Master of Science Thesis

Identification of Large-Scale Structures in Turbulence

Austin Keanu Maui Ramanna

October 31, 2023

Aerodynamics

Faculty of Aerospace Engineering

TU Delft

Delft University of Technology

Identification of Large-Scale Structures in Turbulence

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering at Delft University of Technology

Austin Keanu Maui Ramanna

October 31, 2023

Faculty of Aerospace Engineering · Delft University of Technology

Delft University of Technology

DELFT UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF AERODYNAMICS

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance the thesis entitled **"Identification of Large-Scale Structures in Turbulence"** by **Austin Keanu Maui Ramanna** in fulfillment of the requirements for the degree of **Master of Science**.

Dated: October 31, 2023

Supervisors:

_____
Reader 1

_____
Reader 2

_____
Reader 3

_____
Reader 4

Preface

I want to thank my supervisors Dr N.A.K. Doan and Dr Ir. G.E. Elsinga for developing the Master's thesis topic, supervising and providing datasets and the isosurface to continuous structure MATLAB code. I also want to thank Dr T. Ishihara and Dr A. A. Wray for making their datasets available to the department.

Summary

Currently, a lot of phenomena within fluid mechanics cannot be explained (such as extreme dissipation events and the influence of large-scale motions on shear layers) due to a lack of statistics and general knowledge of the activity at the largest scales. It is hypothesised that there exist approximately uniform flow regions with a similar velocity field at these scales called large-scale structures which would require a reassessment of conventional turbulence theory. The lack of knowledge is caused by the difficulty of reliably identifying these large-scale structures. In homogeneous isotropic turbulence (HIT), this is particularly difficult as these structures seem to lack a preferential alignment leading to abstract geometry and they only appear at high Reynolds numbers (Re) which are computationally expensive to simulate. Presently, simulating high Reynolds numbers is no longer impossible, however, finding these large-scale structures has remained an unexplored area of research.

This thesis aims to develop a reliable method to identify large-scale structures in turbulent flows. Three independent methods are all tested. The clustering algorithm DBSCAN is used to group points together based on coherence in the spatial and velocity domain. Alternatively, a 2D histogram is derived from the velocity orientation and this histogram is fed to both YOLOv7 and U-net, two types of neural networks. YOLOv7 has seen significant success in that object detection can be done live and has widespread use. U-net has previously been used to outperform human detections in the medical field

In the end, only U-net turned out to be applicable in our problem and subsequently, a full methodology around this network was developed. Key processing points are that the U-net is entirely trained on artificial histograms, made with a simple Python program. Another is that 4 different 2D histograms were fed to U-net, mimicking the surface of a globe. Three datasets were used with Taylor Reynolds numbers of $Re_\lambda = 170$ and $Re_\lambda = 1131$, respectively (the two datasets at the large Re were a sub- and downsampled volume of the same underlying set) For post-processing, the detected peaks were converted to structures which were consequently evaluated.

The structures indeed appear to behave as large-scale structures. The largest instances

contain in some cases more than 60% of the total kinetic energy and occupy more than 80% of the computational domain. To assess if these structures were also in the order of the integral length scale, length scales were computed using Minkowski functionals and shapefinders and PCA analysis. Minkowski functionals did, however, not function correctly, producing no realistic results. This was attributed to the many holes in the structures which cannot be handled easily with Minkowski functionals. PCA yielded three perpendicular maximum distances within the structure. These were typically 4-6 standard deviations and in the order of the integral length scale. From the length scales obtained by PCA, it was interesting to see that, at high Re, the structures are significantly larger than at low Re, relative to the integral length scale. Finding a clear explanation for this phenomenon would require further research, particularly doing a detection around $Re_\lambda = 250$.

In conclusion, automated detection is able to identify realistic large-scale structures. Furthermore, the algorithm only requires a handful of user-defined parameters that have been assigned default settings which were not changed for the different datasets. For further research, it will be interesting to see how these detected large-scale structures interact with well-known smaller structures such as shear layers as well as how they scale with Reynolds number.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Turbulence can be considered as a state of the flow that is chaotic. Slight changes in boundary and initial conditions will result in greatly different flow developments, making it difficult to predict the behaviour of a turbulent flow accurately. This chaotic nature means that turbulence research is therefore heavily reliant on heuristics (Nieuwstadt et al., 2016), non-random behaviour, empirically obtained, that can somewhat be predicted based on flow properties. The transition from laminar (characterised as the opposite of turbulence, namely a smooth and organised flow state) to turbulent is a gradual process when increasing the Reynolds number and varies for different flows. The Reynolds number is defined as the ratio between inertial and viscous forces as denoted by Equation 1.1. Note that a Reynolds number is always computed using a particular length scale and generally for turbulence research, the Taylor microscale λ is used which is defined as the transition boundary between the regions of viscous force and kinetic force domination (also in Equation 1.1, U is the root mean square of the overall flow velocity fluctuations and ν the viscosity). Dissipative effects dominate in the region smaller than the Taylor microscale while diffusion occurs at scales larger than the Taylor microscale (Pope, 2000). From around a Reynolds number of Re_λ 250 onward, the flow is generally fully turbulent (in that there is a clear distinction between small and large energy scales)(Elsinga et al., 2017).

$$Re_\lambda = \frac{U\lambda}{\nu} \tag{1.1}$$

Within fully developed turbulent flows certain regions with similar, somewhat uniform flow properties may be present, referred to as structures. These structures vary heavily in size and with several orders of magnitude between the largest and smallest scales. The largest

scales have a length scale and velocity in the same order as the general flow scale (the integral length scale) and velocity (Hickel and Hulshoff, 2020). These large-scale structures are hypothesised to contain most of the kinetic energy in the flow and have an approximately uniform velocity. Small-scale structures are typically defined as structures in the order of $O(10)$ Kolmogorov length scales (the smallest scale in the domain). They are known to have strong gradients and an approximately uniform vorticity (Pope, 2000).

According to Kolmogorov's hypothesis (Kolmogorov, 1941), the small-scales are for the most part isotropic in highly turbulent flows which means that their statistics are largely independent of the spatial orientation. It has also been found that the small scales have their vorticity vector and intermediate eigenvector of the strain rate tensor typically aligned while also having their probability density function of the velocity gradient tensor's second and third invariants in the form of a teardrop (Elsinga and Marusic, 2010). Large-scales, however, are highly anisotropic which makes them harder to identify. This means that they are heavily dependent on the flow domain and in case of homogenous isotropic turbulence, their alignment will be random.

The main goal of this research is to create an efficient and automated method to find these large-scale structures in homogeneous isotropic turbulence (HIT). Three methods are proposed based on findings from literature: clustering datapoints based on the three (spherical) velocity dimensions and three spatial coordinates using DBSCAN (Ester et al., 1996) is the first one. The second method has the datapoints grouped in a 2D histogram based on the velocity angles and then deploys YOLOv7 (Wang et al., 2022), a computer vision algorithm to extract the peaks. The third method replaces YOLOv7 with a U-net (Ronneberger et al., 2015), a semantic segmentation neural network which can detect complex patterns in images. Either way, more efficient access to these structures would allow researchers to spend more resources on analysing the behaviour of these structures and how they interact with smaller scales.

The overall academic value that this identification and subsequent research would bring is further explored in chapter 2 where the overall value that the efficient identification of large-scale turbulent structures will bring is explored. Chapter 3 provides an overview of current identification methods for turbulent structures in general. The limitations of these current methods are then synthesised into the main research question in chapter 4. After this synthesis, methods from the fields of machine and deep learning are considered. Chapter 5 explores various clustering methods, namely DBSCAN and MKNN as well as neural networks, YOLOv7, faster R-CNN and U-net. Chapter 6 relates the methods to the identification of large-scale turbulent structures and establishes sub-questions that they bring with them. The entire detection process is then presented in chapter 7, the methodology. Chapter 8 showcases all the results that are obtained from this methodology and finally, chapter 9 concludes the work and highlights key areas for future work.

Chapter 2

Background

This chapter covers research which is currently limited due to the lack of knowledge on large-scale turbulent structures.

There have been two challenges with detecting large-scale structures in turbulent flows, in flows without a preferential velocity alignment they have abstract shapes hence making them difficult to observe and they are computationally expensive to simulate. (Ishihara et al., 2013) for example have been able to simulate a flow using direct numerical simulation (DNS) with a Reynolds number of $Re_\lambda = 1131$. In these conditions, one would expect to find large-scale structures. Other techniques such as experiments with particle image velocimetry (PIV) could also be used to acquire the data. (Dou et al., 2016) for example, were able to capture turbulent statistics such as kinetic energy and dissipation in a flow of $Re_\lambda = 346$.

Some simulation techniques, instead of capturing a velocity field in a specific volume, track particles instead. The Lattice-Boltzman method (LBM) and simulations involving Lagrangian frames of reference. While these methods are excellent at displaying a flow's time evolution and seeing regions where these particles cluster, they are typically not used to capture instantaneous behaviour, such as flows in HIT. LBM is typically inferior to Navier-Stokes DNS for studying turbulence since it fails to accurately model dissipation (Suss et al., 2023). Lagrangian frames are limited only to homogeneous flows with no spatial development or inhomogeneous but self-similar (Ouellette, 2021).

Within the heuristic exploration of turbulence from Eulerian DNS, large-scales have been shown to play a significant role. When two large-scale structures are close to each other, it is speculated that a shear layer appears between them and in this shear layer, the most

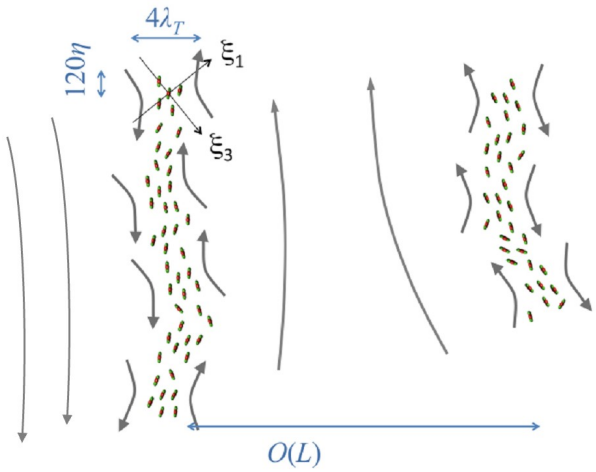


Figure 2.1: Straining motions (Elsinga et al., 2017)

intense vortex stretching (the process by which energy is passed on to smaller scales) and dissipation are present. This is shown in Figure 2.1 where large- and small-scales structures are denoted by the large arrows and green blobs respectively. As a side-note, the shear layers are indicated with the mid-size arrows. Within large-scale structures, the flow is mostly quiescent by comparison (Elsinga et al., 2017). One could then make the argument that the most valuable information on large-scale structures is their geometry and spatial location since this would allow for improved predictions on shear layers between them. It would be fascinating to observe that a shear layer does occur during a collision between two large-scale structures with greatly different velocity magnitudes causing a strong velocity gradient. Furthermore, one could potentially observe whether a large-scale structure is surrounded by shear layers or if shear layers only occur along certain parts of their edge.

The small-scales themselves are also directly influenced by the large-scales. At a Reynolds number of $Re_\lambda = 250$ there was a clear distinction between large- and small- scales and local vorticity was modulated by large-scale velocity fluctuations. Furthermore, the correlation coefficient between the two scales appeared to be primarily dependent on the spatial organisation of the large-scale structures (Fiscaletti et al., 2016). Note that small-scales being affected by large-scales is in direct contrast with the Kolmogorov hypothesis (Kolmogorov, 1941) which assumes small-scales to be nearly isotropic and mostly independent of the large-scales (Pope, 2000). Therefore research on large-scale structures could play a pivotal role in further assessing the validity of this hypothesis and would allow greater insight into scale interactions and dependencies.

Another interesting area of research from (Elsinga et al., 2017) is in the case of a flow being conditioned on extreme dissipation. Large-scales were shown to significantly influence

enstrophy production (vorticity creation) and to some extent extreme dissipation itself. While extreme dissipation is typically not that common (Elsinga et al., 2020), it is still worth studying. It is a strong driver for nonlinear energy transfer in turbulent flows, currently only predictable up to several eddy turnover times before the event, in a limited number of flows (Blonigan et al., 2019). Due to the influence large-scales have on extreme dissipation, one could hypothesise that a better understanding of large-scales would increase our capability to predict extreme dissipation.

A final field worth mentioning in which large-scale instantaneous turbulent structures play a significant role is aeroacoustics. From comparison of DNS simulations with Amiet's theory, it was found that for trailing edge noise at the edge of a symmetrical airfoil, fully turbulent flows cannot be quantified by Amiet's theory due to frozen turbulence being an invalid assumption (Sandberg, 2015). This finding would suggest the need for improved turbulence modelling of large-scale structures. Within aeroacoustics literature, instantaneous large-scale structures are also recognised as the primary noise-generating mechanism due to their coherence (Jordan and Colonius, 2012). One could however hypothesise, that these coherent structures are different from those found by (Elsinga et al., 2017) as they travel far further than the integral length scale and tend to have high vorticity (Miller and Shen, 2018). Note that while the extended travel distance could also occur due to transportation by the mean flow, high vorticity is not commonly associated with large-scale structures.

From the discussion above, being able to identify large-scale structures reliably would be good for turbulence research. There have been instances of large-scale motions affecting the smallest scales, a result in direct contrast with classic literature that can currently not be clearly explained. Large-scale motions were also responsible for enstrophy production and enstrophy is directly related to other flow patterns such as shear layers and strong velocity gradients. Finally, large-scale motions seem to be related to extreme dissipation which is not a well-understood energy transfer phenomenon.

Chapter 3

Turbulence identification

This chapter covers the most significant identification methods for structures within turbulence which have been established by the literature. The work on small scales is discussed initially since this is the widest body of literature. Afterwards, current methods deployed on large scales are covered.

