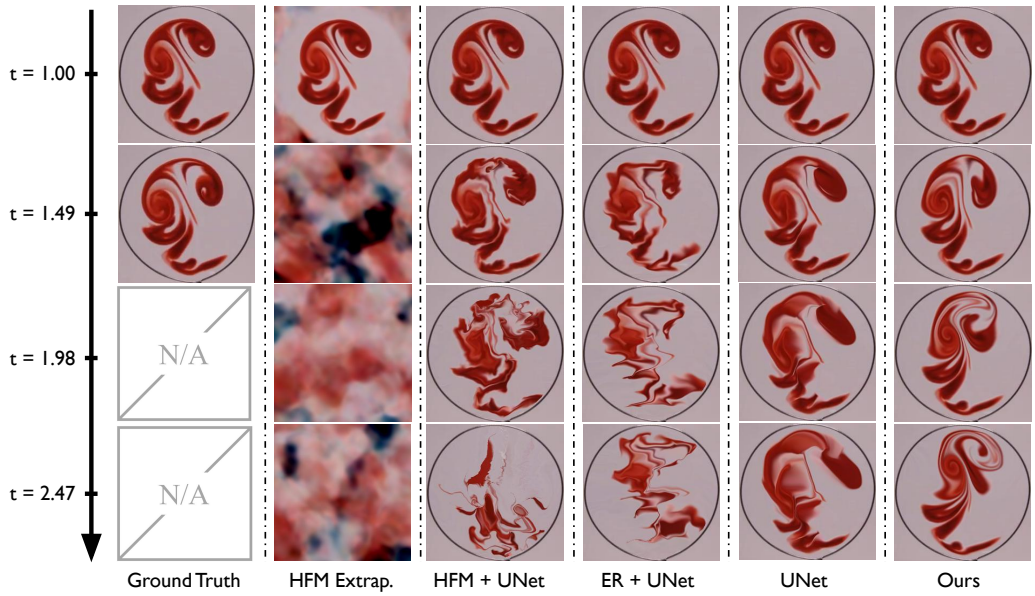

Figure 8: Future predicting capacities of our method compared to benchmarks on a real video sequence. Our method generates a predicted sequence that best matches the input video within its duration and remains visually plausible way beyond its duration.

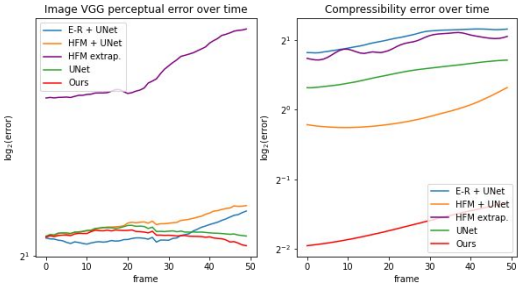

Figure 9: Plots corresponding to Figure 8.

	VGG (avg.)	VGG (final)	Div. (avg)
E-R	2.095	2.205	2.046
HFM+UNet	2.151	2.231	0.940
HFM Extp.	2.980	3.271	1.922
UNet	2.111	2.088	1.447
Ours	**2.093**	**2.045**	**0.318**

Table 2: Data corresponding to Figure 8.

can be seen that in all 3 metrics our method prevails. For prediction results that surpass the duration of the real video, qualitative observations can be made in that our method preserves the vortical structure, generating smooth visualizations over the entire horizon, while other methods end up yielding glitchy patterns.

6 Conclusion & Limitations

In this work, we propose a novel data-driven system to perform fluid hidden dynamics inference and future prediction from single RGB videos, leveraging a novel, vortex latent space. The success of our method in synthetic and real data, both qualitatively and quantitatively, suggests the potential for embedding Lagrangian structures for fluid learning. Our method has several limitations. First, the vortex model is limited to 2D, due to its lack of vortex stretching treatment for 3D vortical flow. Second, our vortex evolution did not take into account the boundary conditions in a physically-based manner, hence it cannot accurately predict flow details around a solid boundary. Third, scaling our method to handle turbulence with multi-scale vortices remains to be explored. We consider two directions for future work. First, we plan to explore the numerical accuracy of our neural vortex representation to improve the current vortex particle methods for scientific computing. Second, we plan to combine our differentiable simulator with neural rendering methods to produce visually appealing simulations from 3D videos.

REPRODUCIBILITY STATEMENT

To ensure reproducibility of our work, we will release the training and testing code, as well as the data to reproduce our results upon publication.

REFERENCES

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29, 2016.

J Thomas Beale and Andrew Majda. High order accurate vortex methods with explicit velocity kernels. *Journal of Computational Physics*, 58(2):188–208, 1985.

Josh Bongard and Hod Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.

