Computers and Fluids 214 (2021) 104770

Contents lists available at ScienceDirect

Computers and Fluids

journal homepage: www.elsevier.com/locate/compfluid

# Multiresolution classification of turbulence features in image data through machine learning

Jesus Pulido[a,b,*], Ricardo Dutra da Silva[c], Daniel Livescu[b], Bernd Hamann[a]

[a] Department of Computer Science, University of California, Davis, CA 95616, USA
[b] Los Alamos National Laboratory, Los Alamos, NM 87544, USA
[c] Department of Informatics, Federal University of Technology, Curtibia, PR 80230-901, Brazil

ARTICLE INFO

*Article history:*
Received 23 November 2019
Revised 4 September 2020
Accepted 11 October 2020
Available online 14 October 2020

*Keywords:*
Turbulence
Vortex detection
Image processing
Machine learning

ABSTRACT

During large-scale simulations, intermediate data products such as image databases have become popular due to their low relative storage cost and fast in-situ analysis. Serving as a form of data reduction, these image databases have become more acceptable to perform data analysis on. We present an image-space detection and classification system for extracting vortices at multiple scales through wavelet-based filtering. A custom image-space descriptor is used to encode a large variety of vortex-types and a machine learning system is trained for fast classification of vortex regions. By combining a radial-based histogram descriptor, a bag of visual words feature descriptor, and a support vector machine, our results show that we are able to detect and classify vortex features at various sizes at multiple scales. Once trained, our framework enables the fast extraction of vortices on new, unknown image datasets for flow analysis.



## 1. Introduction

Today's supercomputers allow scientists to simulate turbulent phenomena using extremely high-resolution grids, producing massive data sets that make it possible to gain new insights into complex turbulent behavior at multiple scales. Unfortunately, there exists a large disparity between compute flops (CPU) and I/O capabilities. This gap has made it unfeasible to save the massive amounts of data generated onto non-volatile space in a reasonable amount of time. These limitations make feature extraction and analysis difficult when performed after a simulation, especially when temporal resolution is low relative to the simulation. It is common to only produce simulation-state outputs at regular intervals of a simulation which incur a huge cost. These state files are often the only data points used for data analysis and feature extraction. These outputs can be hundreds of gigabytes large, requiring large amounts of resources to process them after the simulation has completed. Recent advancements in analysis and visualization techniques have introduced cross-domain in-situ methods that run alongside large simulations, producing reduced-scale intermediate data products (images) that are more reasonable to manage for feature detection and extraction.

In the classical 3D turbulence picture, unsteady vortices of many sizes appear and interact with each other, generating smaller and smaller vortices, up to the scale where viscous effects dominate. The quantitative description of this process has been one of the central objectives of turbulence research, with many questions still open, for example, related to anomalous scaling exponents and intermittency. In order to assist with the understanding of vorticity dynamics, traditionally, vortex extraction in turbulent flows has been a well-studied subject.

Some of the earliest definitions of vortices were given by Jeong et al. [1], where interactions between coherent structures play a role in the development of vortex dynamics. Subsequently, newer topology methods were used to define vortex behavior by Post et al. [2], primarily categorizing feature extraction methods into four types: direct flow, texture-based flow, geometric flow, and feature-based flow visualization.

There have been methods developed and used to detect and extract vortex structures in 2D flow. The Okubo-Weiss method [3,4] has been successfully used on native 2D hydrodynamics and magnetohydrodynamics. The Q-criterion has been used in 3D flows but poses issues when used on 2D datasets. Vortices that are not aligned to a camera's plane-of-view cannot be visually characterized as a vortex. While it is ideal to run many of these detection methods on the original, raw dataset, sizes of output data oftentimes require significant resources that match those where the simulation was executed. More complex methods become prohibitive due to their overhead requirements, and without a pri-

* Corresponding author at: Los Alamos National Laboratory, Los Alamos, NM 87544, USA.
*E-mail addresses:* jpulido@lanl.gov (J. Pulido), rdutra@dainf.ct.utfpr.edu.br (R.D. da Silva), livescu@lanl.gov (D. Livescu), hamann@cs.ucdavis.edu (B. Hamann).

https://doi.org/10.1016/j.compfluid.2020.104770
0045-7930/

ori knowledge of feature locations may need to be processed at a global scale, i.e. computing derivative and tensor products from stored velocity fields. Additionally, our integration with the Cinema framework produces many 2D image-sets as light-weight data products rather than 3D data. For these reasons, we have opted to use the Okubo-Weiss method for 2D training and validation of our classification system.

Aside from the connection with turbulence in general, vortex classification might be particularly useful for understanding the large roll-ups and structure of the flow generated by the Rayleigh-Taylor, Richtmyer-Meshkov, and Kelvin-Helmholtz instabilities. Recently, Zhou has provided a comprehensive review of flow, turbulence, and mixing induced by these instabilities [5–7]. The method proposed here might help extend some of the analysis approaches surveyed in the review. In addition, experimental data results are often only available as 2-D images. In applications such as ICF, where the Rayleigh-Taylor/Richtmyer-Meshkov instabilities are important [5–7], vortex classification may offer a novel tool for flow analysis.

One of the most popular intermediate data products produced today are Cinema databases [8]. Cinema databases most commonly store image-space representations of the data at many different camera perspectives, allowing for instantaneous access to many views of a specific dataset while a simulation is running for post-analysis. For exploratory research, it is sometimes difficult to know prior to the simulation what specific features to look for especially when running many variations of parameters. Furthermore, for massive datasets it is difficult and expensive to save many full-size datasets at fine temporal resolutions. The Cinema framework allows for fine temporal, image outputs of large-scale simulations at a significantly reduced cost. The advent of well-defined image-space features makes Cinema an ideal choice but detection methods must be developed. Recent work by Banesh et al. [9] has been able to successfully use an edge-based contour eddy tracking method on temporal, 2D image data of turbulent ocean currents. Additional tracking and evaluation of these results were done by Gospodnetic et al. [10].

Vortices are typically described by their mathematical behavior, but also contain well defined visual footprints making them an ideal feature to detect and extract in image-space environments. Performing meaningful large-scale extraction of visual features in diverse fields has traditionally been tackled by the use of machine learning algorithms. Without the need of discrete definitions of features, machine learning algorithms take multiple sets of inputs with weighted descriptors to then separate into several types, or classifications. Work in turbulence area is not substantial, but there are several sample works related to turbulence-like datasets.

Zhang et al. [11] used several methods to boost detection results in machine learning for vortex detection. By using local vortex detection algorithms, termed as weak classifiers, and an expert-in-the-loop approach for labeling results, they show to have reduce misclassification rates compared to component classifiers. Similar to this approach, our framework has the capability to enable an expert-in-the-loop approach for labeling but focus on the use of the Okubo-Weiss classifier for this work. Kwon et al. [12] used a learning algorithm to evaluate and select the best image-space representations for large scientific data and was able to show significant improvements compared to manual selections. For the extraction and classification of vortex features in image space, the use of machine learning could significantly improve the efficiency of the extraction. For example, features that would be unclassified by traditional methods could be learned and correctly identified using machine learning techniques.

In this paper, we develop a classification system that can automatically identify, describe, and extract features primary to turbulence datasets in image-space. The focus is on the extraction of vortices, due to their importance for turbulence research. Vortices are also some of the most visually distinct features available. Once trained, our classification system strictly works on image datasets without the use of the original data scalar components. Our contributions are the following:

- an image-space descriptor that operates in linear space for the detection and extraction of vortex-like features;
- a complete training and classification system that enables the low-cost evaluation of image-space datasets.

