Learning to Estimate and Refine Fluid Motion with Physical Dynamics

Mingrui Zhang [1] **Jianhong Wang** [2] **James Tlhomole** [1] **Matthew D. Piggott** [1]

Abstract

Extracting information on fluid motion directly from images is challenging. Fluid flow represents a complex dynamic system governed by the Navier-Stokes equations. General optical flow methods are typically designed for rigid body motion, and thus struggle if applied to fluid motion estimation directly. Further, optical flow methods only focus on two consecutive frames without utilising historical temporal information, while the fluid motion (velocity field) can be considered a continuous trajectory constrained by time-dependent partial differential equations (PDEs). This discrepancy has the potential to induce physically inconsistent estimations. Here we propose an unsupervised learning based prediction-correction scheme for fluid flow estimation. An estimate is first given by a PDE-constrained optical flow predictor, which is then refined by a physical based corrector. The proposed approach outperforms optical flow methods and shows competitive results compared to existing supervised learning based methods on a benchmark dataset. Furthermore, the proposed approach can generalize to complex real-world fluid scenarios where ground truth information is effectively unknowable. Finally, experiments demonstrate that the physical corrector can refine flow estimates by mimicking the operator splitting method commonly utilised in fluid dynamical simulation.

1. Introduction

Fluid flow motion estimation is a topic of interest for many science and engineering fields, including geophysics, oceanology, biology, and environmental engineering. Measuring fluid motion and understanding the underlying dynamics are crucial for exploring complex fluid phenomena in these fields.

One natural way to understand and analyze fluid motion is via visual observation. However, there are generally no strong visible patterns in transparent fluid flow such as water and air. Therefore, visual markers of some description are commonly introduced to allow the optical measurement of the motion. One effective method is to inject these markers into the fluid and record their motion with one or multiple high-speed cameras. By comparing the resulting flow images at different time levels, velocity field information can be extracted. Based on this idea, one of most prominent techniques for fluid motion estimation in experimental fluid mechanics is Particle Image Velocimetry (PIV) (Adrian & Westerweed., 2011). Traditionally, PIV can be regarded as an optical flow estimation problem (Ruhnau et al., 2005) and tackled using variational optical flow methods (D. Heitz, 2010).

With the success of deep learning in optical flow estimation, these methods have also been adopted to solve the corresponding PIV problem. However, pure dense optical flow methods assume brightness constancy and flow smoothness, while the visible tracers in the fluid are driven by fluid dynamics. The missing dynamical information in the estimation model may induce physically implausible results and temporal inconsistency, which would typically be crucial for useful flow diagnosis in rigorous science and engineering applications. In this work we seek to bridge the gap between optical estimation and the governing fluids-based physical models for motion estimation problem. Specifically, a novel unsupervised learning framework is proposed in this paper. The framework is designed as a prediction-correction scheme, which consists of an optical flow based fluid motion predictor and a physical corrector.

The estimation process is inspired by Chorin's projection method, an operator splitting method (Chorin, 1968; Strang., 1968), often used in numerical fluid simulation. The operator splitting method separates the original PDE system into two parts over a time step, separately computes the solution to each part, and then combines the two separate solutions to form a solution to the original system. However, these kinds of methods are limited to physical models, and thus cannot incorporate optical fluid observations, and this can-

[1]Department of Earth Science and Engineering, Imperial College London, UK [2]Department of Electrical and Electronic Engineering, Imperial College London, UK. Correspondence to: Mingrui Zhang <mingrui.zhang18@imperial.ac.uk>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

not be used for fluid motion estimation problem directly. Therefore, here we consider an approach based upon a generalization of the projection method to a generic operator splitting scheme, which incorporates both physical knowledge and fluid observations. In this scheme, the motion predictor first outputs an estimated flow field. This is then refined using a physical based corrector.

For the motion predictor, we note that the Euler-Lagrange equation corresponding to the unsupervised learning formulation is related to the Stokes equation in fluid dynamics. The Stokes equation is a simplified version of the full Navier-Stokes equation, which indicates that the estimation result can in some sense be considered a component of the full equation; thus motivating its inclusion in an operator splitting like scheme. For the physical corrector, the approach taken uses both the velocity field from the previous time level and the current estimate as inputs and is designed to enforce physical consistency and the divergence-free constraint in one shot. We test our resulting estimations methods on both synthetic and real world datasets. The experiments indicate that the proposed unsupervised method can output competitive results on synthetic dataset compared to a supervised method. For the real world dataset without ground truth, the method can achieve reasonable estimation results as noted through benchmarking against those obtained using state-of-the-art open-source fluid motion estimation software. Code is available at: `https://github.com/erizmr/Learn-to-Estimate-Fluid-Motion`.

2. Background

2.1. Fluid Flow

Fluid dynamics is typically modelled using the Navier-Stokes (N-S) equations. There are different variations of the N-S equations depending on the nature of the fluid or its flow. Here we briefly introduce the system that we consider in this work.

Incompressible Fluid Flow. In this work, we only consider incompressible flow, which can be modelled using the Navier-Stokes momentum equation

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{f}, \tag{1}$$

where $\mathbf{u}$ is the velocity, p denotes the pressure field (scalar field), ρ is the fluid density (assumed to be constant), ν is the kinematic viscosity (also assumed to be constant) and $\mathbf{f}$ is the summation of any external forces applied to the fluid body. To represent incompressibility, a divergence-free constraint on the velocity vector field: $\nabla \cdot \mathbf{u} = 0$, should be satisfied.

Transport Equation and Warping. A scalar field I transported in and by the fluid can be described by the advection-diffusion equation

$$\frac{\partial I}{\partial t} + \nabla \cdot (I\mathbf{u}) = D\nabla^2 I, \tag{2}$$

where D is the diffusion coefficient. Given a scalar field I transported in incompressible flow, and assuming it is conserved in the region of interest, i.e., no source terms appear, and that the diffusion coefficient $D = 0$, then with the divergence-free condition, Equation (2) can be simplified to

$$\frac{\partial I}{\partial t} + \mathbf{u} \cdot \nabla I = 0. \tag{3}$$

Note that Equation (3) is consistent with the brightness constancy assumption in Horn and Schunck's optical flow approach (Horn & Schunck., 1981). Therefore, optical flow can be regarded as a special case of fluid flow, i.e., visible markers move in a purely advective manner in fluid flow.

For Equation (2), it can be shown (de Bezenac et al., 2018) that given any initial condition I_0, there exists a unique solution $I(\mathbf{x}, t)$ which can be computed via a convolution between a Gaussian kernel and the initial condition I_0 (shown in Appendix A). Therefore, we can obtain the warping scheme below by discretizing the solution. Using a previous time level scalar field I_t as the initial condition, we can compute the image at the next time level as

$$\hat{I}_{t+1}(\mathbf{x}) = \sum_{\mathbf{y}\in\Omega} k(\mathbf{x} - \mathbf{u}, \mathbf{y}) I_t(\mathbf{y}), \tag{4}$$

where $k(\mathbf{x}-\mathbf{u}, \mathbf{y}) = \frac{1}{4\pi D\delta t} e^{-\frac{1}{4D\delta t}\|\mathbf{x}-\mathbf{u}-\mathbf{y}\|^2}$, δt is the time interval between time levels t and $t+1$. Equation (4) shares similar ideas with the Spatial Transformer Network (Jaderberg et al., 2015), where the $k(\cdot)$ is a sampling kernel. By using this warping scheme, we can approximate the solution of scalar fields such as the vorticity and the concentration of the marker in the fluid, which will be utilised in training the corrector and predictor in this work.

3. Learning to Estimate and Refine Fluid Motion

Problem Statement. Given a sequence of consecutive fluid observation images $\mathbf{I} = (I_1, I_2, I_3, ..., I_T) \in \mathbb{R}^{T\times c\times h\times w}$ as input, where T, c, h, w are number of total time steps, number of image channels, height and width of images. Our goal is to estimate the dense forward flow field (displacement field) for each pair of images, from $\{I_1, I_2\}$ to $\{I_{T-1}, I_T\}$, which is denoted as $\mathbf{u} = (\mathbf{u}_1, ..., \mathbf{u}_{T-1}) \in \mathbb{R}^{(T-1)\times 2\times h\times w}$. In addition, the fluid flow field is assumed to be governed by the incompressible Navier-Stokes Equation (1).

