Received June 8, 2021, accepted July 14, 2021, date of publication July 26, 2021, date of current version August 16, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3100127

Deep Regression Network-Assisted Efficient Streamline Generation Method

JOONG-YOUN LEE[1,2], (Member, IEEE), AND JINAH PARK[2], (Member, IEEE)
[1]Supercomputing Center, Korea Institute of Science and Technology Information, Daejeon 34141, South Korea
[2]School of Computing, Korea Advanced Institute of Science and Technology, Daejeon 34141, South Korea

Corresponding author: Jinah Park (jinahpark@kaist.ac.kr)

This work was supported by Korea Institute of Science and Technology Information (KISTI).

ABSTRACT Streamlining is one of the most frequently utilized visualization methods to analyze the flow structure of computational fluid dynamics (CFD) data. However, it is challenging to find a set of streamlines showing the most prominent flow across the entire flow field due to the heavy computation time required to generate bundles of streamlines. In this paper, we propose an efficient streamline generation method that removes several seed candidates that are predicted as less important using a 3D U-net based regression model. We employ 3D line integral convolution (LIC) volumes that depict the entire flow field for training data of the proposed learning model and evaluate our method using a real-world CFD data set. We find using our model that we can obtain quality of visualization results comparable to that of the ground truth even when more than 90% of the seed candidates are truncated while operating 6.6∼17.1 times faster than the competing method.

INDEX TERMS Deep regression network, dilated convolutional neural network, flow visualization, line integral convolution, streamline.

I. INTRODUCTION

Streamline is a curved line which is instantaneously tangential to the velocity vector in the flow field. It is one of the most frequently utilized visualization methods used to analyze fluid dynamics data because it shows global flow streams intuitively. However, it is challenging to determine which seed locations may generate streamlines showing important flow features, such as the vortex area. Much time is needed to compute bundles of streamlines and find the seed locations for the best streamlines across the entire flow field. Users typically generate them by trial and error, which requires substantial human effort.

Meanwhile, the convolutional neural networks have been researched aggressively in relation to image classification [1]–[4] and segmentation [5]–[8] over the past decade. 3D extensions of these techniques followed to analyze and segment 3D medical volumes such as those from CT, PET, or MRI data intelligently as well [9]–[11]. Many semantic segmentation networks have adopted an encoder-decoder style of architecture based on a fully convolutional network (FCN) [5] for pixel-level classification to segment input images.

The associate editor coordinating the review of this manuscript and approving it for publication was Lei Wei.

This paper introduces a 3D U-net-based deep regression model that predicts the importance of streamlines generated from densely located seed points in a 3D velocity field. Furthermore, we present an efficient streamline generation method that creates bundles of streamlines for the extraction of important flow features using the proposed learning model. We apply dilated convolution instead of the standard convolution for the lower layers of the 3D U-net to address the checkerboard artifacts introduced by U-net's pooling-upsampling architecture.

Unlike most other FCN based models, our network uses regression to predict the floating-point values of importance scores across all voxels, in other words, it undertakes a voxel-wise regression. This step is designed to predict the importance of the streamlines generated from resampled seed points in the input flow field. Our regression model is trained by means of the 3D Line Integral Convolution (LIC) [12], [13] volume, which depicts the global 3D flow of the 3D velocity vector field. Given a streamline σ, line integral convolution consists in calculating the intensity for a pixel located at $x_0 = \sigma(s_0)$ by the following equation [13]:

$$I(x_0) = \int_{s_0-L}^{s_0+L} k(s-s_0)T(\sigma(s))ds. \tag{1}$$

where T denotes an input white noise, $k(s)$ is the convolutional kernel, and L is a length of the streamline segment. Our approach is motivated by the idea that 3D LIC generates volumes capturing flow streams that are visually recognizable by humans such that learning models can recognize them.

We propose an efficient streamline seeding method capable of removing numerous seed candidates that are predicted as less important after the use of the proposed learning model. This greatly reduces the computing time for the streamline calculation if seeds generating uninteresting streamlines are eliminated before the process of particle integration. The deep regression model does not judge which seeds are essential but allows the user to decide this by setting their own range of importance to select the streamlines to be drawn. Furthermore, we present a streamline selection method based on a curved-line clustering algorithm to address the visual cluttering problem as well.

The contributions of this paper are as follows:

- We introduce a deep regression model which captures the global flow trends of the 3D velocity field. We present several optimization techniques that enhance the prediction results as well.
- We propose a novel seed placement method that extracts streamlines showing important flow features in an interactive time using the proposed learning model. To the best of our knowledge, in the context of streamline seeding for 3D flow data, no work has yet done this.

The remainder of this paper is organized as follows: related works are presented in Section 2. An outline of the proposed method is described in Section 3. In Section 4, the 3D U-net based regression model for the learning of flow trends is introduced, while a streamline seeding method using the proposed model is presented in Section 5. Experiments using synthetic and real-world CFD data follow in Section 6. Subsequently, discussions are presented in Section 7. Finally, we conclude the paper in Section 8.

II. RELATED WORKS

This paper deals with the deep learning-based streamline generation method closely related to two major research topics - streamline seeding and deep learning model. This section introduces dozens of related works by dividing them into two categories: streamline visualization and machine learning techniques for flow data analysis.

A. SEED PLACEMENT AND STREAMLINE SELECTION

Various methods have been introduced so far to tackle seed placement and streamline selection problems. Li and Shen [14] presented an image-based streamline generation method to prevent the overlap of streamlines, making it difficult for humans to perceive. Burge *et al.* [15] introduced an importance-driven particle tracing method that reduces visual clutter while revealing important flow structures. Spencer *et al.* [16] suggested a similar method which creates evenly spaced streamlines on the surface in a 3D space.

Several metrics measuring the importance and similarity of streamlines have been proposed as well. Methods based on information entropy theory [17] capable of selecting interesting streamlines among numerous candidates were introduced [18], [19]. Yu *et al.* [20] and Lu *et al.* [21] adopted a curvature and torsion pair to measure the similarity of streamlines, while McLoughlin *et al.* [22] added a new metric, in their case tortuosity, to enhance the measurement of the similarity of curved lines. Chaudhuri *et al.* [23] proposed a novel metric, a box-counting ratio, which measures the complexity of the geometry of curved lines by measuring their space-filling capacities at different scales.

Recently, a survey by Sane *et al.* [24] provided a thorough overview of this research field. They divided the techniques into three different categories - density-based, feature-based, and similarity-based methods - for automated techniques and two categories - interactive tools and domain information - for manual techniques. They evaluated the automated techniques in terms of redundancy, regions of interest, and computation as well.

Even though many studies were introduced to tackle streamline seeding problems so far, most of them assumed that the streamlines were precomputed and disregard this despite the tremendous amount of computation time required. We propose a streamline seeding strategy that reduces the vast streamline computation time using the deep regression network in this paper.

B. FLOW DATA ANALYSIS USING MACHINE LEARNING

Several vortex extraction methods have employed deep learning approaches over the last decade. Many of these had the goal of detecting eddies in oceanographic data. Lguensat *et al.* [25] introduced EddyNet, a deep neural network for the pixel-wise classification of oceanic eddies from sea surface height maps. This network is based on U-net [6], which was developed for the segmentation of biomedical images. Franz *et al.* utilized a combination of a CNN and a RNN to detect and track ocean eddies classified by the Okubo-Weiss method [26], [27]. Bai *et al.* [28] proposed a streampath-based region-based convolutional neural network(SP-RCNN) that detects ocean eddies from streampath images. Duo *et al.* [29] presented a deep learning approach based on a deep residual network (ResNet) [4] and a feature pyramid network (FPN) [30] to detect oceanic mesoscale eddies.

More general flow feature extraction methods have been presented as well. Bin and Yi [31] proposed a CNN-based model which extracts various flow features such as clockwise vortices, anti-clockwise vortices, and saddles. Ströfer *et al.* [32] introduced what they termed a Fluid R-CNN, a model based on an R-CNN [33] that can identify various flow features including 2D recirculation regions, 2D boundary layers and 3D horseshoe vortices. Liu *et al.* [34] proposed a CNN-based shock-wave detection method and a novel loss function to optimize the detection results. Deng *et al.* [35] and Wang *et al.* [36] presented a novel vortex identification method based on a CNN. They utilized the instantaneous vorticity deviation (IVD) to label the ground truth vortex area for the proposed CNN-based model. Kim and Günther [37] proposed a novel CNN-based reference frame extraction method. Li *et al.* [38] presented a method by which to extract flow features using a binary support vector machine (SVM) which performs supervised streamline segmentation. Hong *et al.* [39] employed a Long Short-Term Memory (LSTM) network, which is a type of artificial RNN, to enhance the performance of parallel particle tracing in flow visualization using a novel access pattern estimation approach. A CNN-based flow field reduction scheme was presented as well [40]. This scheme utilized a CNN-based model which learns the flow features of velocity vector fields from bundles of 3D streamlines. Han *et al.* [41] presented FlowNet, a deep learning framework for the clustering and selecting of streamlines/stream surfaces. Jakob *et al.* [42] proposed an extensive fluid flow data set for application to deep learning problems in scientific visualization. The data set was employed in a recent study of CNN-based vortex boundary identification [43].