Data used in this paper is from the public Johns Hopkins Turbulence Database [13]. While the JHTDB hosts many turbulence datasets, the one used corresponds to a Direct Numerical Simulation (DNS) of homogeneous buoyancy-driven turbulence (HBDT) [14] on a $1024^3$ periodic grid. This simulation solves the incompressible Navier-Stokes equations for two miscible fluids with different densities, in a triply periodic domain [15–18]. Both fluids are initialized as random blobs using a characteristic size of about 1/5 of the domain, consistent with the homogeneity assumption. Starting at rest, constant gravitational acceleration causes the fluids to move in opposite directions due to differential buoyancy forces. As turbulence fluctuations are generated and the turbulent kinetic energy increases, features of interest begin to emerge. Nevertheless, stirring by turbulence increases the rate of molecular mixing. After some time, molecular mixing becomes large enough that the buoyancy forces are overcome by dissipation and turbulence starts to decay. Due to the assistance of the buoyancy forces, the turbulence decay is different than classical decay. Visually, one interesting phenomenon is the classification of number and size of vortices that form along a multi-fluid boundary.

The remainder of the paper is organized as follows. Section 2 briefly describes concepts related to the proposed method, which is presented in Section 3. Experimental results are described and discussed in Section 4. Section 5 presents the conclusions and directions for future exploration of the topic.

## 2. Background

In this section, we present some concepts related to the proposed method.

### 2.1. Okubo-Weiss criterion

The Okubo-Weiss criterion for identifying vortexes is $W < 0$, where $W$ is defined as:

$$W = (s_n)^2 + (s_s)^2 - \omega^2 \tag{1}$$

$$s_n = \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}, s_s = \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \tag{2}$$

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}. \tag{3}$$

Here, $s_n$ is the normal component of strain, $s_s$ the shear component of strain, and $\omega$ is the vorticity.

### 2.2. Radial gradient transform

The Radial Gradient Transform (RGT) is used in Takacs et al. [19] and Luo et al. [20] to improve rotation invariance of descriptors. Let $p$ be a point and $g$ a gradient in a given feature image with center $c$. The RGT uses two orthogonal basis vectors to provide a local reference for the gradient $g$. The basis vectors are given by

$$r = \frac{p - c}{|p - c|} \tag{4}$$

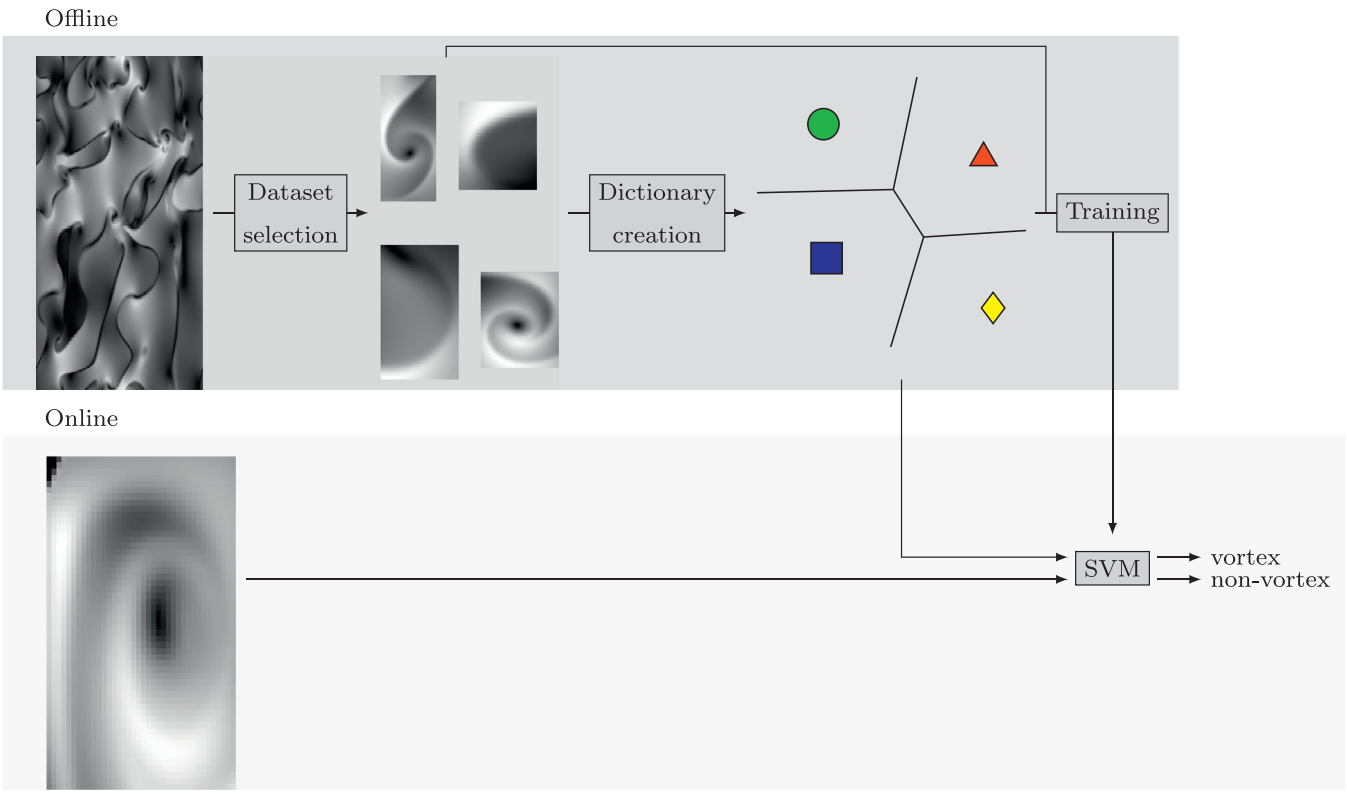



**Fig. 1.** The feature analysis pipeline is composed of an offline training step (dark gray box) and an online classification step (light gray box). The training comprises the creation of a dataset of features that are used to obtain a dictionary of words for describing the features and creating an SVM classifier. The SVM is then used to positively or negatively classify any feature as a vortex.

and

$$t = R_{\pi/2} r \tag{5}$$

such that $R_{\pi/2}$ is the rotation matrix for an angle of 90 degrees. Then, the radial gradient $g'$ is computed as

$$g' = \begin{bmatrix} r^T \\ t^T \end{bmatrix} g. \tag{6}$$

### 2.3. Bag of visual words

A Bag of Visual Words (BOVW) is constructed by using vector quantization of descriptors extracted from image patches. It is used for image classification [21–24]. A BOVW method defines image features as words (or visual words) and counts the occurrences of each feature in an image. A BOVW descriptor of an image is thus a histogram.

The visual words in a histogram form the dictionary (or codebook) of the BOVW model. The visual words must be chosen so that they relate to features that are common among images to make it possible to classify an image/object. The occurrences of the words in one image should make it possible to distinguish that image from another. One expects the distribution of the words to differ significantly. To compute a dictionary one usually performs feature detection and description steps for a set of images, resulting in a set of feature descriptors. These are used to produce a smaller set of descriptors representative for groups of descriptors, those becoming the words. Clustering can be used to compute visual words which, for instance, can be defined as centroids of clusters via a K-means algorithm.

The BOVW descriptor of an image can be computed as the histogram, capturing the frequencies of each word in the image. Given an image, in order to compute the histogram one can perform the feature detection and description steps; subsequently, for each feature descriptor obtained, one accumulates the bin related to the closest word by an appropriate distance metric.

### 2.4. Support vector machines

A support vector machine (SVM) computes a hyperplane that best separates a set of n-dimensional feature descriptors, using labels of two classes [25,26]. The hyperplane is a decision boundary: a new sample can be classified as belonging to one of the two classes according to the side of the hyperplane it is on.

Among all possible hyperplanes, an SVM computes the one that maximizes the margin to the samples in the classes, i.e., it maximizes the distance to the nearest samples in each class, the support vectors. Often a data set is not linearly separable in d-dimensional space. For this reason, an SVM maps $d$-dimensional features to a higher-dimensional space where it is more likely to have a separation hyperplane. This map is performed by a kernel function, e.g., a linear, polynomial or radial basis function. For details concerning the optimization process, we refer to [27–29].

