STARVORTEX: SELF-SUPERVISED GEOMETRIC REPRESENTATION LEARNING OF COLLECTIVE INTEGRAL CURVES FOR LAGRANGIAN VORTICES

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning useful representations of dynamical systems from geometry alone is challenging: individual trajectories provide limited information, while the relevant structure often lies in the collective deformation of neighboring trajectories and should be insensitive to arbitrary spatial orientation. We study this problem in fluid flows and introduce StarVortex, a self-supervised framework for learning geometric representations directly from local bundles of integral curves. We represent each neighborhood as a star consisting of a center curve and symmetrically seeded neighbors, exposing both center-curve transport and relative bundle deformation. Rather than requiring a fully rotation-equivariant architecture, StarVortex combines an exact architectural bias induced by the discrete symmetry of the sampling stencil with a representation-level objective that enforces consistency under arbitrary continuous rotations. Reconstruction and variance–covariance regularization further encourage the latent space to retain informative geometric variation. The resulting representation captures vortex-relevant structure without velocity, vorticity, physics-derived descriptors, or training labels. It supports label-free discovery of vortex regions and provides transferable features for supervised recognition of vortex regions, corelines, and hairpin vortices across pathlines, streamlines, and vortex lines. Experiments and ablations show that explicitly structuring self-supervision around geometric symmetry is critical for learning useful representations, and that the learned features are particularly beneficial when labels are scarce or evaluation shifts beyond the training domain.

1 INTRODUCTION

Vortices organize mixing, transport, and separation in fluid flows, making their identification important to fluid dynamics and scientific visualization. Although a vortex reflects the coordinated motion of neighboring particles, many detectors operate on Eulerian velocity fields. Classical criteria, including Q and λ_2, derive local measures from velocity gradients (Hunt et al., 1988; Jeong & Hussain, 1995). Learning-based detectors likewise use field quantities, either directly or alongside trajectories (Table 1).

For large simulations, storing time-resolved velocity grids is expensive, while reconstructing trajectories from temporally sparse snapshots introduces integration error (Hummel et al., 2016). An alternative is to integrate tracer particles during simulation and store their trajectories (Agranovsky et al., 2014; Bujack & Joy, 2015). Such Lagrangian representations can improve trajectory reconstruction at comparable storage (Sane et al., 2018). In production applications, they have reduced storage to less than 1% of the full velocity-field output (Sane et al., 2022). Experiments with Lagrangian particle tracking also provide trajectories directly (Khojasteh et al., 2021).

Prior work uses these data to reconstruct flow fields and trajectories (Chandler et al., 2015; Li et al., 2022; Han et al., 2022; 2019), and learned trajectory embeddings can reveal Lagrangian coherent structures (MacMillan & Richter, 2021). Our focus is local recognition of vortex regions, corelines, and hairpin vortices from neighboring curves. This requires representations that capture collective geometric structure while remaining insensitive to the spatial orientation of the bundle.

We present StarVortex, which represents each neighborhood as a *star*: a center integral curve surrounded by symmetrically seeded neighbors. A convolution–Transformer encodes sampled positions through signals of center-curve transport and relative bundle deformation. Rotating the neigh-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: **Comparison of deep learning approaches for vortex extraction.** Inputs and supervision refer to the published methods. Our method learns from integral-curve geometry and supports self-supervised training with no labels.

Method	VortexNet Deng et al. (2019)	MVU-Net Deng et al. (2022b)	VortexBoundary Berenjkoub et al. (2020)	VortexViz de Silva et al. (2024)	VortexTransformer Zhang et al. (2025)	**Ours** —
Category	Non-Geometric			Partial-Geometric		**Pure-Geometric**
Data representation	Eulerian velocity-grid			Hybrid		Lagrangian
Input	Patch v	Patch v+ω+IVD	Patch v	Pathline image + $\lvert\omega\rvert$	**Pathline**+IVD	**Integral Curve**
Model	CNN	UNet	UNet	CNN2D+MLP	Transformer	Transformer
native 3D support	✓	✓	✗	✗	✗	✓
label-free learning	✗	✗	✗	✗	✗	✓

bor channel processing encodes the stencil's discrete permutation symmetry, while self-supervised consistency encourages agreement across rotated views. Reconstruction and representation regularization preserve informative geometric variation. The learned features support clustering without training labels and classification with labels. The same representation applies to pathlines, streamlines, and vortex lines.

Our contributions are:

- We introduce a **pure geometric** representation for Lagrangian flow and vortex analysis that operates directly on bundles of sampled, interacting integral curves—pathlines or streamlines—rather than on Eulerian field quantities or hand-crafted physical descriptors. Our integral-curve star representation captures both center-curve transport and the relative deformation of neighboring trajectories. Despite using a simple convolution–Transformer encoder, our method outperforms existing supervised Eulerian non-geometric and hybrid approaches on vortex extraction under matched experimental settings.
- We introduce **self-supervised representation learning for vortex extraction**, built around a symmetry-structured integral-curve stencil. Our formulation combines exact finite-stencil symmetry through parameter sharing with continuous-rotation consistency, reconstruction, and representation regularization, enabling the model to learn vortex-relevant geometric features without prescribing full rotation-equivariant network layers. The resulting approach extracts vortices **without physics-derived ground-truth labels or property inputs**, e.g., IVD (instantaneous vorticity deviation). This further enables fully autonomous vortex extraction directly from particle trajectory flows in settings where IVD cannot be evaluated (Fig. 5 and the supplementary video).
- We further show that the proposed self-supervised objectives are useful beyond label-free vortex extraction: when incorporated as auxiliary objectives in supervised learning, they improve vortex recognition tasks, including coreline extraction and hairpin extraction, with particularly pronounced benefits in **limited-data and domain-shift regimes**.

2 Related Work

Vortex extraction encompasses local vortex criteria (Hunt et al., 1988; Jeong & Hussain, 1995; Chong et al., 1990), reference-frame methods (Günther et al., 2017; Hadwiger et al., 2019), and geometric or topological analysis (Sadarjoen & Post, 1999; Bremer et al., 2016; Zafar et al., 2024a). Learning-based detectors use velocity grids (Deng et al., 2019; 2022b; Berenjkoub et al., 2020) or combine pathlines with vorticity-derived quantities (de Silva et al., 2024; Zhang et al., 2025). Table 1 summarizes their inputs and supervision.

Lagrangian and curve representations support flow reconstruction (Sane et al., 2018; 2022; Han et al., 2019), streamline organization and exploration (Yu et al., 2012; Rössl & Theisel, 2012; Han et al., 2020), and coherent-structure analysis (Haller & Yuan, 2000; Haller, 2015; MacMillan & Richter, 2021). Related learning methods model particle dynamics (Sanchez-Gonzalez et al., 2020; Ummenhofer et al., 2020) or encode trajectories and tractography streamlines (Gao et al., 2020; Li et al., 2018; Chen et al., 2023). We focus on recognizing vortex regions, corelines, and hairpin vortices from local curve bundles, with rotation symmetry supporting self-supervised representation learning. Appendix B provides an extended review of these research directions.

3 STARVORTEX: SYMMETRY-STRUCTURED ENCODING OF INTEGRAL-CURVE STARS

StarVortex represents local flow geometry by a *star*, consisting of a center integral curve and neighboring curves seeded on a symmetric stencil. The encoder combines two forms of symmetry: parameter sharing exploits the exact finite symmetry of the stencil, while a representation-level consistency objective learns invariance to continuous spatial rotations. This avoids requiring the full convolution–Transformer encoder to be $\mathrm{SO}(d)$-equivariant.

3.1 INTEGRAL CURVES

Let $\mathbf{v}(\mathbf{x}, t)$ be a time-dependent velocity field on a domain $D \subset \mathbb{R}^d$, $d \in \{2, 3\}$. All curves we use are *integral curves*, solutions of

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s} = \mathbf{f}\big(\mathbf{x}(s), s\big), \qquad \mathbf{x}(s_0) = \mathbf{x}_0, \tag{1}$$

and differ only in the choice of field $\mathbf{f}$. **Streamlines** use the instantaneous velocity at a fixed time t, $\mathbf{f}(\mathbf{x}, s) = \mathbf{v}(\mathbf{x}, t)$, so s is not physical time; in steady flow they coincide with pathlines. **Pathlines** use $\mathbf{f}(\mathbf{x}, s) = \mathbf{v}(\mathbf{x}, s)$, so s is physical time and the curve is the trajectory of a material particle; they are the only curves that describe material transport in unsteady flow. **Vortex lines** use the vorticity at a fixed time t, $\mathbf{f}(\mathbf{x}, s) = \boldsymbol{\omega}(\mathbf{x}, t)$ with $\boldsymbol{\omega} = \nabla \times \mathbf{v}$, and are defined in 3D only, where the vorticity is a vector.

3.2 STARS OF INTEGRAL CURVES

Construction. Let $G \subset \mathrm{O}(d)$ be a finite group and $\{\mathbf{d}_j\}_{j=1}^K \subset \mathbb{S}^{d-1}$ one of its orbits. Each $g \in G$ induces a permutation σ_g such that

$$g^{-1}\mathbf{d}_j = \mathbf{d}_{\sigma_g(j)}. \tag{2}$$

A star of radius r centered at $\mathbf{c}$ consists of the integral curve $\mathbf{x}_0$ seeded at $\mathbf{c}$ and K neighboring curves $\mathbf{x}_j$ seeded at $\mathbf{c} + r\mathbf{d}_j$. Sampling at $s_1 < \cdots < s_T$ gives

$$\mathbf{X} = \mathcal{T}_{\mathbf{v}}(\mathbf{c}) \in \mathbb{R}^{(K+1)\times T\times d}. \tag{3}$$

We use a regular octagon in 2D and a regular icosahedron in 3D, with full stencil symmetry groups D_8 and I_h, respectively (Table 2). Reflections are included because the target structures do not depend on handedness. Icosahedral discretizations have also been used to represent rotational structure in equivariant models (Cohen et al., 2019; Esteves et al., 2019).

Exact stencil action. A stencil symmetry transforms the spatial vectors and simultaneously relabels the neighboring curves:

$$[\rho(g)\mathbf{X}]_0 = g\mathbf{X}_0, \qquad [\rho(g)\mathbf{X}]_j = g\mathbf{X}_{\sigma_g(j)}, \quad j = 1, \ldots, K. \tag{4}$$

Thus $\rho(g)$ consists of a permutation of neighbor indices together with the corresponding transformation of Cartesian vectors. This is an exact symmetry of the star construction rather than an augmentation heuristic.

Proposition 1 (Exact stencil views) *For* $(g\cdot\mathbf{v})(\mathbf{x}, t) = g\,\mathbf{v}(g^{-1}\mathbf{x}, t)$,

$$\mathcal{T}_{g\cdot\mathbf{v}}(g\mathbf{c}) = \rho(g)\mathcal{T}_{\mathbf{v}}(\mathbf{c}), \qquad g \in G. \tag{5}$$

Proof. If $\mathbf{x}$ is an integral curve of $\mathbf{v}$, then $g\mathbf{x}$ is an integral curve of $g\cdot\mathbf{v}$. Moreover, $g(\mathbf{c} + r\mathbf{d}_{\sigma_g(j)}) = g\mathbf{c} + r\mathbf{d}_j$, so neighbor $\sigma_g(j)$ becomes neighbor j of the transformed star. □

The same identity holds for our fixed-step Runge–Kutta integration, so group-related views can be obtained from a stored star without re-tracing.

Input signal. For the center curve, we use its discrete first derivative with respect to integration time; for each neighboring curve, we use the discrete first derivative of its displacement relative to the center curve. This removes global translation while retaining center transport and relative bundle deformation. The resulting signal $\mathbf{S}$ transforms covariantly under the stencil action, $\mathbf{S} \mapsto \rho(g)\mathbf{S}$.

3.3 HARD SYMMETRY BIAS: SYMMETRY-TIED STAR ENCODER

The encoder Enc_θ reads $\mathbf{S}$ as a sequence of length T using two residual 1D-convolution stages followed by Transformer layers, and produces $\mathbf{z} = [\mathbf{z}_{\mathrm{inv}} \mid \mathbf{z}_{\mathrm{pose}}]$. Downstream classification and clustering use $\mathbf{z}_{\mathrm{inv}}$; the full code is used when reconstruction is required.

Symmetry-tied neighbor processing. The stencil determines exactly which neighboring channels are equivalent under G. We therefore apply the same neighbor-processing map ϕ_θ to every neighbor rather than learning a separate orientation-specific map for each slot.

To distinguish channel permutation from the full spatial action, let P_g denote only the permutation of neighbor indices induced by σ_g. For the neighbor-wise map $\Phi_\theta(\mathbf{S}) = [\phi_\theta(\mathbf{S}_1), \ldots, \phi_\theta(\mathbf{S}_K)]$, parameter sharing gives

$$\Phi_\theta(P_g\mathbf{S}) = P_g\Phi_\theta(\mathbf{S}), \qquad g \in G. \tag{6}$$

Thus the neighbor-index pathway is exactly equivariant to the known discrete permutation. This hard bias prevents symmetry-related slots from learning independent functions and removes unnecessary orientation-specific degrees of freedom.