As we have seen so far, various methods were introduced in recent decades for flow data analysis, including streamline visualization and machine learning techniques for flow data analysis. This paper presents a novel importance metric to measure streamlines to select ones to display while introducing a CNN-based deep learning technique that instantly predicts the metric from the 3D flow field. Our deep learning model is similar to the EddyNet [25] and SP-RCNN [28] for a few reasons. Both EddyNet and our network are based on the U-net [6], but the EddyNet detects local eddies, whereas ours learns global streamlines and predicts their importance. Both SP-RCNN and ours learn flow features from the images generated from the flow field instead of the raw velocity vector fields. However, SP-RCNN relies on the region-based convolutional neural network(R-CNN) [33], [44] while ours are based on the U-Net, which is usually adopted in medical image analysis. Furthermore, both EddyNet and SP-RCNN deal with 2D data only, while our network extends it to 3D.

III. OUTLINE

We present a deep learning-based approach that generates streamlines showing essential flow features efficiently. We employ a deep regression model trained to predict the importance scores of all streamlines instantaneously throughout the entire set of seed candidates in the flow field. A regression model is utilized to remove seed points predicted as less critical from the seed candidates to reduce the computation time required to create streamlines. We achieve quality of the visualization results comparable to that of the ground truth despite the fact that 90% of the seed points are removed in far less time. The workflow of our method is illustrated in Fig. 1. The proposed method can be divided into two processes: training and streamline generation.

A. MODEL TRAINING

The proposed model receives a set of LIC volumes as the training data and the corresponding importance maps as the target data to learn the seed importance distribution in the input 3D flow. The raw flow data are resampled to a rectilinear grid to fit into the proposed deep learning model. The input LIC volumes are created by applying a 3D LIC operation to the resampled flow volumes, while the target data are generated by calculating the importance score of the streamlines that are created from the entire set of resampled voxels such that importance maps with resolution identical to that of the LIC volume are produced.

B. STREAMLINE GENERATION

The trained model predicts the importance score of all voxels in the unseen flow data to remove voxels predicted as being of low importance from the seed candidates. However, because many critical seeds may be concentrated in a few small regions, they may generate visually cluttered and similarly looking streamlines if all of them are drawn together. To overcome this problem, a streamline selection method based on curved-line clustering is added at the end of our approach. Similar streamlines are grouped by k-means clustering using several streamline features, and a few representatives per group are finally drawn. In the subsequent sections, we describe our deep regression model and streamline seeding method in detail.

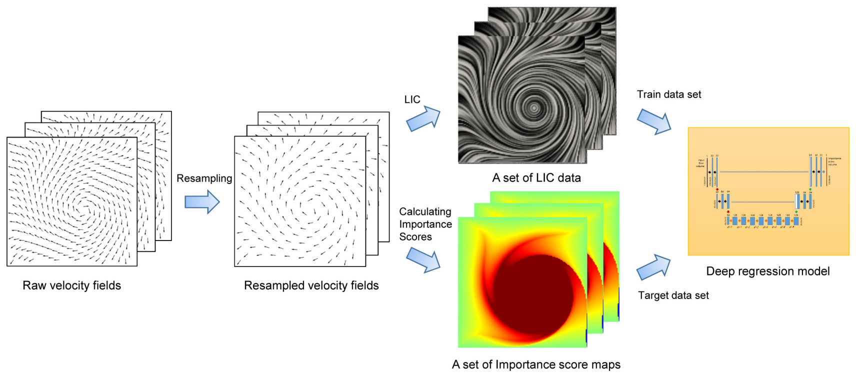
IV. DEEP REGRESSION MODEL FOR THE 3D VELOCITY FIELD

We adopt the encoder-decoder style architecture for the proposed regression model, as this type is commonly utilized for semantic segmentation. Because both models perform voxel-wise inference, we believe that the encoder-decoder architecture is applicable for our purposes here. The proposed deep regression network is modified from the 3D U-net. It receives the LIC volume derived from the 3D velocity field as training data instead of the raw velocity vectors, so to utilize the image recognition capabilities of existing segmentation models.

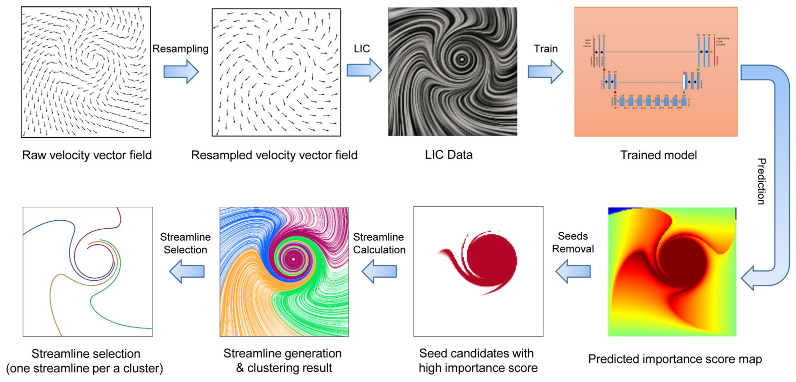
Details about the input training data, the target data to predict, and the architecture of our deep regression model are described in the following subsections - ‘‘LIC Volume’’, ‘‘Importance Score’’ and ‘‘Deep Regression Network’’, respectively.

A. LIC VOLUME

LIC is a well-known dense visualization technique for vector fields, which adopts a low-pass filter to convolve input white noise along pixel-centred symmetrically bidirectional streamlines to exploit spatial correlations in the flow direction [45]. The 2D LIC technique is often used to visualize a 2D velocity field because it shows the global flow trends while not requiring streamline seeding. However, the 3D LIC for flow visualization is very limited due to perceptual challenges such as depth perception, occlusion, and visual complexity [46].



(a) A procedure to train the proposed deep regression model


Raw velocity vector field
Resampling
Resampled velocity vector field
LIC
LIC Data
Train
Trained model
Prediction
Predicted importance score map
Seeds Removal
Seed candidates with high importance score
Streamline Calculation
Streamline generation & clustering result
Streamline Selection
Streamline selection (one streamline per a cluster)

(b) A procedure of the streamline generation based on the trained model

FIGURE 1. Overall procedures of the proposed streamline seeding method. (a): Set of LIC data and the importance score map is created from the raw velocity vector fields. The LIC data is used as train data set while the importance score map is adopted as target data to be predicted by the trained model. (b): The trained model predicts the importance score map from the unseen flow data. Seed candidates predicted as low importance are truncated to reduce total computing time. The streamline selection based on the curved-line clustering is followed.

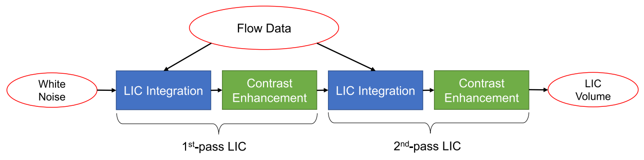
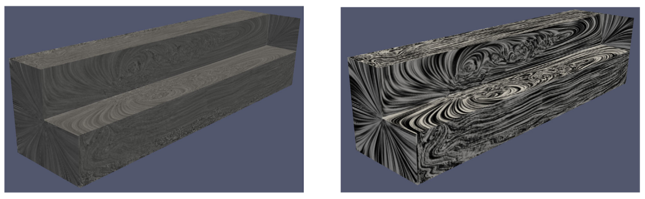


FIGURE 2. Two-pass 3D LIC operation with contrast enhancement.



(a) Basic 3D LIC (b) Two-pass Enhanced 3D LIC

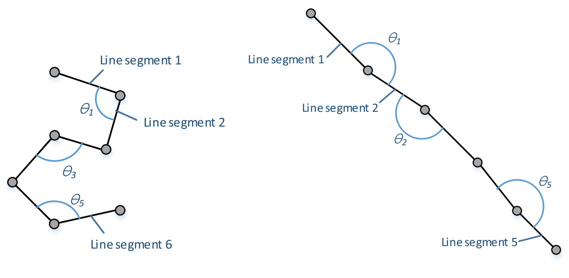
FIGURE 3. LIC volumes with different configurations.

We employ the 3D LIC volume as the training data instead of the raw velocity field in this work. A semantic segmentation network for the 3D volume data is also adopted to analyze the 3D LIC volumes to learn the flow trends in the original velocity fields. Moreover, there is another side benefit related to the memory usage. The LIC volume requires less memory than the velocity vector because the 3D velocity vector consists of three components (u, v, and w), whereas LIC volume requires only one component.

To convert a 3D velocity field into a LIC volume, we employed the 3D version of Enhanced LIC [47], [48]. Because the basic LIC algorithm averages the texel values of white noise that lie on an integrated path of a streamline, the approach often produces blurry images in which not only humans but also deep neural networks can barely recognize. The Enhanced LIC method addresses this problem by means of a two-pass LIC and image enhancement technique. A well-known histogram equalization is employed for the LIC contrast enhancement in this paper. Two-pass LIC performs the LIC operation twice, where the output image of the first execution becomes the input image of the second execution in place of the white noise. Fig. 2 shows a modified version of our LIC procedure and Fig. 3 presents a comparison of the image quality outcomes between the basic LIC and the enhanced LIC approaches.

B. IMPORTANCE SCORE

We introduce a new metric, referred to as the importance score, to measure each streamline's significance. This score



(a) High importance score (b) Low importance score

FIGURE 4. Two line segments with different importance scores.

is calculated by (2).

$$\text{importance score} = \text{twistedness} + \text{coverage} \tag{2}$$

Here, twistedness refers to how much the streamline is twisted, while coverage assesses how far it reaches. Because twistedness and coverage can have different scales, they are standardized by means of z-score normalization, which is less sensitive to outliers than min/max normalization.