## 3. Method

Fig. 1 presents a brief summary of our pipeline. The offline stage comprises the selection of features, the creation of a dictionary to describe what is and what is not a vortex-like feature, and the final training of a classifier. The classifier is then used to predict the class of previously unseen features. Section 3.1 describes how image features were selected by means of the Okubo-Weiss measure in order to compose a dataset for training. Sections 3.2 and 3.3 discuss, respectively, the computation of measures to describe the features by means of Radial Histogram of Gradients and the derived Bag of Visual Words. The training


**Fig. 2.** Various representations of a feature. From left to right: Velocity magnitude, Okubo Weiss, and thresholded Okubo Weiss. The left image is used by our descriptor, and the center and right images are used to decide an initial classification for training.

dataset is then used to compute a classifier that assigns a given input feature as a positive or a negative vortex detection, as described in Section 3.4.

### 3.1. Okubo-Weiss feature selection

One of the most critical components in our pipeline is to select image-space features that will be relevant to domain-specific applications. In this work, we've focused on the extraction of vortex-like features by using a 2-D representation of them. Through the use of the Okubo-Weiss criterion, regions in velocity magnitude image space are extracted and used to train our system.

Most image-space representations used to evaluate scientific data typically employ a rainbow, diverging, or converging colormap. These may introduce biases that may obscure features of interest or introduce artifacts that are not related to known physical features [30]. These chromatic colormaps inherently falsify features perceived by human vision to aid in differentiation. Machine learning algorithms typically require a single-channel input, causing a conversion between a chromatic colormap to a grayscale image to pass-through these falsified features. To remove any feature biases from our training, we select a linear colormap commonly known as a grayscale colorset.

Fig. 2 shows various representations of a vortex-like feature under evaluation by our system for training. When Okubo-Weiss criterion is computed on an original dataset where the velocity fields are available, a negative threshold denotes a high likelihood of a vortex existing at that location. By using a fixed negative threshold, we are able to generate a mask along an entire dataset and use a connected components routine to extract neighboring pixels as full objects, similar to that explained in Fig. 6. An affine bounding-box region is created around each individual object and a velocity magnitude representation is stored for processing further down the pipeline. The extracted Okubo-Weiss objects are presented to the domain researcher for selection.

### 3.2. Radial histogram descriptor

The primary descriptor of a feature is based on capturing local variations of the gradients in a similar process to that of the Histogram of Oriented Gradients (HOG) descriptor [31]. However, HOG descriptors are very sensitive to the orientation of the input feature. Therefore we use an adapted gradient descriptor, as summarized in Fig. 3.

Initially, the gradients of the feature image are computed and the Radial Gradient Transform (RGT) is used to improve rotation invariance. After computing the RGT, the gradient image is resized to an image of 64 $\times$ 64 pixels. By resizing the gradient image, related points in the original image and in the resized image present the same gradient. The gradient field may be deformed if the feature image is resized and then the gradients are computed over

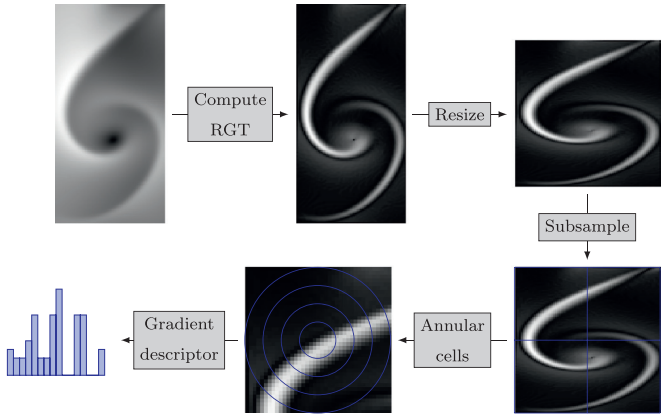



**Fig. 3.** Given a feature, gradient-based descriptors are extracted as words that afterwards will describe the feature as a bag of words. Initially, the RGT is used and the feature is resized to 64 $\times$ 64 pixels. A subsampling process is then performed to obtain windows with different sizes spread all over the feature. Each of such windows is split into annular cells in which gradient descriptors are computed to describe the window. The set of such descriptors for all the windows form the words of the feature.

the resized image. Resizing is performed so that further computation is independent of differences in the height and width, as well as in the resolution of images.

A dense set of windows is then uniformly sampled over the gradient image using vertical and horizontal shifts. The shifts can be smaller than the window size in order to produce overlaid windows. The set of windows is also computed using different sizes to produce a multiscale representation of the feature. In our experiments we used 8 different window sizes starting from 64 with a downsampling factor of 0.9, namely, the windows have sizes 64, 58, 52, 47, 42, 38, 34 and 31.

Each window is subdivided using annular cells, as in Fig. 3, and the histogram of radial gradients is then formed by accumulating the magnitude of the gradients within $k$ bins related to the discretized gradients in $k$ directions. The descriptor of a window is formed by concatenating the directional histograms for each annular cell. In our experiments we have subdivided the windows using two annular cells.

### 3.3. Bag of words descriptor

The final feature descriptor is a Bag of Visual Words (BOVW) [21] with a vocabulary created using the descriptors based on radial gradients. We initially compute the radial descriptors for vortex and non-vortex features, producing a set of points in the space of radial gradient descriptors. A dictionary of $n$ words is then computed using the $k$-means clustering, such that words are the centroids of the clusters. The process is depicted in Fig. 4.

The BOVW descriptor of a feature is a $n$-dimensional vector that counts the occurrence of each word in the feature (Fig. 5). Given a feature, the radial descriptors are computed to produce a set of points in the space of radial gradient descriptors. Each of the points is assigned to the closest word, incrementing the corresponding bin in the final descriptor.

### 3.4. Classification

We train a Support Vector Machine (SVM) [25] based on the BOVW features using a dataset comprised of images extracted from the Johns Hopkins Turbulence Database [13]. The set of images for training is obtained using the process defined in Section 3.1.

Once training is finished, new unknown test images are considered to classify features. Given an image patch input, the SVM pro-

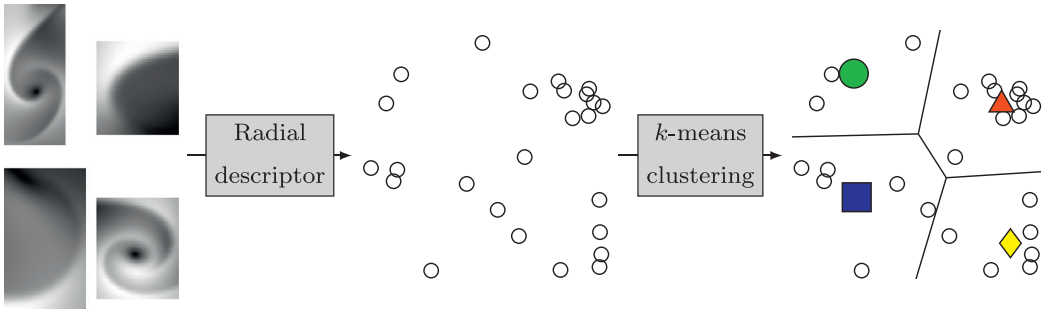



**Fig. 4.** Computation of the dictionary. The gradient descriptors for many features are computed and clustered using a k-means in high-dimensional space. As an example, the centroids of the k-means clustering (square, circle, triangle, diamond) are assigned as the dictionary which will be used for the bag of words.