Importantly, P_g describes only channel relabeling, whereas the physical action $\rho(g)$ in Eq. equation 4 also transforms the Cartesian vector components. The generic convolution–Transformer backbone is not constrained to intertwine this vector transformation with a prescribed feature-space representation. Hence the complete encoder is not claimed to be rotation equivariant; continuous orientation invariance is instead imposed on the learned task representation.

3.4 SOFT SYMMETRY CONSTRAINT: ROTATION-INVARIANT CODES

For each star, we sample a stencil symmetry $g \in G$ and an independent continuous rotation $Q \sim \mathrm{Haar}(\mathrm{SO}(d))$. Let $\kappa(Q)$ rotate only the Cartesian vector components,

$$[\kappa(Q)\mathbf{S}]_j(k) = Q\mathbf{S}_j(k), \tag{7}$$

and construct one transformed view

$$\widetilde{\mathbf{S}}_{g,Q} = \kappa(Q)\rho(g)\mathbf{S}. \tag{8}$$

Here $\rho(g)$ uses an exact symmetry of the discrete stencil, while $\kappa(Q)$ exposes the encoder to orientations beyond the finite set represented by G.

The original and transformed views are encoded with shared weights, and we encourage their invariant codes to agree:

$$\mathcal{L}_{\mathrm{orb}} = 1 - \cos\Big(\mathrm{Enc}_\theta(\mathbf{S})_{\mathrm{inv}}, \mathrm{Enc}_\theta(\widetilde{\mathbf{S}}_{g,Q})_{\mathrm{inv}}\Big). \tag{9}$$

Thus $\mathbf{z}_{\mathrm{inv}}$ is trained to identify stars related by both the exact stencil symmetry and arbitrary continuous rotations, without prescribing the transformation law of intermediate features. The hard architectural bias handles the known discrete correspondence between neighbor slots, while $\mathcal{L}_{\mathrm{orb}}$ extends the desired invariance to continuous orientations.

The complementary roles of hard and soft symmetry constraints. The architectural bias and orbit constraint address different parts of the symmetry problem. Parameter tying encodes the exact finite correspondence between stencil neighbors, whereas $\mathcal{L}_{\mathrm{orb}}$ learns invariance across continuous orientations. Together they reduce the burden of learning a known discrete symmetry while avoiding the need to impose full $\mathrm{SO}(d)$ equivariance on the convolution–Transformer backbone.

3.5 TRAINING REGIMES

Unsupervised representation learning. Without training labels, we learn $\mathbf{z}_{\mathrm{inv}}$ using orbit consistency together with reconstruction and VICReg-style variance/covariance regularization (Bardes et al., 2022). Specifically, an optional SIREN decoder (Sitzmann et al., 2020) $\mathrm{Dec}_\psi(u, \mathbf{z})$ reconstructs each view from its full code with $\mathcal{L}_{\mathrm{rec}} = \sum_j w_j \|\mathbf{S}_j - \mathrm{Dec}_\psi(\cdot, \mathbf{z})_j\|_2^2$. Positive-pair learning can admit collapsed representations (Chen & He, 2021; Grill et al., 2020); we use reconstruction and

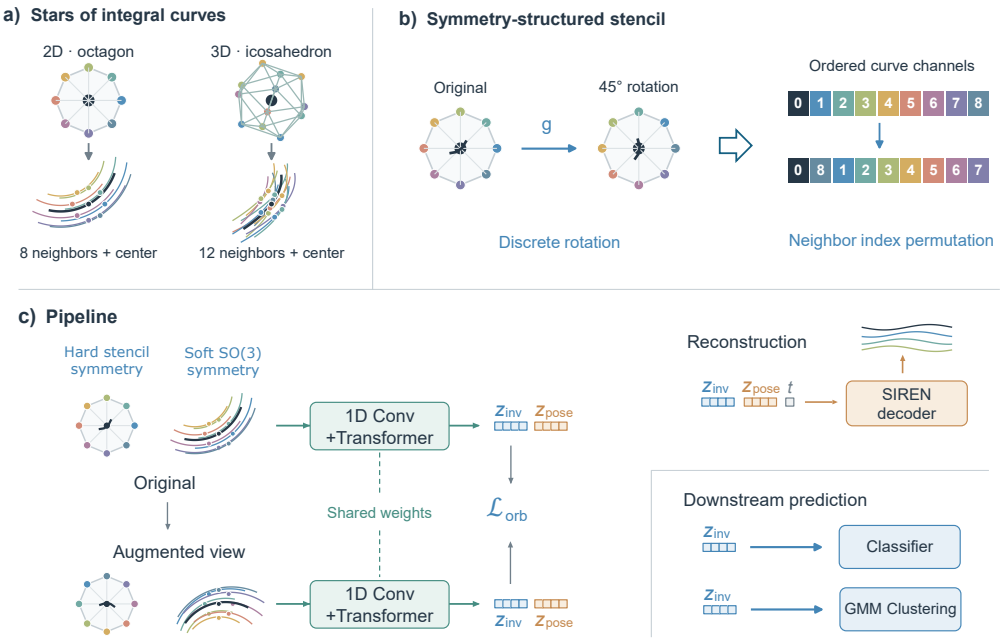

Figure 1: **StarVortex.** (a) A star of curves: a center integral curve and K neighbors seeded on the vertices of a regular octagon (2D) or icosahedron (3D). (b) Each element g of the stencil's symmetry group rotates the samples and induces a permutation of the neighbors. (c) Architecture: a Conv–Transformer encoder maps the star to a code with an invariant part z_{inv}, which the self-supervised objective makes agree across group views, and a pose part z_{pose}, used only by the optional SIREN decoder. z_{inv} feeds a classifier when labels exist and a Gaussian-mixture read-out when they do not.

Table 2: **Instantiations of StarVortex.** (a) Stencil and finite symmetry group. (b) Training objectives and read-outs under different supervision regimes.

(a) Star

	Stencil (K)	G ($\lvert G\rvert$)
2D	octagon (8)	D_8 (16)
3D	icosahedron (12)	I_h (120)

(b) Training

Regime	Objective	Read-out
supervised	$\mathcal{L}_{\text{CE}}$, ($\mathcal{L}_{\text{orb}}$, $\mathcal{L}_{\text{rec}}$)	MLP on $\mathbf{z}_{\text{inv}}$
unsupervised	$\mathcal{L}_{\text{orb}}$, $\mathcal{L}_{\text{rec}}$	GMM on $\mathbf{z}_{\text{inv}}$

variance/covariance regularization to discourage this collapse. We optimize $\mathcal{L}_{\text{orb}}+\lambda_{\text{rec}}\mathcal{L}_{\text{rec}}$ together with these regularizers. For Lagrangian vortex data, we then fit a two-component shared-covariance Gaussian mixture model (GMM) to the ℓ_2-normalized $\mathbf{z}_{\text{inv}}$; clustering the learned representations separates vortex and non-vortex regions and thereby delineates the vortex area. Labels are used only for evaluation.

Supervised learning. With training labels, a linear head or MLP maps $\mathbf{z}_{\text{inv}}$ to the target label and is optimized with the cross-entropy loss $\mathcal{L}_{\text{CE}}$. We additionally evaluate $\mathcal{L}_{\text{orb}}$ and, when used, $\mathcal{L}_{\text{rec}}$ as auxiliary self-supervised objectives, particularly when labeled data are limited.

4 Experiments

4.1 Problem Setup

We take a vortex to be a region of co-rotating fluid particles and study three extraction tasks that share one common form. The input is a cluster (star) of integral curves seeded around a center point and given as sampled positions only, with no velocity, vorticity, or other field attribute. The output is a binary label of the center point. The tasks differ in the curve type, the flow, and what the

Table 3: **3D vortex identification** on ten flow fields. F1, precision, and recall are equal-flow means ± SD over 3 runs. IVD supervision indicates whether IVD-derived labels train the model.

Method	IVD supervision	F1	Precision (%)	Recall (%)
Supervised (IVD-derived labels)				
MVU-Net (Deng et al., 2022b)	✓	0.7470 ± 0.0128	72.622 ± 0.708	79.638 ± 1.667
MVU-Net+IVD (Deng et al., 2022b)	✓	0.8102 ± 0.0049	78.357 ± 0.559	86.365 ± 0.220
VortexBoundary (CNN) (Berenjkoub et al., 2020)	✓	0.4745 ± 0.0085	48.070 ± 0.957	49.665 ± 0.852
VortexViz+Vorticity (de Silva et al., 2024)	✓	0.8293 ± 0.0051	79.273 ± 1.141	89.183 ± 0.675
Ours (supervised)	✓	**0.8879** ± 0.0070	**85.043** ± 0.880	**93.239** ± 0.311
No label supervision				
Geometric features + GMM	✗	0.2408 ± 0.0000	14.892 ± 0.002	**76.178** ± 0.014
Tangent displacement + GMM	✗	0.1323 ± 0.0004	7.374 ± 0.023	73.272 ± 0.141
Ours (unsupervised)	✗	**0.6724** ± 0.0243	**68.864** ± 2.221	69.550 ± 6.053

label means. All ground truth is set or verified by human experts, and all datasets and labels will be released.

4.2 Vortex region identification

We classify whether a pathline star's center lies inside a vortex at its initial time, using reference masks based on instantaneous vorticity deviation (IVD). Across ten 3D flows, self-supervised StarVortex learns and clusters each flow's available pathlines without label supervision. Table 3 compares 3D adaptations of MVU-Net, VortexBoundary, and VortexViz, alongside two traditional geometric representations with the same GMM readout. Self-supervised StarVortex achieves positive-class F1 of 0.6724, compared with 0.2408 for geometric features and 0.1323 for tangent displacements. It also exceeds supervised VortexBoundary (0.4745) but trails MVU-Net (0.7470) and vorticity-augmented VortexViz (0.8293). These self-supervised objectives also benefit supervised coreline recognition, particularly with limited labels or under domain shift (Table 6). Adding orbit consistency and reconstruction increases F1 from 0.7415 to 0.7890 with 500 labels, and from 0.8647 to 0.8916 under domain generalization. Figure 2 shows StarVortex recovering elongated vortices and cylinder-wake shedding patterns. The close-ups reveal overextended predictions at $Re = 160$ and 6400, and missed vortex samples at $Re = 640$. Appendix Figure 11 compares predictions with supervised baselines.

The above results are for 3D flows. For unsteady 2D flows, we report the performance in Table 7 and qualitative examples in Figure 6 of Appendix A, where we observe the same trend: our self-supervised model achieves results comparable to supervised baselines, and our supervised variant surpasses all prior work.

Ablation. Table 4 shows that both hard and soft symmetry mechanisms are important for unsupervised vortex identification. Removing the hard stencil symmetry causes a large drop in F1 from 0.736 to 0.584, while removing the soft $\mathrm{SO}(3)$ symmetry also degrades performance to 0.675, confirming that the two mechanisms provide complementary benefits. The orbit-consistency objective is particularly important: without $\mathcal{L}_{\mathrm{orbit}}$, F1 falls further to 0.566, indicating that explicitly enforcing consistency across symmetry-related views is critical for learning a useful invariant representation. Reconstruction provides an additional complementary signal, with its removal reducing F1 to 0.708. Overall, the ablation supports the design of StarVortex: hard symmetry encodes the known discrete stencil structure, soft symmetry promotes invariance across continuous orientations, and the auxiliary self-supervised objectives further stabilize and enrich the learned vortex representation.