The equation used here to calculate the twistedness of a streamline is expressed below.

$$\text{twistedness} = \sum_{i=1}^{N-1} \left(\frac{\cos\theta_i + 1}{2} \right), \quad 0 \le \theta_i \le \pi \tag{3}$$

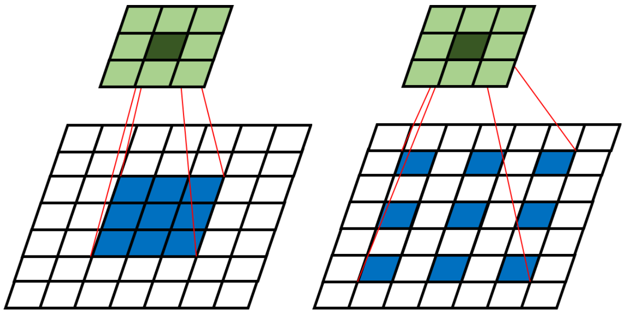
Here, N denotes the number of line segments in the streamline while θ_i indicates the inner angle between the i-th segment and the $i+1$-th segment as illustrated in Fig. 4. Owing to the cosine function, the score increases for a smaller angle and decreases for a larger angle. The cosine values are rescaled from $[-1, 1]$ to $[0, 1]$ so as to avoid the generation of negative scores. In the case of Fig. 4, the streamline (a) has much smaller inner angles than streamline (b); we can interpret this as meaning that line (a) is more twisted than line (b), which in turn means that the twistedness of line (a) is much higher than that of line (b).

As noted above, coverage indicates how far the streamline reaches. This value is calculated using the volume of an axis-aligned bounding box(AABB) of the streamline. When the min point of AABB is (x_0, y_0, z_0) and max point is (x_1, y_1, z_1), the coverage of the streamline is calculated by (4).

$$\text{coverage} = |x_1 - x_0| \times |y_1 - y_0| \times |z_1 - z_0| \tag{4}$$

We employ the proposed importance score as the target value of our network. The streamlines should be calculated from every voxel in the training flow volume in both forward and backward directions. The importance map of the velocity field is generated at the preprocessing time as follows:

1) Create bidirectional streamlines from every voxels in the training data.
2) Calculate the importance scores of each streamline.
3) Assign the calculated score to the seed voxel.
4) Apply normalization to the calculated importance score.



(a) Standard convolution (b) Dilated convolution

FIGURE 5. The illustration of the 2D standard convolution with kernel size 3 × 3 and the dilated convolution using the identical kernel size with a dilated rate *d* = 2.

C. DEEP REGRESSION NETWORK

Since U-net [6] was originally introduced, given its accurate segmentation of images, variants of U-net are often applied in several different areas such as volume segmentation [9], eddy detection [25], and vortex boundary identification [43] among others. U-net is a fully convolutional network for biomedical image segmentation that also adopts the encoder-decoder scheme to enlarge its receptive field quickly by pooling layers. This elegant architecture introduces a long skip connection from the encoder layer to the corresponding decoder layer to feed fine-grained details, which may be lost due to the series of downsampling steps. However, there are checkerboard artifacts introduced by the upsampling operations, which is an unavoidable and severe drawback [49]. To address these problems, several studies have proposed replacing the normal convolutional filters with a dilated convolution strategy [8], [50]–[52]. The dilated convolution [50] refers to a type of convolution that skips pixels on the processing layer to extend the receptive field while holding the number of parameters constant as illustrated in Fig 5.

In this paper, we present a deep regression network based on a dilated 3D U-net approach. Fig. 6 illustrates the architecture of the dilated 3D U-net. It is divided by three parts - the encoder part (A) and the decoder part (B), which are connected to each other via long skip connections, and a dilated convolution part(C). Parts (A) and (B) are equivalent to the corresponding parts in the original U-net. The last one - the dilated convolution part (C) replaces the lower layers of the original U-net with a chain of dilated convolutions to address the checkerboard artifacts. Because dilated convolution enlarges the receptive fields without downsampling feature maps, we do not need to perform upscaling, which may introduce checkerboard artifacts. In Fig. 6, the black box is the input LIC volume while the red box is the volume of importance score, which is the target map. The blue box is a feature map which is assumed to have passed the convolution and pooling filter, and the white box is a copy of the feature map from the contraction path via a long skip connection. The black arrow indicates a series of processes which are in this case 3D convolution, ReLU activation, and batch normalization. The orange arrow is equivalent to the black arrow except that it employs dilated 3D convolution

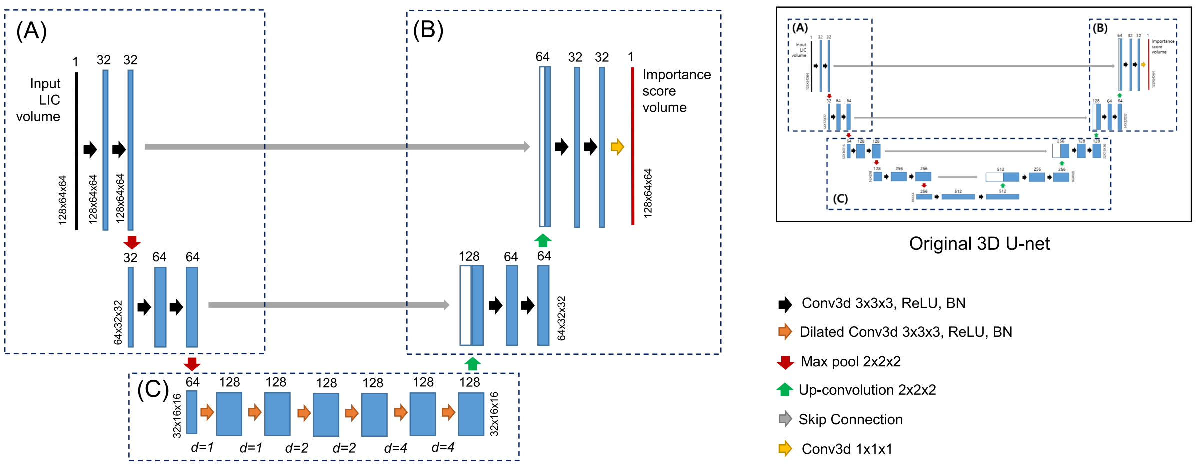


FIGURE 6. **An architecture of the dilated 3D U-net. The image in the box in the upper right corner depicts the architecture of the original 3D U-net.**

instead of the standard 3D convolution, while the "$d = N$" label below the arrow indicates the dilation rate. The red arrow is the 3D max pooling, and the green arrow refers to 3D up-convolution. The yellow arrow is the $1 \times 1 \times 1$ 3D convolution used here, which maps multichannel feature maps to a single-valued target map. The long grey arrow is a long skip connection. The labels beside the boxes present the volume resolution. In this paper, we adopt $128 \times 64 \times 64$ for the input flow volume. The label over the box shows the number of channels of the feature map. Comparing the two architectures (C) of both architectures perform similar operations. Both receive feature maps from the corresponding part (A) and then execute the learning process by enlarging the receptive field twice. The original 3D U-net enlarges the receptive field by pooling, which may lead to checkerboard artifacts. Moreover, it enforces a more extensive network due to the expanded path and duplicated feature maps, as depicted by the white box.

D. DATA AUGMENTATION

Numerous input data are required to train a robust deep learning model which is not overfitted to the training data. Despite the effort to collect CFD data for machine learning [42], such data are not applicable to our work because all of them are 2D flows. Many previous works employed data augmentation schemes [6], [8], [9], [52] to address a lack of training data. This paper employs data augmentation by applying transformation methods such as rotation, flipping, and scaling to the original data. Furthermore, we adopt another augmentation technique: the addition of extra LIC volumes generated by random white noise. Fig. 7 shows LIC volumes generated using different white noises with an identical velocity vector field.

V. STREAMLINE SEED PLACEMENT

In this chapter, we present an efficient seed placement method based on the proposed regression model. We propose a streamline seeding strategy that reduces the vast streamline computation time with the use of the proposed deep regression model. Our model removes numerous less essential seeds, which greatly reduces the computation time for streamline integration and curved-line clustering while maintaining prominent streamlines.

Our method consists of six steps:

1) Resample the input flow volume to fit the deep regression model
2) Predict importance scores using the model
3) Remove seeds predicted as less important. Most uninteresting seeds are truncated during this step.
4) Create streamlines from the seeds remaining after the prior step.
5) Compute the actual importance score from the created streamlines.
6) Perform curved line clustering of the created streamlines to select representatives to be displayed among many similar ones.

The flow volume is resampled in the first step. Because our model can obtain flow volumes with predetermined resolutions only as input data, the input flow data must be resampled to fit the model. To avoid visualizing uninteresting streamlines, seed candidates predicted as less critical are removed in the third step. In the fourth step, streamlines are generated from the seed points which pass the previous step, and accurate importance scores are calculated. Because numerous seed points have been removed by this step, we can save a considerable amount of time during this stage. The accurately calculated scores are utilized for curved-line clustering in the next step. Streamlines to be drawn are determined using the seed selection method based on curved-line clustering to avoid rendering redundant streamlines. During this step, similar streamlines are grouped, and representative streamlines are selected for each group.

