Eurographics Conference on Visualization (EuroVis) 2019
M. Gleicher, H. Leitte, and I. Viola
(Guest Editors)

Volume 38 (2019), Number 3

Robust Reference Frame Extraction from Unsteady 2D Vector Fields with Convolutional Neural Networks

Byungsoo Kim and Tobias Günther

Department of Computer Science, ETH Zurich

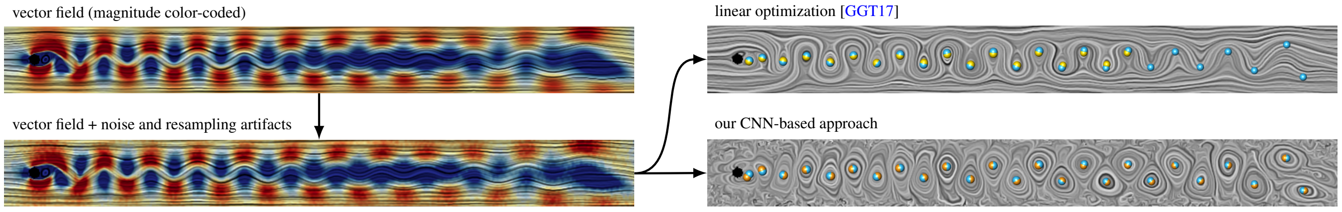


Figure 1: *We developed a novel CNN-based reference frame extraction algorithm that is trained to handle inputs with noise and resampling artifacts. Compared to a linear reference frame optimization [GGT17], our method is more robust to artifacts. The input vector field (w/ and w/o noise) is shown on the left, and the extraction of vortex centers (orange and yellow), compared to a ground truth (blue) is shown on the right. By combining filtering and reference frame extraction via CNNs, vortex extraction becomes more robust. Note that in the experiment above, the CNN has not seen the cylinder flow during training. In fact, it only trained on a synthetic data base that we introduce in the paper.*

Abstract

Robust feature extraction is an integral part of scientific visualization. In unsteady vector field analysis, researchers recently directed their attention towards the computation of near-steady reference frames for vortex extraction, which is a numerically challenging endeavor. In this paper, we utilize a convolutional neural network to combine two steps of the visualization pipeline in an end-to-end manner: the filtering and the feature extraction. We use neural networks for the extraction of a steady reference frame for a given unsteady 2D vector field. By conditioning the neural network to noisy inputs and resampling artifacts, we obtain numerically stabler results than existing optimization-based approaches. Supervised deep learning typically requires a large amount of training data. Thus, our second contribution is the creation of a vector field benchmark data set, which is generally useful for any local deep learning-based feature extraction. Based on Vatistas velocity profile, we formulate a parametric vector field mixture model that we parameterize based on numerically-computed example vector fields in near-steady reference frames. Given the parametric model, we can efficiently synthesize thousands of vector fields that serve as input to our deep learning architecture. The proposed network is evaluated on an unseen numerical fluid flow simulation.

1. Introduction

The robust extraction of vortices remained to this day one of the most difficult problems of unsteady vector field analysis [GT18]. Vortices themselves are studied in many different applications, such as for engine design [RP96, GLT*07], blood flow analysis [KGP*13, OJCJP16] or even in the atmosphere of other planets [HH16]. A vortex is commonly understood as a set of particles rotating around a common point or axis, if the flow is viewed in the correct reference frame [Lug79, Rob91]. This dependence on the reference frame is what makes vortex extraction in unsteady flows difficult. On the one hand, existing methods concentrated on vortex characterizations that give the same result for many different reference frames, such as those arising from a Galilean transformation [Oku70, Hun87, JH95, WSTH07] or from a smooth rotation and translation of the reference frame [Ast79, Hal05, HHFH16]. On the other hand, reference frames have been searched in which topologically relevant structures appear [BPKB14]. Lugt [Lug79] and Robinson [Rob91] argued that the relevant reference frame is the one in which the flow becomes steady. Thus, recently Günther et al. [GGT17] formulated the local search for the steady reference frame as an optimization problem. Their method, however, relies on a regularization that often prevents them from finding a perfectly steady solution. In the limit case of no regularization, their solution depends on second-order derivatives that are extremely difficult to estimate robustly if the data is noisy. Robustness to noise and resampling artifacts is also a major concern when it comes to the local vortex extraction algorithms [GLL91, PR99] that are applied in the resulting frame. An algorithm for the robust extraction of the optimal reference frame that is insensitive to noise and other distortion artifacts is desperately needed in order to achieve a reliable vortex detection in unsteady vector fields.

Inspired from the recent success of machine learning approaches in robustly solving image processing tasks [GBCB16], we propose to train a convolutional neural network (CNN) to locally extract the optimal reference frame, in which a given unsteady 2D vector field becomes steady. We thereby concentrate on the robustness of the extractor by training it on synthetically distorted data, which includes the adding of noise and a prior resampling. A general challenge of machine learning is that it typically requires a large amount of data to successfully generalize. Since machine learning is a rather young field for flow visualization, we are not aware of any large benchmark vector fields that could directly be used for training. A second contribution of our paper is therefore the synthetic generation and the release of a vector field benchmark data set that may be used in the future by other researchers, looking for applications of machine learning in flow visualization. In summary:

- We introduce a novel parametric flow mixture model that is based on Vatistas [VKM91] velocity profiles. After fitting parameters to numerical data, we synthesize a large vector field collection that contains divergence-free and compressible flows.
- We design and evaluate a convolutional neural network to locally extract the optimal reference frame in which a given 2D vector field becomes steady.

In terms of performance our method is slightly slower than a linear optimization [GGT17], but our CNN-based approach has up to 3-4 times lower reconstruction residuals under increasing noise and resampling artifacts, making our method much more robust on real-world data. Fig. 1 gives a first impression of the feature extraction results that are possible on distorted data after computing the optimal reference frame with either our approach or the linear optimization [GGT17].

2. Related Work

In the following sections, we lay the foundations for our work, including a summary of visualization methods that utilize machine learning, and an overview of objective vortex extraction methods.

2.1. Machine Learning in Visualization

Explainable AI. Machine learning and visualization have great potential for synergies. Explainable AI is currently a highly relevant research area that uses visualization to look into the black box that deep learning is often considered to be. We refer to Seifert et al. [SAB*17], Hohman et al. [HKPC18] and Ancona et al. [ACOG18] for an introduction into this promising area.

Deep Learning for Visualization. The opposite direction, i.e., the use of deep learning to solve visualization tasks is still fairly uncharted territory. Frey [Fre17] trained a neural network to determine the best progressive sampling strategy to calculate the similarity of spatio-temporal data sets. In information visualization, Fan and Hauser [FH18] used CNNs to improve the manual brushing of points in scatterplots. Berger et al. [BLL17] explored the use of generative adversary networks to analyze the role of transfer functions in the image synthesis process of direct volume rendering. For a steady 3D vector field, Han et al. [HTW18] used an autoencoder to learn a low-dimensional feature space of a voxel representation of randomly generated streamlines or stream surfaces. They plotted the low-dimensional feature space with t-SNE and selected representative geometric primitives from a density-based clustering.

CNNs for Vortex Extraction. Lguensat et al. [LSF*17] used a classification CNN to identify ocean eddies from sea surface height maps. Not only interested in the detection of ocean eddies but also in their tracking, Franz et al. [FRM*18] trained a classification CNN that receives the Okubo-Weiss [Oku70, Wei91] criterion as input and tracked the structures with optical flow and a spatio-temporal recurrent neural network. Bin and Li [BY18] classified normalized vector field patches of size 9×9 with a classification CNN into clockwise rotating, counterclockwise rotating, saddle type and others. Ströfer et al. [SWXP18] trained a classification CNN to identify specific fluid flow features in the domain, such as recirculation regions, boundary layers and a horseshoe vortex. Deng et al. [DWL*18] trained a CNN to recall the instantaneous vorticity deviation (IVD) of Haller et al. [HHFH16].

All methods above either performed a classification or produced a region-based vortex measure. Since we have line-based vortex extraction in mind, numerical stability is of greater concern to us. Thus, we utilize CNNs to greatly improve the numerical stability by conditioning the network to noisy and resampled inputs, resulting is the first deep learning-based reference frame extraction.

2.2. Objective Vortex Extraction

Over the past decades, dozens of vortex extraction algorithms have been proposed in the flow visualization and fluid mechanics literature. We refer to Günther and Theisel for a recent overview [GT18].