4.3 Vortex coreline identification

We identify vortex corelines from observed streamlines in steady or nearly steady 3D flow; the label states whether the center lies on a coreline. The target is the one-dimensional curve around which the flow swirls. Reference labels are refined by experts from the VTK vortex core filter (Schroeder et al., 2000; VTK Developers, 2026); Appendix C.3 describes annotation and sampling. Figure 7 shows predictions from our supervised model. Although a coreline is by nature an integral curve, the difficulty lies in precisely locating points on it. Our model predicts labels for sampled star center points rather than the curve itself, yet a simple linking algorithm recovers the full coreline from these

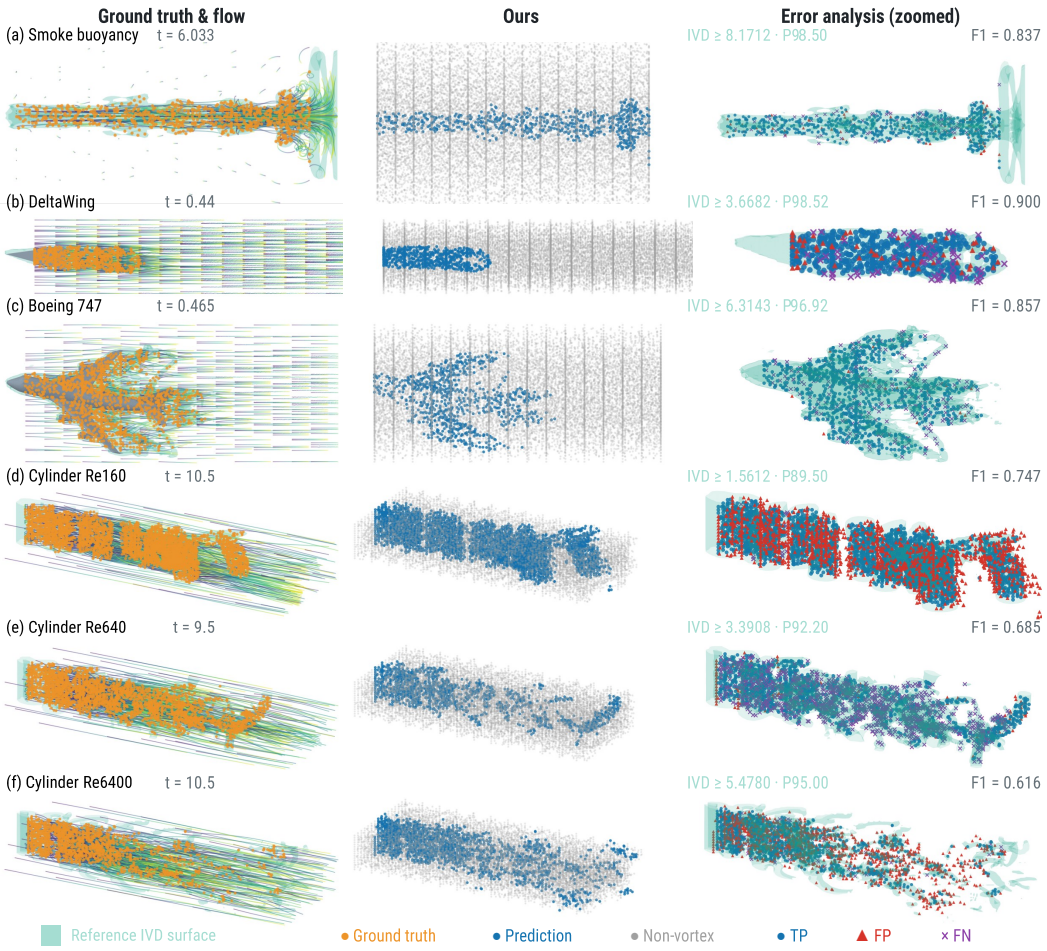

Figure 2: **Vortex-region extraction results.** The first column shows the ground-truth labels and the reference IVD surface defined by domain experts, with pathlines color-coded by time. The second column displays our self-supervised StarVortex predictions, where blue points indicate the predicted vortex region. The final column provides a zoomed-in view of the classification errors.

Table 4: Ablation of unsupervised 3D vortex identification.

	F1	Precision	Recall
Full method	**0.736**	**0.759**	0.745
− hard stencil symmetry	0.584	0.659	0.584
− soft SO(3) symmetry	0.675	0.701	0.704
− $\mathcal{L}_{\text{orbit}}$	0.566	0.648	0.545
− $\mathcal{L}_{\text{recon}}$	0.708	0.643	0.811
− VICReg	0.704	0.757	0.699
k-means read-out instead of GMM	0.664	0.596	**0.845**
Untrained	0.387	0.319	0.541

predictions. Figure 8 compares our method against classical 3D point cloud baselines, PointNet (Qi et al., 2017a) and a 3D convolutional network. Both the close agreement between our predicted point shapes and the ground truth and the F1 scores demonstrate that our architecture effectively captures the swirling behavior of streamline bundles as shape features. Table 8 reports per-flow test performance, including four unseen flows, demonstrating the generalization ability of our method.

Table 5: **Coreline extraction** performance averaged across nine datasets. F1, precision and recall use the positive class ($d < h$); precision and recall are percentages. CD is the Chamfer distance (lower is better). Detailed results are provided in Appendix C.3, Table 8.

Method	F1	CD ↓	Precision (%)	Recall (%)
Conv3D (32^3)	0.7991	1.6786	84.38	83.67
PointNet (Qi et al., 2017a)	0.8431	1.8772	80.08	89.52
Ours	**0.9478**	**0.9019**	**94.63**	**95.06**
VTK (Schroeder et al., 2000) (*source of GT*)†	0.9888	4.7056	98.75	99.04

†The VTK vortex core filter uses a parallel-vectors method (VTK Developers, 2026). Ground truth is derived from its raw output, manually curated and refined by domain experts.

Table 6: **Self-supervised representation-learning objectives improve supervised learning** in coreline extraction, particularly with limited labels and under domain shift. Positive-class F1 scores are averaged over five scenes on a subset of full data.

Loss	All labels	Limited labels (500 / 1000 / 5000)	Domain generalization
$\mathcal{L}_{\text{CE}}$	0.9303	0.7415 / 0.7774 / 0.8389	0.8647
$\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{orb}}$	**0.9333**	0.7528 / 0.7955 / **0.8439**	0.8875
$\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{orb}} + \mathcal{L}_{\text{recon}}$	0.9327	**0.7890** / **0.8042** / 0.8400	**0.8916**

4.4 Hairpin vortex identification

A hairpin vortex consists of a spanwise head connected to two streamwise legs, and arises only in turbulent flow at high Reynolds numbers. Extracting hairpin vortices is a notoriously difficult task. We identify hairpin vortices from vortex lines in 3D wall-bounded turbulent flow; the label states whether the center belongs to a hairpin vortex. Labels come from the manually annotated dataset of Zafar et al. (2026).

We evaluate three wall-bounded flows using three cross-flow splits, training on two flows and testing on the remaining flow. As shown in Table 9, on the turbulent boundary-layer test set, StarVortex is the first hairpin extraction method to learn directly from line bundle shape, and it outperforms all baselines, including the classical 3D shape learning models PointNet Qi et al. (2017a) and a 3D convolutional network, as well as the state-of-the-art non-learning hairpin extractor of Zafar et al. (2026). Details about the hairpin extraction experiment are given in Appendix C.4.

5 Discussion and Limitations

Our results suggest that self-supervised representation learning from rotation symmetry and self-reconstruction helps to expose the vortex structure in the collective geometry of curve bundles. Rotating neighbor channel processing encodes stencil symmetries, while orbit consistency encourages features that persist across rotated views. Reconstruction and variance/covariance regularization retain informative geometric variation. For pathline bundles, this favors descriptors of relative motion and deformation, conceptually resembling finite-time Lyapunov exponent (FTLE) analysis of coherent transport. FTLE measures trajectory stretching objectively (Haller, 2015); establishing such objectivity under time-dependent observer changes for our learned features remains an open question.

The hairpin extraction experiments demonstrate a complementary capability: recognizing hairpin vortices from vortex-line bundle shape. Their characteristic curved head and paired legs form a distinctive spatial organization that supervised learning can discriminate. Together, these findings support integral-curve geometry as a representation of both collective transport and specific vortex morphology.

Limitations. Our self-supervised model trails the strongest supervised detectors in 3D vortex-region identification (Table 3). Hairpin recognition is also limited by the availability of labeled data.

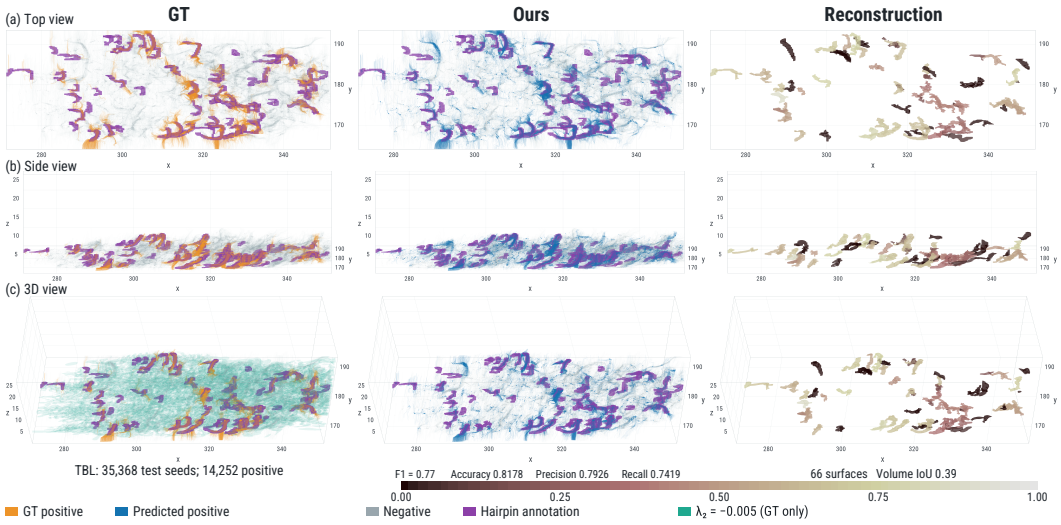

Figure 3: **Hairpin extraction & reconstruction on the TBL flow.** The first column shows ground-truth positive line bundles in orange, human annotations in purple, and the λ_2 isosurface in green. The second column shows StarVortex predictions, with positive samples in blue. The final column shows reconstructed hairpin surfaces, colored by their overlap with the annotations. Rows provide top, side, and 3D views of the same turbulent boundary-layer flow.

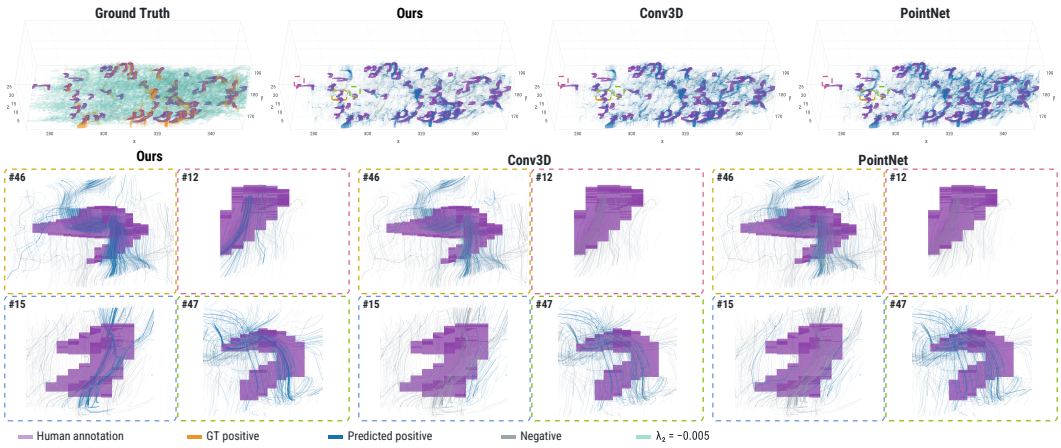

Figure 4: **Hairpin extraction comparison.** The top row shows ground truth followed by predictions from StarVortex, Conv3D, and PointNet (Qi et al., 2017a) on the TBL flow. The panels below show close-ups of four annotated hairpin instances for each model. Purple surfaces mark human annotations, and blue curves mark predicted-positive line bundles.

All evaluated methods perform poorly on Channel flow (Table 9), and limited training diversity may contribute to the learned models' weak cross-flow generalization.

6 Conclusion

We presented StarVortex, which learns vortex structure from local stars of integral curves. Its self-supervised formulation achieves vortex-region identification competitive with supervised detectors. The same representation also yields the first learning-based methods to extract vortex corelines and hairpin vortices from curve geometry alone. Using only sampled positions, StarVortex analyzes coherent structures without velocity, vorticity, or other physical attributes. This makes it directly applicable to particle trajectories produced by Lagrangian fluid simulations, including scattered particle data of the kind illustrated in Appendix Figure 5 and the supplementary video.

AI USE STATEMENT

Large language models assisted with writing and coding. The authors independently verified all code and claims, and take full responsibility for results.

REPRODUCIBILITY STATEMENT

Implementation details, datasets, and hyperparameters are described in Appendix C. The source code, configurations, datasets, and labels will be made publicly available upon acceptance.

REFERENCES

R. J. Adrian, C. D. Meinhart, and C. D. Tomkins. Vortex organization in the outer region of the turbulent boundary layer. *Journal of Fluid Mechanics*, 422:1–54, November 2000. ISSN 1469-7645. doi: 10.1017/s0022112000001580. URL http://dx.doi.org/10.1017/S0022112000001580.

Ronald J. Adrian. Hairpin vortex organization in wall turbulence. *Physics of Fluids*, 19(4), April 2007. ISSN 1089-7666. doi: 10.1063/1.2717527. URL http://dx.doi.org/10.1063/1.2717527.

Alexy Agranovsky, David Camp, Christoph Garth, E. Wes Bethel, Kenneth I. Joy, and Hank Childs. Improved post hoc flow analysis via lagrangian representations. In *2014 IEEE 4th Symposium on Large Data Analysis and Visualization (LDAV)*, pp. 67–75. IEEE, 2014. URL https://ieeexplore.ieee.org/abstract/document/7013206/.

Xue Bai, Changbo Wang, and Chenhui Li. A streampath-based rcnn approach to ocean eddy detection. *IEEE Access*, 7:106336–106345, 2019.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022.

Marzieh Berenjkoub, Guoning Chen, and Tobias Günther. Vortex boundary identification using convolutional neural network. In *IEEE Visualization (VIS)*, pp. 261–265, October 2020. doi: 10.1109/VIS47514.2020.00059.