We adopted the curved-line clustering method to select streamlines from among the many streamlines created in the

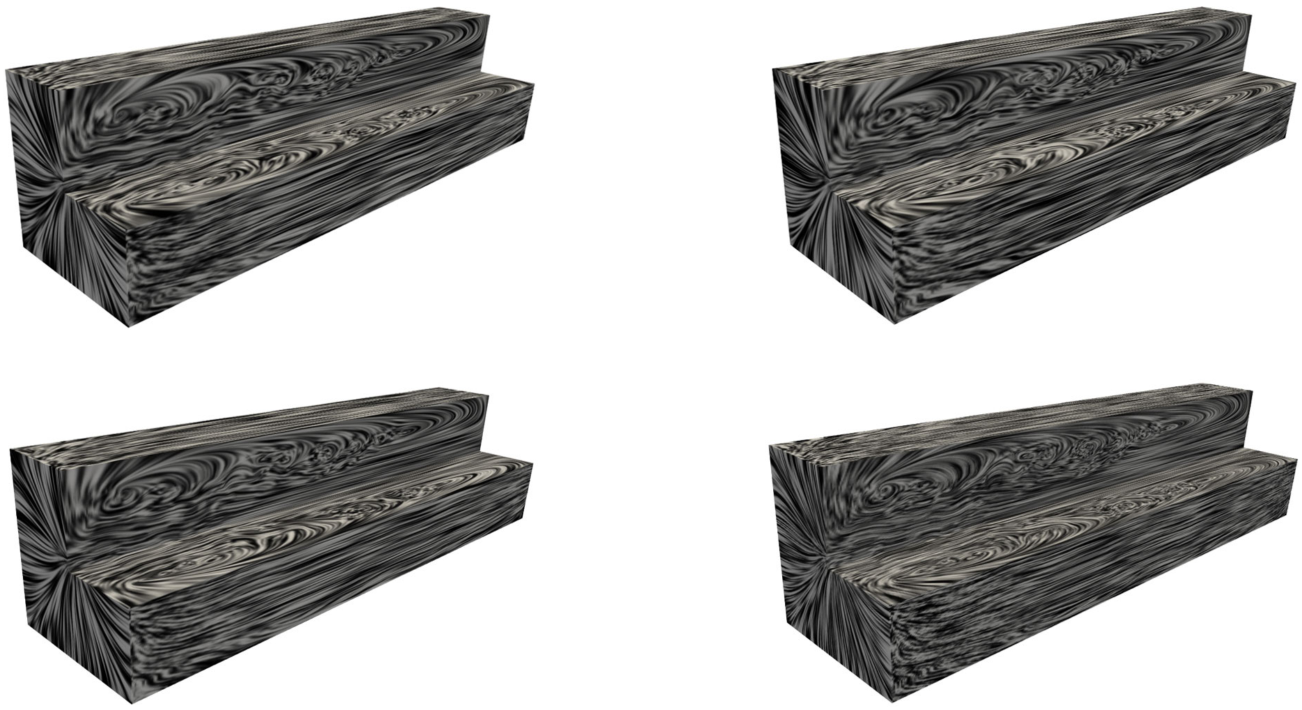


FIGURE 7. Diverse LIC volumes for the data augmentation. A corner of each volume is clipped away to show the inside of the data. The flow patterns in each data are slightly different due to the randomly generated white noises even if they are calculated on the identical flow volume.

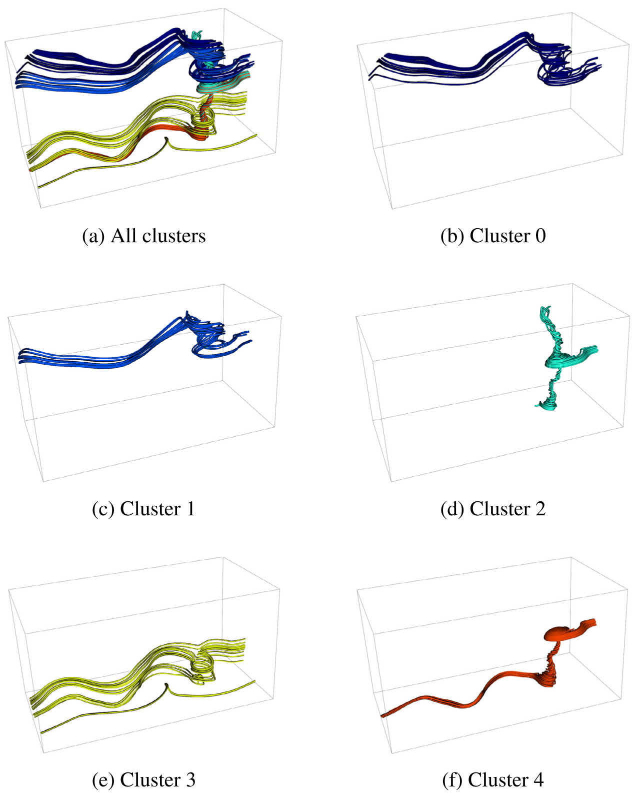


FIGURE 8. Examples of the first k-means clustering using the spatial properties. Streamlines with similar positions are grouped. Half cylinder data set is employed. Color coded by the first cluster ID.

previous step. Our method borrows from the literature [53], specifically one study that employed two-pass k-means clustering using two different properties of spatial and shape for the clustering features. The spatial property consists of the start point, the middle point, and the end point of the streamline, while the shape property includes the linear and angular entropy. Instead of an entropy-based shape property, we adopt an importance-based shape property consisting of twistedness and coverage, which are introduced in relation to the importance score. Twistedness refers to the amount of angular variation along a streamline similar to angular entropy, while the coverage score indicates the amount of spatial occupancy. Fig. 8 depict the examples of the first k-means clustering, while Fig. 9 show the examples of the second k-means clustering.

Similar streamlines are grouped via the curved-line clustering method, and a few representative lines are displayed in the final step. The streamlines with the highest importance scores are selected as the representative streamlines in each cluster. The process of streamline selection is as follows:

1) Extract the spatial and shape properties from the streamlines
2) Perform the first k-means clustering using the spatial properties of the start point, middle point and end point of the streamline
3) Perform the second k-means clustering using the shape properties of the twistedness and coverage scores of the streamline on every cluster generated during the first clustering step
4) Select and display the most important streamlines as representatives of each streamline cluster

TABLE 1. Experiment data information. Four data sets are applied in the experiments. The half cylinder and the Tangaroa data are cropped, while the solar plume and the 3D double gyre data are resized to identical resolution. Every data set is divided into three categories - train, validation, and test as a ratio of 5.6: 1.4: 3.

Data Set	Original Resolution	Converted Resolution	Conversion Method	Number of Data Samples			
				Total	Train	Validation	Test
Half cylinder	640×240×80	128×64×64	Crop	450	252	63	135
Solar plume	512×126×126	128×64×64	Resize	82	46	12	24
Tangaroa	300×180×120	128×64×64	Crop	200	112	28	60
3D double gyre	256×128×128	128×64×64	Resize	100	56	14	30

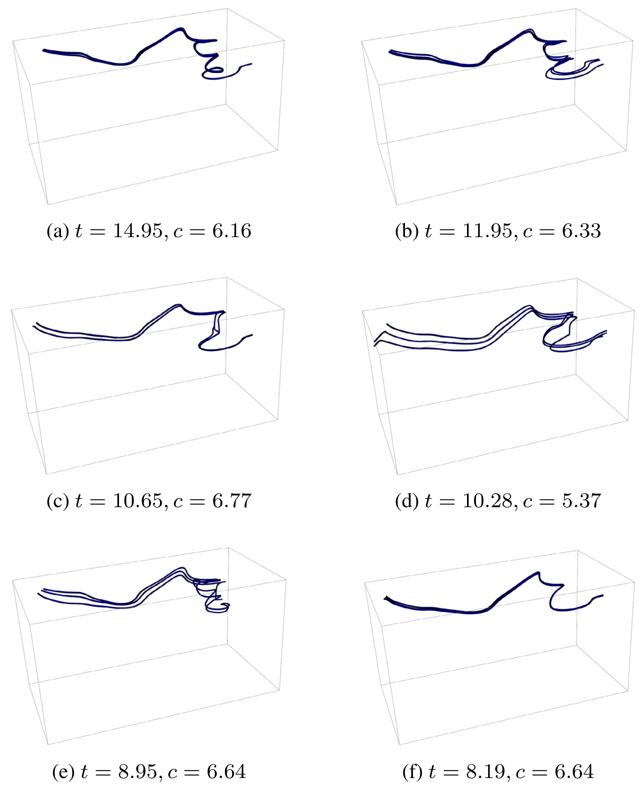

(a) $t = 14.95, c = 6.16$ (b) $t = 11.95, c = 6.33$ (c) $t = 10.65, c = 6.77$ (d) $t = 10.28, c = 5.37$ (e) $t = 8.95, c = 6.64$ (f) $t = 8.19, c = 6.64$

FIGURE 9. Examples of the second k-means clustering using the shape properties. Streamlines with the similar shape in the cluster 0 of Fig. 8 are drawn. *t* denotes the mean twistedness while *c* indicates the mean coverage of the entire streamlines in the cluster.

VI. RESULTS

We evaluated the proposed deep regression model and the streamline visualization method using the list of data sets shown in Table 1. Four flow data sets are used for the experiments. These data sets are, from top to bottom, referred to here as half cylinder [54], solar plume [55], Tangaroa [54], [56], and 3D double gyre [57], [58]. The 3D double gyre data set is synthetically created, but the others are generated by CFD simulations. Every data set is defined on a Cartesian grid and is time-dependent with different time steps.