3.1 Small-scale structures identification

Most effort in developing identification methods has been targeted towards small-scale structures. The Q-criterion (Hunt et al., 1988), Q_s-criterion (Tabor and Klapper, 1994), Δ-criterion (Chong et al., 1990) Λ_2-criterion (Jeong and Hussain, 1995) and swirling strength (Zhou et al., 1999) are all commonly used ways to identify small-scale structures. The Q-criterion is the second invariant of the velocity gradient tensor, the Δ-criterion is derived from both the second invariant and the third invariant R, the Λ_2-criterion is derived from pressure extremes within the velocity field. Finally, the Q_s-criterion is a variation of the Q-criterion with strain taken into account and the swirling strength computation splits the velocity gradient tensor into a symmetric and an asymmetric component which represent strain and vorticity respectively. A threshold is then set for each criterion so that only the structures with the highest vorticity remain.

Some more recent methods such as Omega-Liutex have managed to somewhat mitigate the issue of selecting a suitable threshold by taking the relative vorticity instead of the absolute which is more robust (Liu et al., 2019). In addition, these methods decompose the velocity gradient tensor into rotational and non-rotational components. Doing this prevents shear

contamination which is the process of a high shear being mistaken for high vorticity (Liu and Wang, 2020).

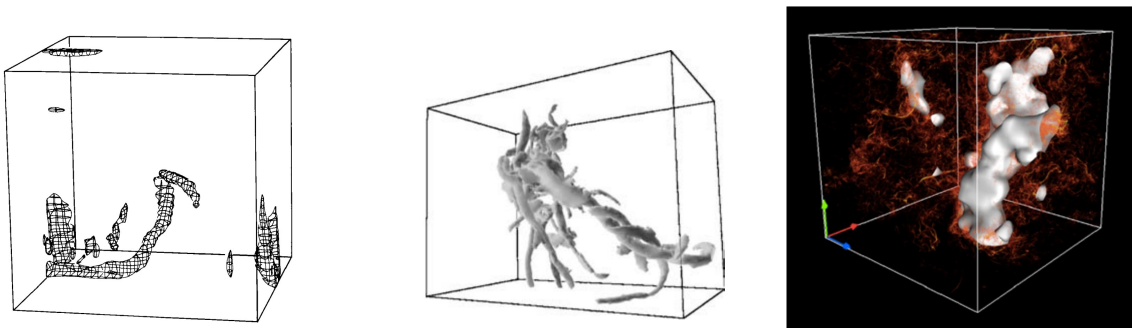
While both the newer and older methods have allowed research to make significant progress in identifying small-scale structures, an extension to large-scales seems unlikely. All of the methods rely on computing velocity gradients which are inherently high for small scales due to their smaller size. Large-scale structures, however, for the most part, consist of quiescent flow and are also not typically associated with high vorticity (Elsinga et al., 2017).

3.2 Velocity decomposition techniques

In turbulence research, it is common practice to decompose a velocity field based on modes and then perform the analysis on these modes. Popular techniques are proper orthogonal decomposition (POD) (Chi et al., 2017), Dynamic Mode Decomposition (DMD) (Yang et al., 2022), Proper Generalized Decomposition (PGD), Fourier analysis (Pope, 2000) and wavelet analysis (Horiguchi et al., 2012). While these decompositions make it easier to look at particular aspects of the flow, such as large velocity amplitudes, for example, they inherently filter out information. Decomposed visualisations will not be able to capture complete interactions between small- and large-scales or the development of a shear layer between two large-scale structures. Furthermore, it is more ambiguous to identify a large-scale structure in this modified space and then revert it to the physical space. A layer (such as an edge of a large-scale structure) in physical space typically corresponds to a broad range in spectral space which means a basic spectral filter (using some cut-off frequency) will not completely capture these layers. While this research has offered noteworthy contributions, it will not be considered in the thesis due to the limitations mentioned above.

3.3 Large-scale structures identification

Large-scale structures were first discovered in DNS through spectral analysis (more on limitations of spectral analysis in section 3.2) in which it was found that they are tube and worm-like, as can be seen in Subfigure 3.1(a) (Siggia, 1981). One could, however, argue that this is not a typical large-scale structure since the Reynolds number Re_λ was only 100 which is not sufficiently high to form a full energy cascade. For some turbulent flows such as wall-bounded or channel flows, large-scale structures have a preferential alignment with the free-stream and are located close to the wall, making them easier to observe (Marusic and Heuer, 2007). This alignment is however, not present in all flows, for example, in homogeneous isotropic turbulence (HIT) (Saltzman et al., 2021) (and potentially also in jet flow as this type of flow appears to become more isotropic further downstream (Pope and



(a) (b) (c)

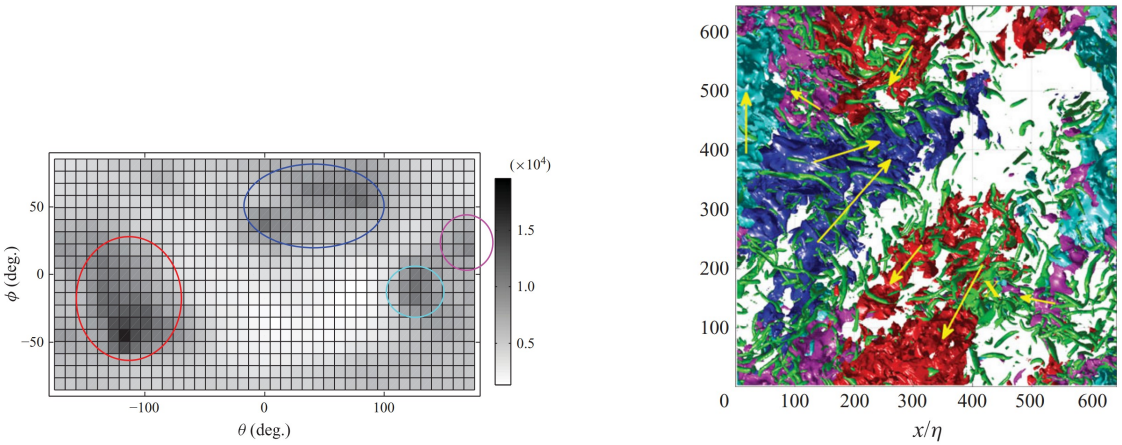
Figure 3.1: Large-scale structures from literature. (a) (Siggia, 1981). (b) (Moisy and Jiménez, 2004). (c) (Ishihara et al., 2013)

Chen, 1982)) the alignment with fixed world coordinates appears to be more random.

While over the years DNS outcome visualisation has improved in terms of Reynolds number and computational domain which allows for analysis in physical space, most research is still done through visual inspection and without standardised procedure. (Ishihara et al., 2013) for example, are able to visualise 'medium'-scale structures as seen in Subfigure 3.1(c). These are the shear layers which are supposedly in between the large-scales and thus only two out of their three dimensions are of the order of the integral length scale (think of the shear layer as a thin sheet between two blobs, the area will have the same dimensions as the blobs but the thickness will be much smaller). The identification procedure is, however, rather limited and not universal: only the structures above a vorticity threshold are detected and such a threshold is not known a priori for a flow. Unlike spectral methods, it does allow for simple backtracking to the original flow field.

Another technique is box-counting, as used by (Moisy and Jiménez, 2004); a method to zoom into a particular part of the field. It can detect regions of high vorticity and high strain, which could be the edges of large-scale structures, see Subfigure 3.1(b). However, the detected layers do not capture significant amounts of turbulent kinetic energy, a quantity closely related to large-scales (Kolmogorov, 1941).

The most relevant procedure so far is arguably the one developed by (Elsinga and Marusic, 2010). Large-scale structures are defined as a region with a uniform velocity which means that the velocity in the region has a similar direction and magnitude. Using this knowledge, they collect all DNS datapoints in a two-dimensional (2D) histogram based on the azimuthal θ and elevation ϕ angles, as shown in Subfigure 3.2(a) with the coordinate system from Figure 3.3. The peaks in the histogram show the imprint of the large-scale structures, which can then be easily traced back to physical space, as shown in Subfigure 3.2(b). All points associated with the peaks from the histogram are plotted in this figure with each



(a) (b)

Figure 3.2: Conversion of peaks from histogram to structures in physical space (Elsinga and Marusic, 2010). (a) Histogram of velocity orientation. (b) Large-scale structures, obtained from the histogram.

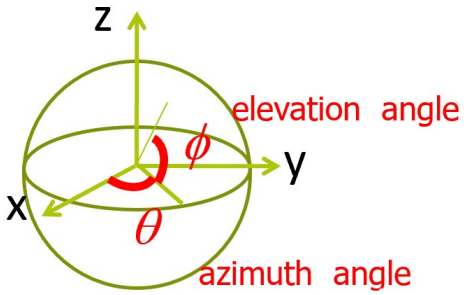


Figure 3.3: Definition of the angles used in the histogram (Elsinga, 2023)

structure having its own colour. Note that the angles can be linked to the surface of a sphere which means θ, ϕ bins around the equator are relatively larger compared to those near the poles. As a type of normalisation, the squares are divided by the cosine of the elevation angle which compensates for this effect.

While the histogram method is a great step in the right direction, there is still a lot of room for improvement. The most glaring issue is what to do with near-zero velocities. Theoretically, the regions also have a similar velocity and could thus appear as peaks in the histogram. Any small perturbation of the velocity has a negligible effect on the velocity magnitude. However, it may significantly affect the local flow direction. It means that these regions have a rather undefined direction that causes them to appear as noise which could potentially distort the figures. In the research of (Elsinga and Marusic, 2010), the velocity magnitude was thresholded at $1.3U_{rms}$ for each datapoint. This value cannot be known a priori and is thus prone to significant variance per dataset.

Another limitation is the histogram resolution. A histogram has a finite resolution, meaning there might be structures that do not appear as peaks due to being overshadowed by larger structures. To mitigate this, the authors divided the domain into five equal slices in the z-direction. While this allows for more detail per each of the five histograms, large clusters which span across multiple slices can no longer be directly identified as one large structure. Furthermore, the histograms also lack spatial resilience; if two independent clusters in space have the same velocity direction, they will appear as one large peak. Furthermore, selecting appropriate bin sizes could play an issue. However, this could potentially be scaled with the size and variation of the dataset. One final issue is that it is a rather manual-intensive process to sample all points belonging to a peak and this extracting is prone to error.

Chapter 4

Intermediate conclusion on turbulence literature

From the literature on turbulence, it can be seen that there is a gap in knowledge regarding large-scale structures. Information on their size, distribution, shape, relative orientation, scaling and velocity is rather limited. Extensive research has been done on small-scale structures and the shear layers they occur in. Yet, the initiation of the shear layers and whether these shear layers always occur at the edge of large-scale structures is barely studied. Furthermore, large-scale structures appear to play a significant role in the development of extreme dissipation and could be the main noise driver of several aeroacoustic mechanisms.

The lack of knowledge can be attributed to the lack of a reliable identification mechanism. Most methods either decompose the velocity field or use dataset-specific filtering. Both of these methods inevitably lead to a loss of information. To be able to generate statistics and draw conclusions on large-scale structures, far more data on them is required.

For further research within the thesis, some main points can be established. Turbulent large-scale structures are of the order of the integral length scale, spatially coherent and have an approximately uniform (nonzero) velocity. They have high kinetic energy but not necessarily high vorticity. The 2D histogram from Elsinga and Marusic (2010) seems to be a step in the right direction as it identifies datapoints belonging to a cluster and allows for simple backtracking. There are still some obstacles associated, however. Initially, a universal method needs to be developed that can remove low and zero-velocity regions without removing actual large-scale structures so that no false clusters appear or valuable data is filtered out. Furthermore, the velocity field volume needs to be carefully modulated so that extremely large-scale structures and smaller large-scale structures can both be identified.

Finally, there needs to be a post-processing mechanism that can check if all the datapoints from a peak belong to the same spatially coherent structure in physical space and if not, separate different structures.

To achieve the main research goal of identifying large-scale instantaneous turbulent structures reliably, the following main research question is posed:

How can large-scale structures be reliably and automatically identified in turbulent flows?

To fully answer this research question, some constraints on this research question are necessary. Large-scale structures are defined as approximately uniform (non-zero) velocity regions, proportional to the integral length scale and spatially coherent. Turbulent flow is considered as flow with a Reynolds number of at least $Re_\lambda = 250$ and only HIT flows will be considered in the thesis as they are canonical and present a challenging case where the large-scale structures are randomly oriented in space.

To achieve reliability in large dataset analysis needed for a full statistical evaluation of the large-scales, the aim should be to automate the identification process as much as possible. One could then consider relying primarily on computer algorithms. Before selecting suitable algorithms, several considerations need to be taken into account first. Inherent to the problem, model performance will be impossible to assess objectively since the definition of a large-scale structure itself is subjective. This means a qualitative analysis between the methods should be performed. The next subsequent chapters will provide research on suitable algorithms. This will allow for the development of sub-research questions associated with the qualitative analysis. The overall conclusion will also include sub-research questions to compare model results with literature as a means of validation.

Chapter 5

Literature on identification techniques

This chapter explores two ways of deploying machine/learning techniques to discover large-scale structures: clustering, instance segmentation and semantic segmentation.

5.1 Clustering

Upon observing the large-scale turbulent structure directly, one can observe that all data-points belonging to the same large-scale structure have roughly similar velocity magnitude and direction and are close to each other in physical space. A clustering algorithm could then potentially group the data points belonging to the same large-scale structure into a cluster. This is done by computing the Euclidean distance between the points in the space of the features. For turbulence, this would be a six-dimensional space consisting of two velocity angles, the velocity magnitude and the three spatial dimensions.