2.2.1. Definition of Objectivity

The most recent vortex extraction methods [HHFH16, GGT17, GT19b, GT19a, HMTR19] aspired to be *objective*. A measure is called *objective* if it remains invariant under a smooth rotation and/or smooth translation of the reference frame [TN65]. Such a transformation transforms a point $(\mathbf{x}, t)$ to the location $(\mathbf{x}^*, t^*)$ with:

$$\mathbf{x}^* = \mathbf{Q}(t)\mathbf{x} + \mathbf{c}(t) , \quad t^* = t - a \tag{1}$$

for a time-dependent rotation matrix $\mathbf{Q}(t)$, a time-dependent translation vector $\mathbf{c}(t)$ and a constant time shift a. Objectivity guarantees that a measure looks the same for different rotations and translations of the observer. Because of the relativity between the motion of the vortex and the motion of the observer, this is equivalent to the guarantee that the measure of a moving feature always look the same if the feature rotates and/or translates. Objectivity is therefore useful, when searching for moving features in unsteady flows.

2.2.2. Reference Frame Transformation of a Vector Field

In order to train a network that recovers a reference frame transformation $\mathbf{Q}(t)$, $\mathbf{c}(t)$ in which a vector field appears steady, we have to generate a large number of unsteady vector fields with a known ground truth transformation. For this, we start from a steady vector field and transform it with many random reference frame transformations, which makes each resulting flow unsteady. In general, a vector field $\mathbf{v}(\mathbf{x},t)$ is transformed into a new frame via [GGT17]:

$$\mathbf{v}^*(\mathbf{x}^*,t^*) = \mathbf{Q}(t)\mathbf{v}(\mathbf{x},t) + \frac{\mathrm{d}\mathbf{Q}(t)}{\mathrm{d}t}\mathbf{x} + \frac{\mathrm{d}\mathbf{c}(t)}{\mathrm{d}t} \tag{2}$$

The evaluation of this equation requires the transformation of $(\mathbf{x}^*,t^*)$ into the old frame $(\mathbf{x},t)$ in order to sample the given (steady) vector field $\mathbf{v}(\mathbf{x},t)$ at the correct location. Since $\mathbf{Q}(t)$ is orthogonal, rearranging Eq. (1) gives:

$$\mathbf{x} = \mathbf{Q}(t)^{\mathrm{T}}\left(\mathbf{x}^* - \mathbf{c}(t)\right), \qquad t = t^* + a \tag{3}$$

By inserting Eq. (3) into Eq. (2), we can uniformly sample the space-time domain $(\mathbf{x}^*,t^*)$ of the transformed unsteady vector field onto a regular grid, which is later fed with the corresponding transformation $\mathbf{Q}(t)$ and $\mathbf{c}(t)$ to the neural network during training.

2.2.3. Objective Region-based Vortex Extraction Methods

In order to derive an objective version of the vorticity tensor, Drouot and Lucius [DL76] and later Tabor and Klapper [TK94] independently proposed the *relative vorticity tensor*, which is viewed in the strain rate basis. The strain rate tensor is objective and thus, all measures viewed in its eigenvector basis become objective as well. Astarita [Ast79] formally proved the objectivity of this approach and further introduced an objective index measure that classifies the domain into regions that either perform extension-like motions or rigid-body-like rotations. Haller [Hal05] introduced the M_z criterion, which first uses the strain rate acceleration tensor to classify the domain into elliptic and hyperbolic regions. By taking a Lagrangian perspective, their approach identifies coherent vortices using particles that stay for a long time in elliptic areas, i.e., they perform rotating motions. Vortex boundaries are also seen as elliptic Lagrangian coherent structures [Hal15]. More recently, Haller et al. [HHFH16] identified coherent vortices objectively using the original vorticity tensor Ω. They introduced a Eulerian measure called instantaneous vorticity deviation (IVD), which subtracts the spatial mean vorticity of the neighborhood, and the integral of IVD along a pathline called Lagrangian averaged vorticity deviation.

2.2.4. Reference Frame Optimization

An objective vortex coreline extractor has recently been proposed by Günther et al. [GGT17]. Since we use their method as baseline in our comparison, we briefly explain the approach in more detail. In the late 1970s, Lugt [Lug79] characterized vortices as closed or spiraling streamlines in a reference frame in which the flow field becomes steady. Similarly, Robinson [Rob91] identified them as closed streamlines in a reference frame that moves with the vortex center. While it was clear early on that not a single reference frame exists in which all vortices become steady [Lug79], since they might move into different directions, it was recognized that the frame has to be searched locally [PC94]. For this reason, Günther et al. [GGT17] solved an optimization problem to find a local reference frame in which the vector field becomes steady. Their goal was to minimize the time partial of the transformed field $\mathbf{v}^*$ earlier shown in Eq. (2) in a local neighborhood U:

$$\int_U \left\| \frac{\partial \mathbf{v}^*(\mathbf{x}^*,t)}{\partial t} \right\|^2 dV \rightarrow min \tag{4}$$

Thereby, the unknowns are the rotation $\mathbf{Q}(t)$ and the translation $\mathbf{c}(t)$ of the reference frame. Important to us is how the rotation $\mathbf{Q}(t)$ and the translation $\mathbf{c}(t)$ can be discretized. When computing the time partial of $\mathbf{v}^*$ in Eq. (2) we only need up to second-order partials [GGT17]. Further, we can restrict the transformation at time t to $\mathbf{Q}(t) = \mathbf{I}$ and $\mathbf{c}(t) = \mathbf{0}$ to obtain our vortex features at the same locations as in the input field $\mathbf{v}$. Thus, the only unknowns are the first-order and second-order derivatives of the rotation and translation, evaluated at time t:

$$\dot{\mathbf{Q}} = \frac{\mathrm{d}\mathbf{Q}(t)}{\mathrm{d}t}, \quad \ddot{\mathbf{Q}} = \frac{\mathrm{d}^2\mathbf{Q}(t)}{\mathrm{d}t^2}, \quad \dot{\mathbf{c}} = \frac{\mathrm{d}\mathbf{c}(t)}{\mathrm{d}t}, \quad \ddot{\mathbf{c}} = \frac{\mathrm{d}^2\mathbf{c}(t)}{\mathrm{d}t^2}$$

In 2D, these are six numbers (two angles and two 2D vectors). Günther et al. [GGT17] have shown that this minimization can be efficiently solved as a linear optimization problem. Recently, they extended their method to affine transformations [GT19a]. Meanwhile, Hadwiger et al. [HMTR19] formulated the search as a global optimization problem. They avoided the choice of a suitable neighborhood and instead regularized the method by the assumption of isometry, i.e., the reference frame transformation is given by an approximate Killing field. Prior to the search for steady reference frames, other approaches have been followed to find a distinguished reference frame for unsteady flow, including the subtraction of the harmonic vector field component found by a Helmholtz-Hodge decomposition [Wie04, WGS07, BPKB14] and reference frames derived from a Galilean-invariant topology that is based on an analysis of the determinant of the Jacobian [BHJ16].

3. Synthetic Generation of Vector Fields

We propose an end-to-end approach that combines preprocessing (smoothing) and feature extraction (reference frame extraction) to compute for a possibly noisy unsteady vector field the reference frame transformation in which the flow becomes steady. To train the model, we first synthetically generate a large data base, containing thousands of vector field patches.

3.1. Parametric Mixture Model for Vector Fields

If enough diverse real-world training data was available, variational autoencoders [KW13] or generative adversarial networks [GPAM*14] could be used to synthesize further training data. In our work, we incorporate experimental observations [VKM91] to formulate an explicit parametric vector field mixture model that is specialized on vortices. As shown later, it generalizes to other flow features. Our analytic model is readily available to everyone, and works without a large flow data base.

3.1.1. Vatistas Vortex Velocity Profile

Since we are later mainly interested in vortices, we make use of Vatistas [VKM91] experimentally-obtained vortex velocity profile.