We propose an unsupervised learning based prediction-correction scheme for the fluid motion estimation problem. Given a time step t, an optical flow based predictor $\mathcal{P}$ provides an estimated flow field $\hat{\mathbf{u}}_t$. Then a physical corrector

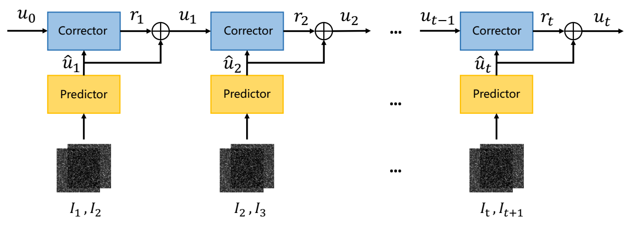

Figure 1. The pipeline of the fluid motion estimation via a prediction-correction based scheme. Given an image pair I_t, I_{t-1} at the current time level t, a predictor first outputs an estimated velocity field $\hat{\mathbf{u}}_t$. Then a corrector computes a refinement $\mathbf{r}_t$ by taking both the velocity at the previous time level, i.e. $\mathbf{u}_{t-1}$ and the current estimate $\hat{\mathbf{u}}_t$ as inputs. By adding the current estimate and the refinement, the corrected $\mathbf{u}_t$ is computed.

takes both $\hat{\mathbf{u}}_t$ and $\mathbf{u}_{t-1}$ (the flow field from the previous time level) as inputs, and outputs a refinement in the form of a corrected flow field $\mathbf{u}_t$:

$$\mathbf{u}_t = \mathcal{P}(I_{t-1}, I_t) + \mathcal{C}(\mathbf{u}_{t-1}, \hat{\mathbf{u}}_t). \tag{5}$$

3.1. Fluid Motion Predictor

The fluid motion estimator is designed based on a variational optical flow approach constrained by the Stokes equation. The unsupervised learning formulation is given by

$$\min_{\mathbf{u}} \int_\Omega \underbrace{\left(\frac{\partial I}{\partial t} + \mathbf{u} \cdot \nabla I\right)}_{Data\ term} + \underbrace{\mu|\nabla \mathbf{u}|^2}_{Smoothness} + \underbrace{p\nabla \cdot \mathbf{u}}_{Divergence} \; d\mathbf{x}. \tag{6}$$

The formula consists of three parts: the data term, the smoothness and the divergence regularizers. The data term is modelled by the brightness constancy assumption (3). For the smoothness and divergence regularizer, these can be identified with the diffusion (or viscous) and pressure gradient terms in the fluid dynamical equations as shown below.

It can be shown (in Appendix B) that the Euler-Lagrange equation of formulation (6) is:

$$-\mu\nabla^2\mathbf{u} + \nabla p = \nabla I, \tag{7}$$

which has the same form as the Stokes equation in fluid dynamics. It is is an approximation/simplification to the Navier-Stokes momentum equation obtained by omitting the inertial component. The ∇I can be interpreted as an external forcing term, which can be determined by optical observations. The forward process of a neural network model has a potential mathematical equivalence to the temporal evolution of a dynamic system (Weinan., 2017). Thus, the training process of the predictor can be regarded as finding the optimal control forces applied on the fluid dynamical system so as to minimize the energy (6). It also suggests that the inference process is solving for a velocity field that is related to the Stokes equation given the optical observations.

3.2. Physical Corrector

Operator Splitting Scheme. The idea of the corrector is motivated by the operator splitting approach common in the numerical solution of PDEs. The approach separates the original equation into two or more parts and computes the solution to each part separately. The separate solutions are then combined to form a solution to the original equation. In incompressible fluid simulation, the flow is often solved using a operator splitting based approach that is often referred to as Chorin's projection method (Chorin, 1968). The idea is first to compute a tentative velocity $\mathbf{u}_t^*$ by neglecting the pressure in the Navier-Stokes momentum equation and then to project the velocity onto the space of divergence free vector fields.

However, Chorin's projection method does not incorporate fluid observations into the scheme, and thus can not be adopted to fluid motion estimation problems directly. In addition, it often involves the solution of a Poisson equation in order to enforce the divergence-free condition, which is expensive compared to conducting neural network based inference. In this work, motivated by Chorin's projection method (a detailed introduction to which is given in Appendix A), we generalise to a generic operator splitting scheme of the form

$$\frac{\mathbf{u}_t^* - \mathbf{u}_{t-1}}{\Delta t} = -\mathbf{u}_{t-1} \cdot \nabla \mathbf{u}_{t-1} + \nu\nabla^2\mathbf{u}_{t-1}, \tag{8}$$

$$\frac{\mathbf{u}_t - \mathbf{u}_t^*}{\Delta t} = -\frac{1}{\rho}\nabla p_t. \tag{9}$$

By adding up the two splitting parts i.e., the Equation (8) and Equation (9), we can recover the incompressible Navier-Stokes equation with temporal discretization

$$\frac{\mathbf{u}_t - \mathbf{u}_{t-1}}{\Delta t} = -\frac{1}{\rho}\nabla p_t + \mathcal{R}_1. \tag{10}$$

For clarity, we use $\mathcal{R}_1$ to denote the advection and viscous terms, i.e., $\mathcal{R}_1 = -\mathbf{u}_{t-1} \cdot \nabla \mathbf{u}_{t-1} + \nu\nabla^2\mathbf{u}_{t-1}$. Since solving for pressure p_t is expensive, we turn to compute the pressure gradient term $\frac{1}{\rho}\nabla p_t$ using the left hand side of Equation (9). Although $\mathbf{u}_t$ is an unknown, we can approximate it using the tentative velocity $\mathbf{u}_t^*$ and velocity estimation $\hat{\mathbf{u}}_t$ from the predictor:

$$\tilde{\mathbf{u}}_t = \mathbf{K}_t \odot \hat{\mathbf{u}}_t + (1 - \mathbf{K}_t) \odot \mathbf{u}_t^*, \tag{11}$$

where $\tilde{\mathbf{u}}_t$ is the approximation to $\mathbf{u}_t$, a weighted average of the estimated and the tentative velocity. $\mathbf{K}_t$ here is a factor that controls the trade-off between the optical estimate and the physical velocity based on an advection-diffusion step. This control factor can be interpreted as the Kalman gain: if $\mathbf{K}_t = 1$, the output only relies on the estimation; while if $\mathbf{K}_t = 0$, the current velocity is computed fully by the physical advection-diffusion step. This can be modelled in a similar manner to the gating mechanisms in recurrent neural networks. $\mathbf{K}_t = \sigma(\mathbf{W}_e * \hat{\mathbf{u}}_t + \mathbf{W}_p * \mathbf{u}_t^* + \mathbf{b}_k)$, where $\mathbf{W}_e$ and $\mathbf{W}_p$ are convolutions and $\mathbf{b}_k$ is the bias, and σ is the sigmoid function.

Substituting Equation (9) and Equation (11) into Equation (10) yields

$$\frac{\mathbf{u}_t - \mathbf{u}_{t-1}}{\Delta t} = \frac{\tilde{\mathbf{u}}_t - \mathbf{u}_t^*}{\Delta t} + \mathcal{R}_1 + \mathcal{R}_2, \tag{12}$$

$$\frac{\mathbf{u}_t - \tilde{\mathbf{u}}_t}{\Delta t} = \mathcal{R}_2. \tag{13}$$

$\mathcal{R}_2$ is used to model the physical residual induced by the predictor and the neglection of the pressure gradient term. In other words, the proposed scheme aims to "project" the errors in the predictor and the velocity field's divergence in one shot.

Dynamics Model. By reformulating Equation (13) as $\mathbf{u}_t = \tilde{\mathbf{u}}_t + \Delta t \mathcal{R}_2$, we can compute the corrected velocity $\mathbf{u}_t$. Note that $\tilde{\mathbf{u}}_t$ can be computed from $\mathbf{u}_t^*$ and $\hat{\mathbf{u}}_t$, and the only unknown is then $\mathcal{R}_2$. $\mathcal{R}_2$ is designed to compensate the missing dynamics induced by the predictor and in neglecting the pressure gradient term. It is assumed that the missing dynamics can be modelled by PDEs and learned from data (Long et al., 2018). We consider an expression of the form

$$\mathbf{\Gamma}(\mathbf{\Psi}(\mathbf{x}, t)) = \sum_{i,j:i+j<q} c_{i,j} \frac{\partial^{i+j} \mathbf{\Psi}}{\partial x^i \partial y^j}(\mathbf{x}, t), \tag{14}$$

which combines spatial derivatives with coefficients $c_{i,j}$ up to a certain differential order q. This is a generic linear combination of partial derivatives, which can be used as a basis to model a wide range of classical physical models, such as the advection-diffusion equation. Thus, $\mathcal{R}_2$ can be modelled using Equation (14), wherein $\mathbf{\Psi}(\mathbf{x}, t) \triangleq \hat{\mathbf{u}}_t(\mathbf{x}, t) - \mathbf{u}_{t-1}$. Accordingly, we redefine the function $\mathbf{\Phi}(\mathbf{u}_{t-1}, \hat{\mathbf{u}}_t) \triangleq \mathbf{\Gamma}\left(\hat{\mathbf{u}}_t(\mathbf{x}, t) - \mathbf{u}_{t-1}\right)$ for conciseness.

Prediction-Correction Scheme. To conclude, we can derive the prediction-correction scheme as:

$$\mathbf{u}_t = \underbrace{\mathbf{K}_t \odot \hat{\mathbf{u}}_t}_{Predictor} + \underbrace{(1 - \mathbf{K}_t) \odot \mathbf{u}_t^* + \mathbf{\Phi}(\mathbf{u}_{t-1}, \hat{\mathbf{u}}_t)}_{Corrector}, \tag{15}$$

where the predictor and corrector, denoted $\mathcal{P}(I_{t-1}, I_t)$ and $\mathcal{C}(\mathbf{u}_{t-1}, \hat{\mathbf{u}}_t)$ respectively, are as mentioned in Equation (5). For the implementation we reformulate Equation (15) as

$$\mathbf{u}_t = \mathbf{u}_t^* + \mathbf{K}_t \odot (\hat{\mathbf{u}}_t - \mathbf{u}_t^*) + \mathbf{\Phi}(\mathbf{u}_{t-1}, \hat{\mathbf{u}}_t). \tag{16}$$

4. Implementation

An overview of the predictor and corrector implementation is shown in Figure 2.