5.1.1 DBSCAN

One clustering algorithm that comes to mind, in particular, is DBSCAN (Ester et al., 1996). This algorithm stands out in that it does not require a predefined number of clusters defined by the user. This is ideal for turbulence since the number of expected large-scale structures per dataset is often not known a priori. The algorithm analyses a predefined radius ϵ in feature space (note that this is, in this case, 6-dimensional due to the three velocity directions and three spatial coordinates) around each point in the dataset. For

other points: it selects a point and checks how many other points there are in its vicinity. If a minimum number of points minPts are found in the radius a cluster is started and each point within this cluster that also meets the threshold is considered a core point. If a point is within another point's radius that meets the threshold but does not meet the threshold itself, then this point is considered a boundary point. Else, if a point is not associated with any cluster, it is considered a noise point. See Subfigure 5.1(a) for a graphic illustration of the different points (Schubert et al., 2017).

(a) (b)

Figure 5.1: DBSCAN illustrations, (a) DBSCAN clustering structure where A is a core point, B and C boundary points and N is a noise point. (Schubert et al., 2017) (b) k-distance graph where the needle is in the bottom right corner with a value of around 1.8 (Bedre, 2023)

So far DBSCAN has only seen limited success in turbulence research (Mowlavi et al., 2022). It was used to identify Lagrangian clusters in Bickley flow, however, at a small ϵ only small structures were identified and at a large ϵ only large ones. While setting the MinPts to 2 times the number of dimensions, selecting a proper value for ϵ does not come with any universal rule and is considered a challenging task. One recommendation is to first compute the average distance of every data point to its k-nearest neighbour where k is an integer chosen by the user. A low value will result in a high variation, this variation can be reduced by increasing the number at the cost of higher computational expense. The graph in Subfigure 5.1(b) can then be computed and its knee is considered the optimal epsilon value. The exact location of this knee can be tuned using the sensitivity parameter S. The higher this value, the more to the left in the graph the knee will be taken (Bedre, 2023). Unfortunately, there is no way to automate selecting a proper S value and hence this could potentially change per dataset. The S value is, however, more robust than epsilon itself. For example, when coarsening the grid, the ideal ϵ value would change as points are relatively further from each other. S would stay roughly constant since coarsening is accounted for in computing the average distance between points. Note that k is also a hyperparameter that needs to be set by the user, however, it is quite robust and can be set to a high value to reduce variation (at the expense of some computational efficiency). To evaluate clustering results from DBSCAN, the primary method arguably has to be a visual inspection. So far it is not yet known whether clustering performance metrics such as the Calinski-Harabasz index (Calinski and Harabasz, 1974) also correspond to a desired identification of large-scale

structures and hence these metrics would have to be validated first.

5.1.2 MKNN

Another clustering algorithm which has previously been used in turbulence is mutual-K nearest neighbours (MKNN) (Ye et al., 2005). This algorithm again has two user-defined hyperparameters, K the number of neighbours (which are points in close vicinity) and threshold λ a regularisation threshold. A larger λ creates smoother clusters at the expense of detail. Unlike DBSCAN which previously struggled with identifying Lagrangian clusters of different densities at the same time, MKNN had no such difficulties (Wei et al., 2022). There are, however, some limitations with MKNN, in that finding appropriate K and λ is more difficult. K will scale with dataset increase and selecting a too-large value will result in increased computational expenses. λ also has no reliable method for identification and typically one selects an appropriate value through trial error: running the clustering multiple times and selecting the most suitable one.

The remainder of the chapter explores the usability of the 2D histogram for automated identification procedures. Local maximum search algorithms will be considered and later a thorough analysis of instance segmentation through neural networks is provided. This section explores both the challenges that come with using such an algorithm and will explore various networks that could be suitable.

5.2 Local maximum search algorithms

Initially, one could think that a local averaging algorithm should easily be able to detect peaks in a 2D histogram. These algorithms are very fast and simple to code. However, such an algorithm is extremely dependent on the size of the local average for the number of detected peaks. The optimal setting will undoubtedly vary per turbulent dataset and thus each dataset will require individual tuning which is far from ideal for a universal procedure. Even more advanced search algorithms such as differential evolution still require a tremendous amount of tuning per dataset. For example in an experimental study, for noise source detection with beamforming, only a handful of tuned settings out of 50 delivered sufficient performance (Malgoezar et al., 2017).

Another theoretical issue with search algorithms is that they typically only find the maximum value, not the entire peak. The algorithm would have to be modified to find the entire peak, however, this would drastically slow it down since now it needs to acquire far more data.

5.3 Instance segmentation

5.3.1 Practical considerations

An approach that will undoubtedly be able to perform the task of identifying peaks in a 2D histogram is to perform a visual inspection through a neural network. For computer visual inspection, there are different levels of complexity (Gupta, 2020). For identifying peaks in a histogram, object detection requires the minimal amount of complexity. This method identifies objects in an image and draws a bounding box around them. It can distinguish different objects and locate them. The downside is that the bounding boxes are not very precise and it is difficult to capture complex objects in a rectangle without also capturing a significant amount of noise. Semantic segmentation is a more complex visual method in that it binarises an image so that each pixel is evaluated. The drawback, however, is that multiple instances of the same class can no longer be distinguished. The most complex visual method is instance segmentation which evaluates each pixel and is also able to distinguish multiple instances of the same class. While all three methods could work after some post-processing (removing noise from the boundary boxes and isolating different instances in a binary map through some form of spatial evaluation) instance segmentation will be deployed in this thesis.

While there are numerous different networks, there is a baseline for all. A neural network is first shown a large number of images which contain the desired object (in this case the peak). The network makes a prediction and is then shown where the objects are and what type of objects they are. The additional information with the correct positions and types is called a label. After thousands of labelled images containing the objects, the algorithm learns to recognise the desired objects. Learning in neural networks occurs through adjustments to the network's parameters, commonly referred to as weights. During training, these weights are iteratively updated to minimise the error between the network's predictions and the actual data. The process also involves a non-linear transformation via an activation function, which introduces the necessary complexity to model non-linear relationships. Once sufficiently trained, the network applies these weights to new data for fast and effective detection. These networks have seen fast success in other fields, they can for example, detect certain tumours with a higher degree of accuracy than certified doctors (Nazir et al., 2021) and they are used in Tesla's autopilot (Bouchard, 2019).

A potential issue that could be pointed out is that neural networks typically require 2000-10000 labelled images to achieve sufficient performance and gathering that amount of labelled 2D histograms would excessive time and is unfeasible. Luckily, however, this problem can be solved by creating artificial images and several approaches are worth mentioning. The first approach is to create Gaussian peaks combined with blur in a randomised manner and fine-tune them to resemble the 2D turbulent histograms. Another more crude way is to

perform data augmentation on existing images, for example, by mirroring or rotation. The downside is that the images will not be independent and thus, one has to act with caution and only generate a limited amount of images this way.

Another possibility which may seem interesting at first is to generate artificial images through a Generative Adversarial Network (GAN) (Goodfellow et al., 2014). GAN networks can learn features from an object and then generate artificial images (commonly used for 'deep fakes' such as the generation of faces from non-existent humans). Theoretically speaking, a GAN network could then generate artificial labelled images to feed the instance segmentation neural network. In practice, this is, however, undesired: GAN networks, just like instance segmentation neural networks, require a significant amount of training data and one could argue that these artificial images do not provide significant added value. This is because the GAN learns certain features of an object and then slightly alters these features to come up with a new representation, essentially it would pass these features to the instance segmentation neural network. The object neural network would then learn its features from the image which when using a GAN will never learn more than the initial GAN. Seeing as instance segmentation networks are optimised for this purpose and generic GANS are not, one is better off feeding the original dataset directly to the instance segmentation neural network. An argument could be made that the GAN would learn very different features and thus provide some value. However, such an investigation is beyond the scope of the present thesis.

One final consideration is inherent with the histograms themselves, some post-processing will undoubtedly be required. The peaks need to be converted to clusters in physical space and then in physical space, they need to be evaluated for spatial coherency. It is also ideal that clusters are extracted from the histogram and that the histogram is then reevaluated to detect new peaks. This extraction would easily allow both larger large-scale structures and smaller large-scale structures to be identified without too much tuning being required. Such an extraction mechanism would, however, have to be designed first and could potentially increase run times.

5.3.2 YOLOv7

The premier algorithm that should be considered is YOLOv7 (Wang et al., 2022) which is part of the YOLO (Redmon et al., 2016) (you only look once) family. These networks scan an image as a whole (this is different from some methods that will be discussed in the next sub-sections) which makes them extremely fast for detection and even allows for live tracking of objects in video. YOLOv7 for example, has a frame-rate between 5 and 160 frames per second depending on the specific model. Due to its simplicity, the network can also be run on a simple mobile GPU, making it highly versatile. As for performance, the algorithm achieves an average precision (AP) of 56.8% on the MS COCO dataset. Note that AP is the

area under the Precision-Recall curve, which plots a model's precision (ratio of true positives to all positive predictions) against recall (ratio of true positives to all actual positives) at different threshold levels. At the time this AP score is the highest performance achieved by a real-time computer vision network (Lin et al., 2015). Another benefit of YOLOv7 is that it is very adaptable to transfer learning; the open-source algorithm starts with the weights from the MS COCO dataset and these are usable for many different tasks. The reason for this being is that image representations often become abstract and simple in the middle layers of a network so that there is little variation per task. Often only the first and last two layers require different weights. The universality of the weights makes YOLOv7 relatively computationally cheap.

YOLO networks consist of three segments, a backbone, a tail and a head. The backbone extracts features from an image, the features are linked and evaluated in the neck and finally, a prediction is made in the head. YOLOv7 has several key features that make it superior to other YOLO networks. The network uses an E-ELAN (extended efficient aggregation network) as the final aggregation layer which can backpropagate (backpropagation is the process of sending information from the prediction accuracy up to the previous layers) information more efficiently. Another advantage is related to scaling (this is done to optimise the model for different inputs or tasks). The output of a layer is first scaled according to the scaling of the input of the layer and then the next layer is scaled with the same factor to match the previous output. This procedure allows for improved scalability. Finally, YOLOv7 uses auxiliary heads at various stages of the model. These allow for early (less accurate) predictions which help improve training times. YOLOv7 has these heads in a coarse-to-fine manner which is unique in that information from the lead head is passed to not only itself but also to the auxiliary heads (Wang et al., 2022).

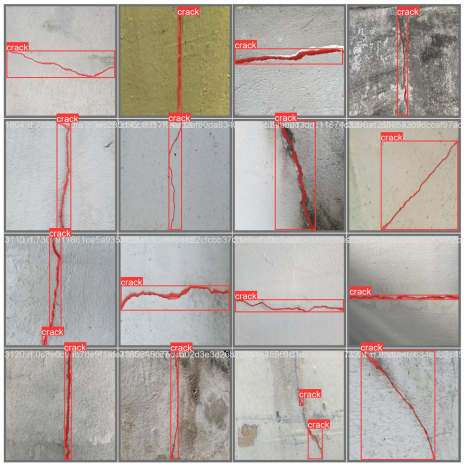


Figure 5.2: Segmentation used for crack detection (Skalski, 2022)

One last valuable feature of YOLOv7 worth mentioning is that it is easily compatible with

instance segmentation. This can be seen in Figure 5.2 (Skalski, 2022).

5.3.3 Faster R-CNN

Typically in the field of computer vision, one had to make the trade-off between a fast algorithm with little detection time (once trained) and one with slower detection time, however, with higher localisation accuracy. The YOLO family of networks have typically been known for their speed, falling into the former category. For accuracy, 'the regions with CNN features' (R-CNN) (Girshick et al., 2016) have typically been the state of the art (note that CNN stands for convolutional neural network). These networks perform a selective search on an image to extract parts of an image which are then warped and fed to a classifier. Initially Alexnet (Krizhevsky et al., 2012) and later on VGG16 (Simonyan and Zisserman, 2015) and ResNet50 (He et al., 2015) were most used. The state of the start in recent years is Faster R-CNN (Ren et al., 2016) which was able to make the algorithm more efficient by making all parts GPU compatible.

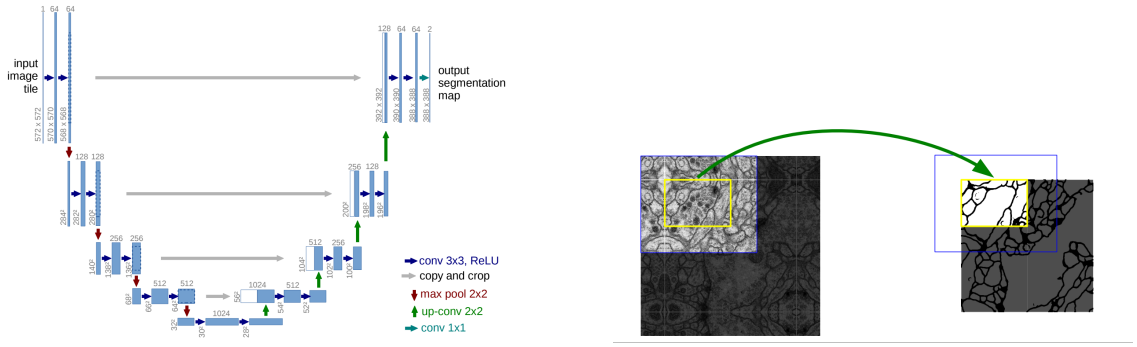
In recent times, however, YOLO networks have been able to beat faster R-CNN on speed and accuracy. YOLOv5 was already able to outperform faster R-CNN (Dwivedi, 2020) and the current YOLOv7 is significantly better than YOLOv5 both in terms of accuracy and speed (Kumar and Srivastava, 2021). Hence for the research in this thesis, the YOLO branch of networks is deemed more suitable than R-CNN's.

5.4 Semantic segmentation

5.4.1 U-net

An interesting alternative to instance segmentation is semantic segmentation. These networks have the advantage of being significantly less complex. An image label can simply be a binary map where 1 indicates it is part of the object. The particular network that will be considered for this task is U-net, as seen in Subfigure 5.3(a). The main feature of this network is that it performs pooling after convolution operations in the encoder part which allows the network to analyse an image at various depths. The information from each level is then taken into account when reconstructing the image in the decoder part, giving the network its U shape.