Peer-Timo Bremer, Andrea Gruber, Janine Bennett, Attila Gyulassy, Hemanth Kolla, Jacqueline Chen, and Ray Grout. Identifying turbulent structures through topological segmentation. *Communications in Applied Mathematics and Computational Science*, 11(1):37–53, January 2016. ISSN 1559-3940. doi: 10.2140/camcos.2016.11.37. URL http://dx.doi.org/10.2140/camcos.2016.11.37.

Roxana Bujack and Kenneth I. Joy. Lagrangian representations of flow fields with parameter curves. In *2015 IEEE 5th Symposium on Large Data Analysis and Visualization (LDAV)*, pp. 41–48. IEEE, 2015. URL https://ieeexplore.ieee.org/abstract/document/7348070/.

Brian Cabral and Leith Casey Leedom. Imaging vector fields using line integral convolution. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pp. 263–270, 1993.

Jennifer Chandler, Harald Obermaier, and Kenneth I. Joy. Interpolation-based pathline tracing in particle-based flow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 21(1):68–80, 2015. doi: 10.1109/TVCG.2014.2325043.

Xinlei Chen and Kaiming He. Exploring simple Siamese representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15745–15753, 2021. doi: 10.1109/CVPR46437.2021.01549.

Yuqian Chen, Chaoyi Zhang, Tengfei Xue, Yang Song, Nikos Makris, Yogesh Rathi, Weidong Cai, Fan Zhang, and Lauren J. O'Donnell. Deep fiber clustering: Anatomically informed fiber clustering with self-supervised deep learning for fast and effective tractography parcellation. *NeuroImage*, 273:120086, 2023. doi: 10.1016/j.neuroimage.2023.120086.

Min S Chong, Anthony E Perry, and Brian J Cantwell. A general classification of three-dimensional flow fields. *Physics of Fluids A: Fluid Dynamics*, 2(5):765–777, 1990.

Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral CNN. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 2019.

Nico Daßler and Tobias Günther. Variational feature extraction in scientific visualization. *ACM Trans. Graph.*, 43(4), July 2024. ISSN 0730-0301. doi: 10.1145/3658219. URL https://doi.org/10.1145/3658219.

Akila de Silva, Nicholas Tee, Omkar Ghanekar, Fahim Hasan Khan, Gregory Dusek, James Davis, and Alex Pang. Vortexviz: Finding vortex boundaries by learning from particle trajectories, April 2024.

Liang Deng, Yueqing Wang, Yang Liu, Fang Wang, Sikun Li, and Jie Liu. A cnn-based vortex identification method. *Journal of Visualization*, 22(1):65–78, February 2019. ISSN 1875-8975. doi: 10.1007/s12650-018-0523-1.

Liang Deng, Wenchun Bao, Yueqing Wang, Zhigong Yang, Dan Zhao, Fang Wang, Chongke Bi, and Yang Guo. Vortex-u-net: An efficient and effective vortex detection approach based on u-net structure. *Applied Soft Computing*, 115:108229, January 2022a. ISSN 1568-4946. doi: 10.1016/j.asoc.2021.108229.

Liang Deng, Jianqiang Chen, Yueqing Wang, Xinhai Chen, Fang Wang, and Jie Liu. Mvu-net: A multi-view u-net architecture for weakly supervised vortex detection. *Engineering Applications of Computational Fluid Mechanics*, 16(1):1567–1586, December 2022b. ISSN 1994-2060. doi: 10.1080/19942060.2022.2104930.

Yitong Deng, Hong-Xing Yu, Jiajun Wu, and Bo Zhu. Learning vortex dynamics for fluid inference and prediction. In *Proceedings of the International Conference on Learning Representations*, March 2023a.

Yitong Deng, Hong-Xing Yu, Diyang Zhang, Jiajun Wu, and Bo Zhu. Fluid Simulation on Neural Flow Maps. *ACM Transactions on Graphics*, 42(6):1–21, December 2023b. ISSN 0730-0301, 1557-7368. doi: 10.1145/3618392. URL https://dl.acm.org/doi/10.1145/3618392.

David J. C. Dennis and Timothy B. Nickels. Experimental measurement of large-scale three-dimensional structures in a turbulent boundary layer. part 1. vortex packets. *Journal of Fluid Mechanics*, 673:180–217, March 2011. ISSN 1469-7645. doi: 10.1017/s0022112010006324. URL http://dx.doi.org/10.1017/s0022112010006324.

Carlos Esteves, Yinshuang Xu, Christine Allen-Blanchette, and Kostas Daniilidis. Equivariant multi-view networks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1568–1577, 2019. doi: 10.1109/ICCV.2019.00165.

Katharina Franz, Ribana Roscher, Andres Milioto, Susanne Wenzel, and Jürgen Kusche. Ocean eddy identification and tracking using neural networks. In *IEEE international geoscience and remote sensing symposium (IGARSS)*, pp. 6887–6890. IEEE, 2018.

Raphael Fuchs, Ronald Peikert, Filip Sadlo, Bilal Alsallakh, and M Eduard Gröller. Delocalized unsteady vortex region detectors. In *VMV*, volume 8, pp. 81–90, 2008.

Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. VectorNet: Encoding HD maps and agent dynamics from vectorized representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Christoph Garth, Alexander Wiebel, Xavier Tricoche, Ken Joy, and Gerik Scheuermann. Lagrangian visualization of flow-embedded surface structures. *Computer Graphics Forum*, 27(3):1007–1014, 2008.

Annalisa Griffa, Leonid I Piterbarg, and Tamay Özgökmen. Predictability of lagrangian particle trajectories: effects of smoothing of the underlying eulerian flow. *Journal of Marine Research*, 62(1):1–35, 2004. doi: 10.1357/00222400460744609.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent – a new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284, 2020.

Tobias Günther and Holger Theisel. The State of the Art in Vortex Extraction. *Computer Graphics Forum*, 37(6):149–173, 2018.

Tobias Günther and Holger Theisel. Objective Lagrangian vortex cores and their visual representations. *IEEE Transactions on Visualization and Computer Graphics*, 31(1):76–85, January 2025. doi: 10.1109/TVCG.2024.3456384.

Tobias Günther, M. Gross, and H. Theisel. Generic objective vortices for flow visualization. *ACM Transactions on Graphics*, 36, 2017. doi: 10.1145/3072959.3073684.

David Ha and Douglas Eck. A neural representation of sketch drawings. In *International Conference on Learning Representations (ICLR)*, 2018.

Markus Hadwiger, Matej Mlejnek, Thomas Theußl, and Peter Rautek. Time-dependent flow seen through approximate observer killing fields. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):1257–1266, 2019. doi: 10.1109/TVCG.2018.2864839.

G. Haller and G. Yuan. Lagrangian coherent structures and mixing in two-dimensional turbulence. *Physica D: Nonlinear Phenomena*, 147(3–4):352–370, December 2000. ISSN 0167-2789. doi: 10.1016/s0167-2789(00)00142-1. URL http://dx.doi.org/10.1016/S0167-2789(00)00142-1.

George Haller. Lagrangian coherent structures. *Annual Review of Fluid Mechanics*, 47(1):137–162, January 2015. ISSN 1545-4479. doi: 10.1146/annurev-fluid-010313-141322. URL http://dx.doi.org/10.1146/annurev-fluid-010313-141322.

George Haller, Alireza Hadjighasem, Mohammad Farazmand, and Florian Huhn. Defining coherent vortices objectively from the vorticity. *Journal of Fluid Mechanics*, 795:136–173, 2016.

Jun Han, Jun Tao, Hao Zheng, Hanqi Guo, Danny Z. Chen, and Chaoli Wang. Flow Field Reduction Via Reconstructing Vector Data From 3-D Streamlines Using Deep Learning. *IEEE Computer Graphics and Applications*, 39(4):54–67, July 2019. ISSN 1558-1756. doi: 10.1109/MCG.2018.2881523. URL https://ieeexplore.ieee.org/abstract/document/8739144.

Jun Han, Jun Tao, and Chaoli Wang. Flownet: A deep learning framework for clustering and selection of streamlines and stream surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 26(4):1732–1744, 2020. doi: 10.1109/TVCG.2018.2880207.

Mengjiao Han, Sudhanshu Sane, and Chris R. Johnson. Exploratory lagrangian-based particle tracing using deep learning. *Journal of Flow Visualization and Image Processing*, 29(3), 2022. URL https://www.dl.begellhouse.com/journals/52b74bd3689ab10b,197ac1601136cd11,494ac9e91b77e1f5.html.

Mengjiao Han, Jixian Li, Sudhanshu Sane, Shubham Gupta, Bei Wang, Steve Petruzza, and Chris R. Johnson. Interactive Visualization of Time-Varying Flow Fields Using Particle Tracing Neural Networks. In *2024 IEEE 17th Pacific Visualization Conference (PacificVis)*, pp. 52–61, April 2024. doi: 10.1109/PacificVis60374.2024.00015. URL https://ieeexplore.ieee.org/abstract/document/10541625.

Mathias Hummel, Roxana Bujack, Kenneth I. Joy, and Christoph Garth. Error estimates for Lagrangian flow field representations. In *EuroVis 2016 – Short Papers*, pp. 7–11. The Eurographics Association, 2016. doi: 10.2312/eurovisshort.20161153.

J. C. R. Hunt, A. A. Wray, and P. Moin. Eddies, streams, and convergence zones in turbulent flows. Report ctr-s88, Center for Turbulence Research, Stanford University, 1988.

Jakob Jakob, Markus Gross, and Tobias Günther. A fluid flow data set for machine learning and its application to neural flow map interpolation. *IEEE Transactions on Visualization and Computer Graphics (Proc. IEEE Scientific Visualization)*, 27(2):1279–1289, 2021. doi: 10.1109/TVCG.2020.3028947.

Jinhee Jeong and Fazle Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285: 69–94, 1995.

Y. Jodai and G. E. Elsinga. Experimental observation of hairpin auto-generation events in a turbulent boundary layer. *Journal of Fluid Mechanics*, 795:611–633, April 2016. ISSN 1469-7645. doi: 10.1017/jfm.2016.153. URL http://dx.doi.org/10.1017/jfm.2016.153.

Babak Kashir, Marco Ragone, Ajaykrishna Ramasubramanian, Vitaliy Yurkiv, and Farzad Mashayek. Application of fully convolutional neural networks for feature extraction in fluid flow. *Journal of Visualization*, 24(4):771–785, August 2021. ISSN 1875-8975. doi: 10.1007/s12650-020-00732-0. URL https://doi.org/10.1007/s12650-020-00732-0.

Jens Kasten, Tino Weinkauf, Christoph Petz, Ingrid Hotz, Bernd R. Noack, and Hans-Christian Hege. Extraction of coherent structures from natural and actuated flows. In *Active Flow Control II*, volume 108 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, pp. 373–387. Springer Berlin Heidelberg, 2010. ISBN 9783642117350. doi: 10.1007/978-3-642-11735-0_24.

Ali Rahimi Khojasteh, Yin Yang, Dominique Heitz, and Sylvain Laizet. Lagrangian coherent track initialization. *Physics of Fluids*, 33(9), 2021.

Byungsoo Kim and Tobias Günther. Robust reference frame extraction from unsteady 2d vector fields with convolutional neural networks. *Computer Graphics Forum*, 38(3):285–295, 2019. ISSN 1467-8659. doi: 10.1111/cgf.13689.

Redouane Lguensat, Miao Sun, Ronan Fablet, Pierre Tandeo, Evan Mason, and Ge Chen. Eddynet: A deep neural network for pixel-wise classification of oceanic eddies. In *IEEE international geoscience and remote sensing symposium (IGARSS)*, pp. 1764–1767. IEEE, 2018.

Haoyu Li, Tianyu Xiong, and Han-Wei Shen. Efficient Interpolation-based Pathline Tracing with B-spline Curves in Particle Dataset. In *2022 IEEE Visualization and Visual Analytics (VIS)*, pp. 140–144. IEEE, 2022. doi: 10.1109/VIS54862.2022.00037. URL https://ieeexplore.ieee.org/abstract/document/9973196/.

Xiucheng Li, Kaiqi Zhao, Gao Cong, Christian S. Jensen, and Wei Wei. Deep representation learning for trajectory similarity computation. In *IEEE 34th International Conference on Data Engineering (ICDE)*, pp. 617–628. IEEE, 2018. doi: 10.1109/ICDE.2018.00062.

Chaoqun Liu, Yisheng Gao, Shuling Tian, and Xiangrui Dong. Rortex—a new vortex vector definition and vorticity tensor and vector decompositions. *Physics of Fluids*, 30(3), March 2018. ISSN 1089-7666. doi: 10.1063/1.5023001. URL http://dx.doi.org/10.1063/1.5023001.

Yang Liu, Yutong Lu, Yueqing Wang, Dong Sun, Liang Deng, Fang Wang, and Yan Lei. A cnn-based shock detection method in flow visualization. *Computers & Fluids*, 184:1–9, 2019.

Theodore MacMillan and David H. Richter. The most robust representations of flow trajectories are Lagrangian coherent structures. *Journal of Fluid Mechanics*, 927:A26, 2021. doi: 10.1017/jfm.2021.768.