4.1. Predictor

We use PWC-Net (Sun et al., 2018) and LiteFlowNet (Hui et al., 2018) as the backbone of the predictor. These two networks have an encoder-decoder like architecture and have been demonstrated to be successful for optical flow estimation problems. We re-train the network on synthetic fluid observation images in an unsupervised way. Due to the lack of annotated data for unsupervised learning, we take the bidirectional (Meister et al., 2018) estimate into consideration to enrich the information used in the training loss. Therefore, two flow fields (the forward and backward flow) are defined respectively for each image pair as $\mathbf{u}^f \equiv (u^f, v^f)^T$ and $\mathbf{u}^b \equiv (u^b, v^b)^T$.

Training Loss. As described in Equation (6), the training loss of the predictor consists of three parts. The total loss is the weighted summation such that

$$L_P = L_d + \lambda_s L_s + \lambda_d L_{div}, \tag{17}$$

where L_d denotes the data term, which is modelled by photometric loss; L_s and L_{div} are the spatial smoothness and divergence-free regularizers. λ_s and λ_d are the weights of the two regularizers respectively.

Data Term. The data term is expressed in terms of the difference between warped and original input images, i.e., photometric loss. The bidirectional photometric loss is thus defined as the sum of these two parts:

$$\begin{aligned} L_d(I_1, I_2, \mathbf{u}^f, \mathbf{u}^b) = \sum_{\mathbf{x} \in P} \sigma\left(I_1(\mathbf{x}) - \hat{I}_1(\mathbf{x})\right) \\ + \sigma\left(I_2(\mathbf{x}) - \hat{I}_2(\mathbf{x})\right), \end{aligned} \tag{18}$$

where $\hat{I}_1(\mathbf{x}) = I_2(\mathbf{x} + \mathbf{u}^f(\mathbf{x}))$, $\hat{I}_2(\mathbf{x}) = I_1(\mathbf{x} + \mathbf{u}^b(\mathbf{x}))$ are the warped image for I_1 and I_2 respectively. $\sigma(\cdot)$ is the generalized Charbonnier penalty function, $\sigma = (x^2 + \epsilon^2)^\gamma$, which is a differentiable, robust convex function (Sun et al., 2014). We use the empirical values $\gamma = 0.45, \epsilon = 10^{-3}$ in this work.

Regularizers. The form of the smoothness and divergence-free regularizer is

$$\begin{aligned} L_s(\mathbf{u}^f, \mathbf{u}^b) + L_{div}(\mathbf{u}^f, \mathbf{u}^b) = \sigma(\nabla \mathbf{u}^f) + \sigma(\nabla \mathbf{u}^b) + \\ \sigma(\nabla \cdot \mathbf{u}^f) + \sigma(\nabla \cdot \mathbf{u}^b). \end{aligned} \tag{19}$$

The gradients and divergence of the velocity fields are approximated using a finite difference approach and computed by the convolution operator with appropriate filters.

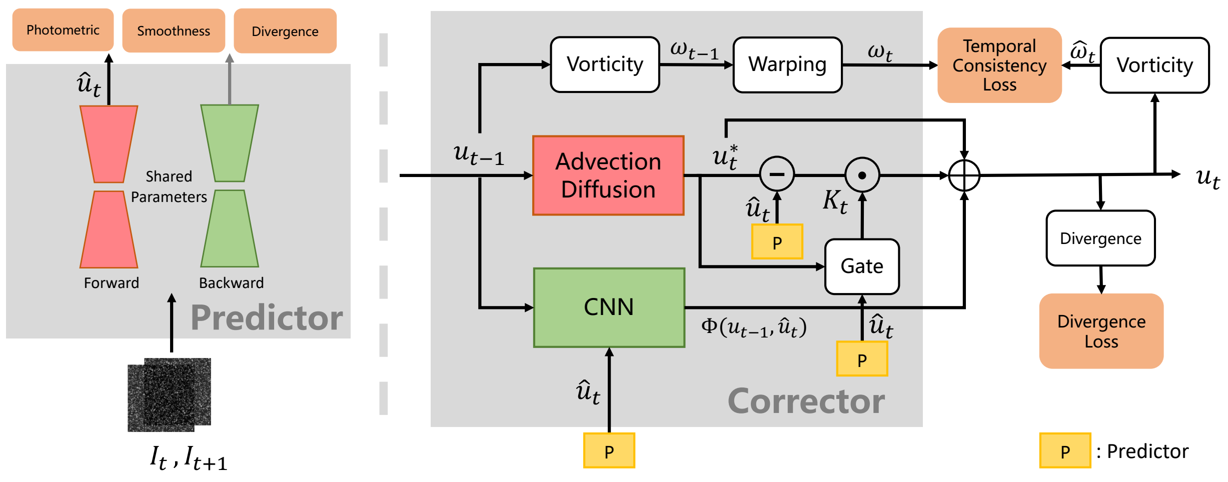

Figure 2. The implementation of the predictor and the corrector described by Equation (16). For the predictor, the red and green blocks represent the network backbone, which is used to perform both forward and backward inference. For the corrector, the red block represents the fluid dynamics described by $\mathcal{R}_1$, which advects and diffuses the previous velocity to the current time level. The green block represents convolution layers which are used to compensate for missing dynamics described by Equation (14). The gate models the control factor $\mathbf{K}_t$. The difference between the warped vorticity and the computed vorticity using the current velocity attempts to enforce temporal consistency. In addition, the divergence is also penalized during training.

4.2. Corrector

Temporal Loss. The training loss of the corrector is composed of two parts – the temporal loss and the divergence loss:

$$L_C = \underbrace{(\hat{\omega}_t - \omega_t)}_{Temporal} + \lambda_d L_{div}, \tag{20}$$

where $\omega = \nabla \times \mathbf{u}$ denotes the flow vorticity, which describes the tendency for fluid parcels to spin in the flow. The temporal loss is modelled by the difference between the warped vorticity field $\hat{\omega}_t$ from the last time level and the vorticity field ω_t computed using the corrected velocity field. As mentioned in section 2.1, warping the vorticity using the scheme (4) can be regarded as solving an advection-diffusion equation, such as the vorticity transport equation (shown in Appendix A). Therefore, reducing the temporal loss tries to enforce temporal consistency between current and previous steps.

5. Experiments

5.1. Datasets

There are two kinds of dataset adopted in this work: synthetic and real world based. The PIV dataset is a synthetic dataset collected by (Cai et al., 2019b). The dataset contains 15,050 particle image pairs with the originating flow field ground truth data obtained from computational fluid dynamics simulations. There are eight different types of flow contained in the dataset, including flow past a backward facing step (back-step) and past a cylinder, both at a variety of Reynolds numbers, DNS-turbulence, sea surface flow driven by a quasi-geostrophic model (SQG), etc. For real world cases, we additionally use two datasets collected in a hydrodynamics laboratory in a shallow water flow past a cylinder configuration. To mimic the complex real world environment, we use imperfect markers in the form of foam and confetti as the visible tracers instead of high quality particles as would be typical in a professional PIV setting. The resulting images with imperfect and sparse tracer coverage are more challenging for fluid motion estimation, which helps extensively test the model's generalization ability.

5.2. Training

Due to the fact that the corrector can be combined with predictor in a non-intrusive way, we can first train the predictor without the corrector. The predictor is trained on the PIV dataset with 12,190 samples used for training and 1505 for testing. We train the model for 40,000 iterations with a batch size of four image pairs using the Adam optimiser. The learning rate is kept at 10^{-4}. After the predictor is trained, the corrector is trained on time-resolved data with the predictor frozen. Here we use the flow type DNS-turbulence for corrector training. All experiments are conducted on a moderate level GPU Nvidia Tesla P100 16GB. We trained two unsupervised models, named UnPwcNet-PIV and UnLiteFlowNet-PIV to indicate that these are the unsupervised version of PwcNet and LiteFlowNet trained on the PIV dataset; along with four supervised models, LiteFlowNet-PIV-S, PwcNet-PIV-S, LiteFlowNet-PIV-SF and PwcNet-PIV-SF for comparison. The 'S' denotes a