Another advantage of U-net is its tune-ability, the network can easily be adapted for the histograms. For example, the original histogram as seen in Subfigure 3.2(a) is a 36x18 image, which means only a limited number of pooling operations can be performed without



(a) (b)

Figure 5.3: Illustrations regarding U-net from (Ronneberger et al., 2015). (a) Original U-net architecture. (b) U-net detection on neuronal structures in EM stacks from (Cardona et al., 2013)

requiring additional padding. Using YOLOv7 for example, the image would have to be converted to a 64x64 before it can be fed to the network. However, one can simply take a U-net divided by 2 and then by 3, allowing the fed image to be unmodified when entering the network (with a 36x18 size and input and output). This U-net also has significantly fewer parameters than YOLOv7, reducing the likelihood of overfitting and drastically reducing training times.

U-net was originally developed for the medical field. Just like turbulence, there are also highly complex structures present, for example as seen in Figure 5.4. Instead of applying manual thresholds, the general direction is to leverage neural networks for the initial detection Azad et al. (2022) as this is cheaper and more accurate. Another note is that similar to turbulence, there is only limited training data available which means artificial data is to be relied upon heavily. In the end, U-net was able to outperform human detection on the dataset of Figure 5.4 from (Cardona et al., 2013), see example detection in Subfigure 5.3(b). Seeing how neural networks have overcome human limitations in the medical field it can be argued that an attempt at turbulence structures is worthwhile. The electron microscopy EM stacks from Figure 5.4 seem far more complex than the peaks in the histogram from Subfigure 3.2(a) which further suggests automated detection is feasible.

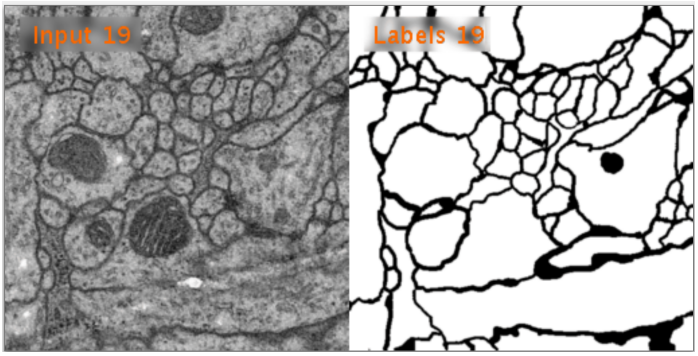


Figure 5.4: Image and label of neuronal structures in EM stacks from (Cardona et al., 2013)

5.5 Conclusion on detection systems

While in theory, a clustering algorithm would be able to identify large-scale structures as they could be grouped based on their approximately uniform velocity and spatial coherence, in practice this will be difficult. Even DBSCAN, which appears to be the most robust of the algorithms has a radius around each point which needs to be tuned and this tuning is very difficult to perform a priori. Implementing the K-distance while making the algorithm more robust, leaves parameters K and S to be tuned.

In the worst-case scenario, the tuning requires as much time as the identification procedure by (Elsinga and Marusic, 2010). While over time metrics might be discovered that can quickly determine if parameters are good, it is anyone's guess since turbulent flows vary heavily when subject to different flow conditions (such as Re). Within the scope of the thesis, however, there is some value to be gained. Identifying large-scale structures through an algorithm such as DBSCAN can act as a verification for other methods when properly tuned on a specific dataset. Nevertheless, despite its limitations an attempt will still be made to reliably identify large-scale structures using DBSCAN.

From this chapter, it can be argued that either YOLOv7 or U-net are the most suitable networks for identifying peaks in a 2D histogram. In either case, the detection system will have to be either semantic or instance segmentation due to the complexity of the structures. To acquire sufficient data, the main method will be to artificially generate 2D histograms using Gaussian peaks. It is difficult to predict how the networks will react to the artificial data since training from 100% artificial data has seldom been done before.

Chapter 6

Literature Synthesis and formulation of research questions

Now that both the literature review on turbulent structure detection and the literature review on suitable methodology have been presented, the research goal can be further elaborated. It was already established that to answer the main research question of 'How can large-scale structures be reliably and automatically identified in turbulent flows?' a key focus lies with reliability. Reliability can be met through automation and for this automation, DBSCAN, YOLOv7 and U-net are considered. Using three independent methods poses two interesting research sub-questions:

1. What is the variation in identified structures between the three methods?

If the variation is minimal and the three methods identify nearly identical structures then all methods become more plausible. The other question is:

2. What is the superior identification algorithm between DBSCAN, YOLOv7 and U-net for large-scale turbulent structure identification?

Assuming that both models can achieve desirable performance (more on performance requirements later in this chapter) given enough tuning or training, superiority can be further evaluated with the following two sub-questions:

2a. Which algorithm is the most robust?

Robustness is defined as maintaining performance when testing on a different dataset. Ideally, not much modification is needed between different datasets so the algorithm is mostly universal.

2b. Which algorithm is the least expensive in terms of training/run time and required data?

Machine and deep learning algorithms such as DBSCAN, YOLOv7 and U-net tend to consume considerable time and computing power; making their use potentially limited. Therefore, the more resource-efficient an algorithm, the more it can be deployed.

As previously mentioned, a vital aspect that cannot be overlooked is the general performance of the algorithms. This can be done by validating the identified structures with observations from the literature:

3. How comparable are the detected large-scale structures compared to those found in the literature?

While this question could directly validate the structures, it is not definitive. Large-scale structures in literature are rather limited and thus, other ways to validate identified large-scale structures with literature are required:

3a. What are the typical large-scale structure dimensions compared to the integral length scale?

It was established that large-scale structures are in the order of the integral length scale, hence this is worth checking.

Chapter 7

Methodology

This chapter presents all the required steps to go from a 3D turbulent velocity field to large-scale structures. The final method was obtained through an iterative process, steps were modified or changed from the initial blueprint to get better performance. U-net turned out to be the only feasible method, hence its methodology is worked out in greater detail in Sections 7.1 through 7.7. Section 7.8 discusses the DNS datasets on which the detections are performed and highlights their key statistics such as length scales. Finally, DBSCAN and YOLOv7 methodologies are also detailed in Sections 7.8 and 7.9, respectively.

7.1 U-net flow diagram

A Flow diagram of the complete U-net detection process can be seen in Figure 7.1. Step 1. pre-processes the velocity field directly obtained from DNS to a format on which U-net can be deployed. This U-net requires some preparation as well, which is covered in Step 2. Step 3. is the actual U-net detection and subsequent post-processing steps to get all the datapoints belonging to each detected structure. In step 4. some general statistics are computed which give an initial estimate of the detection quality. Step 5. is used to isolate continuous structures from the detected point cloud. Finally, in Step 6. length scales are derived from these isolated continuous structures.

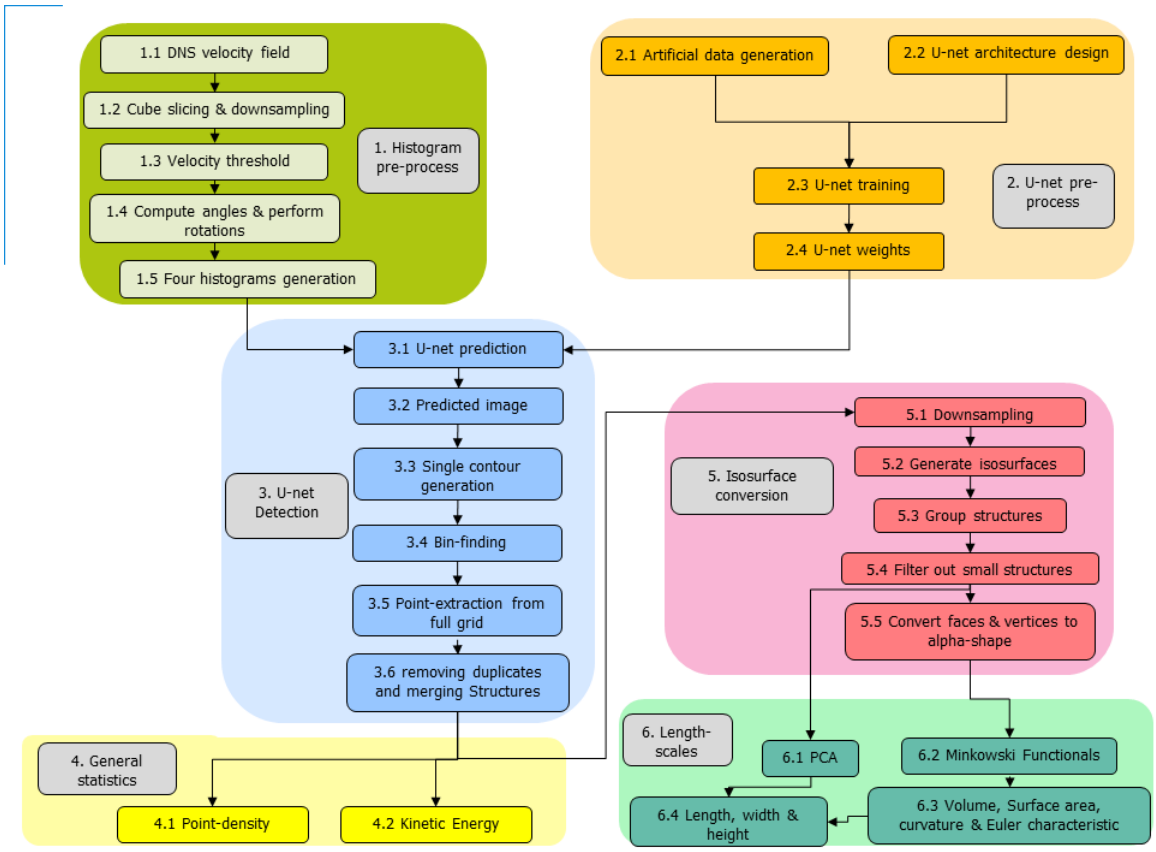


Figure 7.1: Flow diagram U-net methodology

7.2 Step 1: Pre-processing the velocity field

To apply the U-net, some data processing steps need to be taken first. As is done by (Elsinga and Marusic, 2010), the velocity field can be converted to a histogram where the x-axis is the azimuthal angle and the y-axis is the elevation angle. Figure 7.2 is a graphical illustration of Step 1.

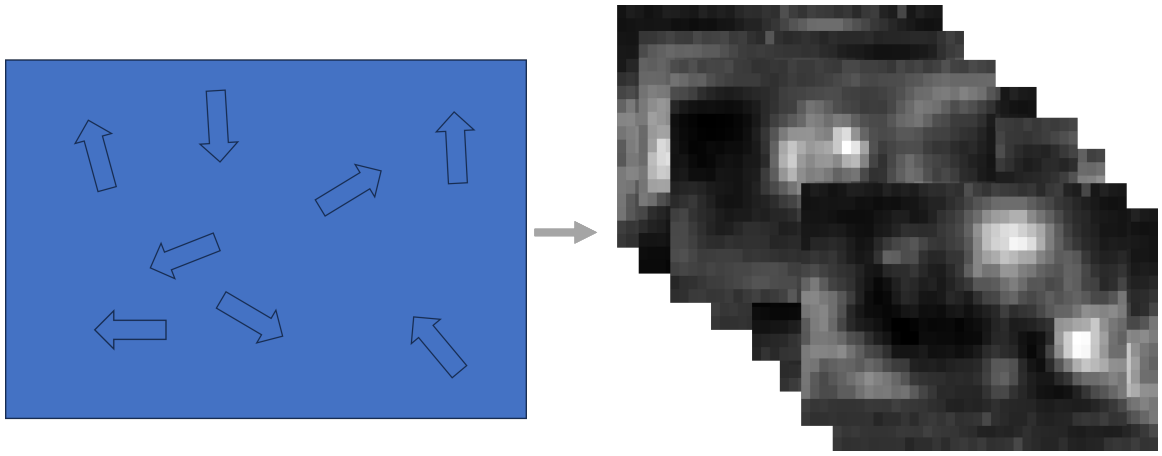


Figure 7.2: Step 1: Pre-processing the velocity field. This illustration shows how the velocity field is converted from physical space into 8 histograms.

7.2.1 Step 1.2: Cube slicing and downsampling

An inherent challenge is that the velocity field is random in terms of orientation and isotropic which means that if the dataset is large enough, each histogram bin is equal and no peaks can be detected. This problem is mitigated by slicing the velocity domain up into smaller cubes and performing the histogram generation and detection on each of them separately. Cubes are preferred over slices to mitigate creating a preferential direction and thus introducing a bias. As the default setting for all runs in the thesis, a slicing of 2 is used, meaning that there are $2^3 = 8$ cubes in total.

As a means of speeding up computations, there is an option in the code to downsample the field. A potential risk of this downsampling is that too much information is lost and no clear structures can be identified in the histogram or visualised as an isosurface. This risk is mitigated by computing the dataset's integral length scale in terms of grid points and making sure that this length scale is represented by at least 5 remaining gridpoints in every direction. As large-scale structures are expected to be of the order of the integral length scale in every direction, this downsampling limit seems sufficient. It allows us to get a general idea of the large-scale structure's location and shape with respect to the integral length scale L. For the default setting here, no downsampling was used.

To validate with (Elsinga and Marusic, 2010) an identification is done without any downsampling and using 5 z-direction cut slices. Both the detection and subsequent structures can then be compared.

7.2.2 Step 1.3: Velocity threshold

Before these angles are computed for each data point, however, applying a threshold filter is still desirable to remove low-velocity datapoints which have an ill-defined velocity direction. As these points have a low velocity, their kinetic energy is also low and thus it is assumed that omitting them will not have significant implications in large-scale structure detection.