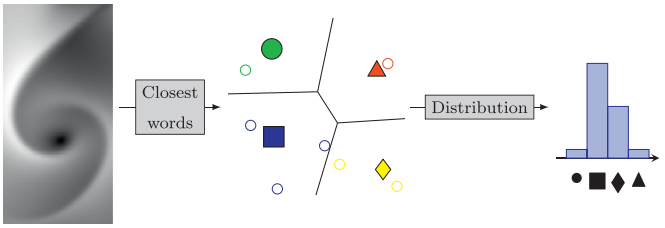



**Fig. 5.** Visual word binning. Given a set of gradient descriptors for a feature, each feature is assigned to the closest dictionary word to form a bag of words descriptor for the feature.

duces a positive or negative response that relates to the distance to the fitted hyperplane used to differentiate between various words. The kernel function used for the fitted hyperplane is not the default linear separable function, but rather a third-degree polynomial function, which allows non-linear classifications. The signed orthogonal distance from the oriented hyperplane is the output response value and is normalized between $-1$ and 1. These distances represent the separation between the visual words "vortex" and "non-vortex".

When investigating a new image, the brute-force, non-efficient method is to consider every single pixel of the new image at the dataset's native resolution. This becomes computationally expensive when considering potentially thousands of individual images, adding up to many billions of pixels. To significantly improve compute performance, first we must select regions from these new images that we want to consider. The selection is performed by a low-cost ellipse region extraction pipeline, as shown in Fig. 6. The extracted region is related to the bounding rectangle that includes the ellipse. It is worth to note that fitting a squared region to an elongated feature could include data from extraneous dynamics unrelated to the vortex. Therefore, the bounding rectangle is better to avoid such interference.

To extract the ellipse regions, we first compute the gradients of an image in the $x$ and $y$ directions. An efficient way of computing this at multiple resolutions is through the use of discrete wavelet transforms (DWT) and using their horizontal and vertical detail coefficients. The Haar DWT high-pass filter that we used approximates a differential for obtaining the gradients. By using coefficients at different scales of the wavelet decomposition it is possible to control the identification from finer to coarser features.

From the gradients of the wavelet transform in a given scale, we perform an edge detection [32]. The curvature at each point $p$ of the edges is then computed as $\kappa(p) = 1/R(a)$, such that $R(a)$ is the radius of the osculating circle at $p$. We then threshold the edge points $p$ using this $\kappa(p)$ value to remove straight edges and maintain curved edges that may be formed by vortices.

Contours given by the connected components of the resulting edges are then extracted and used to fit ellipses [33] which represent regions where vortexes may potentially exist. Once these ellipses are computed, we extract an affine bounding-box region around the ellipse which is considered for classification.

A further filtering step may be applied to avoid extremely elongated ellipses which still may occur due to noisy curvature values. The thresholding of ellipses is computed using the eccentricity (ratio of the minor and major axis lengths). The smaller the ratio the more elongated is the ellipse. Due to the wavelet structure used to generate the gradients in the previous step, contours can be generated at multiple resolutions using wavelet's native hierarchy, allowing the extraction to be performed at multiple scales.

The relatively low computation cost of these routines reduces the computational cost further rather than attempting to perform a classification query for the entire resolution of the image, at every pixel.

## 4. Experiments and results

In the following we present experiments and results achieved by our method. The experiments were performed using turbulence data from the Johns Hopkins Turbulence Database [13], which consists of the velocity components at 1015 time instances in a volume of $1024 \times 1024 \times 1024$ voxels. The time instances are separated by a constant time step of 0.04 and cover the initial state, growth, and long time decay of turbulence. For training, we used 2D slices from 11 time instances (one slice per time instance), starting from time $t = 6$ and up to time $t = 12$, spanning frames numbers 150 to 300. Since the peak of turbulence kinetic energy occurs at $t \approx 11.4$, the training set covers the strong growth regime, as well as the beginning of turbulence decay. The 2D slices contain the direction of gravity and a homogeneous (horizontal) direction, and correspond to a 1/4 of the domain depth in the third (horizontal) direction.

Magnitude images were computed from the velocity components and patches were manually selected to build a dataset for training and testing the SVM classifier. The SVM implementation of OpenCV [34,35] was used with a polynomial kernel of degree 3, gamma value of 1 and coefficient 0. The termination criterion was set to 100 iterations. The dataset contains 447 features: 229 of these features were labeled to the vortex class and 218 features were labeled to the non-vortex class. The vortex patches were chosen so that the vortex center is fully contained inside the patch and nearly located at the center of the patch. Patches of regions without a clear vortex behavior, or even patches depicting a subregion of a vortex such that its center was occluded, were considered as non-vortex regions. Fig. 7 shows some examples of patches in the dataset.

The radial histogram (Section 3.2) was computed using two annular blocks and nine directions. The multiscale representation used 8 different window sizes starting from 64 with a downsampling factor of 0.9, namely, the windows have sizes of 64, 58, 52, 47, 42, 38, 34 and 31. The shift step used for the multiscale repre-

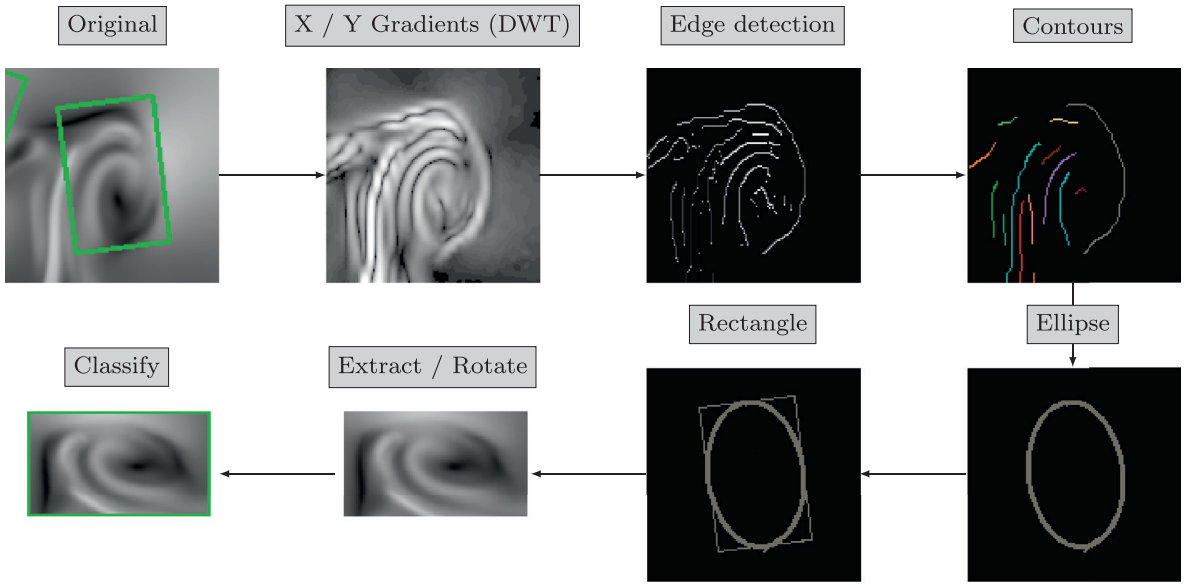
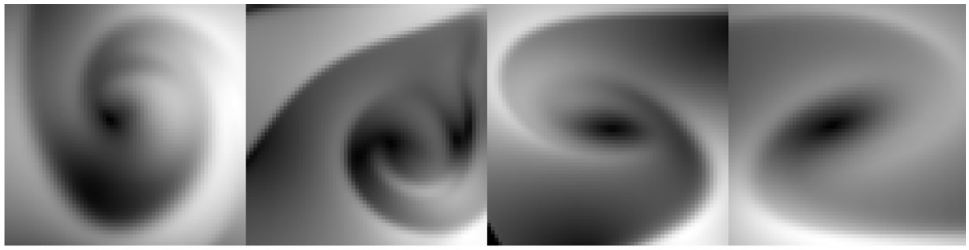
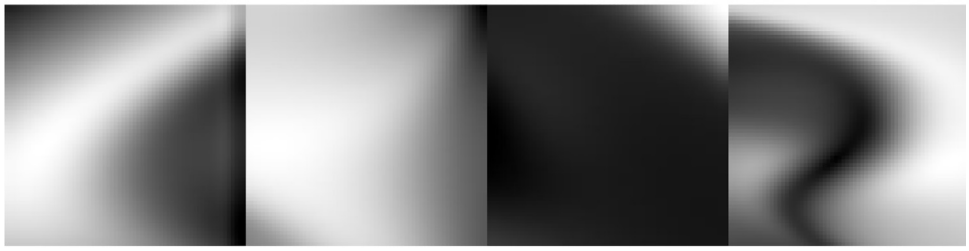



**Fig. 6.** To greatly reduce the problem-set of analyzing new images, low-cost methods are used to extract regions of potential feature activity. A region is extracted by computing an *X* and *Y* axis discrete wavelet transform (DWT) at multiple scales. For each scale, edge detection is applied then used to extract contours. Ellipses are fitted along sets of connected components then a rectangle is extracted. The rectangle region is then ran through our classifier and generates a response.