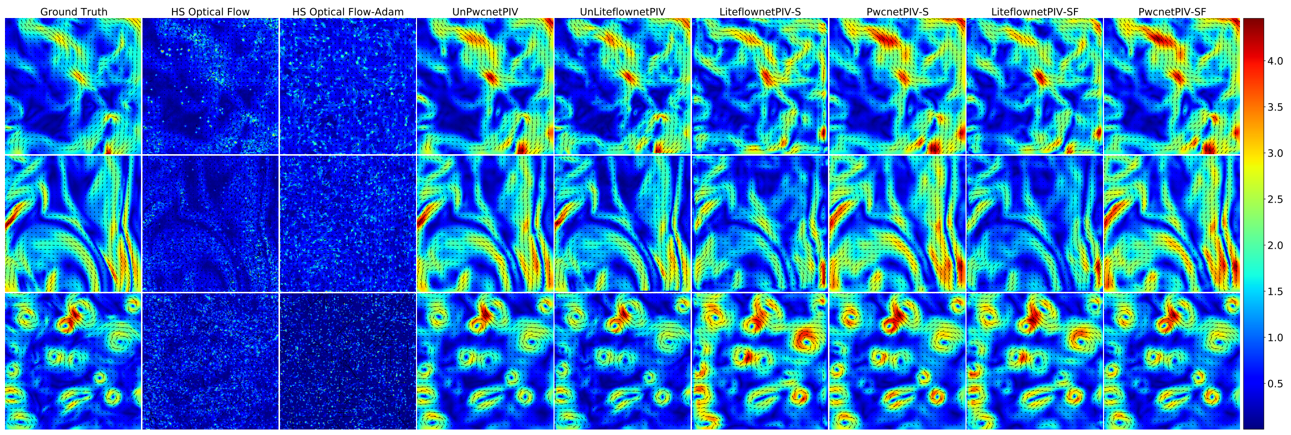

Figure 3. Estimated flow for the samples in the PIV dataset. The color bar describes the magnitude of the displacements in pixels.

supervised model, and 'SF' represents a supervised model fine-tuned with double training iterations. Note the method named UnLiteFlowNet-PIV appeared in the author's previous work (Zhang & Piggott, 2020), with the method proposed here further incorporating physics priors compared to the previous one.

5.3. Results on Synthetic Dataset.

Table 1. Averaged end point error (AEPE) and averaged angular error (AAE) for the PIV dataset, the error unit is set to pixel per 100 pixels for easier comparison. From top to bottom, the first row shows the results of our own GPU implementation of the Horn–Schunck (HS) optical flow approach, the second row shows the results of the HS optical flow solved by an Adam optimizer. The results of the supervised learning models we trained are listed in the next four rows. The final two row shows results of our unsupervised methods introduced in this work.

Methods	Back-Step		Cylinder		JHTDB channel		DNS turbulence		SQG	
	AEPE	AAE	AEPE	AAE	AEPE	AAE	AEPE	AAE	AEPE	AAE
HS Optical Flow	221.1	78.7	88.7	47.3	39.4	34.2	93.7	58.7	116.1	70.0
HS Optical Flow-Adam	200.9	62.6	58.1	25.5	16.1	13.5	57.7	30.4	70.3	35.2
LiteFlowNet-PIV-S	16.4	8.3	15.5	6.8	22.4	18.9	37.6	21.3	44.4	23.5
PwcNet-PIV-S	6.2	3.0	4.9	2.2	13.9	11.7	18.7	10.9	25.0	13.3
LiteFlowNet-PIV-SF	13.7	6.9	13.0	5.6	19.2	16.2	30.8	17.6	36.0	18.9
PwcNet-PIV-SF	**4.7**	**2.4**	**4.1**	**1.8**	12.9	10.8	16.0	9.4	22.1	11.8
UnLiteFlowNet-PIV	9.4	4.0	6.9	3.8	**8.4**	**3.3**	**15.0**	**8.6**	**17.3**	**9.0**
UnPwcNet-PIV	8.2	3.9	7.1	3.9	13.4	11.3	21.5	12.8	25.2	13.5

As shown in 1, both the supervised and unsupervised learning methods outperform the HS optical flow baselines for most cases. UnPwcNet-PIV shows competitive performance compared to PwcNet-PIV-S. UnLiteFlowNet-PIV performs better than its related supervised model LiteFlowNet-PIV-S and even the fine-tuned (double the training iterations) model LiteFlowNet-PIV-SF. This suggests that the proposed unsupervised losses are reasonable and can be used as a substitute to ground truth data for training to some extent. Also, it indicates that the LiteFlowNet backbone is more suitable for the unsupervised learning method, while PwcNet can achieve better performance when trained in a supervised manner.

Refinement by the Corrector. Figure 5 shows the abilities of the corrector for refining the prediction results. It can be observed that the corrector helps to reduce the average end point error and suppress the error fluctuation due to temporal inconsistency.

5.4. Results on Real World Datasets

Visual Comparisons. The results of the models when applied to the Fluid Foam and Fluid Confetti cases are shown in Figure 4. It can be observed that the HS optical flow and HS optical flow-Adam approaches struggle to give a clear estimation of the flow field. There are also obvious outliers around the edges of the foam or confetti. For supervised models, LiteflownetPIV-S and LiteflownetPIV-SF output estimations that conflict with the experimental setting, suggesting that the supervised models with the Liteflownet backbone overfit to the training dataset and show low generalization abilities. For the unsupervised models, the estimation results show reasonable flow structures while vorticity in the wake can also be observed.

Benchmark. Since there is no ground truth for real world datasets, here we used the best results (from our best tuning of the parameters) of the open source fluid motion estimation software PIVlab (Thielicke & Sonntag, 2021) (version 2.53) as a benchmark for our models. PIVlab is a popular Matlab toolbox, adopting correlation based, multi-pass, multi-grid window deformation techniques for fluid motion estimation, which can serve as a reliable baseline. We evaluate both the Fluid Foam and Fluid Confetti real use case datasets using PIVlab. For our models, we choose two unsupervised based models UnLiteFlowNet-PIV and UnPwcnet-PIV for

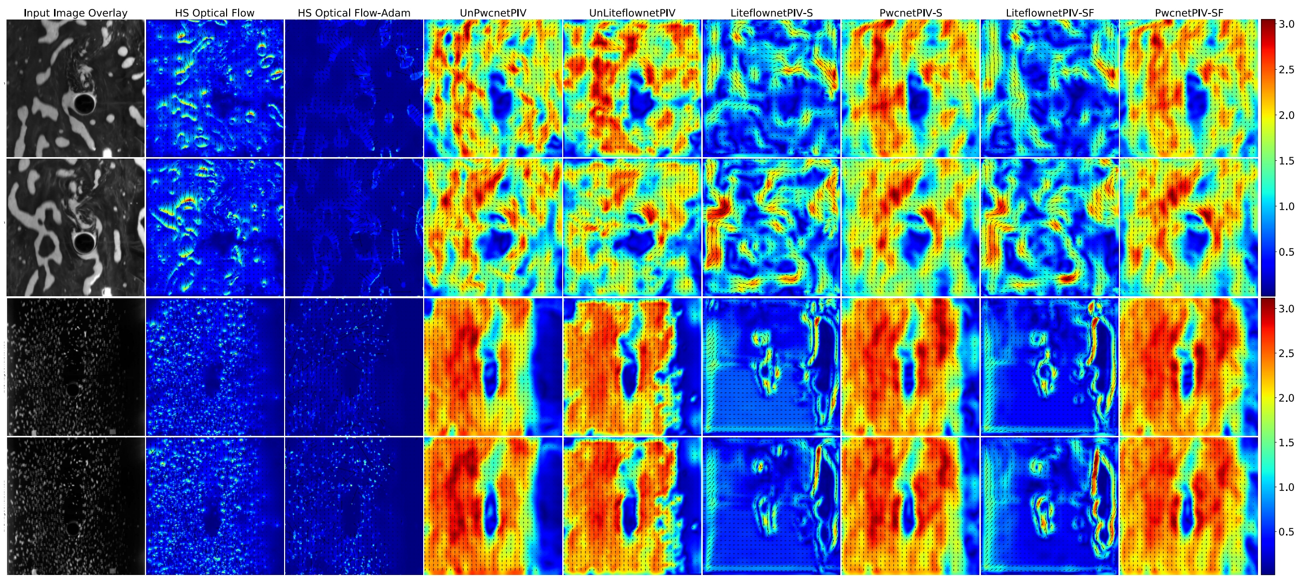

Figure 4. Estimated flow on Fluid Foam (upper two rows) and Fluid Confetti (bottom two rows) dataset. The color bar describes the magnitude of the displacements in pixels.

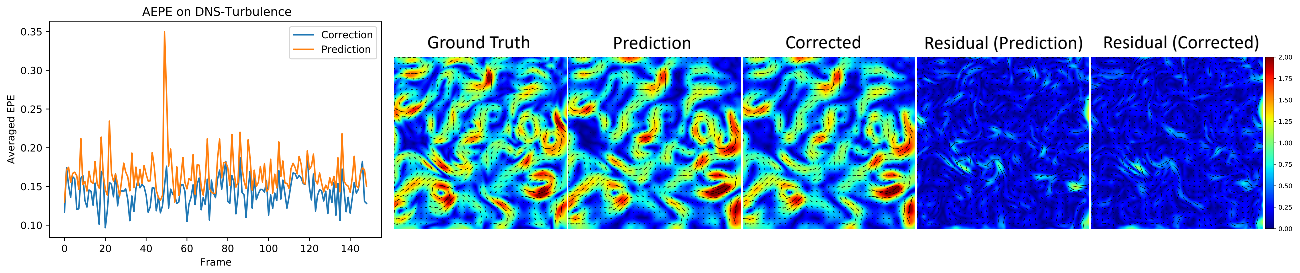

Figure 5. Comparison between prediction-only and the corrected velocities for the *DNS-turbulence* data.

comparisons. Note that the flow estimated velocity can also be interpreted as the displacement per unit time, so we use “displacement” in this paper for conciseness.