7.2.3 Step 1.4: Compute Angles and perform rotations

The 2D histograms are a combination of the azimuthal angle, given by Equation 7.1 and the elevation angle, as defined per Equation 7.2. Note that the azimuthal angle has [-180,180] degrees domain while the elevation angle [-90,90]. By using the directions, the coordinate system becomes essentially spherical.

$$\phi = \arctan 2(v, u) \tag{7.1}$$

$$\theta = \arctan 2\left(\sqrt{u^2 + v^2}, w\right) \tag{7.2}$$

The challenge of having this sphere (of the angles) projected on a 2D image requires several adjustments. Arguably the most striking is that the image is horizontally periodic, a structure can end on one edge and reappear on another. To compensate for this, two images will be subjected to the network, one being untouched and the other being shifted 180 degrees. Near the poles (near an elevation angle of (-) 90° the bins will be relatively larger, therefore a weighting of $\frac{1}{\cos\phi}$ is applied to the joint PDF of the angles. The poles also have a form of interplay in that each vertical edge is connected to the edge shifted 180 degrees horizontally from it (just as Antarctica is round in reality while on world maps it looks like a stroke). However, this is more of a theoretical situation as in reality, datapoints very close to 90 or -90 degrees elevation angle have an ill-defined azimuthal angle and therefore the detection here is inaccurate. Consequently, a high-probability peak spreads over a very large range of azimuthal angles which significantly reduces peak height and increases peak width. Both effects reduce the detectability of the peak.

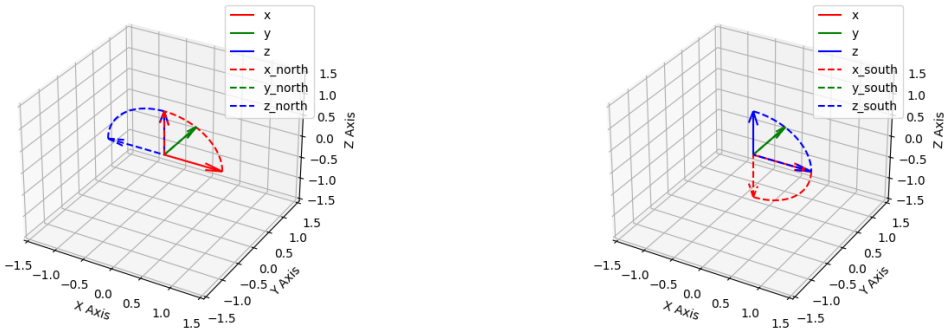
7.2.4 Step 1.5: Four histograms

To still be able to capture structures close to the poles a second set of images with a plus or minus 90-degree vertical shift is also forwarded to the neural network. This transformation

is, however, more complex as it requires the coordinate system singularity to be moved from the pole to the equator. It is done by converting the spherical coordinates to Cartesian coordinates and then performing Equation 7.3 and Equation 7.4 for the north and south poles, respectively. Figure 7.3 shows how the coordinate system is rotated for each of these transformations. Once the shifts are performed, the data points are once again converted back to spherical coordinates. To avoid detecting the same structure twice in different figures, the pole-shifted structures are compared on a point-wise basis to the original two figures. If there is a 60% or more overlap between two structures, only the largest one is kept.

$$x_{northpole} = z, y_{northpole} = y, z_{northpole} = -x \tag{7.3}$$

$$x_{southpole} = -z, y_{southpole} = y, z_{southpole} = x \tag{7.4}$$



(a) (b)

Figure 7.3: Coordinate rotations for the poles. a) North pole, the system is rotated 90° counter-clockwise in the zx-plane. b) South pole, the system is rotated clockwise 90° in the zx-plane

7.3 Step 2: Pre-processing U-net

To prepare for the detection, the U-net first has to be pre-processed. The artificial data is created from scratch as well as the network architecture architecture. There is also a brief section on how the training is performed.

7.3.1 Step 2.1: Artificial data

One of the main challenges of neural networks is to find sufficient data so that they can learn to detect the desired objects. As of the time of doing this research, there is insufficient approximate turbulence with identified large-scale structures to create a large enough dataset. To overcome this limitation, artificial data will be created. Similar to the histograms, this data contains peaks and some noise. The peaks are modelled as ellipses with variable radii and angles. The values of the peaks and noise (as well as the number of peaks per image) are tuned to match those of the histograms. An example image is displayed in Subfigure 7.4(a), in total 10,000 images are created in a similar fashion. On a simple i9 core, the generation takes roughly 10 minutes. If desired more images can easily be produced.

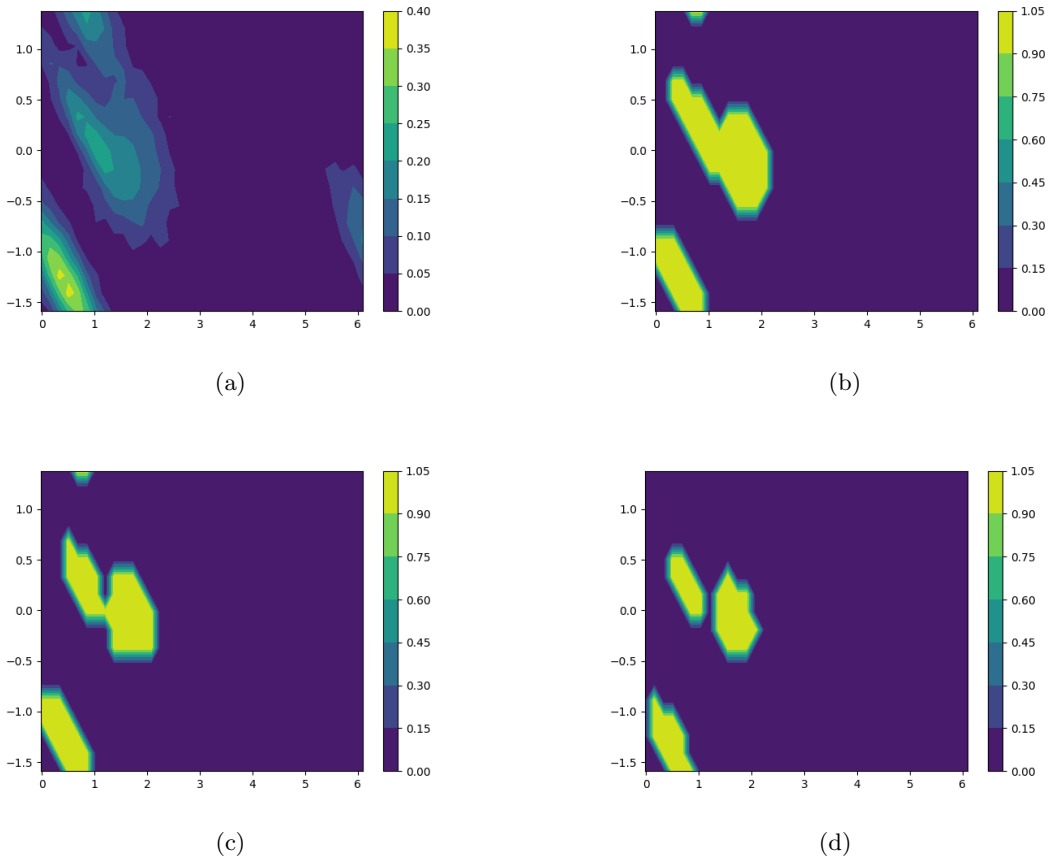


Figure 7.4: Generated histogram image example in (a) artificial image with 3 different label settings in (b) 70% of peak maximum label, (c) 75% of peak maximum label and (d) 80% of peak maximum label. 70% of maximum, for example, means values of 70% of the maximum and higher are included in the label

Arguably the hardest part about this generation is the labels. To train a neural network, the network makes a prediction and is then shown the correct result, this correct result is called the label. A perfect label would mean that one already has perfect knowledge of what a peak would look like and therefore the U-net would be obsolete. The challenge of creating a perfect label is the exact same as being able to detect a structure in the histogram. Using the fact that the peak values in the artificial data are modulated and the locations are known, it is decided to create a boundary around each peak. This peak radius is a sensitive hyperparameter, Subfigure 7.4(b), Subfigure 7.4(c) and Subfigure 7.4(d) show that changing the radius creates very different labels. As the best radius is not known a priori, all three settings will be used to detect structures.

7.3.2 Step 2.2: U-net architecture design

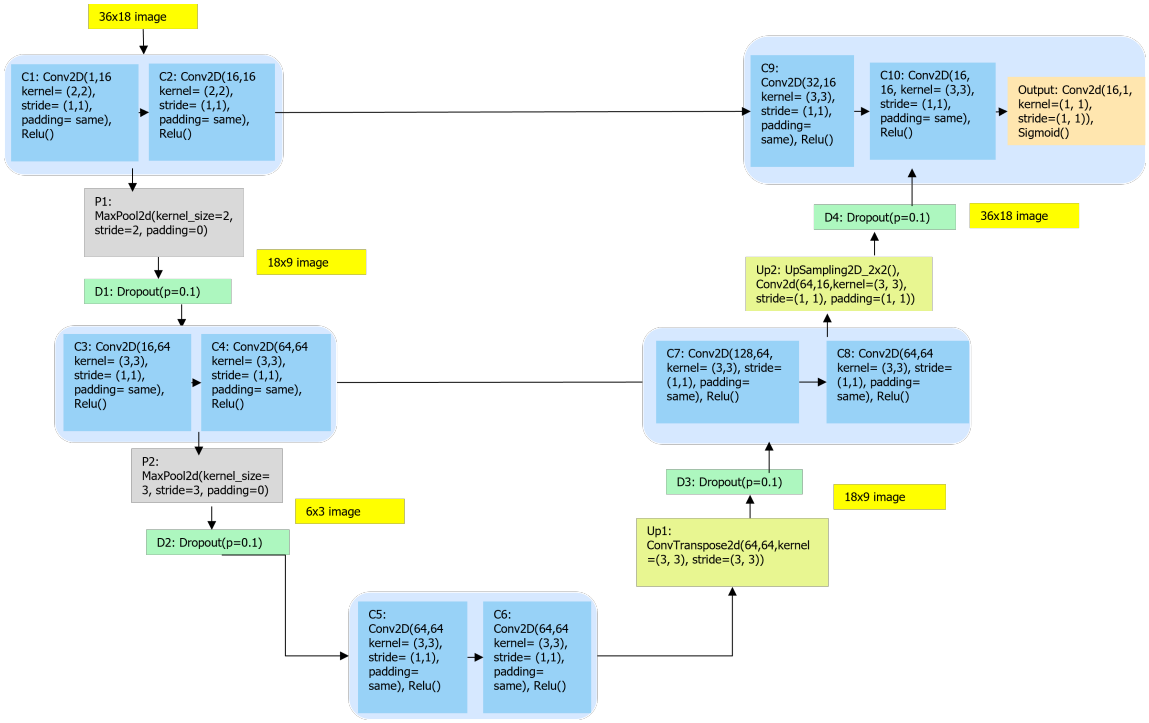


Figure 7.5: Network overview. Blue are convolution followed by ReLu operations, grey max-pooling, green dropout, yellow upsampling followed by convolution as well as convolutional transpose and orange convolution followed by a sigmoid.

The complete modified network is built from scratch with PyTorch and can be seen in Figure 7.5. Note that the network only has three levels of depth instead of the five from the original shown in Subfigure 5.3(a). Furthermore, a 3x3 pooling is used as the second pooling to better match the 36x18 histogram shape as mentioned before. In the upsampling

a transposed convolution allows the image to go from a 6x3 back to a 18x9 shape, essentially mirroring the Maxpooling 3x3. Finally, upsampling followed by a 2D convolution is used instead of a transposed convolution for the last upsampling to keep the risk of checkerboard patterns appearing limited (Odena et al., 2016) while also being more efficient in terms of number of parameters.

Metric	Value
Total parameters	285,473
Trainable parameters	285,473
Non-trainable parameters	0
Total mult-adds (Units. GIGABYTES)	4.39
Input size (MB)	0.26
Forward/backward pass size (MB)	153.50
Params size (MB)	1.14
Estimated Total Size (MB)	154.91

Table 7.1: Model Metrics

Table 7.1 shows the general metrics of the network. Note how there are only around 300.000 parameters which is significantly smaller than YOLOv7 which has 6 to 151 million.

7.3.3 Step 2.3: Training the U-net and step 2.4: U-net weights

The network is trained on an NVIDIA GeForce RTX 3080 Ti Laptop GPU with 16 GB for a batch size of 64 and 250 epochs. The learning rate started at 0.0001 and was updated using the Adams optimiser (Kingma and Ba, 2014). As the network contains so few parameters, it is expected that a run will only take a few minutes and thus it is not deemed necessary to optimise the hyperparameters for training speed. In total four different networks are trained each with a different label radius. They are stored in separate weight files using the PTH file type from PyTorch.

7.4 Step 3: The U-net detection

This step has U-net deployed on the generated histograms from the velocity field. After this detection, contours are used to isolate the different structures. The structures from the different cubes are then merged if deemed identical enough and for each structure, a point cloud (binary grid) in physical space is created. See Figure 7.6 for a graphic overview of Step 3.