Due to the GPU memory constraints, all data sets are resized or cropped to an identical resolution - 128 × 64 × 64 (Table 1). The length of data should be 2^n because there are a couple of pooling operations in the proposed network. Half cylinder and Tangaroa were cropped while the others were resized. In the cropping cases, the most interesting regions in the entire flow field were extracted not to loose the most interesting flow features as shown in Fig. 10. Because most

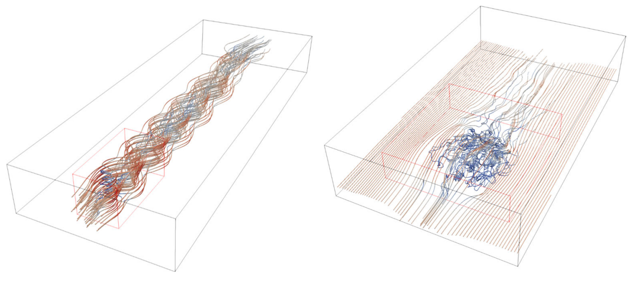

(a) Half Cylinder (b) Tangaroa

FIGURE 10. Visualization of the cropped regions and streamlines in the original data. A red box is the cropped region.

data sets have an asymmetric shape initially, with the axis along the dominant flow longer than the others, we decided to create a test set with an asymmetric shape. The normal distribution with a mean of 0.5, a standard deviation of 0.1 is employed to generate white noise for the LIC computation.

We adopted cross-validation for the experiments. Each data set consisting of multiple time steps is considered as a set of separate data blocks such that they were randomly split into the training and testing data at a ratio of 7:3. Then, the training data was further split into training and validation data at a ratio of 8:2. A ten-fold cross-validation scheme was employed using only the training data. Cross-validation is performed on each data set independently so as not to include too many samples of a single data set in the training set, which may lead to incorrect validation. Augmentation processes were applied only to the training and validation set.

A. DEEP REGRESSION MODEL

We implemented the proposed model using PyTorch 1.7 [59] with CUDA 10.1 [60]. The model is trained on a single computing node with four NVIDIA V100 GPUs with 32GB of memory on the Neuron supercomputer at the KISTI (Korea Institute of Science and Technology Information) supercomputing center whereas it is evaluated on a desktop PC with a single Intel Xeon Gold 6248 CPU with 376GB memory and a single NVIDIA Titan RTX GPU with 24GB memory.

1) MODEL COMPARISON

We evaluated our model relative to the original 3D U-net. In this experiment, the basic LIC strategy served as the input data format, and no augmentation methods were applied. Each learning model was trained using an identical data set, shown in Table 1, for this experiment. Table 2 depicts the elapsed time required to learn the models and predict the

TABLE 2. Comparison of the original 3D U-net and our model. The top two rows depict size of two networks in terms of the number of parameters and receptive field size. The middle four rows show the accuracy of the model for each test data set. Unseen data blocks are employed for this experiments. A basic LIC and no augmentation is applied. Metric is R^2. The last two rows present the speed of the models. Dilated 3D U-net outperforms original 3D U-net even though it runs faster than its competitor.

	Original 3D U-net	Dilated 3D U-net
Number of Parameters	51,779,142	8,417,094
Receptive Field Size	155	131
Half Cylinder	0.6946	0.7045
Solar Plume	0.6915	0.7045
Tangaroa	0.7571	0.7614
3D Double Gyre	0.9247	0.9309
Learning Time	674m 0.78s	556m 36.14s
Prediction Time	0.0847s	0.0699s

importance scores across every voxel in the test data. Both models were trained for 100 epochs. Furthermore, ten-fold cross-validation was applied in this case as well. The dilated 3D U-net outperformed the original 3D U-net across all training data, with the learning time and prediction time also shorter than its competitor despite using far fewer parameters. Both models predict the volume of importance score less than a second, which can take over 10∼60 minutes if naively calculating them using a single CPU. R^2 is employed for the metric to compare the accuracy of the models. The R^2 value is calculated as follows:

$$R^2 = 1 - \frac{\text{residual sum of squares}}{\text{total sum of squares}} = 1 - \frac{\sum_{i=0}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n}(y_i - \bar{y})^2} \tag{5}$$

where y_i is the i-th value of the variable to be predicted, $\hat{y}_i$ is a predicted value of y_i and $\bar{y}$ is a mean value of y_i.

2) INPUT DATA FORMAT

We assessed the proposed model using four different input data formats: velocity vector, basic LIC, enhanced LIC with one-pass enhancement, and the enhanced LIC with two-pass enhancement. The one-pass enhanced LIC performs the LIC integration and contrast enhancement only once, whereas the two-pass enhanced LIC performs them twice, as illustrated in Fig 2. Enhanced LICs outperformed others across all training data, while the basic LIC method surpassed the velocity vector as depicted in Fig. 11. The two-pass version of enhanced LIC showed better results than the one-pass enhanced LIC as well.

3) DATA AUGMENTATION

The proposed augmentation methods were evaluated on our network model using the enhanced LIC data. Fig. 12 shows the evaluation results comparing across different augmentation methods. Four different LIC volumes created by randomly generated white noise were used for LIC augmentation, while eight different transformations were employed for transformation augmentation. Table 3 depicts details of the list of augmentation operations. Combined augmentation was done with both methods simultaneously. All three methods surpassed the baseline on all test data sets. However, there were scant differences noted when comparing the three augmentation methods. LIC augmentation outperforms the transformation method on all test data sets except for the 3D double gyre data set. When comparing LIC and combined augmentation, LIC showed better results on the half cylinder and the Tangaroa data set, whereas the combined method was superior on the 3D double gyre and solar plume data sets. In the case of 3D double gyre, it shows higher accuracy than others. We believe it is because the data is synthetically created whereas the others are generated by the CFD solvers. The proposed model may fit the data easier than the rest. It yields very high accuracy even in the baseline, so that there is only little room for improvement in accuracy from augmentations.

TABLE 3. A list of operations applied on the transformation augmentation.

Type	Operation	# of augmentation
Flip	over X, Y, Z axis	3
Rotation	90°, 180°, 270° along X axis	3
Scaling	2×, 4×	2
Total		8

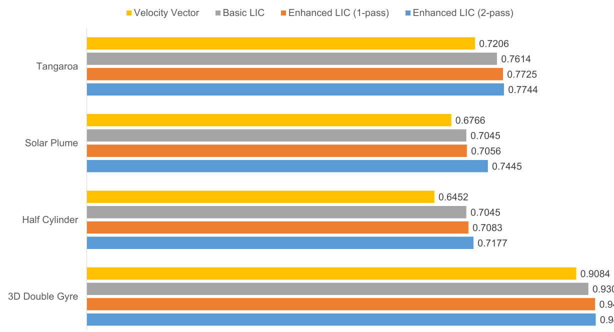


FIGURE 11. Evaluation results comparing across different input data formats. The metric is R^2. Higher is better. Enhanced LIC outperforms others.

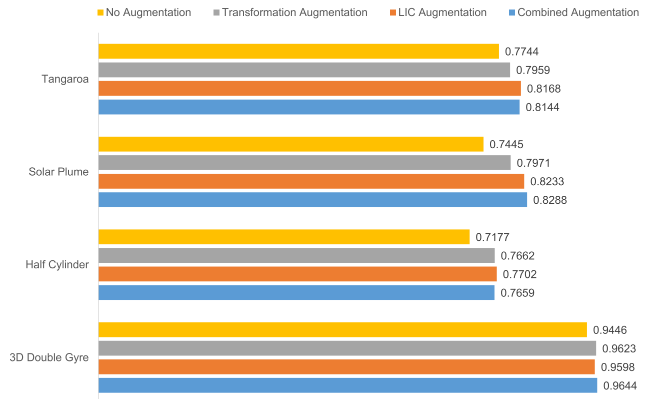


FIGURE 12. Evaluation results comparing across different augmentation methods. The metric is R^2. Higher is better.

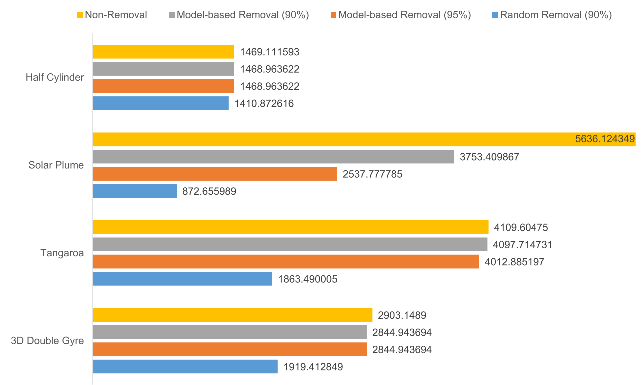


FIGURE 13. Comparison of the total importance score across different seed-removal strategies. Higher is better.