Quantitative Analysis. We adopt three approaches to interpret the results on the real world dataset: wake analysis, displacement distribution analysis and image reconstruction. The results of these approaches are discussed in detail in the following paragraphs.

Wake Analysis. The hydrodynamics experiments performed here are for flow past a cylinder in a shallow water environment. The most interesting part of this problem is the wake dynamics (Karman vortex street). This occurs in the region downstream of the obstacle (i.e., the region in the red bounding box in Figure 6). Therefore, one way to interpret the results of these use cases is to note whether the method can capture wake dynamics properly. To analyze the wake dynamics, we selected a line (shown by the vertical white dashed line in Figure 6) aligned with the position of the cylinder center, i.e. $x = 150$ pixels for the Fluid Foam Dataset and $x = 120$ pixels for the Fluid Confetti Dataset. We extracted the displacements for the x and y directions for all frames (195 for Fluid Foam and 175 for Fluid Confetti Dataset) along the line and compute the averaged displacements across frames. Figure 7 shows that the displacement curves of our two models demonstrate similar tendency and magnitude to that obtained using PIVlab, which implies that our models output reasonable results and are able to achieve competitive performance compared with sophisticated, modern PIV software on these real use cases. In addition, the curves show periodic patterns after the back circle (at around $y = 100$ pixels) of the cylinder, which is compatible to the dynamics expected of a Karman vortex street, indicating that the captured dynamics of all three methods are reasonable qualitatively. Note that due to its underlying methodology PIVlab can only output a sparse displacement field while the outputs of our methods are dense fields. To facilitate comparison, we interpolate the sparse field to a dense field using cubic interpolation. Therefore, the results of PIVlab are potentially over-smoothed, which may not properly describe the small scale dynamics such as vortices right after the cylinder. This may cause the gap between the curves obtained with our methods and PIVlab at the boundary regions.

Displacement Distribution The statistic of the output displacement fields provide another measure which has been adopted by the PIV community for evaluating different algorithms (Kähler et al., 2016), especially when there is no ground truth. Figure 8 shows the comparisons of output displacements distribution for both Fluid Foam (upper plot) and Fluid Confetti dataset (lower plot). It can be observed that the histograms of our methods and PIVlab largely overlap, indicating that output displacement distributions of these methods are similar.

Image Reconstruction. Another way to verify the agreement of the results is to reconstruct future images using the estimated flow. Figure 9 shows the future images reconstruction based on our methods and the PIVLab benchmark. It can be shown that the flow estimated by our methods can help reconstruct images close to ground truth visually, with competitive residual per pixel compared to PIVLab.

6. Related work

Optical Flow Estimation. Deep learning has shown its potential for optical flow estimation (Dosovitskiy et al., 2015; Ilg et al., 2017; Sun et al., 2018; Hui et al., 2018). These methods aim to estimate dense optical flow fields given an image pair using supervised trained deep neural network. There are also some efforts to investigate unsupervised (Yu et al., 2016; Meister et al., 2018; Jonschkowski et al., 2020) or self-supervised (Liu et al., 2019b;a; Jonschkowski et al., 2020) learning of optical flow estimation. All these methods are designed for pure optical flow methods, and mainly focus on the motion of rigid bodies. Moreover, there are works (Lee et al., 2017; Cai et al., 2019b;a) adopting optical methods to fluid flow motion estimation. These methods are limited to supervised learning, while fluid data are difficult to collect and annotate in real world applications. In addition, fluid dynamics are not considered in these methods, thus the physical correctness of the estimation results are not guaranteed. In this work, we leverage physical knowledge and construct a corrector to refine the estimations for

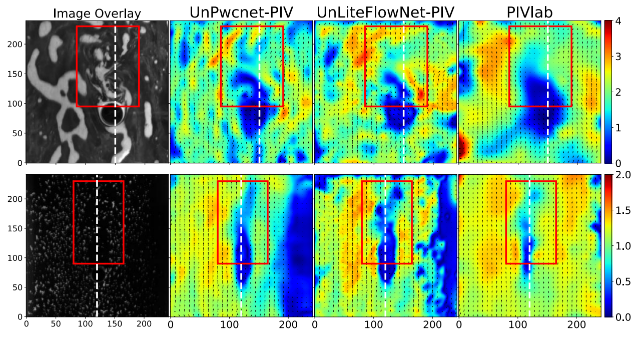

Figure 6. Visual results for Fluid Foam (upper row) and Fluid Confetti (bottom row) Dataset. The color bar describes the magnitude of the displacements in pixels.

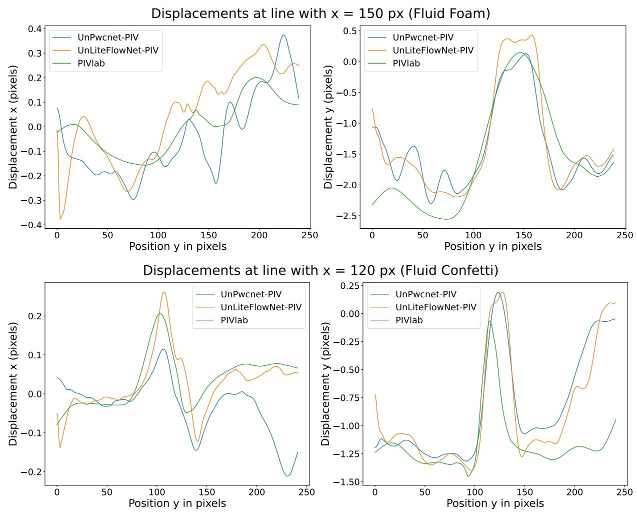

Figure 7. Estimated displacements (left column shows the displacement in the x direction and the right column shows it in the y direction) along the dashed white lines shown in Figure 6.

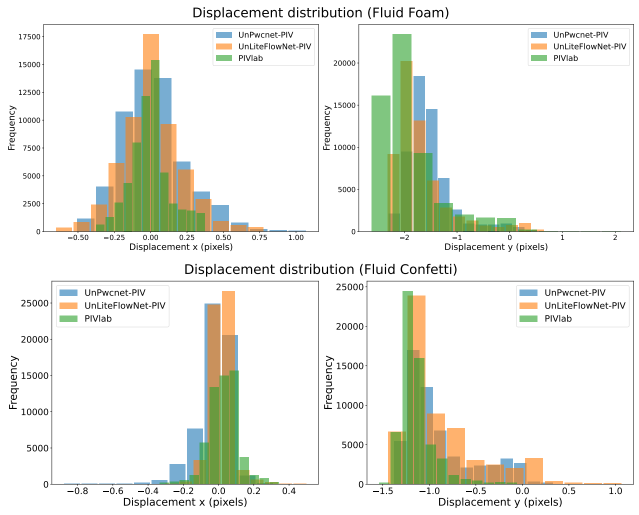

Figure 8. Histograms of output displacements distribution for both Fluid Foam (upper row) and Fluid Confetti dataset (bottom row).

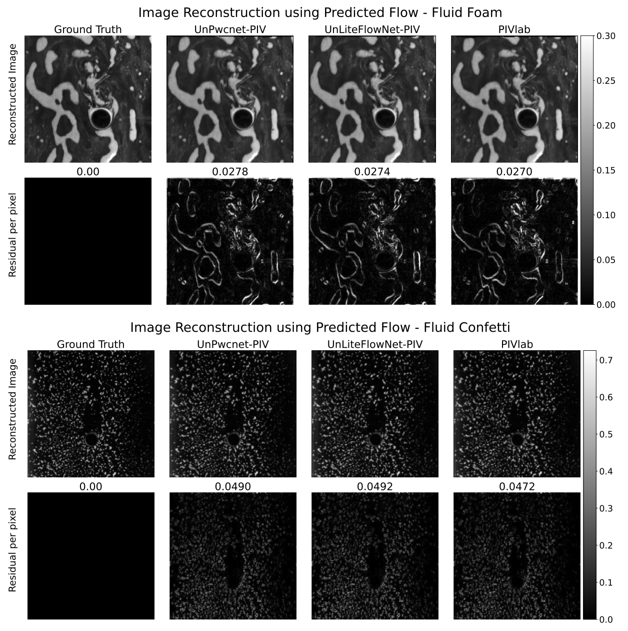

Figure 9. Image reconstruction using the predicted flow on the Fluid Foam and Confetti datasets. The numbers above the second row of images indicate the residual per pixel between the reconstructed and the true images.

physical correctness.

Learning to Solve PDEs. Solving Partial Differential Equations (PDEs) numerically is often slow and inefficient. To improve efficiency, several attempts have been conducted to approximate the solution and response function of PDEs using neural networks (Raissi et al., 2019; Raissi, 2018; Li et al., 2021). The connections between numerical schemes for solving PDEs and residual neural network have also been investigated (Chen et al., 2018; Lu et al., 2018; Weinan., 2017; M.Zhu et al., 2019) for predicting dynamical systems. Also, some studies show that partial derivatives can be approximated with convolutions (Long et al., 2018; Z.Long et al., 2019), which can be used to discretize a broad class of PDEs. In this work, we leverage this idea and use convolutions to model the fluid dynamics.