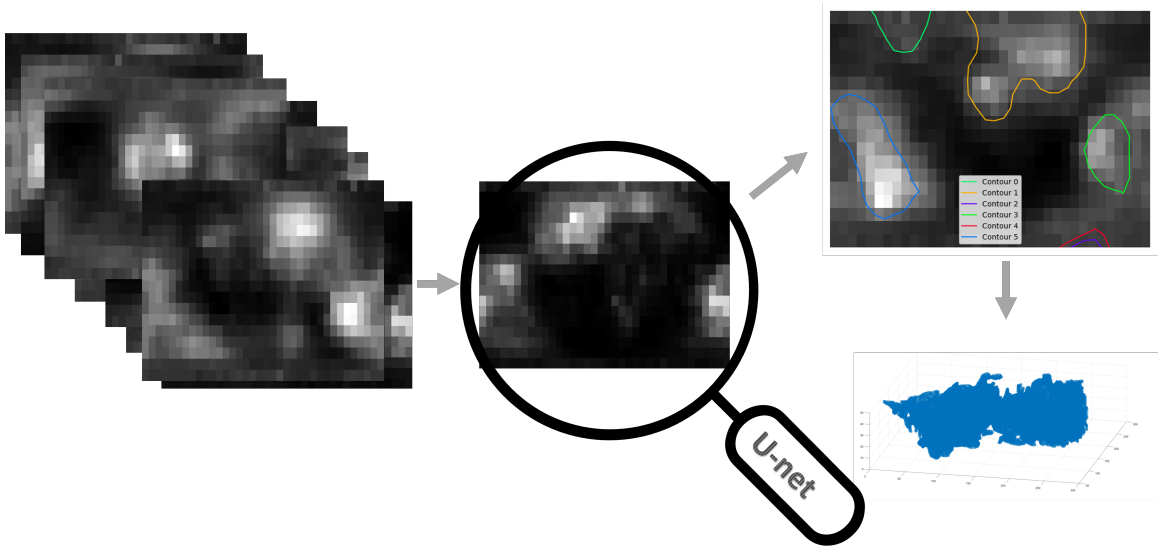


Figure 7.6: Step 3: The U-net detection. This illustration shows how a set of points associated with a large-scale structure is derived from the velocity orientation 2D histogram using U-net.

7.4.1 Step 3.1: U-net prediction and Step 3.2: Predicted Image

All histograms are fed to the U-net as 36x18x1x1 PyTorch tensors which exactly match the network's input layer. The U-net prediction is then also a 36x18 image with values from zero to one. Where values close to one represent the detected peaks. Using the Matplotlib contour function, contours are created around the peaks. This contour function is vital as it is able to distinguish different peaks, essentially creating an instance segmentation network. Hence, multiple independent structures can be detected in the same histogram.

7.4.2 Step 3.3: Single contour generation

To process the image detection and go to actual points, some further steps still need to be taken. All contours are checked with each other per image and then those completely enclosed by a larger one are removed. Contours touching boundaries are also eliminated by checking if they have at least two points along the contour very close to a boundary.

7.4.3 Step 3.4: Bin finding

The next step is to find all the bins enclosed by each contour and then find all the points that belong to these bins. Figure 7.7 is used to verify both steps through visual inspection. The bin search algorithm output which is a list of indices, is compared to the bins enclosed by the contours in the figure which then have to be the same. For the shifted images, the bins are also shifted accordingly.

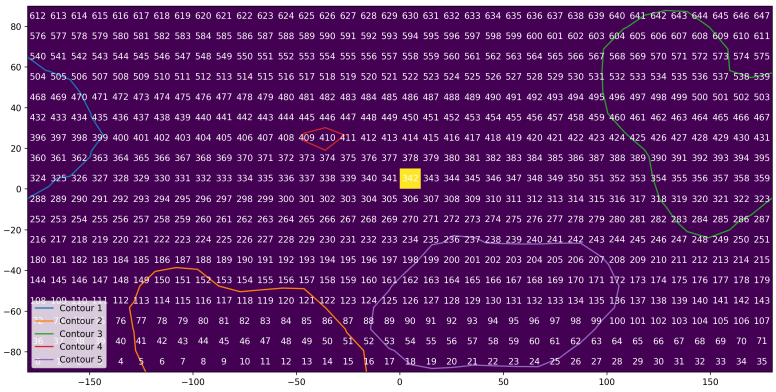


Figure 7.7: Numbering of bins. The contours are from a detection. The numbers inside the contours should match those from the bin index list outputted by the algorithm

7.4.4 Step 3.5: Point-extraction from full grid

To go from bins to points in physical space, all points are again put into bins, this means they are assigned a bin value based on their location. instead of the points from the cube used to make a histogram, all points in the entire velocity domain are considered. The reason for this is that that way, structures are not bound to the size of the cubes and are allowed to cover the entire domain. If a bin index of a point matches the indices of the previously detected bins in the contour, that point is added to the points belonging to the structure associated with the contour. This procedure is done for all structures. The point search algorithm is tested by inputting the edge points of (-180,-90),(-180,90), (180,90) and (180,-90) alongside (0,0) and then seeing if the algorithm output consists of bins 0,612, 647, 35 and 341, as these are the respective edge bins from Figure 7.7

7.4.5 Step 3.6: Removing duplicates and merging structures

To avoid duplicate contours, which is finding the same contour in both original and shifted images, some mitigation is implemented. The smaller of two structures sharing more than 60% of their bins will be removed. Alternatively, the structures could also be merged, however, as structures of different cubes are already merged, they are removed to keep the size of the structures somewhat limited which is computationally efficient.

Instead of linking a cube's histogram to that cube's point in physical space, the histogram is linked to the points of the entire domain in physical space, as mentioned before. The downside of this approach is that the same structure can be detected in multiple cubes. To prevent having these duplicates, two structures are merged if they have at least 85% point overlap. If a structure would sit on a boundary of two cubes it would be detected twice and then those two separate structures would be merged into one. Once all the structures alongside their respective points are found some statistics are computed in Python and afterwards the data is sent to MATLAB for further post-processing.

7.5 Step 4: General statistics

To get a sense of how influential the structures are compared to the rest of the computational domain, the relative kinetic energy and volume fraction are computed. The relative kinetic energy, denoted by Equation 7.5 is the energy inside the structure $E_{k_{\text{structure}}}$ divided by the total kinetic energy in the domain $E_{k_{\text{domain}}}$. Kinetic energy is computed with Equation 7.6 N (total number of points) and i (a single point) are either for a structure or for the entire domain. Equation 7.7 is the volume fraction which is the number of points captured by the structure $N_{\text{structure}}$ divided by the total number of points inside the domain N_{domain}

$$\text{Relative Kinetic Energy} = \frac{E_{k_{\text{structure}}}}{E_{k_{\text{domain}}}} \times 100\% \tag{7.5}$$

$$E_k = \sum_{i}^{N} \frac{1}{2} \left(u_i^2 + v_i^2 + w_i^2 \right) \tag{7.6}$$

$$\text{Volume Fraction} = \frac{N_{\text{structure}}}{N_{domain}} \times 100\% \tag{7.7}$$

7.6 Step 5: Isosurface conversion

While the acquired point cloud from Step 3 should give a good estimate of the detected structure, it does not produce a continuous structure. Step 5 creates these continuous structures from the cloud. Figure 7.8 is a graphic representation.

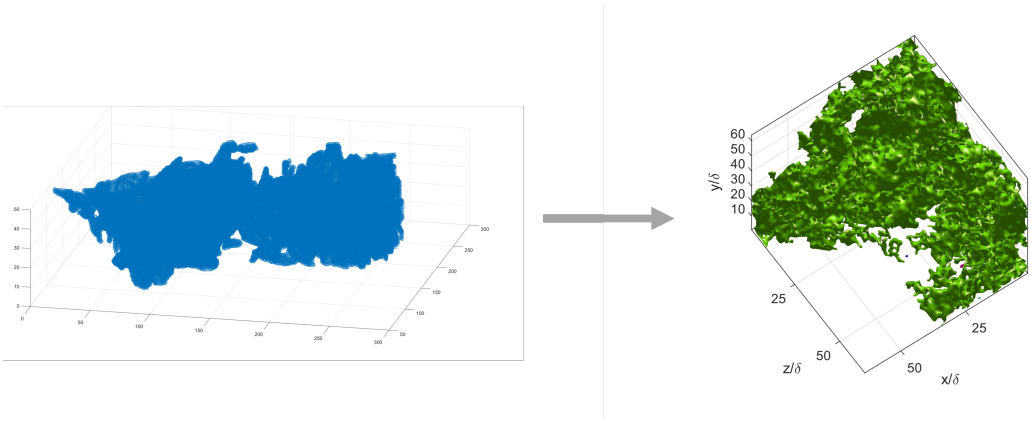


Figure 7.8: Step 5: Isosurface conversion. This illustration shows how the set of points is converted to continuous structures.

7.6.1 Step 5.1: Downsampling, Step 5.2: Generate isosurfaces and Step 5.3: Group structures

Once the points from each of the structures are converted to MATLAB, they are turned into an isosurface. Note that the points will typically not form one continuous structure, rather, they are several and with some noise. Therefore, continuous features, vertices and faces, are extracted from the isosurface. The default downsampling reduces the dataset to a 64^3 grid however, the integral length scale should still be represented by at least 10 gridpoints, hence this may vary.

7.6.2 Step 5.4: Filter out small structures and Step 5.5: Convert faces and vertices to alpha-shape

The smaller structures are filtered out with a filter threshold requiring structures to contain at least 100 points and only the largest ones are taken for further inspection. The sizes of the continuous structures will typically decrease exponentially and therefore only the largest 20% is kept. At this stage the structures can be visualised and compared with those from

the literature, one would expect them to have roughly the same shape. While the vertices and faces representation is useful to isolate continuous structures, they are not accepted by some MATLAB functions such as the 3d party imMinkowski by (Legland, 2023). The structures are therefore first converted to an α-shape which is a thinner version of a convex hull that is particularly suitable for complex and abstract structures. The alpha shape is subsequently binarised to be used for further computations. Note that this alpha shape is not used for PCA which takes the faces and vertices directly.

7.7 Step 6: Length scales

To acquire length scales to get an idea of the size of the detected structures, two methods are used: Minwkowski functionals and shapefinders Equation 7.8 and PCA (Pearson, 1901).

7.7.1 Step 6.2: Minkowski functionals and shapefinders

As a more universal and systematic approach to visual inspection, the length scales of the structures are computed using the method from (Leung et al., 2012). These include Minkowski functionals from Equation 7.8 which are volume V_0, area V_1, mean breadth V_2 and Euler characteristic V_3. If the Euler characteristic is negative, which is the case for structures with many holes, one can take G where $G = 1 - E$. The Minkowski functionals can then be used to compute shapefinders from Equation 7.9 where T is thickness, W is width and L is length, for convex structures, $T \leq W \leq L$. From the shapefinders, simple shapes can be distinguished by acquiring the planarity and filamentarity from Equation 7.10 and plotting them as in Figure 7.9.

$$V_0 = V, \quad V_1 = \frac{S}{6}, \quad V_2 = \frac{1}{3\pi}\int_S \frac{\kappa_1 + \kappa_2}{2}\,\mathrm{d}S, \quad V_3 = \frac{1}{2\pi}\int_S (\kappa_1\kappa_2)\,\mathrm{d}S. \tag{7.8}$$

$$\mathscr{T} = \frac{V_0}{2V_1}, \quad \mathscr{W} = \frac{2V_1}{\pi V_2}, \quad \mathscr{L} = \frac{3V_2}{2V_3} \tag{7.9}$$

$$\mathscr{P} = \frac{\mathscr{W} - \mathscr{T}}{\mathscr{W} + \mathscr{T}}, \quad \mathscr{F} = \frac{\mathscr{L} - \mathscr{W}}{\mathscr{L} + \mathscr{W}} \tag{7.10}$$

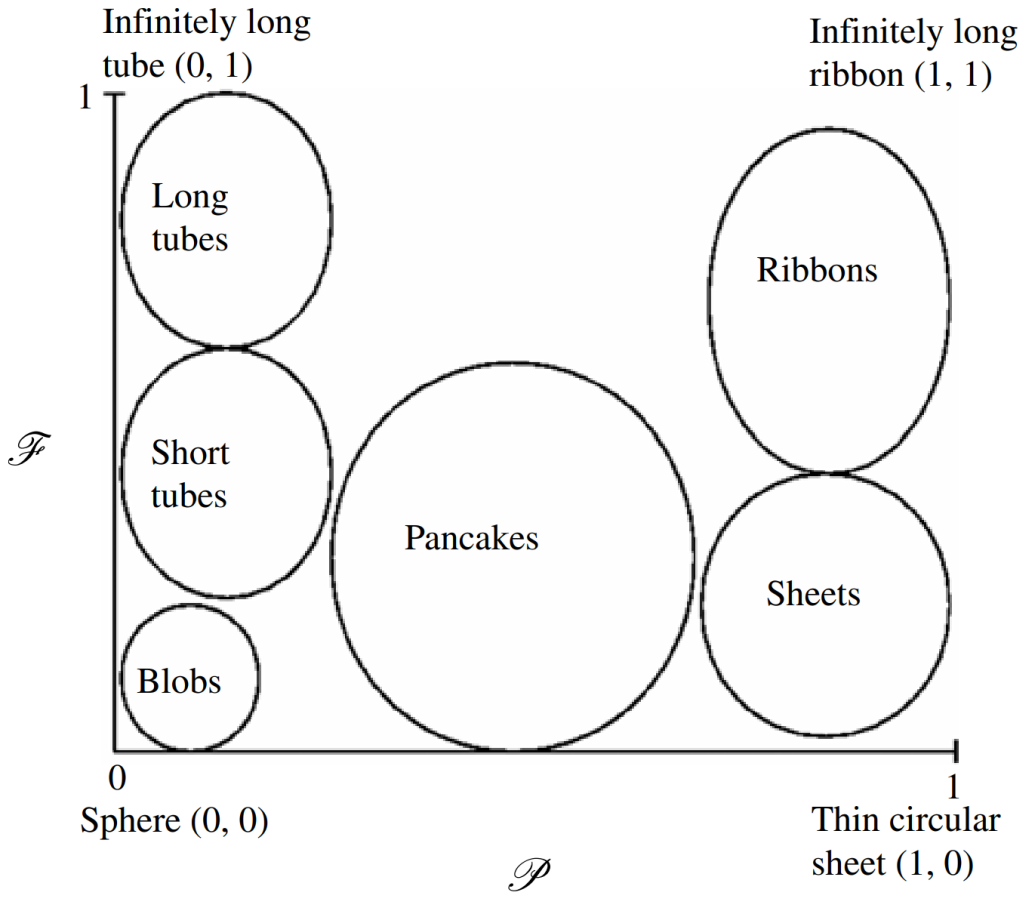


Figure 7.9: Regions of simple shapes (Leung et al., 2012)

To implement these functions into MATLAB for large-scale structures, the third-party package imMinkowski by (Legland, 2023) is used. Table 7.2 shows the outputs and overall, they are as expected. The sphere's volume and area are close to what could be expected from their exact calculations. The mean breadth is twice the radius and the Euler characteristic of 2 is correct for a volume without holes. Note that all three lengths are equal to the radius and that both P and F are roughly zero. For the cylinder, a similar conclusion can be drawn with regard to volume and area. The lengths do not match the radius and height, however, the planarity and filamentarity do correspond with a long tube in Figure 7.9. This is as expected as the lengths are roughly within the same order of magnitude as the exact and should only be used as an estimate. All in all, the MATLAB package does what is expected and is therefore implemented.