Tyson Brochu, Todd Keeler, and Robert Bridson. Linear-time smoke animation with vortex sheet meshes. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 87–95. Citeseer, 2012.

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

Michael B Chang, Tomer Ullman, Antonio Torralba, and Joshua B Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.

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

Mengyu Chu, Lingjie Liu, Quan Zheng, Erik Franz, Hans-Peter Seidel, Christian Theobalt, and Rhaleb Zayer. Physics informed neural fields for smoke reconstruction with sparse data. *ACM Transactions on Graphics (TOG)*, 41(4):1–14, 2022.

Georges-Henri Cottet, Petros D Koumoutsakos, et al. *Vortex methods: theory and practice*, volume 8. Cambridge university press Cambridge, 2000.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, 426:109951, 2021.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. URL `https://arxiv.org/abs/1603.08155`.

ByungMoon Kim, Yingjie Liu, Ignacio Llamas, and Jaroslaw R Rossignac. Flowfixer: Using bfecc for fluid simulation. Technical report, Georgia Institute of Technology, 2005.

Byungsoo Kim, Vinicius C Azevedo, Markus Gross, and Barbara Solenthaler. Lagrangian neural style transfer for fluids. *ACM Transactions on Graphics (TOG)*, 39(4):52–1, 2020.

J Nathan Kutz, Steven L Brunton, Bingni W Brunton, and Joshua L Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

L'ubor Ladickỳ, SoHyeon Jeong, Barbara Solenthaler, Marc Pollefeys, and Markus Gross. Data-driven fluid simulations using regression forests. *ACM Transactions on Graphics (TOG)*, 34(6): 1–9, 2015.

Anthony Leonard. Vortex methods for flow simulation. *Journal of Computational Physics*, 37(3): 289–335, 1980.

Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018.

Chloé Mimeau and Iraj Mortazavi. A review of vortex methods and their applications: From creation to recent advances. *Fluids*, 6(2):68, 2021.

Sang Il Park and Myoung Jun Kim. Vortex fluid for gaseous phenomena. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 261–270, 2005.

Mirta Perlman. On the accuracy of vortex methods. *Journal of Computational Physics*, 59(2):200–223, 1985. ISSN 0021-9991. doi: https://doi.org/10.1016/0021-9991(85)90142-1. URL https://www.sciencedirect.com/science/article/pii/0021999185901421.

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

Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations using gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

Maziar Raissi, Paris Perdikaris, and George 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.

Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. URL https://arxiv.org/abs/1505.04597.

Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science advances*, 3(4):e1602614, 2017.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning*, pp. 4470–4479. PMLR, 2018.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pp. 8459–8468. PMLR, 2020.

Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

Andrew Selle, Nick Rasmussen, and Ronald Fedkiw. A vortex particle method for smoke, water and explosions. In *ACM SIGGRAPH 2005 Papers*, pp. 910–914. 2005.

Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin. Accelerating eulerian fluid simulation with convolutional networks. In *International Conference on Machine Learning*, pp. 3424–3433. PMLR, 2017.

Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. In *International Conference on Learning Representations*, 2019.

Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. *Advances in neural information processing systems*, 30, 2017.

Steffen Weißmann and Ulrich Pinkall. Filament-based smoke with vortex shedding and variational reconnection. In *ACM SIGGRAPH 2010 papers*, pp. 1–12. 2010.

Steffen Wiewel, Moritz Becher, and Nils Thuerey. Latent space physics: Towards learning the temporal evolution of fluid flow. In *Computer graphics forum*, volume 38, pp. 71–82. Wiley Online Library, 2019.

G.S. Winckelmans and A. Leonard. Contributions to vortex particle methods for the computation of three-dimensional incompressible unsteady flows. *Journal of Computational Physics*, 109(2): 247–273, 1993. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.1993.1216. URL `https://www.sciencedirect.com/science/article/pii/S0021999183712167`.

You Xie, Erik Franz, Mengyu Chu, and Nils Thuerey. tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Transactions on Graphics (TOG)*, 37(4):1–15, 2018.

Shiying Xiong, Xingzhe He, Yunjin Tong, and Bo Zhu. Neural vortex method: from finite lagrangian particles to infinite dimensional eulerian dynamics. *arXiv preprint arXiv:2006.04178*, 2020.

Liu Yang, Dongkun Zhang, and George Em Karniadakis. Physics-informed generative adversarial networks for stochastic differential equations. *SIAM Journal on Scientific Computing*, 42(1): A292–A317, 2020.

Mingrui Zhang, Jianhong Wang, James Tlhomole, and Matthew D Piggott. Learning to estimate and refine fluid motion with physical dynamics. *arXiv preprint arXiv:2206.10480*, 2022.