(a) vortex



(b) non-vortex

**Fig. 7.** Examples of patches in the dataset. The first row contains examples of the class vortex and the second row contains examples of the class non-vortex. These visually distinct features are necessary to enable the creation of a large range of words during training.

sentations was of 8 pixels and the dictionary was computed using 128 words (Section 3.3).

Dictionary creation, training and classification steps were performed on a system with an Intel Core i7 6700k running at 4.00 GHz and conducted as a single-threaded operation on a CPU implementation. To measure pre-processing performance, when ingesting 1833 images, dictionary creation took approximately 104 seconds and training 64 seconds per iteration. This dictionary creation and training step results in rates of 17.6 patches-per-second for the dictionary and 28.6 patches-per-second for training. When conducting classification of a new dataset, a patch size of about $100 \times 100$ pixels took about 0.05 seconds using an already trained system.

The classification experiments are reported using precision (*P*), recall (*R*) and F-measure (*F*), defined respectively by Eqs. 7–9. The measures are defined according to the number of true positives *TP* (a vortex feature that was classified as a vortex), false positives *FP* (a non-vortex feature classified as a vortex), true negative *TN* (a non-vortex feature classified as a non-vortex) and false negatives *FN* (a vortex feature classified as a non-vortex).

$$\mathrm{P} = \frac{TP}{TP + FP}, \tag{7}$$

$$\mathrm{R} = \frac{TP}{TP + FN}, \tag{8}$$

$$\mathrm{F} = 2\frac{\mathrm{P} \cdot \mathrm{R}}{\mathrm{P} + \mathrm{R}} \tag{9}$$

In order to evaluate the proposed method, the dataset of 447 images was partitioned into disjoint training and testing subsets

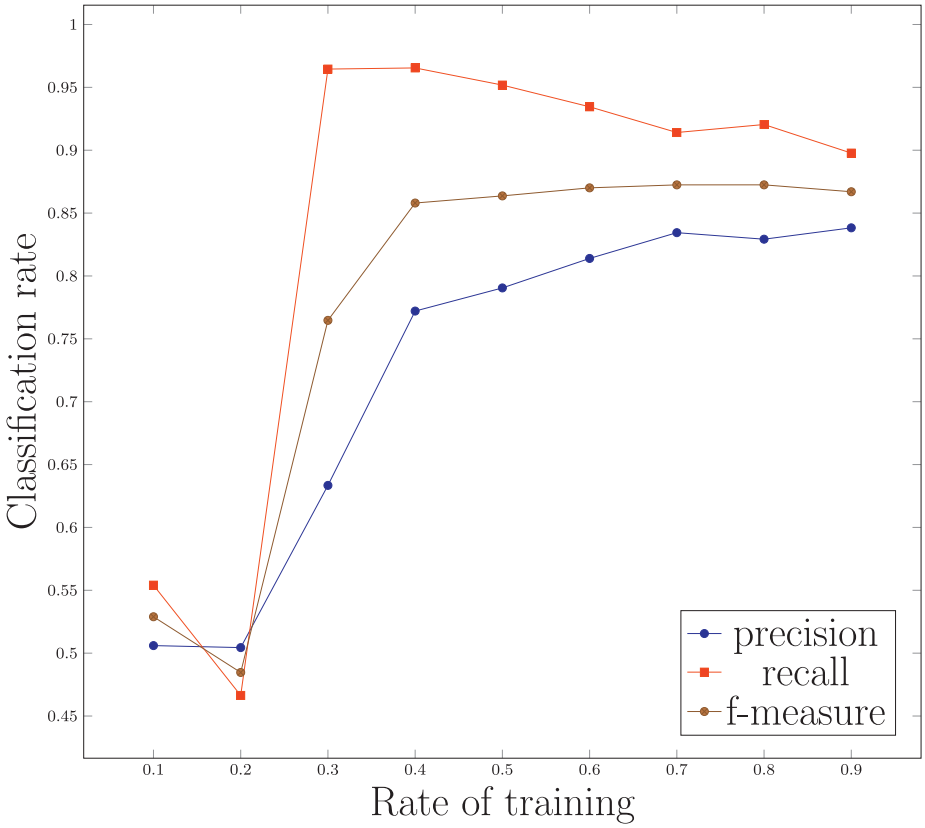



**Fig. 8.** Classification results for different sizes of the training sets. Precision, Recall and F-measure are computed as the training rate changes. The balance between false positives and false negatives was achieved when 70% of the dataset was used for training.

**Table 1**
Real versus predicted classification results obtained for the best and the average classifiers. These results were obtained with a training rate of 0.7 and 100 runs of random disjoint train and test partitions.

| | | Best | | Average | |
|---|---|---|---|---|---|
| | | Predicted | | Predicted | |
| | | Positive | Negative | Positive | Negative |
| Real | Positive | 65.00 | 0.00 | 62.18 | 2.82 |
| | Negative | 11.00 | 54.00 | 17.38 | 47.62 |

that were randomly chosen. The graph in Fig. 8 summarizes the average results for each measure given different sizes for the training set. Training and testing steps were performed 100 times for each partition and the average value for each evaluating measure was computed. The training set size varies from rates of 0.1 to 0.9, that is, from 10% to 90% of the 447 images. The subset of testing images is formed by 10% of the images (rate of 0.1).

It can be noticed from precision and recall curves that the classifier performs poorly between rates 0.1 and 0.2. In fact, we noticed the classifiers obtained at such low rates would classify all test samples either as a vortex or a non-vortex. From the precision curve, it can be noticed the number of false positives decreases as the rate of training images approaches 0.7 and then it becomes stable. The number of false negatives can be analyzed from the recall curve, which slightly decays as the precision improves. The f-measure averages both curves and from this measure it is possible to note that the classifier becomes stable at a rate of 0.7.

The confusion matrices for the best classifier and for the average classifier are summarized in Table 1. These results were obtained with a training rate of 0.7 and 100 runs of random disjoint train and test partitions. The rows correspond to the real classes and the columns to the predicted classes. The best classifier achieved a precision of 0.85 and a recall of 1.0, since not a single vortex sample of the testing subset was misclassified. The f-measure was of 0.92. The average precision, recall and f-measure values are, respectively, 0.78, 0.96 and 0.86.

Figs. 9 –11 show a large spectrum of classified samples along with the confidence of the classification. The confidence of the classification is based on the distance of a sample to the classification boundary returned by SVM. The values were normalized using the maximum and minimum distances so that a value of 1.0 represents high confidence and a value of 0.0 represents low confidence. Fig. 9 depicts samples that were correctly classified as vortices. The scale under the samples shows a confidence distribution from higher confidence (dark blue) towards lower confidence samples (light blue). Fig. 10 shows examples of samples that were misclassified as vortices. The confidence values show that these samples are very close to the classification boundary. The problem may happen because there is not enough data to compute a discriminative descriptor. We have noticed that many misclassified samples are very low-resolution samples. That can be noticed from the pixelation effect in many images of Fig. 10. The first image to the left, for instance, is only 19 $\times$ 26 pixels.