	Sphere	Cylinder
R	100	10
h	-	100
V_0 exact	$\frac{4}{3}\pi(100^3)$	$\pi(10^2) \times 100$
V_1 exact	$4\pi(100^2)$	$2\pi \times 10 \times 100 + 2\pi(10^2)$
V_0	4.19×10^6	7.90×10^4
V_1	2.09×10^4	5.31×10^3
V_2	134	314
V_3	2	73
T	100.01	7.44
W	99.76	10.8
L	100.2	235
P	-0.0012	0.180
F	0.002	0.91

Table 7.2: Properties of both a sphere and cylinder, exactly as determined by the Minkowski functionals and shapefinders. See how V0 and V0 match the exact values, how T, W and L are all equal to R for the sphere and how P and F match in a thin tube from Figure 7.9.

7.7.2 Step 6.1: Length Scales from Principal Component Analysis

A limitation of the Minkowski functionals and shape finders is that they become inaccurate for structures with many holes. As not much is known about turbulent structures, it is also not known if they contain a lot of holes. Therefore, an alternative method, principal component analysis (PCA) (Pearson, 1901) is also used. In this method, the eigenvectors of the covariance matrix are computed and sorted based on the largest eigenvalues. The first eigenvector will indicate the direction of the largest variance in the dataset, the second eigenvector will be perpendicular to this one and have the second highest variance and so on. Taking the first three eigenvectors $PC1$, $PC2$ and $PC3$ then results in a 3D coordinate

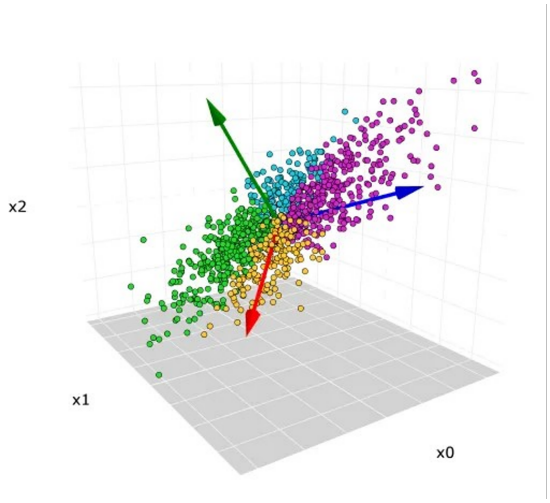


Figure 7.10: Illustration of a PCA coordinate system (Cheng, 2022). See how instead of a fixed x,y,z the new coordinate system matches data in terms of perpendicular directions with the highest variance.

system. An example of such a coordinate system can be seen in Figure 7.10

In the context of this thesis, PCA is applied to each detected structure using the MATLAB built-in function from the Machine Learning Toolbox. Unlike imMinkowski this function is able to directly take the structure vertices as input. Once a principal coordinate system is created, the distances from the two furthest points in each direction alongside the standard deviations (thus 3 distances and 3 standard deviations) are computed. The isosurface generation and subsequent feature extraction remove noise points, therefore preventing outliers from contaminating this length-scale computation. As the data originated from a HIT turbulence field, there is no preferential direction and the data does not need to be normalised to prevent a bias in the PCA analysis.

7.8 Datasets

This section briefly goes over the three datasets that are used throughout the thesis. the Low Re da

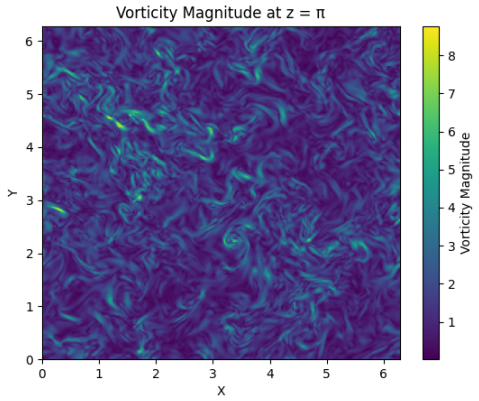


Figure 7.11: Vorticity plot from the low Re dataset provided by Dr A. A. Wray (CTR 2002, private communication)

7.8.1 Low Re

To develop the detection software and perform debugging, the algorithm is first tested on a low Reynolds number (see snapshot in Figure 7.11), $Re_\lambda = 170$ dataset provided by Dr A. A. Wray (CTR 2002, private communication) which is relatively small and therefore computationally less expensive. The 256^3 dataset has a Reynolds number of $Re_\lambda = 170$ and was also used by (Elsinga and Marusic, 2010). Regarding length scales, the Kolmogorov length scale is 0.3978, the Taylor scale of 10.2 and the integral length scale of 200 grid spacing, all measured in grid spacing. It would be expected that the large-scale structures have dimensions of roughly 200 and would at the very least be significantly larger than the Taylor scale of 10.2. Note that the boundaries are periodic, which means that structures could end on one end and then continue on the opposite end. The downsampled structures in MATLAB are computed on a 64^3 grid which means that the integral length scale L covers $200/(256/32) = 50$ gridpoints.

7.8.2 High Re subvolume

The high Reynolds number subvolume was provided by Ishihara from(Ishihara et al., 2007) and has a Reynolds number of $Re_\lambda = 1311$ (see example in Figure 7.12. It is a 512^3 cube which is part of the full domain 4096^3. This means that the boundaries of this smaller cube are not periodic. The integral length scale is found to be $L = 1.09$ while the full domain attains 0 to 2π in each direction. The integral length corresponds to $1.09/(2\pi/4096) = 710$ gridpoints. As the integral length scale is far larger than the dimensions of the cube, it is expected that only portions of large-scale structures can be detected. The Taylor

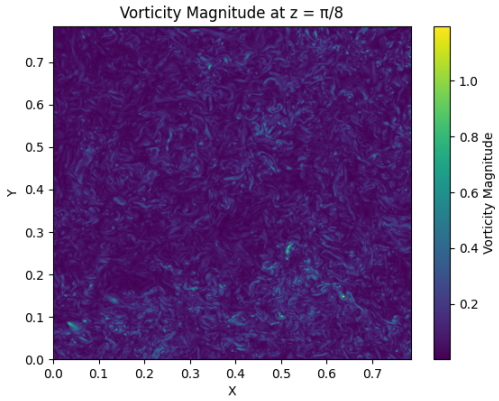


Figure 7.12: Vorticity plot from the high Re subvolume dataset (Ishihara et al., 2007)

length scale is $\lambda = 0.0339$, corresponding to 44 gridpoints. Finally, the Kolmogorov scale is $\eta = 0.00051$, (corresponding to 0.66 gridpoints). The downsampled structures in MATLAB are computed on a 64^3 grid which means that the integral length scale of L corresponds to $710/(512/32) = 88$ gridpoints on the downsampled grid.

7.8.3 High Re downsampled full dataset

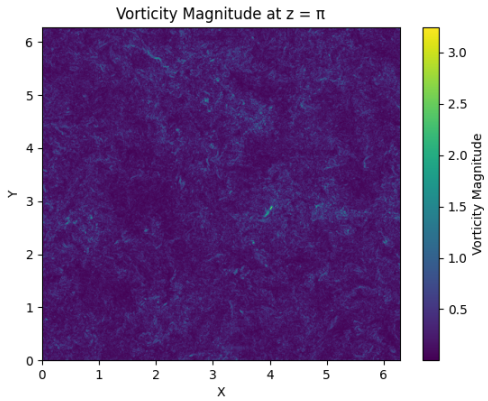


Figure 7.13: Vorticity plot from the high Re full volume dataset (Ishihara et al., 2007)

A limitation of taking only one cube from Ishihara is that there is a chance the cube does not or barely contains any large-scale structures. To overcome this, the whole 4096^3 volume is down-sampled to a 512^3 cube, meaning a factor 8 downsampling (see example image in Figure 7.13). The integral length scale is then covered by 89 gridpoints while the Taylor

length scale is covered by 3 datapoints. For the MATLAB 64^3 grid, the integral length scale consists $L = 12$ gridpoints.

7.9 YOLOv7

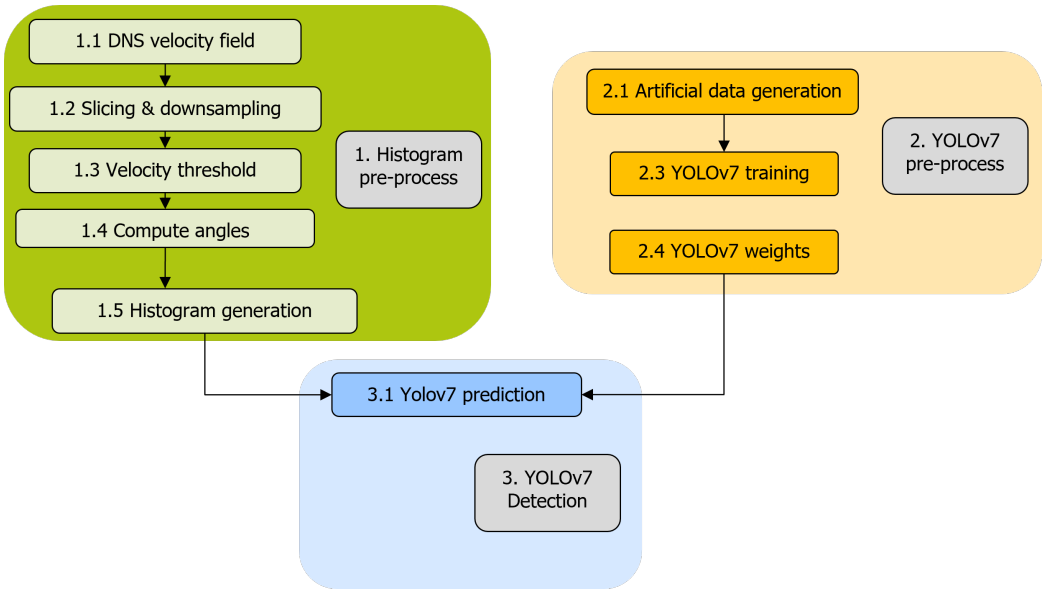
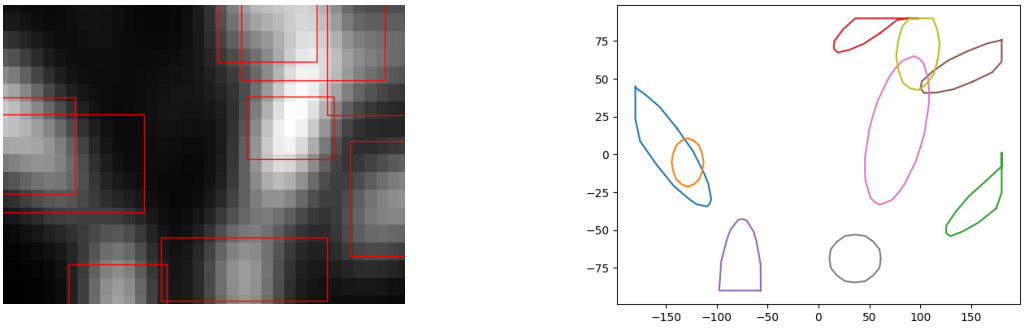


Figure 7.14: Flow diagram of YOLOv7 implementation

Figure 7.14 shows the entire methodology that has been developed for the YOLOv7 implementation. Note that the post-processing pipeline is significantly shorter than for U-net. The reason for this will be explained in the next chapter. Like U-net, YOLOv7 is also trained on the artificial data and then shown a real histogram. The labels are, however, changed as YOLOv7 either takes polygons or bounding boxes. Instead of a binarised image. These polygons are also artificially generated circumferencing the peak from a certain distance. The bounding boxes are also drawn depending on the size and location of the peak. Subfigure 7.15(a) shows the artificial data as an image with bounding boxes and Subfigure 7.15(b) has the bounding boxes replaced with the polygons at the same locations. Note that, unlike U-net, YOLOv7 is able to distinguish different structures by itself and does not need contours. For the training, YOLOv7 (Wang et al., 2022) is trained using transfer learning, which is far more efficient than training from scratch. In total 2,000 bounding box labelled and 5,000 polygon-labelled images are generated. 55 epochs on the GPU are selected with a batch size of 16.



(a) (b)

Figure 7.15: YOLOv7 labels, note how the two produce labels at the same locations corresponding to the peaks in the image. (a) Bounding boxes (b) Polyons

7.10 DBSCAN

The DBSCAN methodology, as shown in Figure 7.16 is radically different from the two neural networks. To get a better understanding of the flow field, simulations are done on the entire grid as well as a slice out of five cuts from the z-axis, the same as the one by (Elsinga and Marusic, 2010). For downsampling, factors 1 and 4 are used. After applying the velocity threshold, computing the velocity direction with magnitude and the spatial coordinates, the data is normalised, so that the range is between 0 and 1. To give a higher weighting to the spatial coordinates, they are multiplied by a factor of 10 so that their data range is between 0 and 10. This penalises points for being further apart so that the detected structures are more spatially coherent.