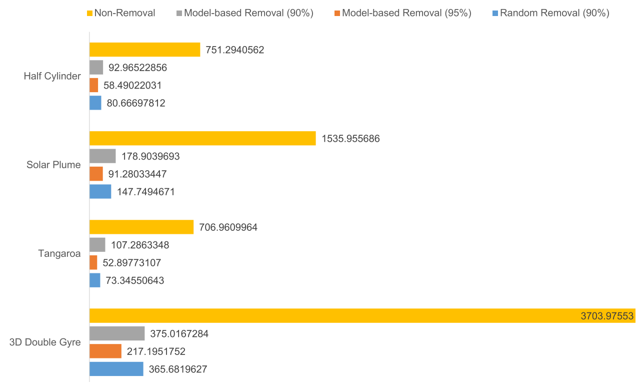
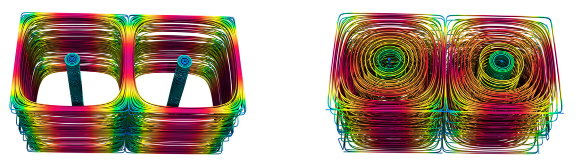


FIGURE 14. Comparison of the streamline creation time across different seed-removal strategies. The metric is seconds.

B. STREAMLINE SEEDING

We conducted experiments to evaluate the proposed model in terms of accuracy and speed. This experiment assessed the summation of the importance score of the 100 most essential streamlines across different seed-removal strategies which are non-removal (ground truth), the model-based seed-removal, and random seed-removal. The non-removal strategy generates all streamlines from all seed candidates without any seed truncation. The second strategy removes numerous seed points predicted as less important by the proposed model. We set the fraction of removal to 90% and 95% in this experiment. The last strategy removes seed candidates randomly. Only 90% removal is included for this strategy, which is the minimum removal fraction of the model-based experiments. We also measured the total process time required to generate streamlines to assess the reduction in the amount of time across the strategies, as mentioned earlier. Because the ground truth does not remove seeds, it records the exact maximum score.

We obtained a commensurable score in the experiments of the model-based seed-removal strategy compared to the ground truth across all test data sets as shown in Fig. 13. In the half cylinder data case, the model-based strategy using 90% and 95% removal levels showed scores nearly identical to that of the ground truth. The difference was less than 0.1%. It also scored around 98% of the ground truth on the half cylinder and Tangaroa data sets despite the fact that they utilized only 5∼10% of all seed points. In the solar plume case, we obtained less accurate results compared to the previous trials, i.e., 67% and 45% of the ground truth for the 90% seed-removal and 95% seed-removal strategies, respectively. However, these results are very competitive compared to the random removal strategy, which removes 90% of the seeds randomly. The proposed model-based strategy outperformed its competitors by 1.5 times and 2.2 times at 95% removals and 90% removal, respectively. In terms of speed, any of the seed-removal strategies surpassed the non-removal case as depicted in Fig. 14. It performed 12.8∼17.1 and 6.6∼9.9 times faster than the baseline with the 95% removal and 90% removal strategies, respectively.



(a) Original seed-removal strategy (b) Modified seed-removal strategy

FIGURE 15. Comparison of streamlines of 3D double gyre data using different seed-removal strategies. 100 streamlines are drawn.

Fig. 16 shows the visualization results of the streamlines generated in the experiments. The data blocks are chosen from among the unseen blocks. Streamlines in the 90% seed-removal and the 95% seed-removal cases, which are based on the proposed model, are more similar to the non-removal case than to the random removal case. However, many similar streamlines are shown to be overlapped in a small region when we draw the most important streamlines only. The streamline selection method based on curved-line clustering is helpful to overcome this problem. Fig. 17 shows the results. These images present more diverse flow features than the previous cases. Because the two model-based strategies produce nearly identical results, we omit one - 90% removals - which preserves more seeds. For the half cylinder and solar plume cases, the three strategies show similar results except that random removal generates more streamlines of less importance. For Tangaroa, the non-removal and random removal strategies reveal less essential flows, whereas the proposed method eliminates them. This occurs because the proposed model predicts less important seed points correctly and removes them from the seed candidates. However, this method is not applicable to all data sets. In the 3D double gyre case, our strategy's resulting images do not show streams between the core of the flow and the outer flows because it removes most of the less important streamline seeds such that no seeds remain in that area. This drawback can be resolved by slightly modifying the proposed seed-removal method. When removing seed points predicted as less important, 1∼2% of those are randomly chosen and kept as seed candidates. This slight change provides much richer streamline results, as shown in Fig. 15.

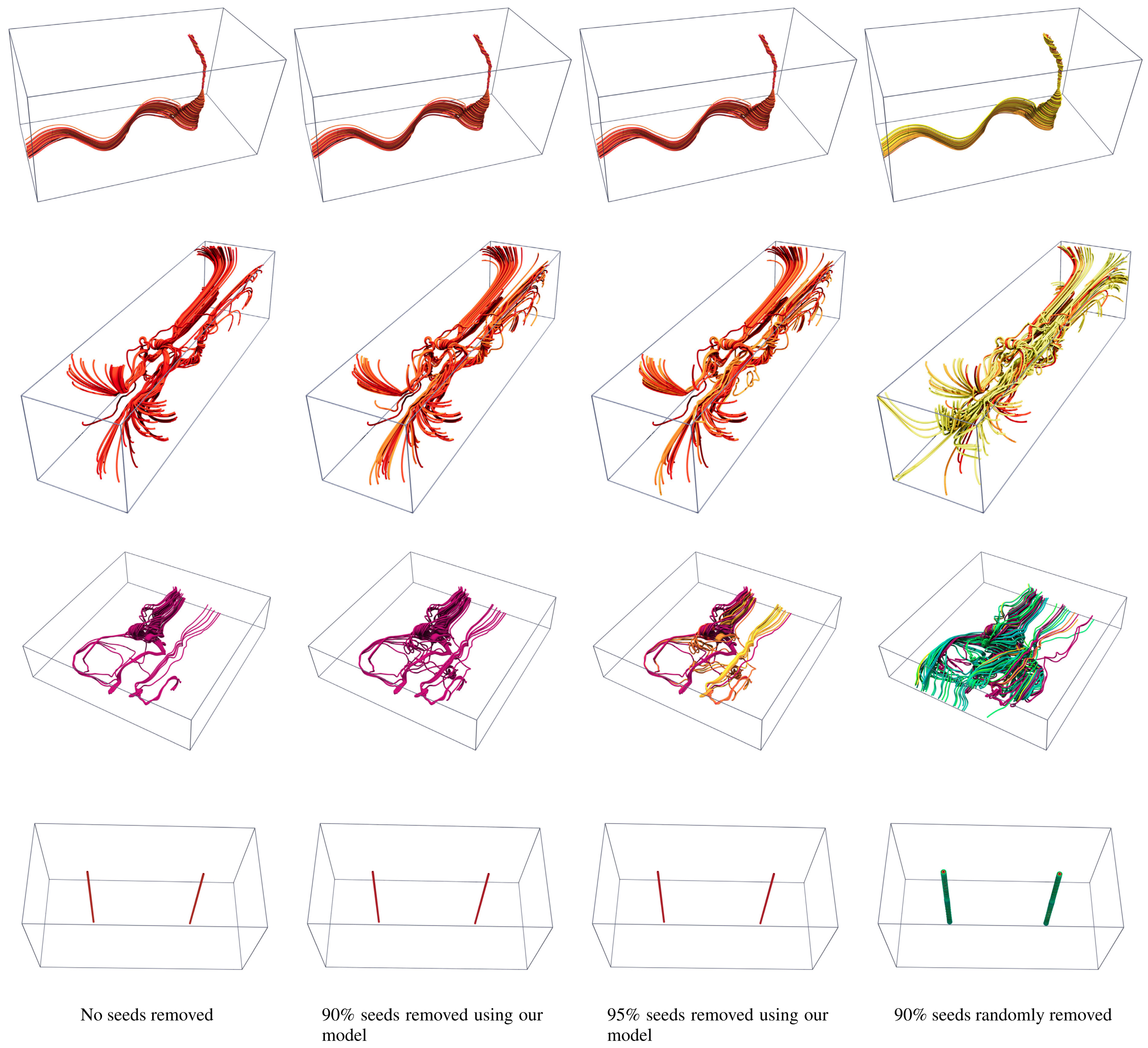

No seeds removed
90% seeds removed using our model
95% seeds removed using our model
90% seeds randomly removed

FIGURE 16. Comparison of visualization results showing the top 100 streamlines with the highest importance score across different seed-removal strategies. Color coded by the importance score. One data per row: 100th time step of the half cylinder, the fifth time step of the solar plume, 100th time step of the Tangaroa, and the 1st time step of the 3D double gyre, which are all unseen data sets.

VII. DISCUSSION

A. RECEPTIVE FIELD SIZE

In order to determine the importance score precisely, we calculate the required number of layers, ensuring that the perceptrons in the network can cover a large area of the flow field. If the receptive field is too small, the model can only utilize local information around the voxel and will thus never learn the global stream flow information. A receptive field that is too large may lead to lower accuracy due to the excessive influence of distant voxels, which may be unrelated. Additionally, it can yield a vast neural network that requires too many resources to process. However, there are no related studies of the optimal size of the receptive field for this network. In this paper, we heuristically determine that the receptive field of the network should enable the perceptron at the centroid of the volume to see the entire flow field.

Considering the findings of earlier studies [61], [62], the size r_0 of the receptive field of the neural network is expressed as:

$$r_0 = \sum_{l=1}^{L}\left((k_l - 1)\prod_{i=1}^{l-1} s_i\right) + 1 \tag{6}$$

where L is the number of layers, k_l is the kernel size of layer l, and s_i is the stride of layer i.