With r_c being the radius with maximal velocity, the tangential flow velocity $v_0(r)$ of a rotationally-symmetric unit vortex is:

$$v_0(r) = \frac{r}{2\pi r_c^2 \left((\frac{r}{r_c})^{2n} + 1 \right)^{\frac{1}{n}}} \,. \tag{5}$$

This equation was taken from Bhagwat and Leishman [BL00], who formulated the model for varying vortex radii. Note that Vatistas' vortex model contains other well-known vortex models as special cases, such as the Kaufmann vortex [Kau62] (for $n = 1$) and the Rankine vortex (for $n \to \infty$). Thus for increasing n, the behavior near the critical point becomes linear. For $n = 2$, Vatistas model is similar to the Lamb-Oseen vortex model [Ose12]. Note that Eq. (5) denotes the radius r in absolute values, rather than being relative to r_c as in the literature [BL00]. The different vortex models are shown in Fig. 2. We leave the core radius r_c and the shape exponent n as degrees of freedom in the parametric model.

3.1.2. Parametric Mixture Model

Based on Vatistas velocity profile $v_0(r)$ in Eq. (5), we define a steady flow primitive $\mathbf{v}_p$ with a critical point at $\mathbf{t} = (t_x, t_y)$ as:

$$\mathbf{v}_p(x,y) = \begin{bmatrix} d_x & c_x \\ -c_y & d_y \end{bmatrix} \begin{pmatrix} x - t_x \\ y - t_y \end{pmatrix} \cdot \frac{v_0(\sqrt{(x-t_x)^2 + (y-t_y)^2})}{\sqrt{(x-t_x)^2 + (y-t_y)^2}} \tag{6}$$

with the physical meaning that $\mathbf{c} = (c_x, c_y)$ describes the vortical motion and $\mathbf{d} = (d_x, d_y)$ denotes the in-flow and out-flow. In total, each primitive has eight degrees of freedom: c_x, c_y, d_x, d_y, t_x, t_y, r_c, and n. Depending on the parameterization, a range of different flow structures appear, including vortices, sinks, sources and saddles. Different instances of the model are shown in Fig. 3. By restricting the above parameters, it would be possible to incorporate additional domain knowledge, for instance, by concentrating on divergence-free flows only. We used the full range of all eight parameters to test the limits of a general model.

Since our vector field patches might contain multiple critical points, we use a mixture model of m flow primitives:

$$\mathbf{v}(x,y) = \sum_{p=1}^{m} \mathbf{v}_p(x,y) \,, \tag{7}$$

which in general results in $8m$ parameters. To find physically plausible parameter configurations, we fit the above mixture model to

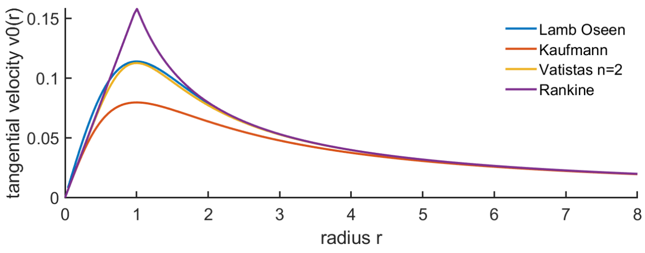


Figure 2: *Plot of common vortex velocity profiles. Starting from the center, the tangential velocity increases up until a certain maximum (here at $r_c = 1$). Afterwards, it decays with increasing distance. The Vatistas model [VKM91] contains Kaufmann [Kau62] for $n = 1$ and Rankine for $n \to \infty$ as special cases. For $n = 2$, Vatistas is similar to the Lamb-Oseen model [Ose12].*

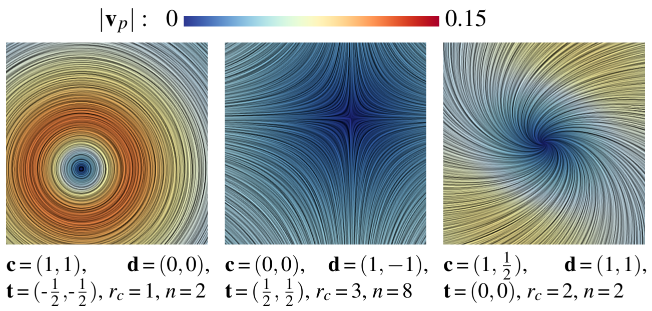

$|\mathbf{v}_p|$: 0
0.15
$\mathbf{c}=(1,1)$, $\mathbf{d}=(0,0)$, $\mathbf{t}=(-\frac{1}{2},-\frac{1}{2})$, $r_c=1$, $n=2$
$\mathbf{c}=(0,0)$, $\mathbf{d}=(1,-1)$, $\mathbf{t}=(\frac{1}{2},\frac{1}{2})$, $r_c=3$, $n=8$
$\mathbf{c}=(1,\frac{1}{2})$, $\mathbf{d}=(1,1)$, $\mathbf{t}=(0,0)$, $r_c=2$, $n=2$

Figure 3: *Examples of steady 2D flows, generated by our model in Eq. (6). Here, shown for the spatial domain $\mathcal{D} = [-2,2]^2$.*

patches of a numerical data set, which we describe next. Other approaches to formulate a model are discussed later in Section 6.

3.1.3. Parameter Space Fitting

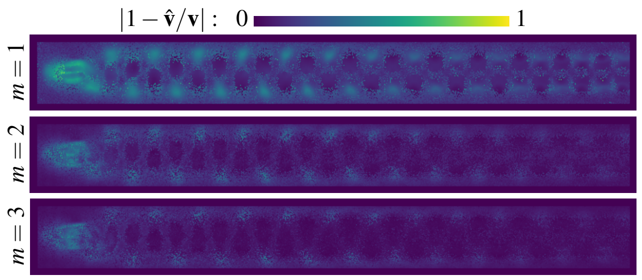
Eq. (7) gives rise to a large number of different vector fields, which leads to two questions to answer: how can we restrict the parameter space to physically-plausible flows and are there flows that cannot be captured by the model? To answer these questions, we fit the above model to numerically simulated vector fields. Since we are aiming for a vector field collection that represents flows in optimal near-steady reference frames, we first extract the optimal reference frame with the approach of Günther et al. [GGT17]. Since the parameters in Eq. (7) are non-linear, we first use 200 iterations of simulated annealing, which is followed by further 200 iterations of gradient descent to settle into the local minimum. Fitting a vector field requires a distance metric between vector fields. In this paper, we use an L_1 distance with gradients, which was introduced by Kim et al. [KCAT*18] in the context of velocity reconstruction using CNNs. They introduced two metrics: one for divergence-free flows and one for compressible flows. Since we do not constrain the parameter range in our parametric model in Eq. (6), we opt for the more general distance metric for compressible flows:

$$\mathcal{L}(\hat{\mathbf{v}}, \mathbf{v}) = ||\hat{\mathbf{v}} - \mathbf{v}||_1 + \lambda ||\nabla \hat{\mathbf{v}} - \nabla \mathbf{v}||_1 , \tag{8}$$

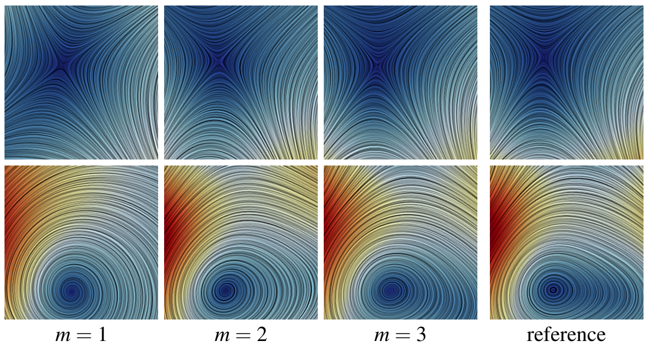
where $\hat{\mathbf{v}}$ and $\mathbf{v}$ are the vector fields to compare, and λ is a weight for the gradient difference. In practice, we use $\lambda = 1$, as it gives lower residuals than for $\lambda = 0$ or for the L_2-norm (Mean Squared Error), as shown in Appendix A. Fig. 4a shows a heat map in the CYLINDER flow, displaying for each voxel in the domain, how closely we could fit the numerical data with our parameteric mixture model. It becomes apparent that obstacles are not well represented in the model, but the remainder of the domain is approximated well. Below in Fig. 4b, individual patches of the domain are shown, displaying how closely the mixture model matches the numerical data for the varying numbers of model components m. For $m = 3$, we achieve a high accuracy, and we thus use $m = 3$ models later during the vector field synthesis. Finally, Fig. 4c shows histograms of the individual parameters. The histograms are used in the next section to sample further vector fields. In Appendix B, additional fitting results are shown for the BOUSSINESQ vector field.