R. Peikert and M. Roth. The "Parallel Vectors" operator-a vector field visualization primitive. In *Proceedings Visualization '99 (Cat. No.99CB37067)*, pp. 263–270, October 1999. doi: 10.1109/VISUAL.1999.809896.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 652–660, 2017a.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, volume 30, 2017b.

Xuerong Qin, Erik van Sebille, and Alexander Sen Gupta. Quantification of errors induced by temporal resolution on lagrangian particles in an eddy-resolving model. *Ocean Modelling*, 76: 20–30, 2014. doi: 10.1016/j.ocemod.2014.02.002.

Ali Rahimi Khojasteh, Sylvain Laizet, Dominique Heitz, and Yin Yang. Lagrangian and Eulerian dataset of flow over a circular cylinder at Reynolds number 3900. Recherche Data Gouv, dataset, 2021. URL `https://doi.org/10.15454/GLNRHK`.

Peter Rautek, Matej Mlejnek, Johanna Beyer, Jakob Troidl, Hanspeter Pfister, Thomas Theußl, and Markus Hadwiger. Objective observer-relative flow visualization in curved spaces for unsteady 2D geophysical flows. *IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE Scientific Visualization 2020)*, 27(2):283–293, 2021. doi: 10.1109/TVCG.2020.3030454.

Peter Rautek, Xingdi Zhang, Bernhard Woschizka, Thomas Theußl, and Markus Hadwiger. Vortex lens: Interactive vortex core line extraction using observed line integral convolution. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):55–65, 2024. doi: 10.1109/TVCG.2023.3326915.

Christian Rössl and Holger Theisel. Streamline embedding for 3d vector field exploration. *IEEE Transactions on Visualization and Computer Graphics*, 18(3):407–420, 2012. doi: 10.1109/TVCG.2011.78.

M. Roth and R. Peikert. A higher-order method for finding vortex core lines. In *IEEE Visualization (VIS)*, pp. 143–150, 1998. doi: https://doi.org/10.1109/visual.1998.745296.

I. Ari Sadarjoen and Frits H. Post. *Geometric Methods for Vortex Extraction*, pp. 53–62. Springer Vienna, 1999. ISBN 9783709168035. doi: 10.1007/978-3-7091-6803-5_6. URL `http://dx.doi.org/10.1007/978-3-7091-6803-5_6`.

Saroj Sahoo, Y. Lu, and Matthew Berger. Neural flow map reconstruction. *Computer Graphics Forum*, 41(3):391–402, 2022. doi: 10.1111/cgf.14549.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8459–8468, 2020.

Sudhanshu Sane, Roxana Bujack, and Hank Childs. Revisiting the evaluation of in situ Lagrangian analysis. In *Eurographics Symposium on Parallel Graphics and Visualization (EGPGV)*, pp. 63–67. The Eurographics Association, 2018. doi: 10.2312/pgv.20181096.

Sudhanshu Sane, Chris R. Johnson, and Hank Childs. Demonstrating the viability of Lagrangian in situ reduction on supercomputers. *Journal of Computational Science*, 61:101615, 2022. doi: 10.1016/j.jocs.2022.101615.

William J. Schroeder, Lisa S. Avila, and William Hoffman. Visualizing with vtk: A tutorial. *IEEE Computer Graphics and Applications*, 20(5):20–27, 2000. doi: 10.1109/38.865875.

Shawn C Shadden, Francois Lekien, and Jerrold E Marsden. Definition and properties of lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3–4):271–304, 2005. doi: 10.1016/j.physd.2005.10.007.

Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7462–7473, 2020.

David Sujudi and Robert Haimes. Identification of swirling flow in 3-d vector fields. In *Proceedings of the 12th Computational Fluid Dynamics Conference*, pp. 792–799, 1995. doi: https://doi.org/10.2514/6.1995-1715.

Artur P. Toshev, Gianluca Galletti, Fabian Fritz, Stefan Adami, and Nikolaus A. Adams. LagrangeBench: A Lagrangian fluid mechanics benchmarking suite. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, volume 36, 2023.

Benjamin Ummenhofer, Lukas Prantl, Nils Thuerey, and Vladlen Koltun. Lagrangian fluid simulation with continuous convolutions. In *International Conference on Learning Representations (ICLR)*, 2020.

VTK Developers. vtkVortexCore class reference. VTK documentation, 2026. URL https://vtk.org/doc/nightly/html/classvtkVortexCore.html. Accessed September 26, 2026.

Yueqing Wang, Liang Deng, Zhigong Yang, Dan Zhao, and Fang Wang. A rapid vortex identification method using fully convolutional segmentation network. *The Visual Computer*, 37(2):261–273, 2021.

Tino Weinkauf, Jan Sahner, Holger Theisel, and Hans-Christian Hege. Cores of swirling particle motion in unsteady flows. *IEEE Transactions on Visualization and Computer Graphics*, 13(6): 1759–1766, 2007.

Y. Wu and K. T. Christensen. Population trends of spanwise vortices in wall turbulence. *Journal of Fluid Mechanics*, 568:55–76, November 2006. ISSN 1469-7645. doi: 10.1017/s002211200600259x. URL http://dx.doi.org/10.1017/S002211200600259X.

Tengfei Xue, Yuqian Chen, Chaoyi Zhang, Alexandra J. Golby, Nikos Makris, Yogesh Rathi, Weidong Cai, Fan Zhang, and Lauren J. O'Donnell. TractCloud: Registration-free tractography parcellation with a novel local-global streamline point cloud representation. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, Lecture Notes in Computer Science, pp. 409–419. Springer, 2023a. doi: 10.1007/978-3-031-43993-3_40.

Tengfei Xue, Fan Zhang, Chaoyi Zhang, Yuqian Chen, Yang Song, Alexandra J. Golby, Nikos Makris, Yogesh Rathi, Weidong Cai, and Lauren J. O'Donnell. Superficial white matter analysis: An efficient point-cloud-based deep learning framework with supervised contrastive learning for consistent tractography parcellation across populations and dMRI acquisitions. *Medical Image Analysis*, 85:102759, 2023b. doi: 10.1016/j.media.2023.102759.

Hongfeng Yu, Chaoli Wang, Ching-Kuang Shene, and Jacqueline H. Chen. Hierarchical streamline bundles. *IEEE Transactions on Visualization and Computer Graphics*, 18(8):1353–1367, 2012. doi: 10.1109/TVCG.2011.155.

Adeel Zafar, Zahra Poorshayegh, Di Yang, and Guoning Chen. Topological separation of vortices. In *2024 IEEE Visualization and Visual Analytics (VIS)*, pp. 311–315. IEEE, October 2024a. doi: 10.1109/vis55277.2024.00070. URL http://dx.doi.org/10.1109/VIS55277.2024.00070.

Adeel Zafar, Di Yang, and Guoning Chen. Extract and characterize hairpin vortices in turbulent flows. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):716–726, 2024b. ISSN 2160-9306. doi: 10.1109/tvcg.2023.3326603. URL http://dx.doi.org/10.1109/TVCG.2023.3326603.

Adeel Zafar, Zahra Poorshayegh, Lei Si, Di Yang, and Guoning Chen. Hairpin vortices extraction in turbulent boundary layer flows. *IEEE Transactions on Visualization and Computer Graphics*, 32(7):7288–7300, 2026. doi: 10.1109/TVCG.2026.3698734.

L. Zhang, Q. Deng, R. Machiraju, A. Rangarajan, D. Thompson, D. K. Walters, and H.-W. Shen. Boosting techniques for physics-based vortex detection. *Computer Graphics Forum*, 33(1):282–293, 2014. ISSN 1467-8659. doi: 10.1111/cgf.12275.

Weihan Zhang and Jun Tao. Automatic semantic alignment of flow pattern representations for exploration with large language models. *IEEE Transactions on Visualization and Computer Graphics*, 32(1):835–845, 2026. doi: 10.1109/TVCG.2025.3634650.

X. Zhang, P. Rautek, and M. Hadwiger. Vortextransformer: End-to-end objective vortex detection in 2d unsteady flow using transformers. *Computer Graphics Forum*, 44(2):e70042, 2025. doi: https://doi.org/10.1111/cgf.70042. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.70042.

Xingdi Zhang, Markus Hadwiger, Thomas Theußl, and Peter Rautek. Interactive exploration of physically-observable objective vortices in unsteady 2d flow. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):281–290, 2022. doi: 10.1109/TVCG.2021.3115565.

Xingdi Zhang, Peter Rautek, and Markus Hadwiger. Generic Variational Spacetime Optimization of Vortex Core Manifolds. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–11, Los Angeles CA USA, July 2026. ACM. ISBN 979-8-4007-2554-8. doi: 10.1145/3799902.3811230. URL https://dl.acm.org/doi/10.1145/3799902.3811230.

J. Zhou, R. J. Adrian, S. Balachandar, and T. M. Kendall. Mechanisms for generating coherent packets of hairpin vortices in channel flow. *Journal of Fluid Mechanics*, 387:353–396, May 1999. ISSN 1469-7645. doi: 10.1017/s002211209900467x. URL http://dx.doi.org/10.1017/S002211209900467X.

Junwei Zhou, Duowen Chen, Molin Deng, Yitong Deng, Yuchen Sun, Sinan Wang, Shiying Xiong, and Bo Zhu. Eulerian-Lagrangian Fluid Simulation on Particle Flow Maps. *ACM Transactions on Graphics*, 43(4):1–20, July 2024. ISSN 0730-0301, 1557-7368. doi: 10.1145/3658180. URL https://dl.acm.org/doi/10.1145/3658180.

Lu Zhu and Li Xi. Vortex axis tracking by iterative propagation (vatip): a method for analysing three-dimensional turbulent structures. *Journal of Fluid Mechanics*, 866:169–215, March 2019. ISSN 1469-7645. doi: 10.1017/jfm.2019.75. URL http://dx.doi.org/10.1017/jfm.2019.75.

Table 7: **2D vortex-region extraction.** F1, precision and recall are computed for the vortex class from pooled test samples within each field, then averaged equally across Boussinesq and Cylinder. Neural models use a single run; the two traditional baselines are averaged over three clustering seeds. IVD supervision indicates training with vorticity-deviation labels. Unsupervised methods cluster test inputs, using labels only to match clusters for scoring. GMM denotes a Gaussian mixture model.

Method	IVD supervision	F1	Precision (%)	Recall (%)
Supervised with IVD-derived labels				
MVU-Net (Deng et al., 2022b)	✓	0.7933	**73.712**	86.384
VortexBoundary (CNN) (Berenjkoub et al., 2020)	✓	0.7654	67.263	90.401
VortexViz+Vorticity (de Silva et al., 2024)	✓	0.7621	73.526	81.262
VortexTransformer (Zhang et al., 2025)	✓	0.3080	18.375	97.998
VortexTransformer+IVD (Zhang et al., 2025)	✓	0.6240	57.867	71.655
Ours (supervised)	✓	**0.8024**	67.178	**99.843**
No label supervision during training				
Geometric features + GMM	✗	0.3515	22.636	85.455
Tangent displacement + GMM	✗	0.3046	18.732	**82.987**
Ours (unsupervised)	✗	**0.6828**	**81.427**	70.157

A Additional Experimental Results and Visualizations

Scattered particle data. Figure 5 uses particle trajectories of a cylinder wake at $Re = 3900$ (Rahimi Khojasteh et al., 2021); we use only particle positions from this dataset, without supplying velocity or vorticity to the geometric clustering. Stars do not require a grid: where the stencil vertices around a center particle are not occupied by particles, we interpolate between nearby particle trajectories to reconstruct the center curve and its 12 neighbor curves. With geometric features and a GMM, the particles in the shed vortex form one cluster that persists as they move downstream across the five snapshots. Trajectory geometry alone therefore separates vortical fluid from its surroundings in irregular particle distributions, a setting in which grid-based criteria cannot be evaluated without first reconstructing a velocity field.

Coreline predictions. Figures 7 and 9 compare StarVortex predictions with the human-refined corelines; F1 scores refer to the positive class at the selected frames. Isolated and nearly straight cores, as in the delta wing, the square cylinder, and the steady tornado, are recovered almost completely, with F1 between 0.95 and 0.98. Most errors occur in the half-cylinder wakes, where cores bend, branch, and cross: false positives lie close to the reference coreline inside the vortex tube, and false negatives concentrate at junctions and at the ends of cores. The lowest score, 0.88 on $Re = 640$, coincides with the most fragmented core network.

Comparison with other learning-based coreline extraction baselines. Figure 8 shows the same frames for StarVortex, the convolutional baseline, and PointNet; it uses the earlier StarVortex checkpoint and the 32^3 convolutional baseline, and the fixed-checkpoint evaluation on additional flows is reported in Section 4.3. Both baselines scatter false positives away from the cores; on the steady tornado, the convolutional model leaves gaps along the core, and PointNet predicts core samples outside the IVD surface. StarVortex produces continuous, tightly localized corelines and the highest F1 in all five frames, with the largest margins on $Re = 320$ (0.93 versus 0.69 and 0.76) and on the steady tornado, which is absent from training (0.97 versus 0.75 and 0.85).