Fig. 11 depicts samples that were correctly classified as non-vortices. The scale under the samples shows a confidence distribution from lower confidence (light red) towards higher confidence samples (dark red).

In many observational or experimental studies, full data information is not available. Taking vortex identification as a generic example, we show that our method can be useful for flow analysis in the absence of full data, provided that a complementary dataset can be used for training. Thus, through training and classification, the high-level identification of regions of interest is made possible on image datasets rather than the original data. In addition,

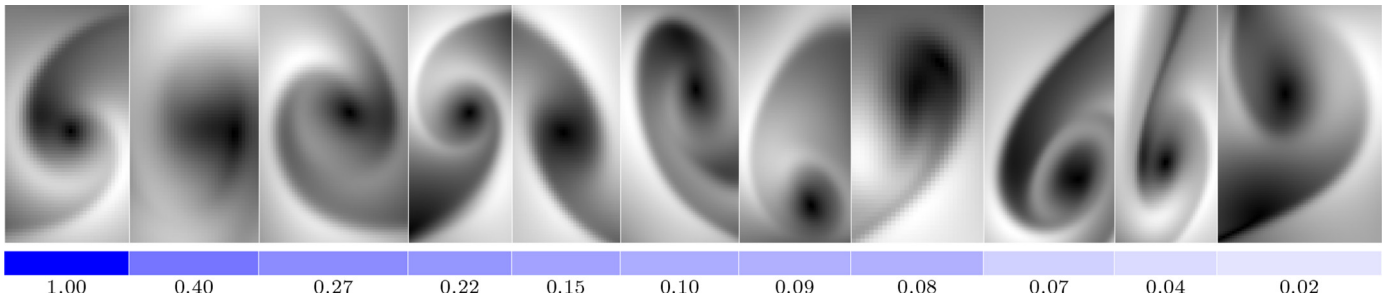



**Fig. 9.** Examples of vortices classified using the proposed method and the confidence obtained for each classification. Since the dataset was built with the center of the vortex usually at the center of the image, images with such an aspect are classified with higher confidence.

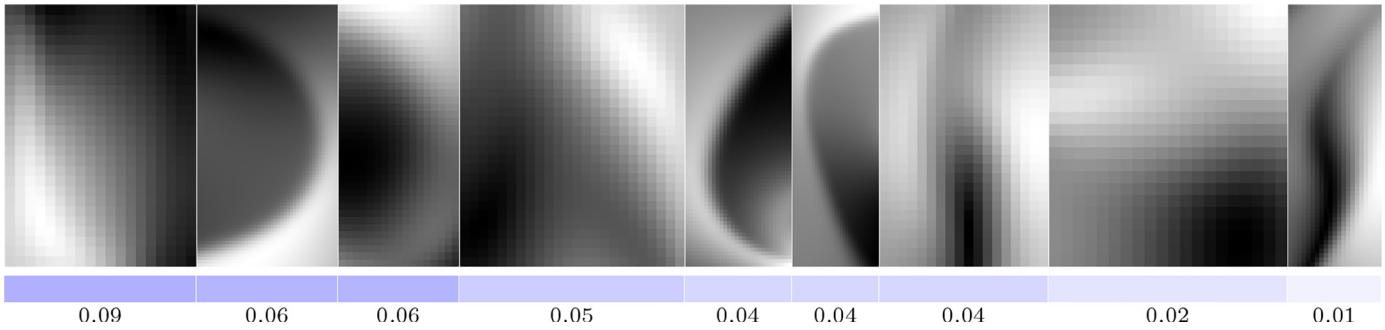



**Fig. 10.** Misclassified samples and the confidence value. The misclassified samples were usually features with very low resolution or with a high level of curvature, which happens often when the feature is close to a vortex but the vortex center is not inside the feature image.

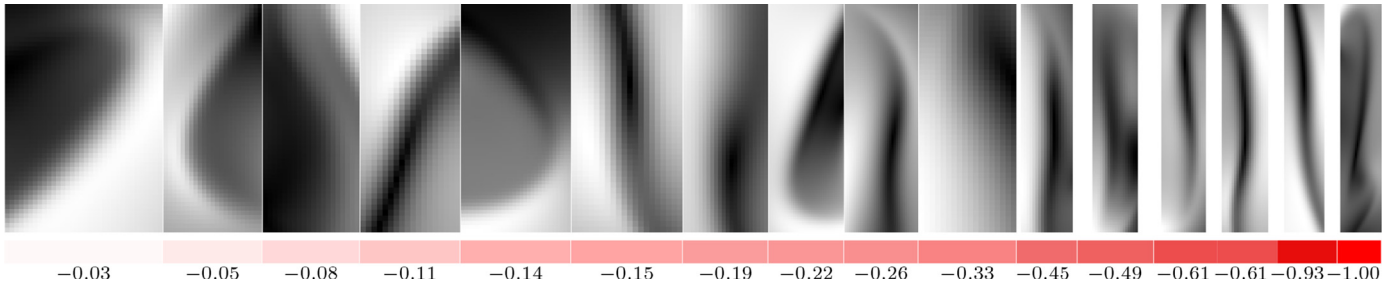



**Fig. 11.** Examples of non-vortices classified using the proposed method and the confidence of the classification. The images show regions with high curvature but that do not contain a vortex center. Usually, these are regions at the boundary of the mixing fluids.

for very large datasets, the use of image analysis and classification can accelerate the data exploration process. The results show that the method was effective to discriminate vortex-like regions from other boundary regions that, even with a strong gradient response, do not present much revolving behavior or the center of the vortex inside of it. We note that some of the false positives and true negatives lying close to the decision boundary of the classifier present the following characteristics: they have a curved behavior very similar to the one in vortices, and they come from low-contrast regions or very small resolution regions. We believe these issues can be solved in part by including more samples with different behavior in the dataset so that the classifier can achieve a better generalization and we can have a better understanding of the limits regarding resolution and contrast limits. In the next section, we explore the use of the method for datasets outside the domain of calibration.

### 4.1. Detection against test datasets

The detection performance of our trained system was tested among three other datasets, previously not used for training or validation, with various levels of mixed quantities. These three datasets correspond to the same data series in the JHTDB, but are from numerical times $t = 8$, 10, and 12, corresponding to frames 200, 250, and 300. While this new test subset is from the same data series, the extracted 2D slices are from a different depth-level in the domain that was not used for training to represent a visually different but similar testing dataset. During extraction, a domain depth of 1/2 was used for training and a depth of 1/4 is used to extract a testing dataset. These time frames are before and after kinetic energy peak ($t \approx 11.4$) and cover the interval when turbulence is most active. However, the amount of molecular mixing progressively increases with time, so that the images are not equivalent among the three time instances.

Fig. 12 at $t = 8$ presents a scenario where features are clearly defined and not yet in a very 'mixed' state. As shown, the clearly defined features allow for the automatic extraction using the ellipse detector to work well, and produce correct classifications. As previously mentioned, features that are too small in size or too elongated are filtered out before being processed by the classifier. Finally, candidate regions classified as vortices are drawn in red while the ones classified as non-vortices are drawn in red.

(a) (b) (c)

**Fig. 12.** A new test dataset (a) with well-defined features represented as velocity magnitude. Our ellipse detector first extracts regions to evaluate (b). After simple filtering of features, a classification response is generated (c) to show that our method is able to identify vortex-like regions (red rectangles) at various resolutions. . (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

(a) (b) (c)

**Fig. 13.** A more mixed dataset (a) represented as velocity magnitude. Our ellipse extracted regions (b) shows the extraction of many candidate regions before being filtered out. Finally, the classification response is shown (c) for regions that fall within the vortex and non-vortex criteria. Our method is able to identify these regions at various resolutions. .

Fig. 13 at $t = 10$ examines a time close to the turbulent kinetic energy peak, increasing the complexity of vortex features. The automatic ellipse detector begins to extract many more regions of interest but those with a diverging eccentricity ratio are removed by a threshold before being processed by the classifier. Finally, the regions that are classified are done so correctly despite overlapping regions.