3.1.4. Sampling of the Parameter Space

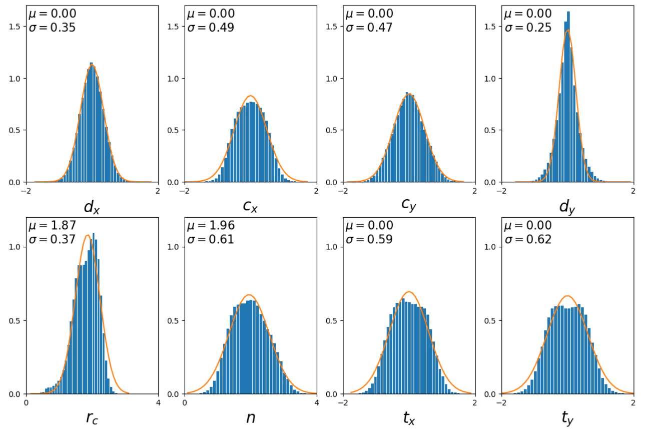
The histograms in Fig. 4c characterize the distribution of model parameters needed in order to generate further vector field patches



(a) *Heat maps of the fitting residual for varying number of mixture model components m. The higher m, the lower the residual. Obstacles are not yet contained in the parametric flow model.*



(b) *Fitting results for selected vector field patches. The fitting improves with more degrees of freedom, i.e., for a higher number of models m.*



(c) *Histograms of the individual model parameters, showing the near-Gaussian distribution of the individual parameter values.*

Figure 4: *Fitting results for the* CYLINDER *flow, showing that our parametric flow mixture model approximates numerical data well.*

of similar type. We individually approximate the distribution of each model parameter (c_x, c_y, d_x, d_y, t_x, t_y, r_c, and n) by a separate Gaussian distribution $\mathcal{N}$. The mean and standard deviation of each component are shown in Fig. 4c. In order to sample more flows, similar to the CYLINDER flow, the correlation between the random variables would have to be accounted for. In order to span a wider range of vector fields, we purposefully neglect the correlations and sample each variable independently. This means, we sample each Gaussian in Fig. 4c, and insert the resulting parameters into the parametric model in Eq. (7), resulting in a new analytic vector field. Since the sampling process is computationally cheap, we can quickly synthesize thousands of vector field patches in order to train a convolutional neural network for reference frame extraction, as we describe next.

4. Deep Learning of Reference Frame Extraction

4.1. Overview

In this paper, we propose a convolutional neural network that spans two stages of the visualization pipeline: the filtering and the feature extraction. By combining both in an end-to-end fashion, the reference frame extractor can easily be conditioned to handle noisy inputs and data with resampling artifacts. During our supervised learning, we teach the network pairs of unsteady vector fields and their corresponding optimal reference frame transformation. These training examples are generated by transforming a steady vector field into an unsteady reference frame. By distorting the input data, the network learns to undo artifacts, which greatly improves robustness and helps in subsequent feature extraction tasks, such as the detection of vortex centers, which we demonstrate later. The following sections introduce the network architecture and the generation of the training data in more detail.

4.2. Architecture

We build our network as a form of typical CNN as seen in Fig. 5. The first part of our CNN consists of 3D convolutional kernels. In this part, the dimension reduction is performed for the feature extraction with a kernel size of $3\times3\times3$ and strides of $2\times2\times2$, followed by a batch normalization and a rectified linear unit (ReLU) layer. The size of the filter is doubled starting from 64 to maximally 1024. With input 2D unsteady fields $\mathbf{v}^*\in\mathbb{R}^{T\times H\times W\times 2}$, the number of convolutional layers is calculated by $n = log_2\max(H,W)-2$, and the output dimension of last convolutional layer would be $max(\frac{T}{2^n},1)\times\frac{H}{2^n}\times\frac{W}{2^n}\times min(64^n,1024)$. The second part of our network exploits fully connected layers instead of convolutional filters for the final inference from identified high-level vortex features. Batch normalization and a ReLU layer are followed same as convolutional layers, and a dropout with the probability of 0.1 is used to avoid overfitting. Note that we infer the first and second-order derivatives of the reference frame transformation $\dot{\mathbf{Q}},\ddot{\mathbf{Q}},\dot{\mathbf{c}},\ddot{\mathbf{c}}$ in 2D. Here we denote them as a 6-dimensional parameter vector $[\dot{\theta},\ddot{\theta},\dot{x},\dot{y},\ddot{x},\ddot{y}]$.

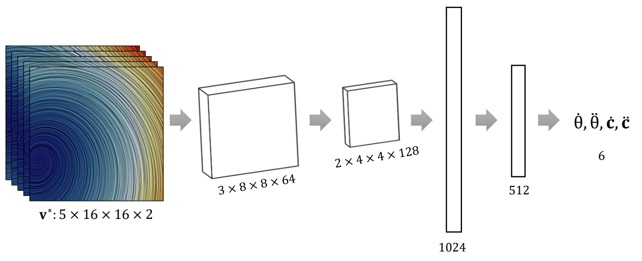

$\mathbf{v}^*: 5\times16\times16\times2$
$3\times8\times8\times64$
$2\times4\times4\times128$
1024
512
$\dot{\theta},\ddot{\theta},\dot{\mathbf{c}},\ddot{\mathbf{c}}$
6

Figure 5: *Our CNN Architecture. The numbers below each box represent the dimension of feature maps.*

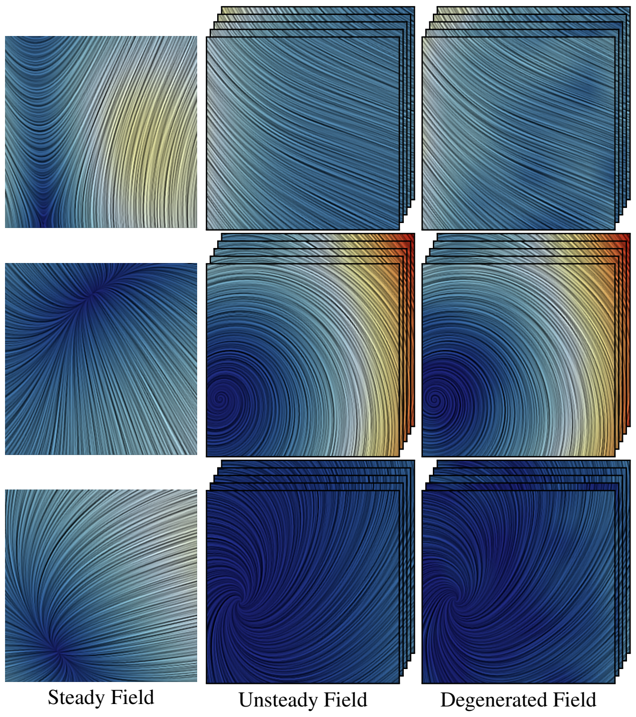


Figure 6: *Examples of training data pairs. From left to right, steady field, transformed unsteady field and distorted field. You can see some spots on magnitude plots of degenerated flows.*

4.3. Data Sets

We synthesize our data set based on Vatistas velocity profile as described in Section 3. Once a 2D steady field is generated by a super-position of three models with sampled parameters, we build unsteady vector fields by applying a reference frame transformation, followed by degrading with varying degrees of additive uniform noise (i.e., white noise) and down-up resampling. For simplicity we chose white noise during training, but we also test the network on a data set that was corrupted with Gaussian noise.

4.3.1. Training Data

As described in Section 2.2.2, we transform each 2D steady vector field to an unsteady 2D vector field. Note that parameters for the rotation and translation $[\dot{\theta}, \ddot{\theta}, \dot{x}, \dot{y}, \ddot{x}, \ddot{y}]$ are uniformly sampled. Then, we degrade it by adding uniform noise and down-up resampling, which is designed to simulate a general degeneration in a data acquisition system. The number of samples is 30,000 and the data set for training and testing is split into the ratio of 9:1. Some examples of the training data are illustrated in Fig. 6. We refer to Section C for details about each operation.

4.3.2. Numerical Data for Validation

After training, we have validated our neural network on the CYLINDER data set. As the input window size of our CNN is fixed, we have evaluated this data set by sliding a lookup window over the entire domain. In order to evaluate the CYLINDER flow, which has a resolution of $5 \times 80 \times 640$, we slide a lookup window of size $5 \times 16 \times 16$ with the stride 1 and sequentially predict parameters for the center pixel. (e.g., 159 steps for 40,625 windows with a batch size of 256).