B Extended Related Work

B.1 Vortex Extraction

Vortex Region Extraction. Classical detectors threshold scalar fields computed from velocity gradients. The non-objective criteria Q (Hunt et al., 1988), λ_2 (Jeong & Hussain, 1995), and Δ (Chong et al., 1990) depend on the observer (Günther & Theisel, 2018). Instantaneous and Lagrangian-averaged vorticity deviation (IVD and LAVD) are objective (Haller et al., 2016), but region extrac-

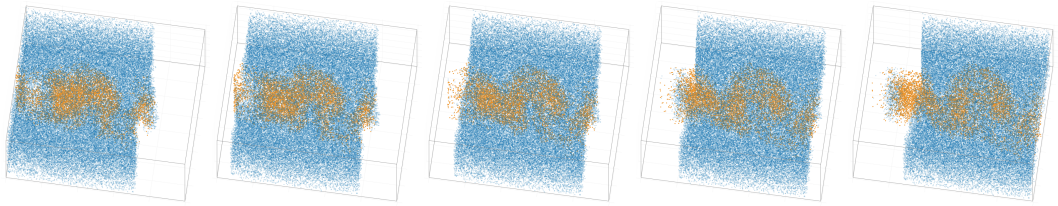

Figure 5: **Cylinder-wake clustering.** The five panels follow the same particles at successive times in a cylinder wake at $Re = 3900$. Orange and blue denote the two clusters from a geometric baseline. Each particle retains its cluster assignment across panels, and the gray box marks the same observation volume.

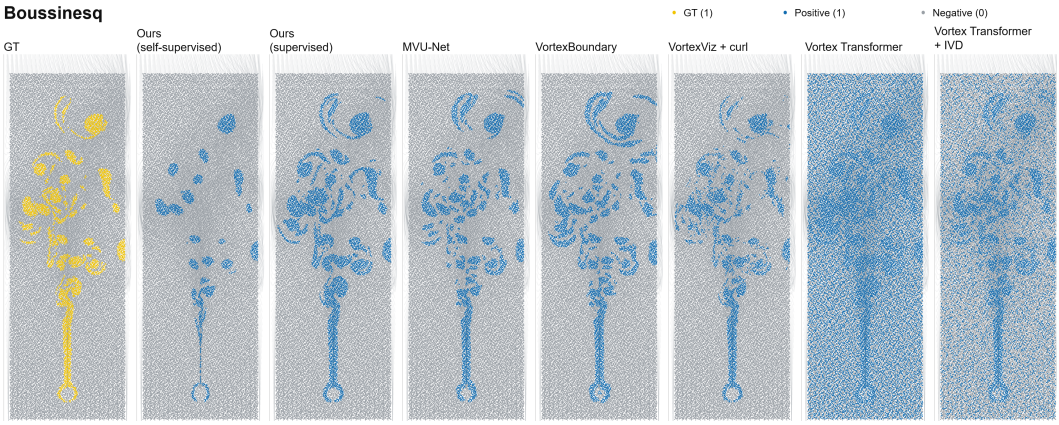

Figure 6: **Vortex region extraction on 2D Boussinesq flow.** First column: human-refined ground truth. Remaining columns: Ours (self-supervised), Ours (IVD-supervised), and other baseline results.

tion still requires an averaging domain and a threshold (Appendix C.2). Delocalized criteria also integrate local vortex measures along trajectories (Fuchs et al., 2008).

Learning-based Vortex Region Extraction. Learning-based region extractors use field quantities as inputs and learn from criterion-derived or expert-provided labels. Boosted classifiers (Zhang et al., 2014), CNNs (Deng et al., 2019; Kashir et al., 2021; Wang et al., 2021), and U-Nets (Deng et al., 2022b;a) classify or segment grid-based flow features; Berenjkoub et al. (2020) train on analytic Vatistas profiles but are limited to 2D and need a separate optimization step. VortexViz (de Silva et al., 2024), a detector using pathline images, classifies each pathline image separately and adds the accumulated curl as input; VortexTransformer (Zhang et al., 2025) processes groups of pathlines but still takes IVD as input. Related networks detect ocean eddies (Lguensat et al., 2018; Franz et al., 2018; Bai et al., 2019) and shocks (Liu et al., 2019). As Table 1 shows, the compared published configurations use at least one Eulerian quantity.

Vortex Coreline Extraction. A coreline is the curve around which the flow swirls. Parallel vectors (PV) methods (Sujudi & Haimes, 1995; Peikert & Roth, 1999; Roth & Peikert, 1998) locate alignments between velocity and acceleration or higher-order derivatives, and have been extended to unsteady flow (Weinkauf et al., 2007); their raw output is fragmented and noisy and must be filtered (Daßler & Günther, 2024). Objective methods instead take corelines as critical points in an optimal reference frame (Günther et al., 2017; Hadwiger et al., 2019; Rautek et al., 2021; Zhang et al., 2022; Rautek et al., 2024), as pathlines (Günther & Theisel, 2025), or as space-time manifolds (Zhang et al., 2026). These methods use the velocity field and its derivatives. The winding-angle method (Sadarjoen & Post, 1999) instead clusters looping streamlines in steady 2D flow and returns isolated core points. Kim & Günther (2019) learn reference frames for vortex extraction in 2D. We learn 3D coreline labels directly from curve geometry.

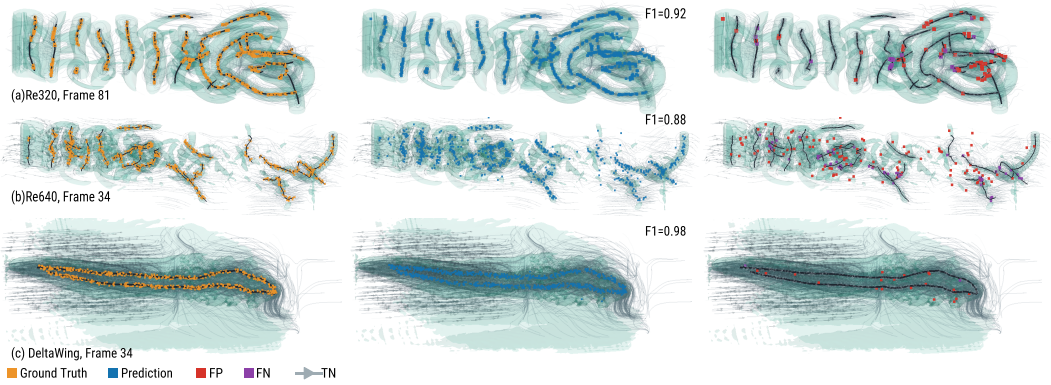

Figure 7: **Coreline extraction results.** The first column shows human-refined ground-truth corelines in orange. The second column shows StarVortex predictions as blue core samples. The final column highlights false positives in red and false negatives in purple. Rows show selected frames from three flows; background streamlines and translucent IVD surfaces provide spatial context.

Table 8: **Coreline extraction performance.** Positive-class F1 and precision by dataset. CD_h: Chamfer distance normalized by grid spacing (lower is better). Fully unseen excludes both training and checkpoint selection.

Dataset	Fully unseen	Conv3D (32^3)			PointNet			Ours		
		F1	$\mathrm{CD}_h \downarrow$	Precision	F1	$\mathrm{CD}_h \downarrow$	Precision	F1	$\mathrm{CD}_h \downarrow$	Precision
Cylinder ($Re = 160$)	No	0.8467	2.9331	0.7648	0.8079	3.7889	0.6992	**0.9333**	**1.8826**	**0.9311**
Half-cylinder ($Re = 320$)	No	0.7839	2.4621	0.7235	0.7471	3.1379	0.6982	**0.9182**	**1.8725**	**0.9139**
Half-cylinder ($Re = 640$)	No	0.8292	1.0658	0.7661	0.7562	1.2492	0.6923	**0.8831**	**0.9524**	**0.8551**
Delta wing	No	0.9558	0.6186	0.9292	0.9385	0.7272	0.9024	**0.9711**	**0.5909**	**0.9631**
Square cylinder	No	0.9384	1.2842	0.9015	0.9319	1.1622	0.9103	**0.9670**	**0.7488**	**0.9648**
Steady tornado	No[†]	0.8089	0.5443	0.9478	0.8456	1.5207	0.8845	**0.9764**	**0.5225**	**0.9564**
RFC bend	Yes	0.8070	4.9913	0.6764	0.8440	3.9833	0.7780	**0.9792**	**0.6201**	**0.9664**
RFC tilt	Yes	0.9540	0.6981	0.9120	0.9541	0.7640	0.9191	**0.9885**	**0.5172**	**0.9875**
Moving vortex ring	Yes	0.2676	0.5096	0.9727	0.7628	0.5613	0.7230	**0.9137**	**0.4099**	**0.9786**

[†]Steady tornado was absent from training but was included in the original test set used for checkpoint selection. The three fully unseen flows were evaluated only after all checkpoints were fixed.

Self-supervised objectives improve supervised learning. We further evaluate whether the proposed self-supervised representation-learning objectives can benefit supervised vortex recognition. Table 6 reports positive-class F1 scores for coreline extraction under full supervision, limited-label settings, and domain generalization. Adding the orbit-consistency objective $\mathcal{L}_{\mathrm{orb}}$ consistently improves over cross-entropy training alone across all settings. The benefit is particularly pronounced when labeled data are scarce: with only 500 training labels, adding both $\mathcal{L}_{\mathrm{orb}}$ and $\mathcal{L}_{\mathrm{recon}}$ improves F1 from 0.7415 to 0.7890. The same combination also gives the strongest domain-generalization performance, improving F1 from 0.8647 to 0.8916. In contrast, under full supervision the improvement is smaller, from 0.9303 to 0.9333 with $\mathcal{L}_{\mathrm{orb}}$. These results indicate that the geometric structure learned through the self-supervised objectives provides a useful regularization signal for supervised recognition, with the largest gains arising when direct supervision is limited or the model must generalize beyond the training domain.

Hairpin Vortex Extraction. Hairpin vortices, each comprising a spanwise head joined to two quasi-streamwise legs, dominate wall-bounded turbulence (Zhou et al., 1999; Adrian et al., 2000; Adrian, 2007; Wu & Christensen, 2006; Dennis & Nickels, 2011; Jodai & Elsinga, 2016). Region criteria such as λ_2, λ_{ci} (Zhou et al., 1999), and Rortex (Liu et al., 2018) do not separate neighbouring hairpins, so existing extractors add topological segmentation (Bremer et al., 2016; Zafar et al., 2024a) or axis tracking (Zhu & Xi, 2019), followed by geometric shape tests (Zafar et al., 2024b; 2026); all of them depend on a scalar criterion and its threshold. No learning-based hairpin extractor exists. Ours is the first, and it takes only the geometry of vortex lines as input.

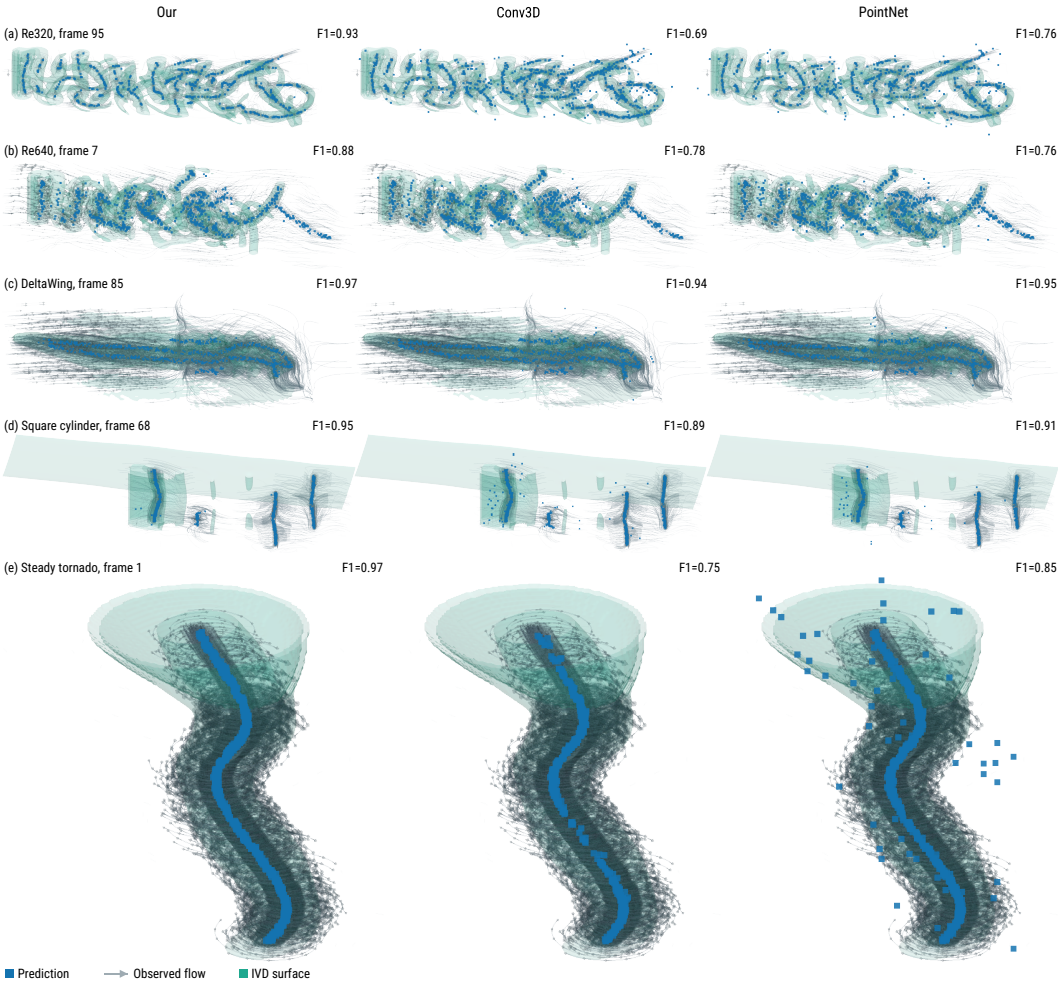

Figure 8: **Coreline comparison.** Columns show predictions from StarVortex, Conv3D, and PointNet (Qi et al., 2017a), with blue points indicating predicted core samples. Each row shows a selected frame from one of five flows. The methods share sample locations, background streamlines, reference IVD surfaces, and viewpoints for direct spatial comparison.