Learning to Enhance Fluid Simulation. Various studies have investigated the application of deep learning techniques to enhance fluid simulation, in terms of either efficiency or accuracy. Several studies have sought to accelerate simulation using deep neural networks. For solving the incompressible Euler equations, convolution networks are trained for divergence-free corrections (Tompson et al., 2017). In turbulent flows modelling (Kochkov et al., 2021), deep neural networks are applied to accelerate both direct numerical simulation and large eddy simulation of turbulent flows. To reduce numerical errors, a differentiable physics network is proposed which can interact with iterative PDE solvers (Um et al., 2020). Existing studies have also explored the abilities to learn to simulate complex physics simulations using of graph neural networks, in both Eulerian (Pfaff et al., 2021) and Lagrangian (Sanchez-Gonzalez et al., 2020) perspectives. Graph neural networks are also utilized to process unstructured mesh data and speed-up fluid flow prediction (de Avila Belbute-Peres et al., 2020) in computational fluid dynamics. However, these methods focus on simulation only; they do not support the processing of observations at each time step, which is required in fluid motion estimation scenarios.

7. Conclusion

Summary. We present here an unsupervised learning approach for solving the problem of fluid flow motion estimation. The proposed approach shows significant promise and potential advantages for fluid flow estimation. It yields competitive results when compared to existing supervised learning based methods, and even outperforms them for some difficult flow cases. Furthermore, the unsupervised learning approach shows robust generalization ability when dealing with complex real-world flow scenarios, though trained purely on synthetic data. In addition, the proposed physical corrector is able to refine the estimates obtained from the predictor by incorporating temporal information and fluid knowledge.

Limitation and Future Work. There are several limitations of the approach presented in this paper. First, there are only limited training and testing fluid observation data available, which is not enough for extensive test of the corrector. To address this, we aim to collect and generate more time-resolved data of different flow types for further investigation. Second, the initial and boundary conditions are not known for the physical model in the corrector. Third, the current model relies on observations at every time step. We plan to further investigate the prediction-correction scheme for forecasting tasks. This extension will require network architectures such as recurrent neural networks (RNNs) to better handle temporal information. Finally, we would like to explore more learning based methods inspired by numerical simulation, to bridge the communities between machine learning and scientific computing.

Acknowledgements. This work is sponsored by the Chinese Scholarship Council and Imperial College London (under a pump priming research award from the Energy Futures Lab, Data Science Institute and the Gratham Institute). JT wishes to acknowledge PhD research funding from the Government of Botswana.

References

Adrian, R. and Westerweed., J. *Particle Image Velocimetry*. Cambridge University Press, 2011.

Cai, S., Liang, J., Gao, Q., Xu, C., and Wei., R. Particle image velocimetry based on a deep learning motion estimator. *IEEE Transactions on Instrumentation and Measurement*, 2019a.

Cai, S., Zhou, S., Xu, C., and Gao., Q. Dense motion estimation of particle images via a convolutional neural network. *Experiments in Fluids*, 60, 2019b.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud., D. K. Neural ordinary differential equation. *Advances in neural information processing systems*, 2018.

Chorin, A. J. Numerical solution of the navier-stokes equations. *Math. Comp.*, pp. 745–762, 1968.

D. Heitz, E. Memin, C. S. Variational fluid flow measurements from image sequences: synopsis and perspectives. *Experiments in Fluids*, 48:369–393, 2010.

de Avila Belbute-Peres, F., Economon, T. D., and Kolter, J. Z. Combining differentiable pde solvers and graph neural networks for fluid flow prediction. *International Conference on Machine Learning*, 2020.

de Bezenac, E., Pajot, A., and Gallinari., P. Deep learning for physical processes: Incorporating prior scientific knowledge. *International Conference on Learning Representations*, 2018.

Dosovitskiy, A., P. Fischer, E. I., Hausser, P., Hazirbas, C., Golkov, V., der Smagt, P. V., Cremers, D., and Brox., T. Flownet: Learning optical flow with convolutional networks. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2758–2766, 2015.

Horn, B. and Schunck., B. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

Hui, T., Tang, X., and Loy., C. Liteflownet: A lightweight convolutional neural network for optical flow estimation. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8981–8989, 2018.

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox., T. Flownet 2.0: Evolution of optical flow estimation with deep networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2:2758–2766, 2017.

Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu., K. Spatial transformer networks. *Proceedings of the 28th International Conference on Neural Information Processing Systems*, 2:2017–2025, 2015.

Jonschkowski, R., Stone, A., Barron, J. T., Gordon, A., Konolige, K., and Angelova., A. What matters in unsupervised optical flow. *European Conference on Computer Vision*, 2020.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., and Hoyer, S. Machine learning accelerated computational fluid dynamics. *arXiv:2102.01010*, 2021.

Kähler, C. J., Astarita, T., Vlachos, P. P., Sakakibara, J., Hain, R., Discetti, S., Foy, R. L., and Cierpka1, C. Main results of the 4th international piv challenge. *Experiments in Fluids*, 57(97):97–167, 2016.

Lee, Y., Yang, H., and Yin., Z. Piv-dcnn: cascaded deep convolutional neural networks for particle image velocimetry. *Experiments in Fluids*, 58:8981–8989, 2017.

Li, Y., Perlman, E., Wan, M., Yang, Y., Meneveau, C., Burns, R., Chen, S., Szalay, A., and Eyink., G. A public turbulence database cluster and applications to study lagrangian evolution of velocity increments in turbulence. *Journal of Turbulence*, 9, 2008.

Li, Z., Kovachki, N., Azizzadenesheli, K., B. Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *International Conference on Learning Representations*, 2021.

Liu, P., King, I., Lyu, M., and Xu., J. Ddflow: Learning optical flow with unlabeled data distillation. *AAAI*, 2019a.

Liu, P., Lyu, M., King, I., and Xu., J. Selflow: Self-supervised learning of optical flow. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4571–4580, 2019b.

Long, Z., Lu, Y., Ma, X., and Dong, B. Pde-net: Learning pdes from data. *International Conference on Machine Learning*, 2018.

Lu, Y., Zhong, A., Li, Q., and Dong., B. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. *International Conference on Machine Learning*, pp. 3282–3291, 2018.

Meister, S., Hur, J., and Roth., S. Unsupervised learning of optical flow with a bidirectional census loss. *The Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

M.Zhu, B.Chang, and C.Fu. Convolutional neural networks combined with runge-kutta methods. *International Conference on Learning Representations*, 2019.

Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., and Battaglia, P. W. Learning mesh-based simulation with graph networks. *International Conference on Learning Representations*, 2021.

Raissi, M. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, pp. 932–955, 2018.

Raissi, M., Perdikaris, P., and Karniadakis, G. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, pp. 686–707, 2019.

Ruhnau, P., T. Kohlberger, C. S., and Nobach., H. Variational optical flow estimation for particle image velocimetry. *Experiments in Fluids*, 38:21–32, 2005.

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. W. Learning to simulate complex physics with graph networks. *International Conference on Machine Learning*, 2020.

Strang., G. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, pp. 506–517, 1968.

Sun, D., Roth, S., and Black., M. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *International Journal of Computer Vision*, 106:115–137, 2014.

Sun, D., Yang, X., Liu, M.-Y., and Kautz., J. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8934–8943, 2018.

Thielicke, W. and Sonntag, R. Particle image velocimetry for matlab: Accuracy and enhanced algorithms in pivlab. *Journal of Open Research Software*, 9, 2021.

Tompson, J., Schlachter, K., Sprechmann, P., and Perlin, K. Accelerating eulerian fluid simulation with convolutional networks. *International Conference on Machine Learning*, 2017.

Um, K., Brand, R., Fei, Y. R., Holl, P., and Thuerey, N. Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers. *Advances in neural information processing systems*, 2020.

Weinan., E. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, pp. 1–11, 2017.

Yu, J., Harley, A., and Derpanis., K. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. *European Conference on Computer Vision*, pp. 3–10, 2016.

Zhang, M. and Piggott, M. D. Unsupervised learning of particle image velocimetry. *ISC High Performance*, 2020.

Z.Long, Y.Lu, and B.Dong. Pde-net2.0: Learning pdes from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, 2019.

A. Fluid dynamics

Here we introduce the key methods and equations of fluid dynamics in this work.

Stokes Equation. The Stokes equation

$$-\mu\nabla^2\mathbf{u}+\nabla p=\mathbf{f}, \tag{21}$$

where μ is the dynamic viscosity, is an approximation/simplification to the Navier–Stokes momentum equation obtained by omitting the inertial part. It is usually used to model fluid flow with a low Reynolds number, where advective inertial forces are small compared with viscous forces. In this work, we demonstrate that the output of the motion predictor can be interpreted as the solution of a Stokes-like equation.