Patch Normalization Since the training data set is defined in the 2D unit domain (i.e., $\bar{\mathcal{X}} \times \bar{\mathcal{Y}} \times \bar{\mathcal{T}} = [-1,1]^3$), input patches of numerical data sets are also scaled into this domain. Thus, given velocity patch slices $\mathbf{v}_p : \mathcal{X} \times \mathcal{Y} \times \mathcal{T} \to \mathcal{X} \times \mathcal{Y}$, defined in the domain $\mathcal{X} \times \mathcal{Y} \times \mathcal{T} = [x_{min}, x_{max}] \times [y_{min}, y_{max}] \times [t_{min}, t_{max}]$, we compute $\bar{\mathbf{v}}_p$ in the unit domain via:

$$\bar{\mathbf{v}}_p(\mathbf{x}) = \begin{bmatrix} \frac{t_{max}-t_{min}}{x_{max}-x_{min}} & 0 \\ 0 & \frac{t_{max}-t_{min}}{y_{max}-y_{min}} \end{bmatrix} \mathbf{v}_p(\mathbf{x}). \quad (9)$$

Furthermore, we globally normalize the magnitude of the training data to $[-1,1]$ during training. Thus, re-scaled patches from the validation data set are finally normalized by the same factors that have been applied to the training data before it was fed to the network.

4.4. Implementation

We implemented our convolutional neural network (CNN) using Keras [C*15] with Tensorflow [ABC*16] as backend. The networks are trained on each data set for 300 epochs using an Adam optimizer [KA15] with a learning rate of 0.001 and the mean squared error (MSE) loss function. We chose a batch size of 256. Finally, the model is selected, which shows the minimum test error during training. All visualizations were created with the visualization toolkit Amira [SWH05].

5. Result

In the following sections, we first apply our network to synthetic data generated with our parametric mixture model, before testing it on the reference frame extraction from unseen numerical data. Afterwards, the performance is discussed.

5.1. Training and Testing on Synthetic Data

As baseline, we compare our method with the linear reference frame optimization by Günther et al. [GGT17] on a synthetic data set, generated as described in Section 3.1.4 with a 1% noise-to-signal ratio in the normalized domain. We compute the mean-squared error on the inferred parameter vector of the test split, and our CNN shows not only visually more plausible results it also obtains up to two orders of magnitude lower time partial residuals than the linear optimization method, as shown in Fig. 7. Our method therefore passes the first test: outperform the baseline on synthetic flows similar to the ones seen during training.

5.2. Validation of Network on Numerical Data

Next, we apply our network to unseen numerical data to evaluate how well the network generalized, given its synthetic training data.

5.2.1. Robustness

By feeding a patch of voxels into the network, the CNN has enough information to learn proper smoothing and filtering kernels, such that noise and resampling artifacts can be compensated. To test the

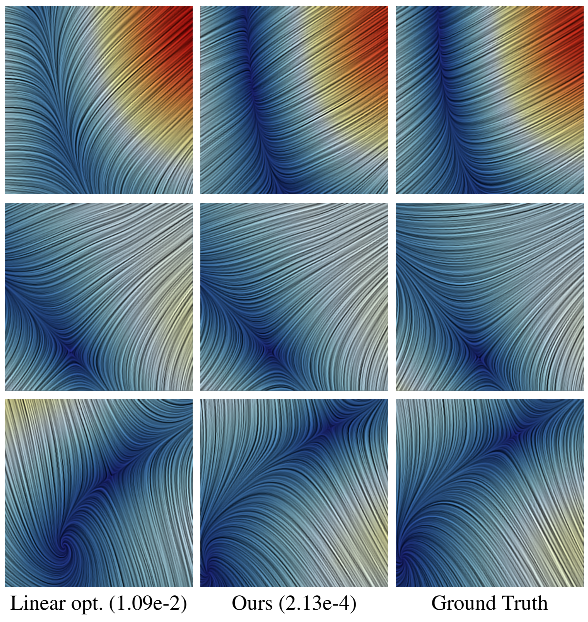


Figure 7: *Results of vortex extraction on test splits. The numbers in parenthesis are MSE, and our method robustly handles degenerated flows where linear optimization method fails.*

numerical robustness, we introduce varying degrees of noise and resampling artifacts into the unseen validation data, and compare the network output to the linear optimization.

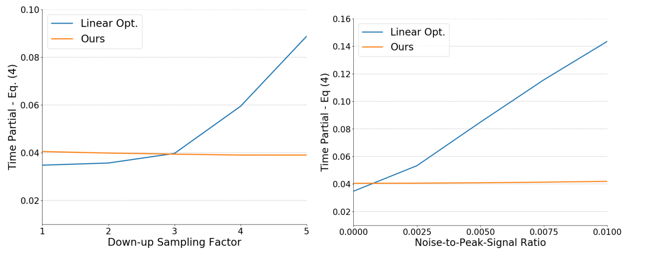
Error Metric. As error metric, we use the reference frame transformation obtained by the network or by the linear optimization, respectively, in order to transform the non-distorted unsteady vector field back into its steady frame. In the resulting frame, we calculate the time partial derivative, which ideally becomes zero, i.e.,

$$\mathbf{v}_t(\mathbf{x},t) = \mathbf{v}_t^*(\mathbf{x},t) + \dot{\mathbf{Q}}\mathbf{v}(\mathbf{x},t) + \ddot{\mathbf{Q}}\mathbf{x} + \ddot{\mathbf{c}} - [\mathbf{J}(\mathbf{x},t) + \dot{\mathbf{Q}}] \cdot [\dot{\mathbf{Q}}\mathbf{x} + \dot{\mathbf{c}}] \tag{10}$$

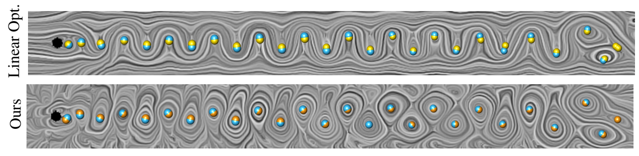
If $|\mathbf{v}_t| = 0$, the network successfully found the reference frame transformation, despite the presence of noise.

Quantitative Experiment. We use the CYLINDER flow as benchmark. In this vector field, the swirling strength of the vortices reduces over time, due to numerical dissipation and viscosity, which means that vortices become weaker. As the angular velocity magnitude decreases down the flow, the influence of the artificially added noise becomes stronger, causing errors in the linear optimization [GGT17] that are discussed later in Section 6 and can be seen in Fig. 1. In consequence, not all vortices are detected. Since our CNNs have seen resampling artifacts and noise during training, they robustly recover the reference frame in all areas of the domain even for different levels of degeneration as seen in Fig. 8.

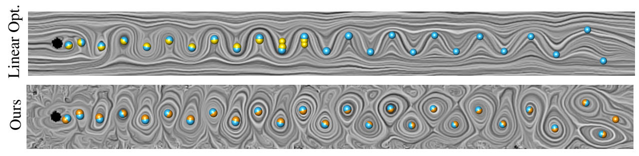
Noise Type Experiment. In Fig. 9, we applied our network (trained on 1% uniform noise and with resampling artifacts) to a data set with 1% of Gaussian noise. Since the results are very similar, our network also seems to handle this type of noise well. For higher noise magnitudes, differences will start to appear and then it will be better to train the network with Gaussian noise.