Fig. 14 at $t = 12$, during the initial stages of the turbulence decay stage samples a more mixed state with small and complex vortices dominating the visual space. Many ellipses are automatically extracted but are further reduced by eccentricity thresholds, removing mainly the smallest, most elongated regions. The result of this trimming produces a subset of regions that are likely to contain vortex activity, and are then able to be classified correctly. This set of results also shows the ability to detect various sizes of vortices correctly within a single dataset.

### 4.2. Characterizing detected objects

To characterize the detection of vortices, we measure their properties and plot the distribution of their regions. The detected regions are defined by rectangles (enclosing the detected ellipses) with angle, width (minor axis) and height (major axis) properties associated to them.

The diagonal of a rectangle is given by

$$c = \sqrt{a^2 + b^2} \tag{10}$$

such that $a$ is the region width and $b$ is the region height. We define the normalized quantity as the *characteristic length scale* (CLS)

$$CLS = \left( \frac{c - c_{\min}}{c_{\max} - c_{\min}} \right) 2\pi \tag{11}$$

such that $c_{\min}$ and $c_{\max}$ correspond to the minimum and maximum normalized values among all classifications of a time series.

The distributions of the CLS at four simulation timesteps $t =$ 5.44, 7.68, 9.84, and 12, corresponding to frames 136, 192, 246, and 300, covering the early and late growth as well as the beginning of turbulence decay, are shown in Fig. 15. As the heavy and light fluid regions start moving due to differential buoyancy forces, the total number of vortices detected in the data increases. Initially, vortices are detected across all scales as shown by the distribution of CLS from 0 to $2\pi$. At very early times, vortices are produced mainly by the Kelvin-Helmholtz instability acting at the interface between the initially segregated pure fluids. These vortices are the size of the pure fluid regions. Due to vortex stretching and interaction with the nearby flow, the vortices then start breaking up. As the turbulence cascade is established, the characteristic vortex size decreases.

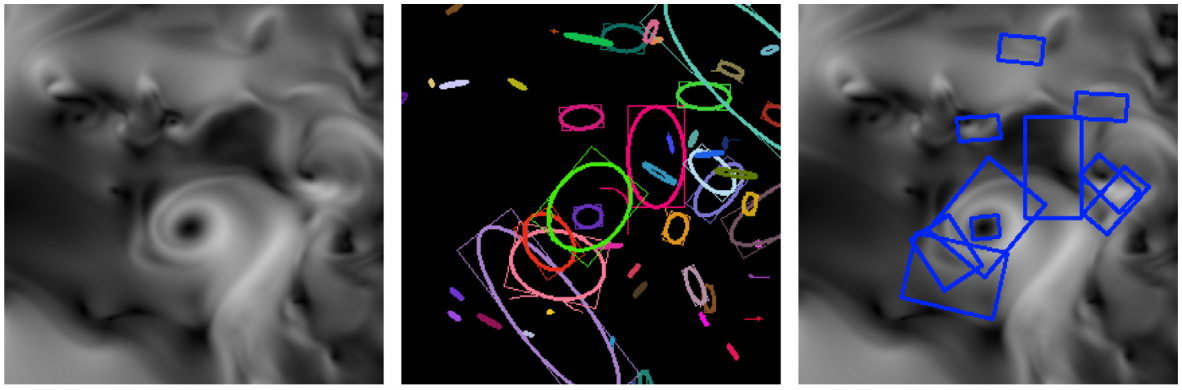



**Fig. 14.** A well-mixed dataset (a) with fine vortex features is represented as velocity magnitude. We are able to extract many candidate regions in (b) based on our ellipse detector and classify them correctly in (c), showing extractions at various resolutions.

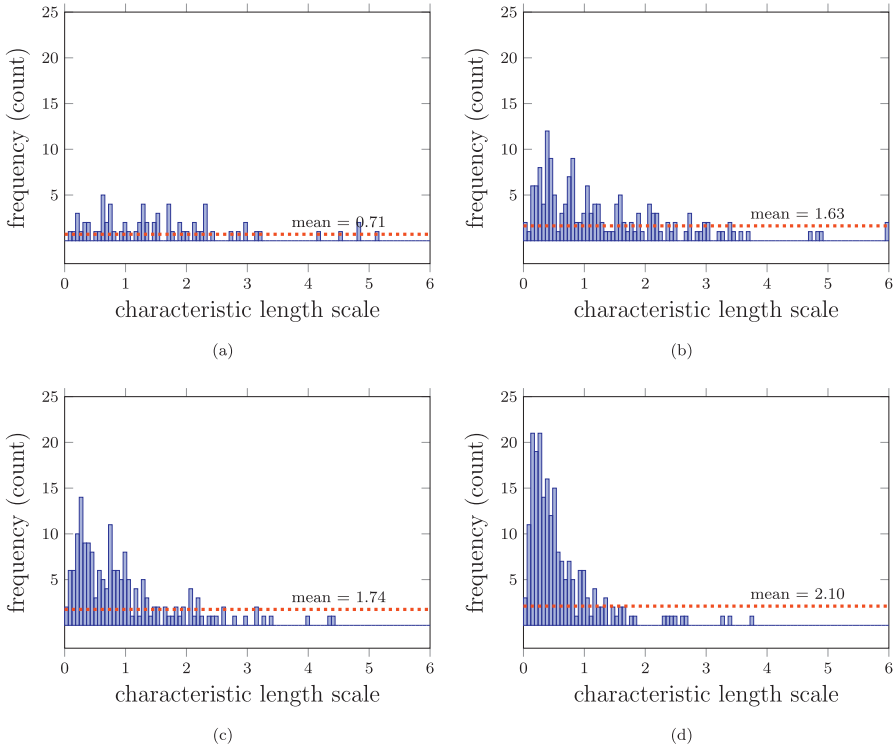



**Fig. 15.** Vortex characterization at four different steps. Time steps $t = 5.44$ (a), 7.68 (b), 9.84 (c), and 12 (d) show the progression of detected vortex features. As the simulation progresses, vortices are detected at larger rates and smaller sizes. The mean of detected objects (red) signifies the increase in vortex appearances as it relates to the increased mixed nature of the simulation as it progresses over time. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

## 5. Conclusions

As current and future supercomputers produce insurmountable amounts of data year-over-year, it has become necessary to use approaches for data reduction to alleviate bandwidth limitations and facilitate the study of new simulations. The generation of Cinema image databases has been a recent popular approach capable of generating high temporal fidelity snapshots of simulations that otherwise would be skipped by conventional restart state writes. Performing data analysis on these massive restart files is prohibitive due to required resources, making the analysis of high-fidelity image database data a prominent approach.

We presented a method for classification of features in flow dataset that uses a 2D approach to discriminate between vortex and non-vortex features. In the method we propose the use of radial gradients to produce a Bag of Words descriptor for the features, which are classified using a Support Vector Machine. The image features are detected using an ellipse detection method. In such a manner our contributions include an image-space method for the detection and classification of vortex-like features and a complete training and classification system that enables the low-cost evaluation of image-space datasets. Our experiments show that the method is able to positively classify vortex features with minor losses while keeping a low rate of positively misclassified features.

Improvements of our method should be possible by increasing dataset size using more samples and exploring methods for data augmentation through geometrical transformations. We intend to examine these possibilities in future research It is also possible to consider a generalization of the method to 3D space. A possibility is to consider adjacent slices of 3D simulation data, in order to use 2D methods expanded with only minor adjustments. By detecting vortices in all dimensions, i.e., in the classes of *XY*-, *YZ*-, and *ZX*-slicing planes, it would be viable to triangulate 3D bounding boxes of regions containing detected vortices. Further, recent improvements were made to the Cinema standard enabling the generation of floating-point image data during visualization. By encoding floating-point image data, it would be possible to reduce a single gray-scale channel representation of image data, quantized from to 255, and encode them at much higher precision.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## CRediT authorship contribution statement

**Jesus Pulido:** Conceptualization, Methodology, Software. **Ricardo Dutra da Silva:** Methodology, Software, Writing - original draft. **Daniel Livescu:** Data curation, Writing - original draft. **Bernd Hamann:** Writing - review & editing, Supervision.