B.2 Lagrangian Representations

Particle advection underlies line integral convolution (Cabral & Leedom, 1993), finite-time Lyapunov exponents (FTLE) (Shadden et al., 2005; Garth et al., 2008; Kasten et al., 2010), and Lagrangian coherent structures (Haller & Yuan, 2000; Haller, 2015). Because pathlines integrated on temporally sparse velocity grids accumulate error (Griffa et al., 2004; Qin et al., 2014; Hummel et al., 2016), Lagrangian representations store integral curves computed in situ (Agranovsky et al., 2014; Bujack & Joy, 2015) and can improve reconstruction accuracy at comparable storage (Sane et al., 2018; 2022). Interpolation methods reconstruct new pathlines from particle data (Chandler et al., 2015; Li et al., 2022). Lagrangian particle tracking recovers trajectories from observations (Khojasteh et al., 2021); particle trajectory datasets support its evaluation (Rahimi Khojasteh et al., 2021).

B.3 Learning on Particles and Curves

Learning on Lagrangian particles. The machine learning community already treats Lagrangian particle data as a learning target. Graph network simulators (Sanchez-Gonzalez et al., 2020) and continuous convolutions (Ummenhofer et al., 2020) learn fluid dynamics directly on spatial particles, and LagrangeBench (Toshev et al., 2023) provides 2D and 3D particle-based benchmarks for such learned solvers. Particle and neural flow maps also carry the state of fluid simulators (Zhou et al.,

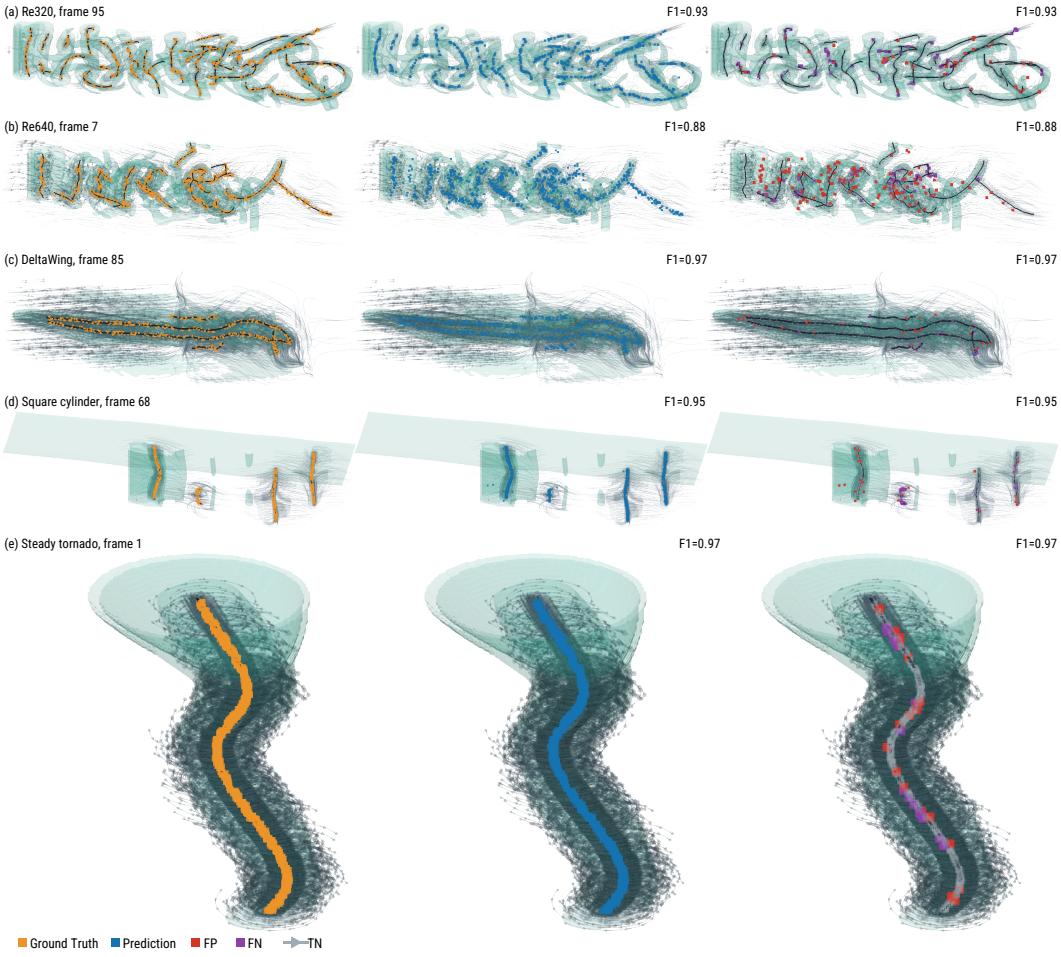

Figure 9: **Coreline extraction results.** The first column shows human-refined ground-truth corelines in orange. The second column shows StarVortex predictions as blue core samples. The final column highlights false positives in red and false negatives in purple. Rows show selected frames from five flows; background streamlines and translucent IVD surfaces provide spatial context.

2024; Deng et al., 2023b), and vortex dynamics can be learned from videos (Deng et al., 2023a). These methods predict how particles move from one frame to the next; however, none analyzes the structures that the trajectories reveal.

Learning on line geometry. In flow visualization, streamlines have been clustered (Yu et al., 2012), embedded in a low-dimensional space (Rössl & Theisel, 2012), and encoded by autoencoders for visual exploration (Han et al., 2020) or for retrieval by language queries (Zhang & Tao, 2026). Other networks reconstruct vector fields from streamlines (Han et al., 2019), predict trajectories from in situ flow maps (Han et al., 2022; 2024), or upsample and reconstruct flow maps (Jakob et al., 2021; Sahoo et al., 2022). Outside fluid dynamics, curves are learned directly as point sequences. Sketch-RNN (Ha & Eck, 2018) models pen strokes as sequences of offsets, VectorNet (Gao et al., 2020) encodes road maps and agent trajectories as polylines with a hierarchical graph network, and t2vec (Li et al., 2018) embeds GPS trajectories for similarity search. Streamlines from diffusion MRI tractography are clustered and parcellated by point-cloud networks (Chen et al., 2023; Xue et al., 2023b;a). These works establish the usefulness of learned curve representations; our focus is local vortex-region, coreline, and hairpin recognition.

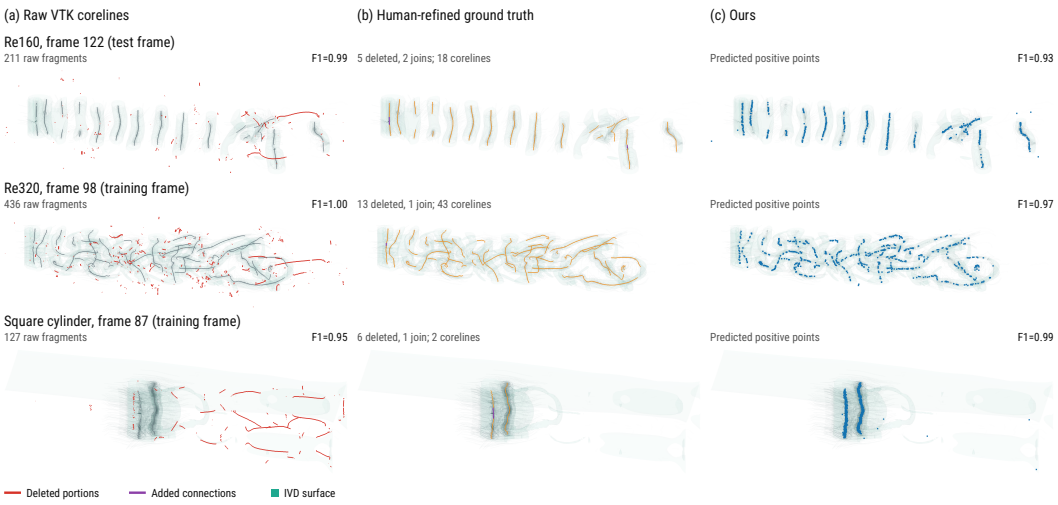

Figure 10: **Coreline annotation.** The first column shows raw VTK corelines, with discarded segments in red. The second column shows the human-refined ground truth, with added connections in purple. The third column shows StarVortex predictions as blue core samples. Rows illustrate one test frame and two training frames.

Table 9: Hairpin Extraction performance under three **cross-flow evaluation settings**. Each flow is used for testing, with the other two used for training. Bold values indicate the highest F1 for each test flow.

Test set	Model	F1	Accuracy (%)	Precision (%)	Recall (%)
TBL	Ours	**0.7664**	81.78	79.26	74.19
	Conv3D (16^3)	0.7215	74.52	64.47	81.91
	Conv3D (32^3)	0.7231	75.93	67.39	78.02
	PointNet Qi et al. (2017a)	0.6696	69.68	59.70	76.23
	PointNet++ Qi et al. (2017b)	0.6618	65.29	55.14	82.76
	Zafar et al. (2026)	0.4155	39.68	60.89	31.53
Channel	Ours	**0.4102**	69.51	70.99	28.84
	Conv3D (16^3)	0.4022	72.66	72.33	27.85
	Conv3D (32^3)	0.4077	72.40	69.95	28.77
	PointNet (Qi et al., 2017a)	0.3619	71.56	69.81	24.43
	PointNet++ (Qi et al., 2017b)	0.3596	69.68	55.14	26.68
	Zafar et al. (2026)	0.2930	41.87	48.77	20.94
Couette	Ours	**0.7960**	88.88	82.51	76.89
	Conv3D (16^3)	0.7699	83.99	77.80	76.20
	Conv3D (32^3)	0.7840	87.94	79.20	77.61
	PointNet (Qi et al., 2017a)	0.6259	83.32	85.14	49.48
	PointNet++ (Qi et al., 2017b)	0.6498	79.68	62.78	67.35
	Zafar et al. (2026)	0.4470	49.55	58.16	36.30

C Experimental Details

All labels are set or verified by hand; the subsections below describe annotation, data splits, and training. Unless stated otherwise, aggregate scores give each flow equal weight. The protocols distinguish clustering of available flows from prediction on held-out frames or flows.

C.1 Implementation and task-specific objectives

The three-dimensional encoder has two residual one-dimensional convolutional stages with 64 and 128 channels, followed by three Transformer layers. Each Transformer layer uses width 128, four attention heads, and feedforward width 256. The latent mean has 16 invariant-code and 12 pose-

code coordinates. The sinusoidal representation network (SIREN) decoder has four hidden layers of width 128 and sine frequency 30. The region model has 586,477 parameters, including the decoder. Coreline classification reads the first 16 mean coordinates; the implemented hairpin classifier reads all 28 mean coordinates. Thus the supervised implementations share the star encoder but differ in their readout and auxiliary objectives. Table 10 summarizes the optimization settings of each task. All models are implemented in PyTorch, and each run uses a single NVIDIA V100 GPU with 32 GB of memory; one coreline training run takes about 70 s.

Normalization. Inputs are the center curve and each neighbor relative to the center. We divide by one root-mean-square scale for the center and one shared scale for the neighbors, then clip vector magnitudes at $5\sqrt{3}$. Supervised experiments estimate these scales from training data only; region clustering estimates them from the unlabeled data.