(a) *Plots of the time partial for comparison on different types of flow degeneration. The left plot shows the robustness of each method on different levels of resampling without noise, and the right plot shows the one on different noise levels without resampling.*



(b) *Results of the reference frame optimization on the resampled* CYLINDER *data set with the factor of 5.*



(c) *Results of the reference frame optimization on noise-added* CYLINDER *data set. The ratio of noise to the maximum magnitude of flow is 0.01 (1%).*

Figure 8: *Study of robustness for different levels and types of degeneration in the* CYLINDER *flow. Our method shows robust performances for various degradations. In this experiment, the network was trained for* 1% *noise and with resampling artifacts.*

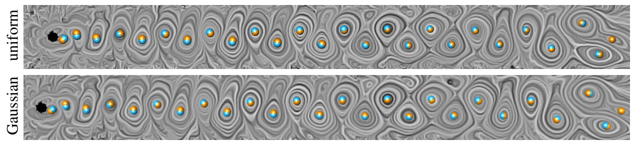
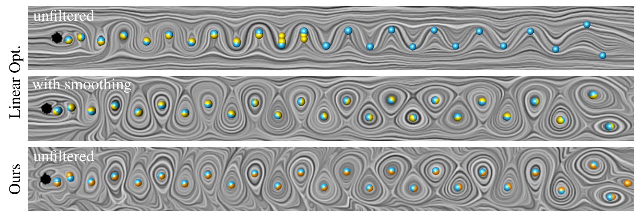
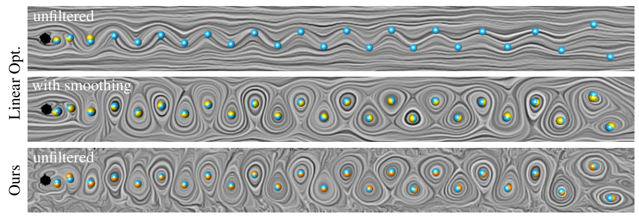


Figure 9: *Results of our CNN-based reference frame optimization on two different types of noise in the* CYLINDER *flow. During training only uniform noise was seen by the network (*1% *magnitude).*

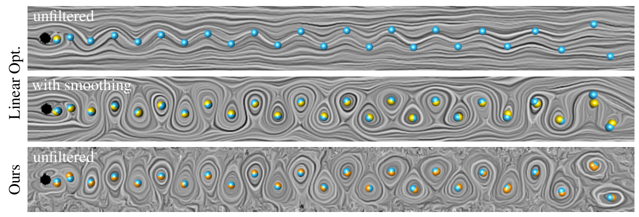
Noise Magnitude Experiment. In order to study the behavior of our network under a stronger influence of noise, we retrained the network and introduced for each patch up to 10% of white noise (uniformly sampled from 0% to 10%) and applied our CNN at testing time to data with 1%, 5% and 10% of noise. The results are shown in Fig. 10. In this comparison, we also tested how much prior Gaussian smoothing (3 iterations using a 7×7 filter kernel) helps the linear method, i.e., the preprocessing is done explicitly before the reference frame extraction. For small noise magnitudes, smoothing helps, however, smoothing also affects the position of the vortex core. Especially for large noise ratios, the CNN-based method recovers the reference frame better. Without prior smoothing the linear method removes almost no ambient motion, keeping the dominant downstream ambient motion of the input flow.



(a) *Results of the reference frame optimization on noise-added* CYLINDER *data set. The ratio of noise to the maximum magnitude of flow is 0.01 (1%).*



(b) *Results of the reference frame optimization on noise-added* CYLINDER *data set. The ratio of noise to the maximum magnitude of flow is 0.05 (5%).*



(c) *Results of the reference frame optimization on noise-added* CYLINDER *data set. The ratio of noise to the maximum magnitude of flow is 0.1 (10%).*

Figure 10: *Comparison of robustness of vortex extraction methods on different levels and types of degeneration on* CYLINDER *flow. Our method shows robust performances for various degradation, although it has not seen them during training. For the noise experiments, we also show results of the linear method [GGT17] after applying Gaussian smoothing to the noisy input vector field.*

5.2.2. Temporal Coherence

To demonstrate the temporal consistency of our results, we show the paths of vortex cores in space-time in Fig. 11. While the vortex corelines resulting from the linear optimization are disconnected and partially missing on noisy data, our method finely extracts the corelines, which are almost identical to the results of the linear optimization on the original data.

5.3. Performance

In the following, we provide timing measurements of our CNN-based approach, compared to the linear optimization [GGT17]. All measurements were taken with an 8GB NVIDIA GTX 1080 GPU and an Intel i7-6700K CPU at 4.00 GHz with 32 GB memory. Generating the 30,000 training patches took in total about 37 seconds (24 seconds to compute the unsteady flow patches and 13 seconds to compute the steady ground truth). The training takes roughly 10 minutes and the size of the resulting network model is 20.9 MB in HDF5 file format. Our method takes 22ns on average for a feed-forward evaluation of a single patch (5.63ms per batch). Our method takes a constant evaluation time for a single batch. However, the batch size is limited by the GPU memory, and it makes our computation time linear in the number of local patches over the whole domain. For instance, in the CYLINDER data set a slice batch with resolution $5 \times 80 \times 640$, takes about 0.893s, while the linear optimization method takes 0.366s. Note, however, that our local patch estimation is scalable to the number of available GPUs.

6. Discussion

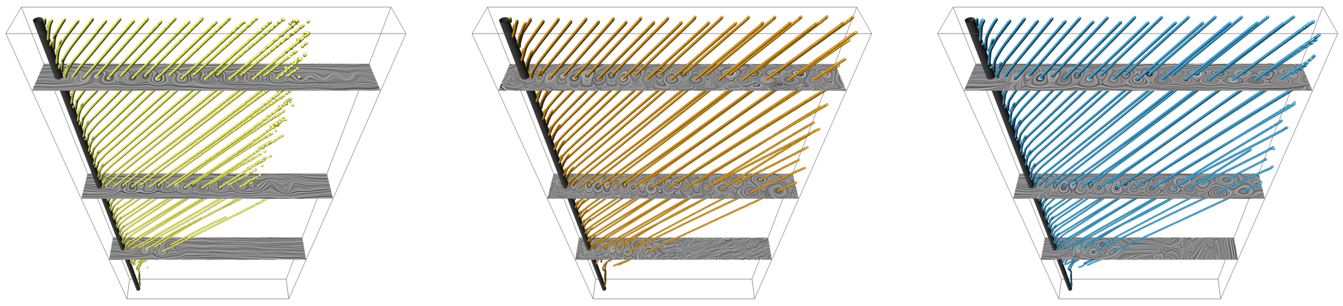
Limitations. The generalization capacity of a supervised learning approach is always limited to the data that was seen during training. At present, our network has not seen obstacles or boundary data. In the future, we plan to expand the training data to also include wall velocity profiles. As shown in Appendix B, our parametric mixture model is not yet expressive enough to approximate turbulent and small-scale structures accurately. The optimal choice of model parameters m also depends on the patch size, which we currently assume to be constant. In the future, we would like to explore scale-space approaches that look for optimal frames at different patch sizes. Since the time partial residual can be evaluated, the best patch size could be selected automatically. Further, we concentrated on 2D time-dependent data. A natural next step is the extension to 3D.

Choice of Basis. There are several other options to form the parametric vector field model than the approach we described in Section 3.1.2. Standard basis functions would be imaginable, such as monomial, Chebyshev and Fourier basis functions. While those options have an orthogonal basis, our primitives have a strong prior, since they are derived from models that were fit to experimental data [VKM91]. In the context of vector field design, radial basis functions have been used to combine primitives [ZMT06], which gives more intuitive results than adding them up. Whether this is closer to observational data or whether it helps a CNN to pick up the flow patterns remains to be explored in future work.

Noise in Linear Method. Our experiments have shown that noise can have a significant impact on the linear method [GGT17]. Even though, the linear method fits a reference frame transformation to a spatial neighborhood (which has a spatial smoothing effect), there is no smoothing over time. Further, the second-order accurate finite differences are particularly prone to noise. Adding noise of only 1% to $\mathbf{v}$ in the CYLINDER flow already increased the noise level in the time partial $\mathbf{v}_t$ by 14%. This increase is also dependent on the temporal resolution, since for higher grid resolution the noise in the time domain is more high-frequent. We found in Fig. 8 that smoothing can help to some extent. Our CNN-based approach removes the noise implicitly. The global optimization of Hadwiger et al. [HMTR19] also requires the time partial $\mathbf{v}_t$. It remains to be tested, whether a high noise level on $\mathbf{v}_t$ also influences their result.