## Acknowledgments

This work has been co-authored by employees of Triad National Security, LLC which operates Los Alamos National Laboratory (LANL) under Contract no. 89233218CNA000001 with the U.S. Department of Energy/National Nuclear Security Administration. D.L. acknowledges funding from the Laboratory Directed Research and Development (LDRD) program at LANL under project 20190059DR.

## References

[1] Jeong J, Hussain F. On the identification of a vortex. J Fluid Mech 1995;285:69–94. doi:10.1017/S0022112095000462.
[2] Post FH, Vrolijk B, Hauser H, Laramee RS, Doleisch H. The state of the art in flow visualisation: feature extraction and tracking. Comput Graphics Forum 2003;22(4):775–92. doi:10.1111/j.1467-8659.2003.00723.x.
[3] Shivamoggi B.K., Heijst G.J.F., Kamp L.P.J.. The Okubo-Weiss criteria in two-dimensional hydrodynamic and magnetohydrodynamic flows. 2011. arXiv:1110.6190
[4] Chang YL, Oey LY. Analysis of STCC eddies using the Okubo–Weiss parameter on model and satellite data. Ocean Dyn 2014;64(2):259–71. doi:10.1007/s10236-013-0680-7.
[5] Zhou Y. Rayleigh–Taylor and Richtmyer–Meshkov instability induced flow, turbulence, and mixing. I. Phys Rep 2017;720–722:1–136. doi:10.1016/j.physrep.2017.07.005.
[6] Zhou Y. Rayleigh–Taylor and Richtmyer–Meshkov instability induced flow, turbulence, and mixing. II. Phys Rep 2017;723–725:1–160. doi:10.1016/j.physrep.2017.07.008.
[7] Zhou Y, Clark TT, Clark DS, Gail Glendinning S, Aaron Skinner M, Huntington CM, et al. Turbulent mixing and transition criteria of flows induced by hydrodynamic instabilities. Phys Plasmas 2019;26(8):080901. doi:10.1063/1.5088745.
[8] Ahrens J, Jourdain S, OLeary P, Patchett J, Rogers DH, Petersen M. An image-based approach to extreme scale in situ visualization and analysis. In: SC '14: proceedings of the international conference for high performance computing, networking, storage and analysis; 2014. p. 424–34. doi:10.1109/SC.2014.40.
[9] Banesh D, Schoonover J, Ahrens J, Hamann B. Extracting, visualizing and tracking mesoscale ocean eddies in two-dimensional image sequences using contours and moments. Workshop on visualisation in environmental sciences (EnvirVis); 2017.
[10] Gospodnetic P, Banesh D, Wolfram P, Peterson M, Hagen H, Ahrens J, et al. Ocean current segmentation at different depths and correlation with temperature in a MPAS-ocean simulation Workshop on scientific visualisation (SciVis) IEEE VIS; 2018.
[11] Zhang L, Deng Q, Machiraju R, Rangarajan A, Thompson D, Walters DK, et al. Boosting techniques for physics-based vortex detection. Comput Graph Forum 2014;33(1):282–93. doi:10.1111/cgf.12275.
[12] Kwon O.H., Crnovrsanin T., Ma K.L.. What would a graph look like in this layout? A machine learning approach to large graph visualization. 2017. arXiv:1710.04328.
[13] Li Y, Perlman E, Wan M, Yang Y, Burns R, Meneveau C, et al. A public turbulence database cluster and applications to study Lagrangian evolution of velocity increments in turbulence. J Turbul 2008;9(31).
[14] Livescu D., Canada C., Kanov K., Burns R., IDIES, Pulido J.. Homogeneous buoyancy driven turbulence data set. 2014. Los Alamos National Laboratory Report LA-UR-14-20669, http://turbulence.pha.jhu.edu/docs/README-HBDT.pdf.
[15] Livescu D, Ristorcelli JR. Buoyancy-driven variable-density turbulence. J Fluid Mech 2007;591:43–71.
[16] Livescu D, Ristorcelli JR. Variable-density mixing in buoyancy-driven turbulence. J Fluid Mech 2008;605:145–80.
[17] Livescu D. Numerical simulations of two-fluid turbulent mixing at large density ratios and applications to the Rayleigh–Taylor instability. Phil Trans R Soc A 2013;371:20120185.
[18] Livescu D. Turbulence with large thermal and compositional density variations. Annu Rev Fluid Mech 2020;52:309–41.
[19] Takacs G, Chandrasekhar V, Tsai SS, Chen D, Grzeszczuk R, Girod B. Fast computation of rotation-invariant image features by an approximate radial gradient transform. IEEE Trans Image Process 2013;22(8):2970–82.
[20] Luo Z, Chen J, Takiguchi T, Ariki Y. Rotation-invariant histograms of oriented gradients for local patch robust representation. In: Asia-pacific signal and information processing association annual summit and conference; 2015.
[21] Csurka G, Dance CR, Fan L, Willamowski J, Bray C. Visual categorization with bags of keypoints. In: Workshop on statistical learning in computer vision, ECCV; 2004. p. 1–22.
[22] Fei-Fei L, Perona P. A Bayesian hierarchical model for learning natural scene categories. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), 2; 2005. p. 524–531 vol. 2.
[23] Feng J, Jiao LC, Zhang X, Yang D. Bag-of-visual-words based on clonal selection algorithm for SAR image classification. IEEE Geosci Remote Sens Lett 2011;8(4):691–5.
[24] Zhu Q, Zhong Y, Zhao B, Xia G, Zhang L. Bag-of-visual-words scene classifier with local and global features for high spatial resolution remote sensing imagery. IEEE Geosci Remote Sens Lett 2016;13(6):747–51.
[25] Ben-Hur A, Horn D, Siegelmann HT, Vapnik V. Support vector clustering. J Mach Learn Res 2002;2:125–37.
[26] Osuna E, Freund R, Girosit F. Training support vector machines: an application to face detection. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition; 1997. p. 130–6.
[27] Boser BE, Guyon IM, Vapnik VN. A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on computational learning theory. COLT '92. New York, NY, USA: Association for Computing Machinery; 1992. p. 144–52. ISBN 089791497X. doi:10.1145/130385.130401.
[28] Cortes C, Vapnik V. Support-vector networks. Mach Learn 1995;20(3):273–97. doi:10.1023/A:1022627411411.
[29] Schölkopf B, Smola AJ, Williamson RC, Bartlett PL. New support vector algorithms. Neural Comput 2000;12(5):1207–45. doi:10.1162/089976600300015565.
[30] Moreland K. Diverging color maps for scientific visualization. In: Bebis G, Boyle R, Parvin B, Koracin D, Kuno Y, Wang J, et al., editors. Advances in visual computing. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. p. 92–103. ISBN 978-3-642-10520-3.
[31] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), 1; 2005. p. 886–93.
[32] Mallat S. A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way. 3rd . Orlando, FL, USA: Academic Press, Inc.; 2008. ISBN: 0123743702, 9780123743701
[33] Fitzgibbon A, Fisher R. A buyer's guide to conic fitting. In: Proceedings of the 6th British conference on machine vision (Vol. 2). BMVC '95. Surrey, UK, UK: BMVA Press; 1995. p. 513–22. ISBN 0-9521898-2-8. http://dl.acm.org/citation.cfm?id=243124.243148
[34] Itseez. Open source computer vision library. https://github.com/itseez/opencv; 2018.
[35] Bradski G, Kaehler A. Learning OpenCV: Computer Vision in C++ with the OpenCV Library. 2nd . O'Reilly Media, Inc.; 2013. ISBN: 1449314651, 9781449314651