Vorticity Transport Equation. The vorticity is a pseudovector field (scalar field in 2D) that describes the tendency for fluid parcels to spin in the flow. Here we consider the incompressible vorticity transport equation that takes the form

$$\omega_t+\mathbf{u}\cdot\nabla\omega=\nu\nabla^2\omega, \tag{22}$$

where $\omega=\nabla\times\mathbf{u}$ denotes the vorticity, and ν is the kinematic viscosity. This is closely related to the full incompressible Navier-Stokes equation for homogeneous flow, describing the evolution of the fluid's vorticity over time. By using the vorticity transport equation, we can evolve the vorticity field while avoiding the need to solve the coupled pressure field that appears in the full Navier-Stokes system.

Chorin's Method. Chorin's projection method can be considered as an operator splitting based approach. The idea is first to compute a tentative velocity $\mathbf{u}_t^*$ by neglecting the pressure in the Navier-Stokes momentum equation, i.e. by solving

$$\frac{\mathbf{u}_t^*-\mathbf{u}_{t-1}}{\Delta t}=-\mathbf{u}_{t-1}\cdot\nabla\mathbf{u}_{t-1}+\nu\nabla^2\mathbf{u}_{t-1}, \tag{23}$$

and then projecting the obtained velocity $\mathbf{u}_t^*$ onto the space of divergence free vector fields using the update

$$\frac{\mathbf{u}_t-\mathbf{u}_t^*}{\Delta t}=-\frac{1}{\rho}\nabla p_t. \tag{24}$$

Note that if we add up both sides of Equation (23) and (24), the incompressiable Navier-Stokes momentum equation is recovered with the tentative velocity $\mathbf{u}_t^*$ cancelled.

According to Helmholtz decomposition, the velocity $\mathbf{u}_t^*$ can be decomposed into a divergence-free (solenoidal) part and an irrotational part:

$$\mathbf{u}_t^*=\mathbf{u}_{\text{sol}}+\mathbf{u}_{\text{irrot}}, \tag{25}$$

where $\nabla\cdot\mathbf{u}_{\text{sol}}=0$. The $\mathbf{u}_{\text{sol}}$ is the velocity $\mathbf{u}_t$ we would like to solve. Therefore, by taking the divergence of both sides of Equation (24), we obtain the equation

$$\frac{\rho}{\Delta t}\nabla\cdot\mathbf{u}_t^*=\nabla^2 p_t, \tag{26}$$

which is a Poisson equation for the unknown pressure p_t. Following solution of this equation, the updated velocity can be computed as

$$\mathbf{u}_t=\mathbf{u}_t^*-\frac{\Delta t}{\rho}\nabla p_t. \tag{27}$$

Warping Scheme and Advection-Diffusion Equation.

Theorem 1. For any initial condition $I_0\in L^1(\mathbb{R}^2)$ with $I_0(\pm\infty)=0$, there exists a unique global solution $I(x,t)$ to the advection-diffusion equation

$$I(\mathbf{x},t)=\int_{\mathbb{R}^2}k(\mathbf{x}-\mathbf{u},\mathbf{y})I_0(\mathbf{y})d\mathbf{y}, \tag{28}$$

where $k(a,b)=\frac{1}{4\pi Dt}e^{-\frac{1}{4Dt}\|a-b\|^2}$ is a Gaussian distribution density with mean $\mathbf{x}-\mathbf{u}$ and variance $2Dt$, $\mathbf{u}$ is the velocity field. Equation (28) indicates that the scalar field $I(x,t)$ can be computed via a convolution between a Gaussian kernel and the initial condition I_0. The full proof of the theorem can be found in de Bezenac et al. (2018).

B. Euler-Lagrangian Equation Derivation

Here we show the derivation of the Euler-Lagrange equation of formulation (6). Given $\mathbf{u} = (u, v)$, denoting $f_i = \frac{\partial f}{\partial x_i}, f_{ij} = \frac{\partial f}{\partial x_i \partial x_j}$, we rewrite Equation (6) as:

$$\begin{aligned}\mathcal{F}(u,v) &= \int_\Omega \mathcal{L}(u, v, u_x, u_y, v_x, v_y)\, d\mathbf{x} \\ &= \int_\Omega (I_t + uI_x + vI_y) + \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) \\ &\quad + p(u_x + v_y)\, d\mathbf{x}.\end{aligned} \tag{29}$$

According to the definition of Euler-Lagrange equation for several functions of several variables with single derivative, we can get the following system of equations for u and v:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial u} - \frac{\partial}{\partial x}(\frac{\partial \mathcal{L}}{\partial u_x}) - \frac{\partial}{\partial y}(\frac{\partial \mathcal{L}}{\partial u_y}) = 0, \\ \frac{\partial \mathcal{L}}{\partial v} - \frac{\partial}{\partial x}(\frac{\partial \mathcal{L}}{\partial v_x}) - \frac{\partial}{\partial y}(\frac{\partial \mathcal{L}}{\partial v_y}) = 0. \end{cases} \tag{30}$$

By simplifying the equations above, we can get

$$\begin{cases} 2\mu(u_{xx} + u_{yy}) + p_x = I_x, \\ 2\mu(v_{xx} + v_{yy}) + p_y = I_y. \end{cases} \tag{31}$$

Rewriting the system of equations above, we can get

$$-2\mu\nabla^2\mathbf{u} + \nabla(-p) = \nabla(-I), \tag{32}$$

which has the same form as the Stokes equation, where μ is the dynamic viscosity constant, $\nabla(-p)$ and $\nabla(-I)$ are the pressure gradient term and the external force term, respectively. Considering that μ is a constant we can omit the constant 2 in front of it. For the force terms, the minus sign only denotes the direction of the forces and thus they are omitted in Equation (7) of the main paper for clarity.

C. Further Generalization Tests

Although the Fluid foam and Fluid confetti dataset mentioned in this paper have already served to test the generalization abilities of the models, we adopt additional test cases from the Particle Image Velocimetry community. We consider two examples: "Jet Flow" (shown in Figure 10) and "Karman" (shown in Figure 11).

D. Dataset

PIV Dataset. There are eight different types of flow in the PIV dataset. The types "Back-step" and "Cylinder" also contain data for different Reynolds numbers. The detailed descriptions for each type are shown in Table 2.

Fluid Foam Dataset. The setting for Fluid Foam can be summarized as follows: cylinder diameter = 50mm, frame rate = 60fps, water depth = 90mm, 1px = 0.0002174m. A sample of the collected images is shown in Figure 12.

Fluid Confetti Dataset. Fluid Confetti is another dataset that we collected using a similar setting as for the Fluid Foam case. The difference is that the visible tracers are small pieces of white confetti (small squares of paper around 2mm in width), and the lighting has also been set up so the tracers could be clearly observed in the images. For the camera, a Hero7 camera with 4k resolution was set up at an angle and a 2.7k resolution Hero5 camera was set up vertically. The camera takes photos with an interval 1/60 second.