C.2 Vortex Region Extraction

Instantaneous vorticity deviation. The vorticity of the fluid is $\boldsymbol{\omega} = \nabla \times \mathbf{v}$, and its spatial mean over a domain $U(t)$ is

$$\bar{\boldsymbol{\omega}}(t) = \frac{\int_{U(t)} \boldsymbol{\omega}(\mathbf{x}, t)\, dV}{\mathrm{vol}(U(t))},$$

where $\mathrm{vol}(\cdot)$ denotes volume in 3D and area in 2D. The instantaneous vorticity deviation (IVD) (Haller et al., 2016) is the deviation of the vorticity at a point from this mean,

$$\mathrm{IVD}(\mathbf{x}, t) = \left|\boldsymbol{\omega}(\mathbf{x}, t) - \bar{\boldsymbol{\omega}}(t)\right|.$$

Labels. Turning IVD into a vortex region mask requires the averaging domain $U(t)$ and a threshold, and neither transfers across flows. A small domain makes $\bar{\omega}$ follow the vortex itself, so the deviation vanishes inside large vortices; a large domain mixes several vortices and the background shear into one average. The IVD magnitude also depends on the velocity scale and vortex strength of each flow, so a fixed threshold either merges neighboring vortices or misses weak ones. An expert therefore inspects the masks and sets the threshold per flow and frame until the extracted regions match the visible vortices; this manual tuning is the step our learned model removes.

Data and splits. Table 11 lists the source-frame indices defining each supervised split; indices are zero-based. Every flow contributes 14 windows to the unlabeled clustering pool, including windows excluded from supervised fitting. Supervised training, validation, and testing use chronologically ordered windows. Eight flows use six training, two validation, and two test windows; Cylinder ($Re = 160$) and Half-cylinder ($Re = 6400$) use two windows per split. Each pathline spans 12 source-frame intervals, using 48 fourth-order Runge–Kutta updates of one-quarter source timestep and retaining 32 positions. Consequently, different start indices do not guarantee disjoint physical-time support. We propose a 16^3 seed grid with an 8% boundary inset and trace 12 neighbors at radius $0.5h$, where h is the minimum grid spacing. Incomplete or out-of-domain stars are excluded identically from all comparisons.

Table 10: StarVortex optimization settings. All tasks use AdamW with weight decay 10^{-4} and gradient-norm clipping at 1.

	Vortex region	Coreline	Hairpin
Training signal	self-supervised	supervised	supervised
Training length	6,000 updates	4,000 updates	20 epochs
Batch size	512 stars	512 stars	85 seeds $\times$ 3 lengths
Learning rate	5×10^{-5} enc., 5×10^{-4} dec.	10^{-3}	10^{-4}
Schedule	constant	400 warmup + cosine	2,000 warmup + cosine
Head dropout	–	0.15	0.15
Readout	GMM, 16-d invariant code	MLP, 16-d invariant code	MLP, 28-d code

Evaluation. The supervised splits contain 198,221 training, 75,726 validation, and 75,792 test samples across ten flows. All methods are evaluated on the same test locations and labels. Region F1, precision, and recall treat vortex samples as the positive class. For each run, we average each metric equally across the ten flows, then report the mean and sample standard deviation over three runs.

Table 11: Vortex region identification: source-frame starts for supervised training, validation, and testing.

Flow	Training	Validation	Testing
Channel	32, 44, 56, 69, 81, 93	105, 118	130, 142
Cylinder ($Re = 160$)	77, 88	100, 111	123, 134
Half-cylinder ($Re = 640$)	16, 21, 26, 31, 36, 42	47, 52	57, 62
Half-cylinder ($Re = 6400$)	77, 88	100, 111	123, 134
Tangaroa	41, 56, 72, 87, 102, 118	133, 148	164, 179
Delta wing (resampled)	35, 48, 61, 74, 87, 100	113, 126	139, 152
Delta wing (lattice Boltzmann)	47, 65, 83, 101, 119, 137	155, 173	191, 209
F22	32, 44, 56, 69, 81, 93	105, 118	130, 142
Boeing 747	40, 55, 71, 86, 101, 117	132, 147	163, 178
Buoyant smoke	32, 44, 57, 69, 81, 94	106, 118	131, 143

Self-supervised training and clustering. Each flow is trained independently with the settings in Table 10; The variance-floor penalty is disabled. Checkpoints are considered every 500 updates, including initialization. The invariant means undergo a float16 roundtrip and row-wise ℓ_2 normalization before clustering.

Comparators. The geometric baseline (**Geometric features + GMM**) standardizes speed, curvature, and torsion sequences and reduces them to two principal components. The displacement baseline (**Tangent displacement + GMM**) standardizes center and relative-neighbor tangent displacements without dimensionality reduction. Both use the same 13-curve samples, normalization before GMM fitting, and silhouette-based GMM selection as the learned representation. Their three runs vary GMM initialization seeds (7068, 7168, 7268), whereas neural-model runs vary training seeds. These standard deviations therefore measure different sources of variability. The supervised baselines are three-dimensional adaptations trained at the same sparse reference locations, rather than exact reproductions of the original dense two-dimensional protocols. MVU-Net (Deng et al., 2022b) and VortexBoundary (Berenjkoub et al., 2020) receive velocity; their augmented variants additionally receive vorticity and the label-defining vorticity deviation. VortexViz (de Silva et al., 2024) receives a rasterized center pathline, optionally with cumulative vorticity along that pathline. Models train for at most 100 epochs with AdamW, weight decay 10^{-4}, and early stopping after 20 unimproved epochs. Learning rates are 10^{-4}, 10^{-3}, and 5×10^{-4} for MVU-Net, VortexBoundary, and VortexViz, respectively. Checkpoints maximize validation average precision, and decision thresholds maximize validation positive-class F1.

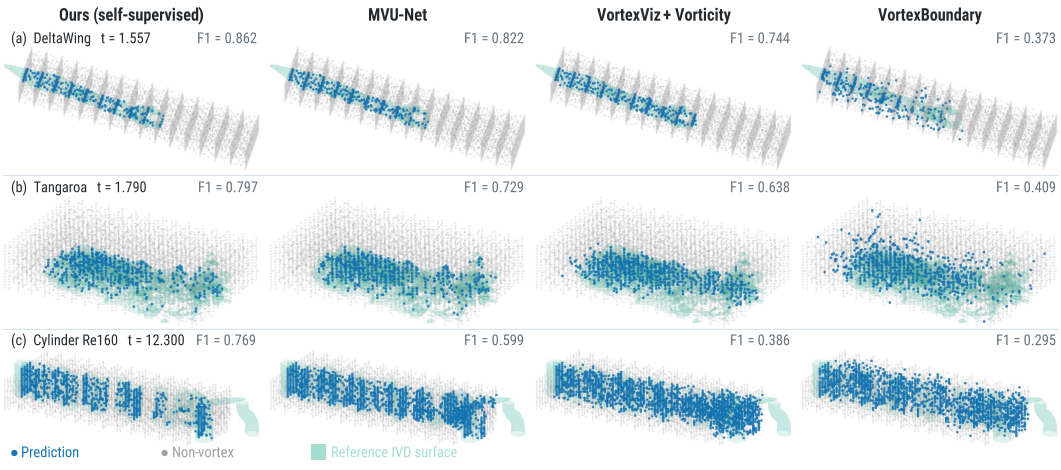

Figure 11: **Vortex-region extraction comparison.** The first column shows our self-supervised StarVortex predictions; the remaining columns show predictions from MVU-Net (Deng et al., 2022b), VortexViz (de Silva et al., 2024), and VortexBoundary (Berenjkoub et al., 2020). Blue and gray points denote predicted vortex and non-vortex samples, respectively. The translucent surface visualizes the expert-selected IVD threshold, and each row uses the same sample locations and viewpoint.

Table 12: Coreline identification: zero-based training, original test, and unseen-flow test frame indices.

Flow	Training	Testing
Cylinder ($Re = 160$)	75, 76, 82, 83, 90, 97, 98, 104, 105, 112, 113, 119, 120, 127, 134, 135, 141, 142	81, 95, 108, 122, 136
Half-cylinder ($Re = 320$)	75, 76, 82, 83, 89, 90, 97, 98, 105, 112, 113, 119, 120, 127, 128, 134, 135, 142	81, 95, 108, 122, 136
Half-cylinder ($Re = 640$)	1, 8, 16, 23, 30, 38, 45, 52, 60, 67	7, 21, 34, 47, 61
Delta wing (resampled)	9, 10, 25, 26, 43, 59, 60, 76, 77, 93, 94, 110, 111, 127, 144, 145, 160, 161	24, 55, 85, 115, 146
Square cylinder	7, 14, 16, 35, 45, 56, 65, 66, 75, 85, 87, 94, 95	15, 68, 86
Steady tornado	–	0
Unseen flows, evaluated after checkpoint selection		
RFC bend	–	2, 9, 16, 22, 29
RFC tilt	–	2, 9, 16, 22, 29
Moving vortex ring	–	2, 9, 16, 22

C.3 Vortex Coreline Extraction

Labels. We first run the vortex core filter of VTK (Schroeder et al., 2000; VTK Developers, 2026) on every frame. Flow experts then inspect the raw corelines frame by frame, remove spurious segments, merge fragments of the same core, and add cores the filter missed (Figure 10); red in the figure marks all raw segments absent from the final labels, including those removed by automatic filtering before manual refinement. Unlike a vortex region, a coreline specifies the one-dimensional curve around which the flow swirls; in 3D, it need not pass through a velocity critical point. A sample is positive if its exact distance to the continuous human-refined coreline segments is strictly below h. The VTK comparison measures agreement with reference labels refined from VTK output.

Data and splits. Table 12 lists the original 77 training and 24 test frames. Training and test times are interleaved; the steady tornado appears only in the original test set. Three unseen flows, the bent and tilted rotating-cylinder constructions (RFC bend, RFC tilt) and a moving vortex ring, are evaluated after checkpoint selection on the frames listed in the same table. The unmodified rotating-cylinder construction and moving-ring frame 29 were excluded during annotation review, before this evaluation. The bent and tilted cases share a construction family. Each frame stores the complete instantaneous observed velocity field; the steady case uses its original velocity. For each frame, 3,000 proposals mix 25% coreline-tube samples, 35% samples in the h–$8h$ shell, and 40% grid samples above mean vorticity deviation. We integrate bidirectional streamlines of total arc length 0.5 and retain 65 positions. Validity requires successful integration of the center and three 12-neighbor shells at radii $0.25h$, $0.5h$, and h; the network reads only the 13 curves at radius h. Accepted counts are 224,801 training, 70,317 original-test, and 41,860 additional-test samples. This label-conditioned sampling targets neighborhoods of candidate cores and does not estimate uniform whole-volume accuracy.

Optimization and selection. StarVortex uses class-weighted cross-entropy alone, with no reconstruction or view-consistency loss (Table 10). The original protocol selects the checkpoint with highest pooled test macro-F1 every 200 updates. Seeds 11, 23, and 37 give pooled test macro-F1 0.95535, 0.95606, and 0.95485, respectively (0.95542 ± 0.00061). These pooled scores differ from the equal-flow averages in the main table. The nine-flow comparison fixes StarVortex at seed 23/update 4,000, PointNet (Qi et al., 2017a) at seed 11/update 3,600, and the 32^3 convolutional model at seed 11/update 3,800. No additional-flow scores enter these choices. PointNet and the convolutional model use the same 13 curves, resampled to 32 positions and centered/scaled by the maximum Euclidean radius. The convolutional resolution study uses 8^3, 16^3, and 32^3 grids with otherwise fixed settings; all three results are retained. Table 8 reports per-flow scores.

C.4 Hairpin Vortex Extraction

Labels and data. We use the annotated hairpin dataset of Zafar et al. (2026) without modification. The three folds hold out Channel, turbulent boundary layer, and Couette flow in turn. All models use the same labeled seed locations. StarVortex traces 12 icosahedral neighbors at initial radius $2h$ and resamples each vortex line to 32 positions. The radius is reduced when required by domain

containment or integration validity. The convolutional and PointNet baselines retain their original seven-curve representation, so the comparison also changes the neighborhood representation.

Objectives and optimization. Training combines weighted cross-entropy with weight 0.1 on the Jensen–Shannon divergence between the three length-specific class distributions. StarVortex additionally reconstructs the original signal and two discrete symmetry views; its remaining settings are listed in Table 10. The negative-class weight is multiplied by 1.2 before normalization using training labels. All models use seed 96611, 20 epochs, AdamW weight decay 10^{-4}, threshold 0.5, and three-length majority voting. The convolutional and PointNet baselines use batches of 170 seeds, learning rate 2.5×10^{-4}, dropout 0.35, and gradient clipping at 5. PointNet also retains its feature-transform orthogonality regularizer.

From bundle predictions to surfaces. We first remove isolated positive predictions. A predicted-positive bundle is retained only if the $8h$-wide cube centered at its seed contains at least ten other positive seeds and at least 70% of the seeds in that cube are predicted positive, where h denotes the minimum mean grid spacing across axes. Both criteria exclude the query seed and are evaluated on the original predictions, so the result does not depend on the order in which bundles are processed. We then group the retained bundles by spatial connectivity and, for each group, build a cell mask on the native grid using a local λ_2 threshold calibrated on the training data. A cell is selected only if it lies within $4h$ of a near-seed, subthreshold curve segment and passes the threshold check at every point along its connection to the nearest curve sample. We keep the connected components of the mask that are adjacent to a seed and extract their surfaces with marching cubes. Finally, we discard any component whose volume is at least 80% covered by a component that has already been retained.