7. Conclusion

In this paper, we developed a convolutional neural network that extracts the reference frame in which a given unsteady 2D vector field becomes steady. In such a reference frame, features such as vortices are no longer hidden by ambient motion. To increase the robustness to noise and resampling artifacts, we trained our network on distorted inputs, which significantly improves the quality over a



(a) *Linear optimization on noisy data.* **(b)** *Our CNN-based approach on noisy data.* **(c)** *Linear optimization on original data.*

Figure 11: *Comparison of vortex corelines in space-time in the* CYLINDER *flow. Despite the noise and resampling artifacts our CNN recovers the reference frame well, so that vortex corelines can be extracted as cleanly as with the linear optimization [GGT17] on the original data.*

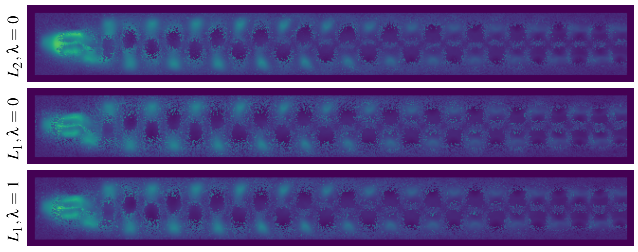


Figure 12: *Heat maps of the fitting residual for different vector field distance functions. The mean relative distances are 0.2028, 0.1947 and 0.1906, from top to bottom.*

linear optimization. In order to generate the required training data, i.e., patches of vector fields, we developed a parametric vector field mixture model that is based on Vatistas' experimentally obtained vortex velocity profile. To parameterize the model, we fitted it to numerical data and afterwards sampled thousands of training data sets. Our work shows the great potential of deep learning for end-to-end combinations of preprocessing and feature extraction.

In the future, we would like to increase the generality of our method, for instance by training on other classes of reference frame invariances, such as affine invariance. We would like to incorporate further training data to obtain more generality, such as for modeling obstacles and boundaries. Finally, we plan to further improve our parametric mixture model by studying other basis functions and by automatically selecting the patch size in a scale-space approach.

Acknowledgements. This work was supported by the Swiss National Science Foundation Ambizione grant no. PZ00P2_180114.

Appendix A: Comparison of Distance Metrics

For the fitting of our parametric mixture model to numerical data, we used the distance metric of Kim et al. [KCAT*18], cf. Eq. (8). Following their recommendation, we chose the L_1 distance with the gradient weight $\lambda = 1$. Fig. 12 presents fitting results for other parameter choices, showing that the chosen distance metric is best.

Appendix B: Further Fitting Results

In addition to the CYLINDER flow, we fitted our parametric mixture model to a more turbulent BOUSSINESQ flow. This data set contains

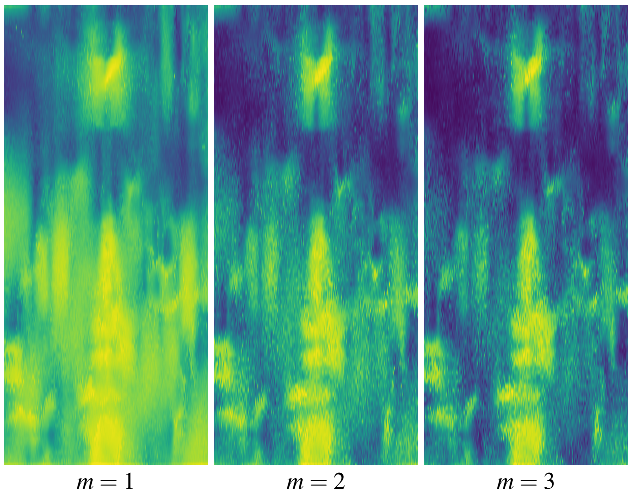


Figure 13: *Heat maps of the fitting residual in the* BOUSSINESQ *data set for varying number of mixture model components m.*

smaller structures that are more difficult to fit with only $m = 3$ mixture model components. From the results in Fig. 13, we conclude that the optimal choice of m is data-dependent.

Appendix C: Parameters for data synthesis

Table 1 lists the parameters of our reference frame transformations. The domain is scaled to the unit box, and the resolution is set to $5 \times 16 \times 16$ both for computational efficiency and since objective reference frame transformations are local operations. In fact, the temporal derivative to minimize only depends on up to second-order derivatives [GGT17]. Given the unit domain and its resolution, parameters for the transformation are set at a moderate level.

PARAMETER	DESCRIPTION	VALUE
Ω	Domain	$[-1,1] \times [-1,1] \times [-1,1]$
T, H, W	Resolution of the domain Ω (time, height and width)	$5 \times 16 \times 16$
$\dot{\theta}$	Range of the first-order derivative of the rotation $\dot{\mathbf{Q}}$	$[-0.3, 0.3]$
$\ddot{\theta}$	Range of the second-order derivative of the rotation $\ddot{\mathbf{Q}}$	$[-0.01, 0.01]$
$\dot{x}, \dot{y}$	Range of the first-order derivative of the translation $\dot{\mathbf{c}}$	$[-0.3, 0.3] \times [-0.3, 0.3]$
$\ddot{x}, \ddot{y}$	Range of the second-order derivative of the translation $\ddot{\mathbf{c}}$	$[-0.01, 0.01] \times [-0.01, 0.01]$
α	Range of uniform noise	$[-0.01, 0.01]$
β	Down-up resampling factor	0.5

Table 1: *Common parameters for data synthesis and analysis.*

References

[ABC*16] ABADI M., BARHAM P., CHEN J., CHEN Z., DAVIS A., DEAN J., DEVIN M., GHEMAWAT S., IRVING G., ISARD M., ET AL.: Tensorflow: a system for large-scale machine learning. In *OSDI* (2016), vol. 16, pp. 265–283. 6

[ACOG18] ANCONA M., CEOLINI E., OZTIRELI C., GROSS M.: Towards better understanding of gradient-based attribution methods for deep neural networks. In *6th International Conference on Learning Representations (ICLR 2018)* (2018). 2

[Ast79] ASTARITA G.: Objective and generally applicable criteria for flow classification. *Journal of Non-Newtonian Fluid Mechanics 6*, 1 (1979), 69–76. 1, 3

[BHJ16] BUJACK R., HLAWITSCHKA M., JOY K. I.: Topology-inspired galilean invariant vector field analysis. In *2016 IEEE Pacific Visualization Symposium (PacificVis)* (2016), IEEE, pp. 72–79. 3

[BL00] BHAGWAT M. J., LEISHMAN J. G.: Correlation of helicopter rotor tip vortex measurements. *AIAA journal 38*, 2 (2000), 301–308. 4

[BLL17] BERGER M., LI J., LEVINE J. A.: A generative model for volume rendering. *arXiv preprint arXiv:1710.09545* (2017). 2

[BPKB14] BHATIA H., PASCUCCI V., KIRBY R. M., BREMER P.-T.: Extracting features from time-dependent vector fields using internal reference frames. *Computer Graphics Forum (Proc. EuroVis) 33*, 3 (2014), 21–30. 1, 3

[BY18] BIN T., YI L.: Cnn-based flow field feature visualization method. *International Journal of Performability Engineering 14*, 3 (2018), 434. 2

[C*15] CHOLLET F., ET AL.: Keras: Deep learning library for theano and tensorflow. *URL: https://keras. io/k 7*, 8 (2015). 6

[DL76] DROUOT R., LUCIUS M.: Approximation du second ordre de la loi de comportement des fluides simples. lois classiques déduites de lâĂŹintroduction dâĂŹun nouveau tenseur objectif. *Archiwum Mechaniki Stosowanej 28*, 2 (1976), 189–198. 3

[DWL*18] DENG L., WANG Y., LIU Y., WANG F., LI S., LIU J.: A cnn-based vortex identification method. *Journal of Visualization* (Oct 2018). 2

[FH18] FAN C., HAUSER H.: Fast and accurate CNN-based brushing in scatterplots. *Computer Graphics Forum 37*, 3 (2018), 111–120. 2

[Fre17] FREY S.: Sampling and estimation of pairwise similarity in spatio-temporal data based on neural networks. In *Informatics* (2017), vol. 4, Multidisciplinary Digital Publishing Institute, p. 27. 2

[FRM*18] FRANZ K., ROSCHER R., MILIOTO A., WENZEL S., KUSCHE J.: Ocean eddy identification and tracking using neural networks. In *Poster at IEEE International Geoscience and Remote Sensing Symposium* (2018). 2

[GBCB16] GOODFELLOW I., BENGIO Y., COURVILLE A., BENGIO Y.: *Deep learning*, vol. 1. MIT press Cambridge, 2016. 2

[GGT17] GÜNTHER T., GROSS M., THEISEL H.: Generic objective vortices for flow visualization. *ACM Transactions on Graphics (Proc. SIGGRAPH) 36*, 4 (2017), 141:1–141:11. 1, 2, 3, 4, 6, 7, 8, 9