For the receptive field of the centroid voxel in the flow volume to cover the entire flow field, the size of the receptive field r_0 should exceed the maximum length of the flow volume.

For flow data of size W × H × D, it is necessary find L, which satisfies the following equation:

$$r_0 = \sum_{l=1}^{L}\left((k_l - 1)\prod_{i=1}^{l-1} s_i\right) + 1 \geq \max(W, H, D) \tag{7}$$

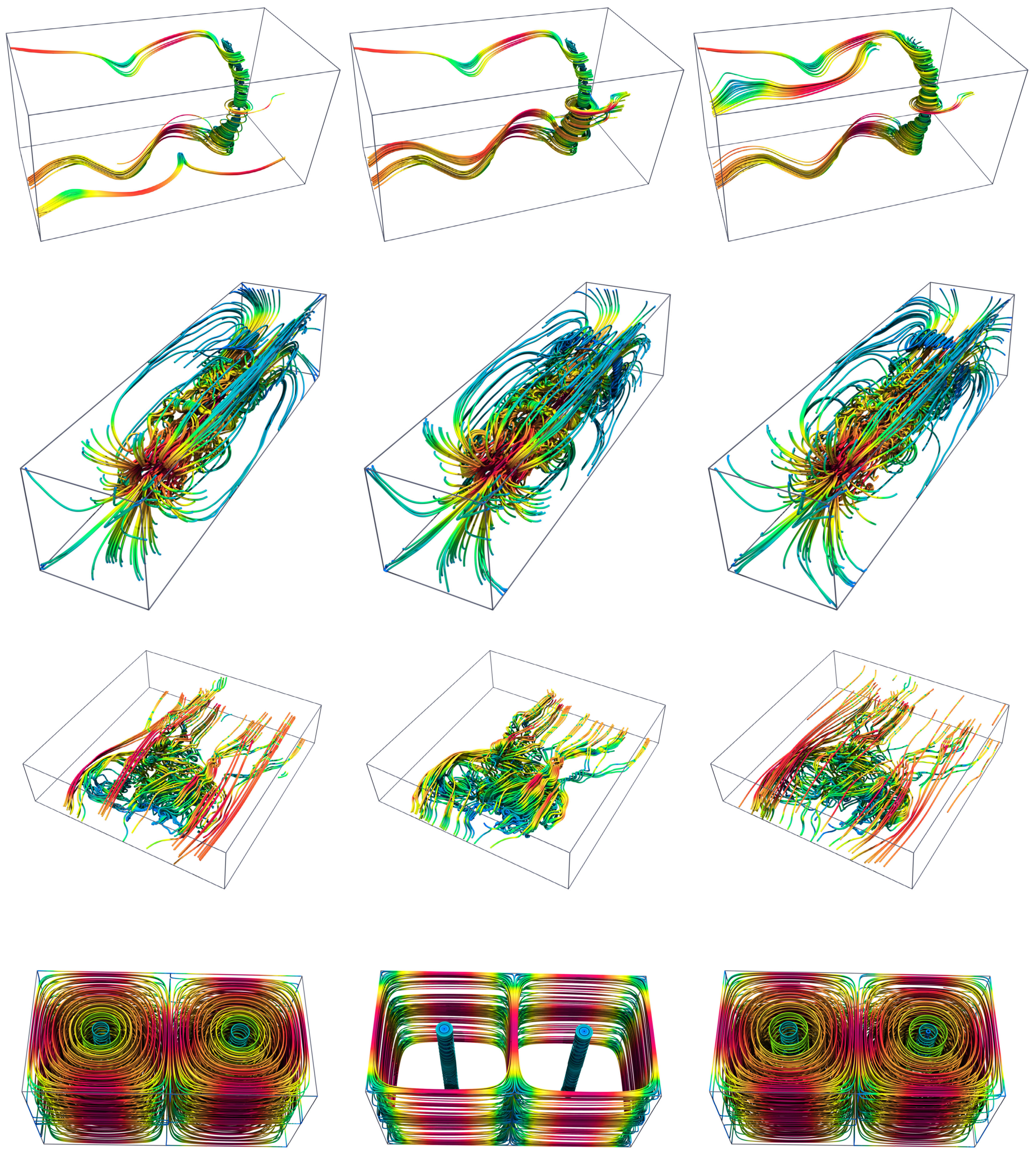


FIGURE 17. **Visualization results of the streamline selection method based on the curved-line clustering across different seed-removal strategies. The 3D double gyre case employed the modified seeds removal method while others adopted the original strategy. Color coded by the velocity magnitude. One data per row: 100th time step of the half cylinder, the fifth time step of the solar plume, 100th time step of the Tangaroa, and the 1st time step of the 3D double gyre.**

Assuming that the size of the volume data is $128 \times 64 \times 64$, the required number of layers in the dilated 3D U-net is 14. This leads to the network architecture shown in Fig. 6. Table 2 shows a comparison of the two networks in terms of total number of parameters and the receptive field size. We assume a resolution of the LIC volume of $128 \times 64 \times 64$.

B. LIMITATIONS

The proposed network has limitations, despite the fact that it helps to reduce the time required to generate streamlines revealing significant flow features. Because LIC performs voxel-wise flow integration, it could not accurately depict local flows compared to the standard streamline calculation that performs intra-voxel integration. It may be helpful to generate LIC volumes with much higher resolution by super-sampling the input velocity field to overcome this problem. However, this requires a more extensive deep regression network, which demands more memory. Although the magnitude of the velocity vector is one of the most important features when analyzing flow data, the proposed method does not exploit it because the LIC operation performs normalization on the velocity vectors naturally. We may need to extend our regression network to utilize multivariate volume data by adding the velocity magnitude to the input training volume.

The test data set in the experiments is a set of distinct time steps generated using a CFD solver identical to that of the training data set. We believe that it is possible to ensure high accuracy for a relatively small network because both training and test data are from the same modality. A much more extensive network would be needed to obtain such accuracy on unseen data generated using a completely different modality.

C. APPLICATIONS

The proposed network can be employed to visualize massive amounts of time-variant flow data efficiently, with calculations done by a remotely located high-performance computer. Such data types are too vast to visualize using a desktop PC. We can employ the proposed model to learn the flow trend of the data set using a temporally sampled data set on a remote server. The trained model can evaluate every time step in the original data and determine which time steps are worth downloading. Moreover, we can utilize the model to accelerate the drawing of streamlines on a desktop PC. Apart from streamline seeding, we believe that there are more applications for our network, such as vortex identification, flow data compression, or the upscaling of velocity vector fields.

VIII. CONCLUSION

In this paper, we proposed a deep regression model to learn flow trends in the velocity field to predict flow importance levels across complete flow fields. We presented an efficient visualization method for streamlines based on the proposed network as well. Curved-line clustering is employed for streamline selection to group similar streamlines. Moreover, the proposed methods were evaluated using a single synthetic data set and three different real-world CFD data sets. We obtained quality of visualization results comparable to that of the ground truth despite the fact that more than 90% of the seed candidates were truncated in 6.6∼17.1 less time compared to the proposed method's competitors.

We adopted the dilated 3D U-net strategy in this work, but other FCN-based networks should be applicable as well. We will soon apply another CNN-based deep regression network, such as pyramid dilated convolution or a residual network, to enhance the proposed model. We will improve LIC augmentation using more diverse LIC images generated by various random distributions with varied statistic values as well. Due to the memory constraint in this case, the input flow data were resampled to a smaller resolution. This can be resolved if a distributed deep learning technique is employed. We plan to expand our deep regression network using a distributed framework to overcome the size problem.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds., San Diego, CA, USA, May 2015, pp. 1–14.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[5] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham, Switzerland: Springer, 2015, pp. 234–241.

[7] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[8] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[9] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2016*, S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, Eds. Cham, Switzerland: Springer, 2016, pp. 424–432.

[10] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-Net: Fully convolutional neural networks for volumetric medical image segmentation," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 565–571.

[11] H. R. Roth, H. Oda, X. Zhou, N. Shimizu, Y. Yang, Y. Hayashi, M. Oda, M. Fujiwara, K. Misawa, and K. Mori, "An application of cascaded 3D fully convolutional networks for medical image segmentation," *Comput. Med. Imag. Graph.*, vol. 66, pp. 90–99, Jun. 2018.

[12] B. Cabral and L. C. Leedom, "Imaging vector fields using line integral convolution," in *Proc. 20th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*. New York, NY, USA: Association for Computing Machinery, 1993, pp. 263–270.

[13] D. Stalling and H.-C. Hege, "Fast and resolution independent line integral convolution," in *Proc. 22nd Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*. New York, NY, USA: Association for Computing Machinery, 1995, pp. 249–256.

[14] L. Li and H.-W. Shen, "Image-based streamline generation and rendering," *IEEE Trans. Vis. Comput. Graphics*, vol. 13, no. 3, pp. 630–640, May/Jun. 2007.

[15] K. Burger, P. Kondratieva, J. Krüger, and R. Westermann, "Importance-driven particle techniques for flow visualization," in *Proc. IEEE Pacific Vis. Symp. (PacificVis)*, Mar. 2008, pp. 71–78.

[16] B. Spencer, R. S. Laramee, G. Chen, and E. Zhang, "Evenly spaced streamlines for surfaces: An image-based approach," *Comput. Graph. Forum*, vol. 28, no. 6, pp. 1618–1631, Sep. 2009.

[17] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.