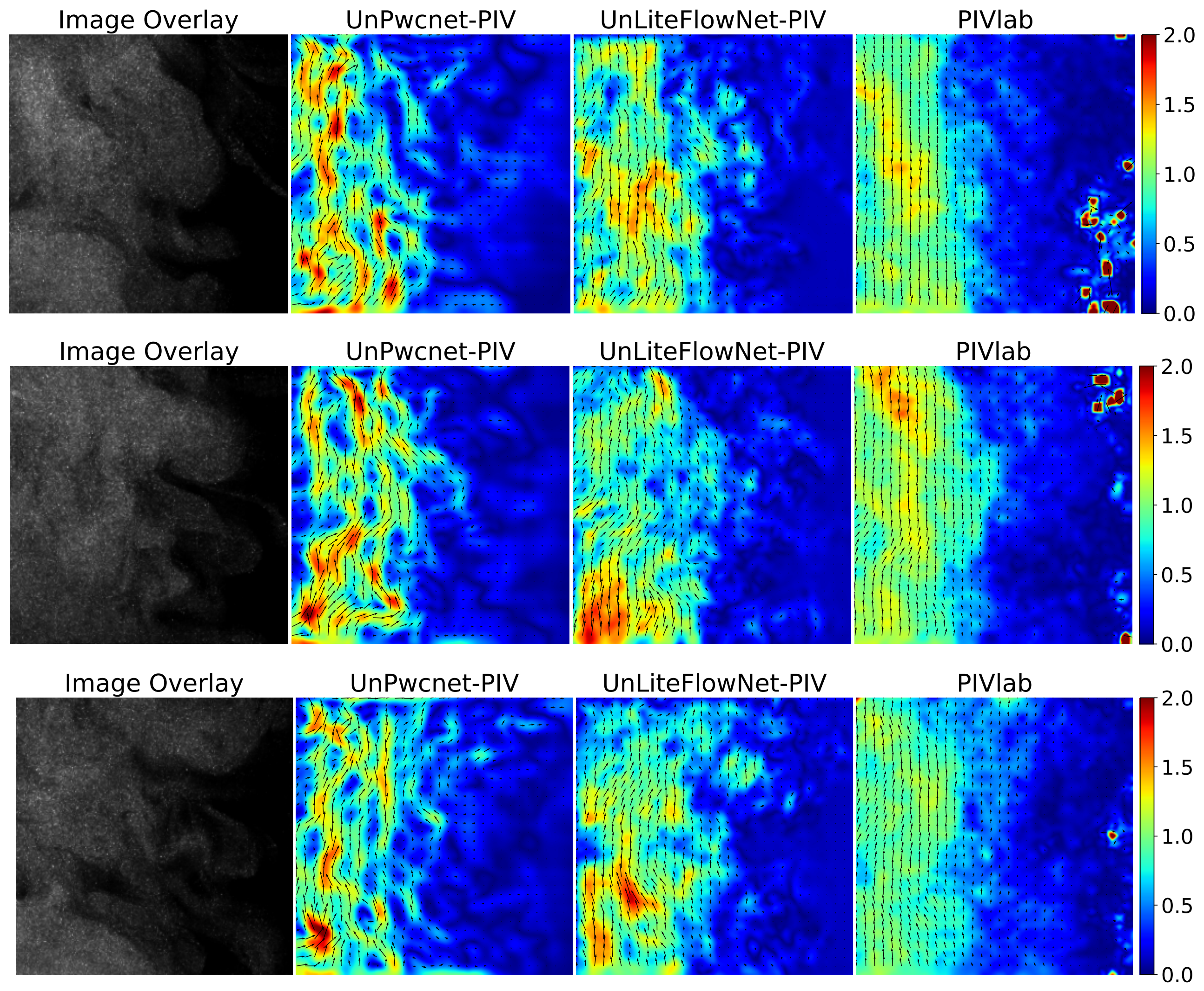

Figure 10. Extra real use case "Jet Flow" from 3th PIV challenge `https://www.pivchallenge.org/`. The figure shows that our methods can capture the dynamics properly with more small scale detail comparing to the over-smoothed PIVlab results.

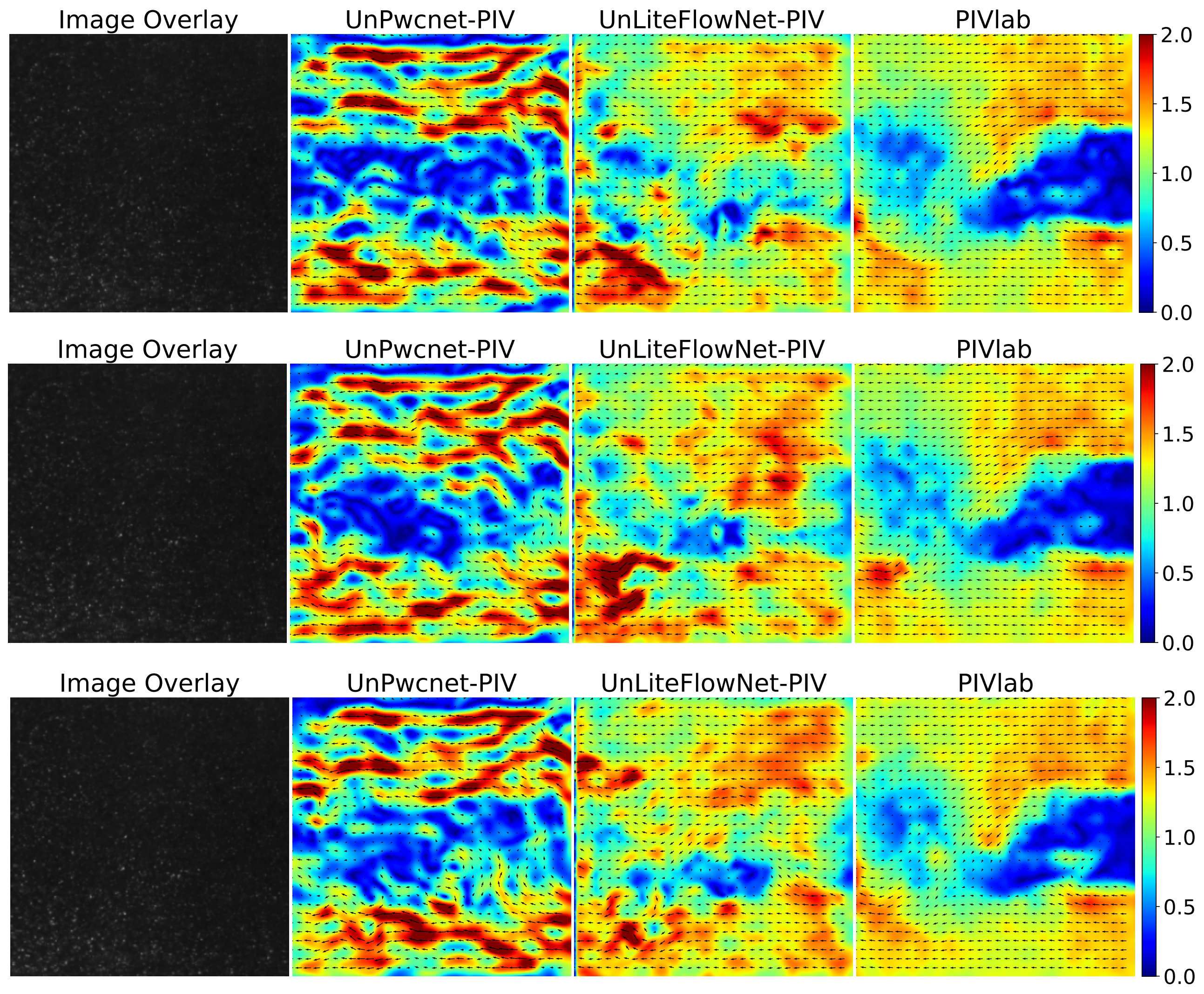

Figure 11. Extra real use case "Karman" from PIVlab. It is observed that the model UnLiteFlowNet-PIV can still capture the wake after the obstacle, although the UnPwcnet-PIV outputs noisy results.

Table 2. Detailed description of the PIV dataset considered, from (Cai et al., 2019b). dx refers to the particle displacements considered between two image frames in units of number of pixels. Re refers to the Reynolds numbers considered. 'JHTDB' implies that the data was taken from the Johns Hopkins turbulence databases (Li et al., 2008). Refer to (Cai et al., 2019b) for further details.

Type Name	Description	Condition	Quantity
Uniform	Uniform flow	$\|dx\| \in [0, 5]$	1000
Back-step	Flow past a backward facing step	Re = 800	600
		Re = 1000	600
		Re = 1200	1000
		Re = 1500	1000
Cylinder	Flow past a circular cylinder	Re = 40	50
		Re = 150	500
		Re = 200	500
		Re = 300	500
		Re = 400	500
DNS-turbulence	Homogeneous and isotropic turbulent flow	-	2000
SQG	Sea surface flow driven by SQG model	-	1500
Channel flow	Channel flow provided by JHTDB	-	1600
JHTDB-mhd1024	Forced MHD turbulence provided by JHTDB	-	800
JHTDB-isotropic1024	Forced isotropic turbulence provided by JHTDB	-	2000

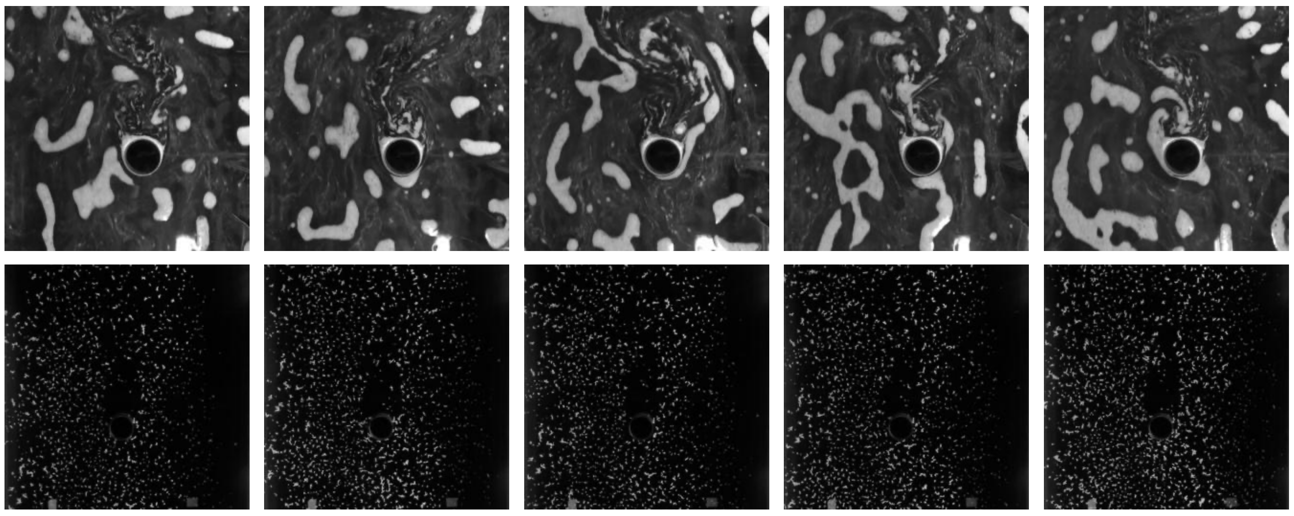

Figure 12. Samples of Fluid Foam (upper row) and Fluid Confetti (bottom row) dataset. The white foam/confetti in the image serve as the visible tracers.