[GLL91] GLOBUS A., LEVIT C., LASINSKI T.: A tool for visualizing the topology of three-dimensional vector fields. In *Proc. IEEE Visualization* (1991), pp. 33–40. 2

[GLT*07] GARTH C., LARAMEE R. S., TRICOCHE X., SCHNEIDER J., HAGEN H.: Extraction and visualization of swirl and tumble motion from engine simulation data. In *Topology-based Methods in Visualization*, Visualization and Mathematics. Springer Berlin Heidelberg, 2007, pp. 121–135. 1

[GPAM*14] GOODFELLOW I., POUGET-ABADIE J., MIRZA M., XU B., WARDE-FARLEY D., OZAIR S., COURVILLE A., BENGIO Y.: Generative adversarial nets. In *Advances in neural information processing systems* (2014), pp. 2672–2680. 3

[GT18] GÜNTHER T., THEISEL H.: The state of the art in vortex extraction. *Computer Graphics Forum 37*, 6 (2018), 149–173. 1, 2

[GT19a] GÜNTHER T., THEISEL H.: Hyper-objective vortices. *IEEE Transactions on Visualization and Computer Graphics* (2019), to appear. 2, 3

[GT19b] GÜNTHER T., THEISEL H.: Objective vortex corelines of finite-sized objects in fluid flows. *IEEE Transactions on Visualization and Computer Graphics (Proc. IEEE Scientific Visualization 2018) 25*, 1 (2019), 956–966. 2

[Hal05] HALLER G.: An objective definition of a vortex. *Journal of Fluid Mechanics 525* (2005), 1–26. 1, 3

[Hal15] HALLER G.: Lagrangian coherent structures. *Annual Review of Fluid Mechanics 47* (2015), 137–162. 3

[HH16] HADJIGHASEM A., HALLER G.: Geodesic transport barriers in jupiter's atmosphere: A video-based analysis. *SIAM Review 58*, 1 (2016), 69–89. 1

[HHFH16] HALLER G., HADJIGHASEM A., FARAZMAND M., HUHN F.: Defining coherent vortices objectively from the vorticity. *Journal of Fluid Mechanics 795* (2016), 136–173. 1, 2, 3

[HKPC18] HOHMAN F. M., KAHNG M., PIENTA R., CHAU D. H.: Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics* (2018), 1–1. 2

[HMTR19] HADWIGER M., MLEJNEK M., THEUSSL T., RAUTEK P.: Time-dependent flow seen through approximate observer killing fields. *IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE Scientific Visualization 2018) 25*, 1 (2019), 1257–1266. 2, 3, 8

[HTW18] HAN J., TAO J., WANG C.: Flownet: A deep learning framework for clustering and selection of streamlines and stream surfaces. *IEEE Transactions on Visualization and Computer Graphics* (2018), to appear. 2

[Hun87] HUNT J. C. R.: Vorticity and vortex dynamics in complex turbulent flows. *Transactions on Canadian Society for Mechanical Engineering (Proc. CANCAM) 11*, 1 (1987), 21–35. 1

[JH95] JEONG J., HUSSAIN F.: On the identification of a vortex. *Journal of Fluid Mechanics 285* (1995), 69–94. 1

[KA15] KINGMA D., ADAM J. B.: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)* (2015), vol. 5. 6

[Kau62] KAUFMANN W.: Über die Ausbreitung kreiszylindrischer Wirbel in zähen (viskosen) Flüssigkeiten. *Ingenieur-Archiv 31*, 1 (1962), 1–9. 4

[KCAT*18] KIM B., C. AZEVEDO V., THUEREY N., KIM T., GROSS M., SOLENTHALER B.: Deep Fluids: A Generative Network for Parameterized Fluid Simulations. *arXiv preprint arXiv:1806.02071* (2018). 4, 9

[KGP*13] KÖHLER B., GASTEIGER R., PREIM U., THEISEL H., GUTBERLET M., PREIM B.: Semi-automatic vortex extraction in 4D PC-MRI cardiac blood flow data using line predicates. *IEEE Transactions on Visualization and Computer Graphics (Proc. IEEE SciVis) 19*, 12 (2013), 2773–2782. 1

[KW13] KINGMA D. P., WELLING M.: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013). 3

[LSF*17] LGUENSAT R., SUN M., FABLET R., MASON E., TANDEO P., CHEN G.: Eddynet: A deep neural network for pixel-wise classification of oceanic eddies. *CoRR abs/1711.03954* (2017). 2

[Lug79] LUGT H. J.: The dilemma of defining a vortex. In *Recent developments in theoretical and experimental fluid mechanics*. Springer, 1979, pp. 309–321. 1, 3

[OJCJP16] OELTZE-JAFRA S., CEBRAL J. R., JANIGA G., PREIM B.: Cluster analysis of vortical flow in simulations of cerebral aneurysm hemodynamics. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (Jan 2016), 757–766. 1

[Oku70] OKUBO A.: Horizontal dispersion of floatable particles in the vicinity of velocity singularities such as convergences. *Deep Sea Research and Oceanographic Abstracts 17*, 3 (1970), 445–454. 1, 2

[Ose12] OSEEN C.: Über die wirbelbewegung in einer reibenden flüssigkeit. *Ark. Mat. Astro. Fys. 7* (1912). 4

[PC94] PERRY A. E., CHONG M. S.: Topology of flow patterns in vortex motions and turbulence. *Applied Scientific Research 53*, 3 (1994), 357–374. 3

[PR99] PEIKERT R., ROTH M.: The "parallel vectors" operator – a vector field visualization primitive. In *Proc. IEEE Visualization* (1999), pp. 263–270. 2

[Rob91] ROBINSON S. K.: Coherent motions in the turbulent boundary layer. *Annual Review of Fluid Mechanics 23*, 1 (1991), 601–639. 1, 3

[RP96] ROTH M., PEIKERT R.: Flow visualization for turbomachinery design. In *Proceedings of the 7th Conference on Visualization '96* (Los Alamitos, CA, USA, 1996), VIS '96, IEEE Computer Society Press, pp. 381–384. 1

[SAB*17] SEIFERT C., AAMIR A., BALAGOPALAN A., JAIN D., SHARMA A., GROTTEL S., GUMHOLD S.: *Visualizations of Deep Neural Networks in Computer Vision: A Survey*. Springer International Publishing, Cham, 2017, pp. 123–144. 2

[SWH05] STALLING D., WESTERHOFF M., HEGE H.-C.: Amira: A highly interactive system for visual data analysis. In *The Visualization Handbook*. Elsevier, 2005, pp. 749–767. 6

[SWXP18] STRÖFER C. M., WU J., XIAO H., PATERSON E.: Data-driven, physics-based feature extraction from fluid flow fields. *arXiv preprint arXiv:1802.00775* (2018). 2

[TK94] TABOR M., KLAPPER I.: Stretching and alignment in chaotic and turbulent flows. *Chaos, Solitons & Fractals 4*, 6 (1994), 1031–1055. 3

[TN65] TRUESDELL C., NOLL W.: *The nonlinear field theories of mechanics*. Handbuch der Physik, Band III/3, e by Flugge, S., (ed.) , Springer-Verlag, Berlin, 1965. 2

[VKM91] VATISTAS G. H., KOZEL V., MIH W.: A simpler model for concentrated vortices. *Experiments in Fluids 11*, 1 (1991), 73–76. 2, 3, 4, 8

[Wei91] WEISS J.: The dynamics of enstrophy transfer in two-dimensional hydrodynamics. *Physica D: Nonlinear Phenomena 48*, 2-3 (1991), 273–294. 2

[WGS07] WIEBEL A., GARTH C., SCHEUERMANN G.: Computation of localized flow for steady and unsteady vector fields and its applications. *IEEE Transactions on Visualization and Computer Graphics 13*, 4 (2007), 641. 3

[Wie04] WIEBEL A.: *Feature Detection in Vector Fields Using the Helmholtz-Hodge Decomposition*. Diploma thesis, Univ. Kaiserslautern, 2004. 3

[WSTH07] WEINKAUF T., SAHNER J., THEISEL H., HEGE H.-C.: Cores of swirling particle motion in unsteady flows. *IEEE Transactions on Visualization and Computer Graphics (Proc. Visualization) 13*, 6 (2007), 1759–1766. 1

[ZMT06] ZHANG E., MISCHAIKOW K., TURK G.: Vector field design on surfaces. *ACM Transactions on Graphics (ToG) 25*, 4 (2006), 1294–1326. 8