[18] S. Furuya and T. Itoh, "A streamline selection technique for integrated scalar and vector visualization," presented at the IEEE Vis. Poster Session, Oct. 2008.

[19] S. Marchesin, C.-K. Chen, C. Ho, and K.-L. Ma, "View-dependent streamlines for 3D vector fields," *IEEE Trans. Vis. Comput. Graphics*, vol. 16, no. 6, pp. 1578–1586, Nov. 2010.

[20] H. Yu, C. Wang, C.-K. Shene, and J. H. Chen, "Hierarchical streamline bundles," *IEEE Trans. Vis. Comput. Graphics*, vol. 18, no. 8, pp. 1353–1367, Aug. 2012.

[21] K. Lu, A. Chaudhuri, T.-Y. Lee, H.-W. Shen, and P. C. Wong, "Exploring vector fields with distribution-based streamline analysis," in *Proc. IEEE Pacific Vis. Symp. (PacificVis)*, Feb. 2013, pp. 257–264.

[22] T. McLoughlin, M. W. Jones, R. S. Laramee, R. Malki, I. Masters, and C. D. Hansen, "Similarity measures for enhancing interactive streamline seeding," *IEEE Trans. Vis. Comput. Graphics*, vol. 19, no. 8, pp. 1342–1353, Aug. 2013.

[23] A. Chaudhuri, T. Lee, H. Shen, and R. Wenger, "Exploring flow fields using space-filling analysis of streamlines," *IEEE Trans. Vis. Comput. Graphics*, vol. 20, no. 10, pp. 1392–1404, Oct. 2014.

[24] S. Sane, R. Bujack, C. Garth, and H. Childs, "A survey of seed placement and streamline selection techniques," *Comput. Graph. Forum*, vol. 39, no. 3, pp. 785–809, Jun. 2020.

[25] R. Lguensat, M. Sun, R. Fablet, P. Tandeo, E. Mason, and G. Chen, "EddyNet: A deep neural network for pixel-wise classification of oceanic eddies," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2018, pp. 1764–1767.

[26] A. Okubo, "Horizontal dispersion of floatable particles in the vicinity of velocity singularities such as convergences," *Deep Sea Res. Oceanogr. Abstr.*, vol. 17, no. 3, pp. 445–454, 1970.

[27] J. Weiss, "The dynamics of enstrophy transfer in two-dimensional hydrodynamics," *Phys. D, Nonlinear Phenomena*, vol. 48, nos. 2–3, pp. 273–294, 1991.

[28] X. Bai, C. Wang, and C. Li, "A streampath-based RCNN approach to ocean eddy detection," *IEEE Access*, vol. 7, pp. 106336–106345, 2019.

[29] Z. Duo, W. Wang, and H. Wang, "Oceanic mesoscale eddy detection method based on deep learning," *Remote Sens.*, vol. 11, no. 16, p. 1921, Aug. 2019.

[30] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 936–944.

[31] T. Bin and L. Yi, "CNN-based flow field feature visualization method," *Int. J. Perform. Eng.*, vol. 14, no. 3, p. 434, 2018.

[32] C. M. Ströfer, J.-L. Wu, H. Xiao, and E. Paterson, "Data-driven, physics-based feature extraction from fluid flow fields using convolutional neural networks," *Commun. Comput. Phys.*, vol. 25, no. 3, pp. 625–650, 2019.

[33] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[34] Y. Liu, Y. Lu, Y. Wang, D. Sun, L. Deng, F. Wang, and Y. Lei, "A CNN-based shock detection method in flow visualization," *Comput. Fluids*, vol. 184, pp. 1–9, Apr. 2019.

[35] L. Deng, Y. Wang, Y. Liu, F. Wang, S. Li, and J. Liu, "A CNN-based vortex identification method," *J. Vis.*, vol. 22, no. 1, pp. 65–78, 2019.

[36] Y. Wang, L. Deng, Z. Yang, D. Zhao, and F. Wang, "A rapid vortex identification method using fully convolutional segmentation network," *Vis. Comput.*, vol. 37, no. 2, pp. 261–273, Jan. 2020.

[37] B. Kim and T. Günther, "Robust reference frame extraction from unsteady 2D vector fields with convolutional neural networks," *Comput. Graph. Forum*, vol. 38, no. 3, pp. 285–295, 2019.

[38] Y. Li, C. Wang, and C.-K. Shene, "Extracting flow features via supervised streamline segmentation," *Comput. Graph.*, vol. 52, pp. 79–92, Nov. 2015.

[39] F. Hong, J. Zhang, and X. Yuan, "Access pattern learning with long short-term memory for parallel particle tracing," in *Proc. IEEE Pacific Vis. Symp. (PacificVis)*, Apr. 2018, pp. 76–85.

[40] J. Han, J. Tao, H. Zheng, H. Guo, D. Z. Chen, and C. Wang, "Flow field reduction via reconstructing vector data from 3-D streamlines using deep learning," *IEEE Comput. Graph. Appl.*, vol. 39, no. 4, pp. 54–67, Jul. 2019.

[41] J. Han, J. Tao, and C. Wang, "FlowNet: A deep learning framework for clustering and selection of streamlines and stream surfaces," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 4, pp. 1732–1744, Apr. 2020.

[42] J. Jakob, M. Gross, and T. Günther, "A fluid flow data set for machine learning and its application to neural flow map interpolation," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 1279–1289, Feb. 2021.

[43] M. Berenjkoub, G. Chen, and T. Gunther, "Vortex boundary identification using convolutional neural network," in *Proc. IEEE Vis. Conf. (VIS)*, Oct. 2020, pp. 261–265.

[44] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[45] Z. Liu and R. Moorhead, "Visualizing time-varying three-dimensional flow fields using accelerated UFLIC," in *Proc. 11th Int. Symp. Flow*, Jan. 2004, pp. 1–10.

[46] R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, and D. Weiskopf, "The state of the art in flow visualization: Dense and texture-based techniques," *Comput. Graph. Forum*, vol. 23, no. 2, pp. 203–221, 2004.

[47] A. Okada and D. L. Kao, "Enhanced line integral convolution with flow feature detection," *Proc. SPIE*, vol. 3017, pp. 206–217, Apr. 1997.

[48] B. Loring, H. Karimabadi, and V. Rortershteyn, "A screen space GPGPU surface LIC algorithm for distributed memory data parallel sort last rendering infrastructures," in *Numerical Modeling of Space Plasma Flows ASTRONUM-2014* (Astronomical Society of the Pacific Conference Series), vol. 498, N. V. Pogorelov, E. Audit, and G. P. Zank, Eds. San Francisco, CA, USA: Astronomical Society of the Pacific, Oct. 2015, p. 231.

[49] A. Odena, V. Dumoulin, and C. Olah. (2016). Deconvolution and checkerboard artifacts. Distill. [Online]. Available: https://distill.pub/2016/deconv-checkerboard and https://distill.pub/2016/deconv-checkerboard/

[50] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. ICLR*, 2016, pp. 1–13.

[51] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2881–2890.

[52] S. K. Devalla, P. K. Renukanand, B. K. Sreedhar, G. Subramanian, L. Zhang, S. Perera, J.-M. Mari, K. S. Chin, T. A. Tun, N. G. Strouthidis, T. Aung, A. H. Thiéry, and M. J. A. Girard, "DRUNET: A dilated-residual U-Net deep learning network to segment optic nerve head tissues in optical coherence tomography images," *Biomed. Opt. Exp.*, vol. 9, no. 7, pp. 3244–3265, Jul. 2018.

[53] C.-K. Chen, S. Yan, H. Yu, N. Max, and K.-L. Ma, "An illustrative visualization framework for 3D vector fields," *Comput. Graph. Forum*, vol. 30, no. 7, pp. 1941–1951, Sep. 2011.

[54] S. Popinet, "Free computational fluid dynamics," *ClusterWorld*, vol. 2, no. 6, pp. 1–7, 2004.

[55] J. Clyne, P. Mininni, A. Norton, and M. Rast, "Interactive desktop analysis of high resolution simulations: Application to turbulent plume dynamics and current sheet formation," *New J. Phys.*, vol. 9, no. 8, p. 301, Aug. 2007.

[56] S. Popinet, M. Smith, and C. Stevens, "Experimental and numerical study of the turbulence characteristics of airflow around a research vessel," *J. Atmos. Ocean. Technol.*, vol. 21, no. 10, pp. 1575–1589, Oct. 2004.

[57] S. C. Shadden, F. Lekien, and J. E. Marsden, "Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows," *Phys. D, Nonlinear Phenomena*, vol. 212, nos. 3–4, pp. 271–304, Dec. 2005.

[58] T. Wilde, C. Rössl, and H. Theisel, "Recirculation surfaces for flow visualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 1, pp. 946–955, Jan. 2019.

[59] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, vol. 32. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035.

[60] NIVIDA. (2021). *CUDA Toolkit*. [Online]. Available: https://developer.nvidia.com/cuda-toolkit and https://developer.nvidia.com/cuda-toolkit

[61] H. Le and A. Borji, "What are the receptive, effective receptive, and projective fields of neurons in convolutional neural networks?" 2018, *arXiv:1705.07049*. [Online]. Available: https://arxiv.org/abs/1705.07049

[62] A. Araujo, W. Norris, and J. Sim. (2019). Computing receptive fields of convolutional neural networks. Distill. [Online]. Available: https://distill.pub/2019/computing-receptive-fields