For DBSCAN two key hyperparameters are of interest, the minimum number of points $minPTS$ and the search radius around a point ϵ. For the minimum number of points, twice the number of dimensions is used. To get a proper estimate for ϵ, the nearest neighbours and their respective distances are first computed using the scikit-learn (Pedregosa et al., 2011) package and plotted as an x-y graph. From this graph, the needle is computed using the Kneed package (Arvai, 2021), and the obtained value serves as the initial ϵ radius. The 6 features, 2 velocity angles, 1 velocity magnitude and 3 spatial coordinates for all the points are then implemented into DBSCAN using the aforementioned hyperparameters. Once more the scikit-learn (Pedregosa et al., 2011) package is used, now to directly implement DBSCAN. To fine-tune the algorithm, the needle sensitivity S is altered, this allows to shift of the needle upwards or downwards on the graph. An optimal setting can be chosen based on trial & error and visual inspection.

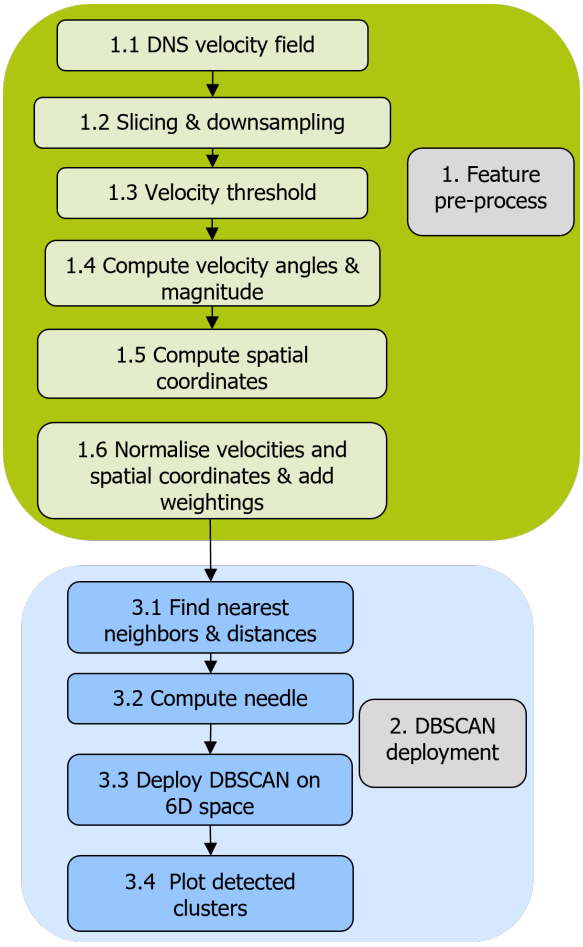


Figure 7.16: Flow diagram of DBSCAN implementation

Two downsides of this DBSCAN implementation are that there is no way to account for the periodicity of the velocity direction and scikit-learn only runs on CPU which is less efficient than GPU. For the former one can do a new simulation with the velocity shifted and then analyse if certain clusters have now become larger. For the CPU limitation, downsampling can be used to reduce the computational demands.

Chapter 8

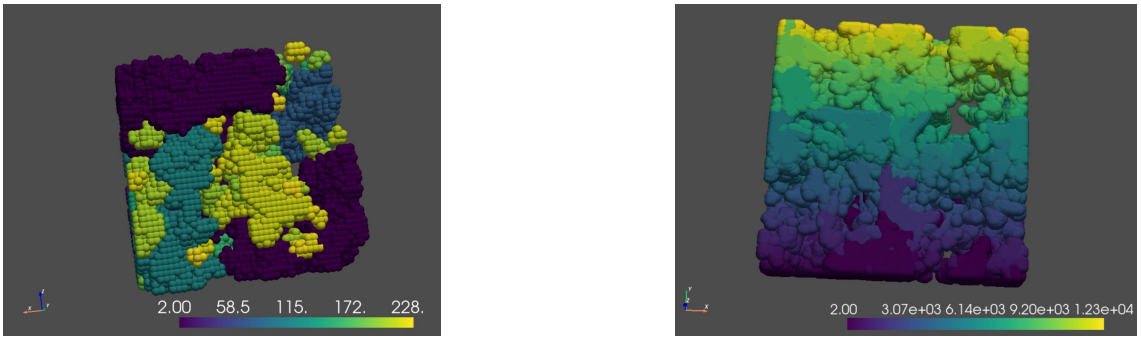
Results

This chapter presents the results of deploying the algorithms on the low Reynolds number DNS simulation, the high Reynolds number subvolume and the high Reynolds number downsampled In the early stages, it became clear only U-net would be feasible for the detection. Nevertheless, the results from DBSCAN and Yolov7 are still presented and discussed in order to show why they were unfeasible. For U-net, some initial parameter tweaking and validation were done on the low Reynolds number dataset. Other than that, all three dataset experiments with U-net contain detections and structures (which can be seen in the appendices) and general statistics, including kinetic energy, volume fraction and length scales. The effects of downsampling on structure dimensions were analysed for each dataset separately.

8.1 Low Re DBSCAN

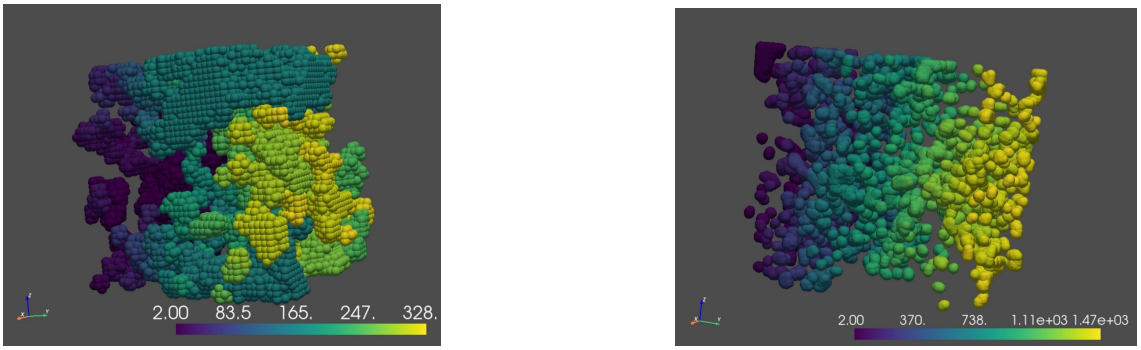
Figure 8.1 shows the results of deploying DBSCAN on the low Re dataset where Subfigure 8.1(a) is coarsened by a factor 4 to reduce computation times. Needle sensitivity S was tuned to produce realistic-looking structures which are somewhat similar to those in Subfigure 3.2(b) found by (Elsinga and Marusic, 2010). Using the same settings on the unfiltered grid as in Subfigure 8.1(b) shows completely different structures.

The same procedure of first tuning S on a filtered grid and then applying the same settings on the unfiltered is also done in Figure 8.2. Again, the structures in the two figures look completely different.



(a) (b)

Figure 8.1: DBSCAN output on a slice. (a) 4x coarsening. (b) Uncoarsened.



(a) (b)

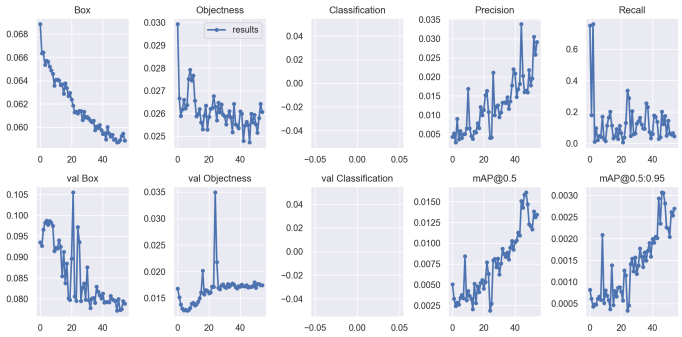
Figure 8.2: DBSCAN output on the full domain. (a) 4x coarsening. (b) Uncoarsened.

From these findings, it can be concluded that while DBSCAN is able to show structures, they are in most cases not the right structures as they look very different compared to those found by Elsinga and Marusic (2010). Finding the right structures requires significant knowledge of them a priori so that from multiple runs, the best settings can be selected. The hyperparameters are so sensible that give completely different outputs for the same dataset when filtering or slicing sensitive. As there is currently very limited knowledge of large-scale structures, the DBSCAN algorithm was not continued for the remainder of the thesis.

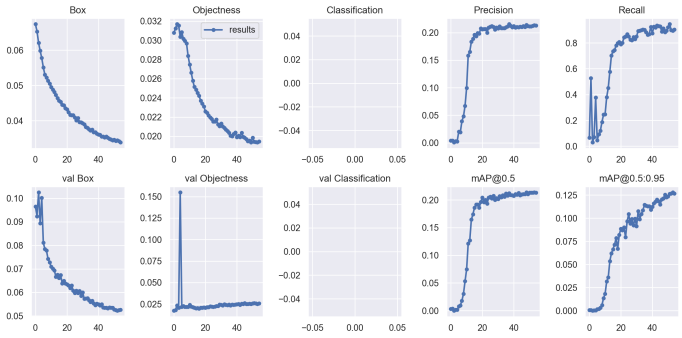
8.2 Low Re YOLOv7

Arguably even more underwhelming are the YOLOv7 results shown in Figure 8.3. Transfer learning training YOLOv7 for 55 epochs returns a MAP of 1.5%, as seen for the bounding box (Figure 8.3). The polygons do a lot better. However, the highest achieved MAP is still only 22%. Note that while both runs have not fully converged yet (see how val Box is still decreasing in both cases), an argument could be made that they are nearing conversion and will not get significantly better. Running the 55 epochs on datasets of 3000-5000 images with multiple objects in each figure already takes several hours, even on the GPU. Subfigure 8.3(c) shows a detection on a histogram derived from the low Re dataset. The detection highlights a location where there are clearly no peaks present and the confidence of the prediction is also low at 12% (one would hope for values of $70 - 90\%$). This suggests that YOLOv7 with these weights is very far from being a reliable detection system.

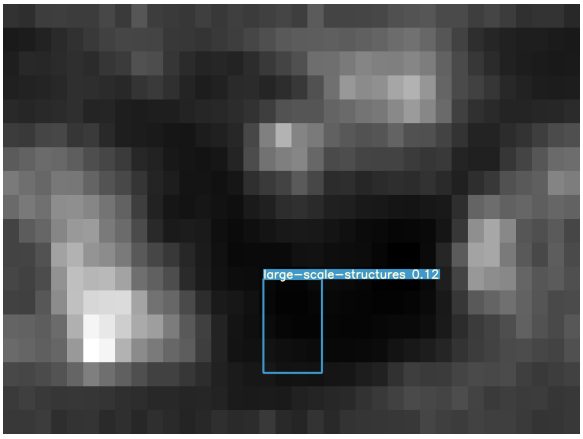
The low performance of both the bounding box and polygon datasets could be caused by the fact that the histograms are very different from the objects YOLOv7 was trained, for real-life objects. YOLOv7 is probably optimised to capture information at way finer levels than are present for the histogram. It is also worth mentioning that YOLOv7 has three input channels as it is trained for coloured images while the histograms only require 1 channel. Furthermore, YOLOv7 only accepts images such as .png or.jpg files while in reality, each histogram is only 36x18 datapoints. These datapoints could easily be saved with multiple histograms as arrays in a single file. Since trying to tweak the YOLOv7 is highly complex and training times are very large, efforts were diverted to U-net.



(a)



(b)



(c)

Figure 8.3: YOLOv7 outputs. (a) Bounding box. (b) Polygon. (c) Detection using weights from polygon dataset training. These graphs are taken directly from YOLOv7 and hence there is a lot of additional information. The main statistics useful for this thesis are val Box which shows the networks' convergence and mAP@0.5 which is a measure of accuracy.

8.3 Low Re U-net

This section presents all the results of the U-net detection on the low Re dataset. This was the original dataset the entire U-net method was developed from, hence some additional results are presented. These extra results include a subsection on training times, a subsection on the performance of the different labels from Figure 7.4, a subsection on the effect of the contour selection, a direct validation with literature and a subsection on hole analysis. A subsection on downsampling and the general statistics of the detected structures are presented for all three datasets. Furthermore, detections and subsequent structures of the low Re case can be found in Appendix A

8.3.1 Training times

Before the network can be deployed on the histograms, it has to be trained. The total training sees the network converge and the total training time is roughly 5 minutes per run. Therefore, the presumption that training times would be low is justified and no further effort is spent optimising training speed. Creating cubes and subsequent histograms, performing the detection and finding the points associated with each structure were all done in one code on the CPU and took roughly 10 minutes when using 8 cubes and a coarsening factor 4. Merging similar structures took roughly another 10 minutes, creating the isosurface and isolating the continuous structures took around 5 minutes for a downsampling factor of 8. Lowering the downsampling or coarsening factor drastically increases the MATLAB computation time exponentially. Computation times for acquiring Minkowski functionals, shapefinders and PCA lengths were all negligible.

To validate with literature, another identification was also done using five slices cut in the z-direction instead of 8 cubes. For these, no downsampling or coarsening was used and contours were directly related to points inside the slice, not the entire domain. Going from velocity field to large-scale structures in terms of points took 1-2 minutes. However, the MATLAB code without downsampling took roughly 10-12 hours to isolate all continuous surfaces.

8.3.2 U-net effect of different label size

Figure 8.4 shows how the predictions stack up versus the real image. One can argue that from the three predictions, Subfigure 8.4(b) matches the original image in Subfigure 8.4(a) most closely. Furthermore, it is preferred for the detection to capture all of the bins associated with a few additional empty ones than to miss out. The reason for this is that the

empty bins will contain only a few points and these can easily be removed in later stages. Therefore, the 70% of the maximum label setting is kept for all subsequent detections.

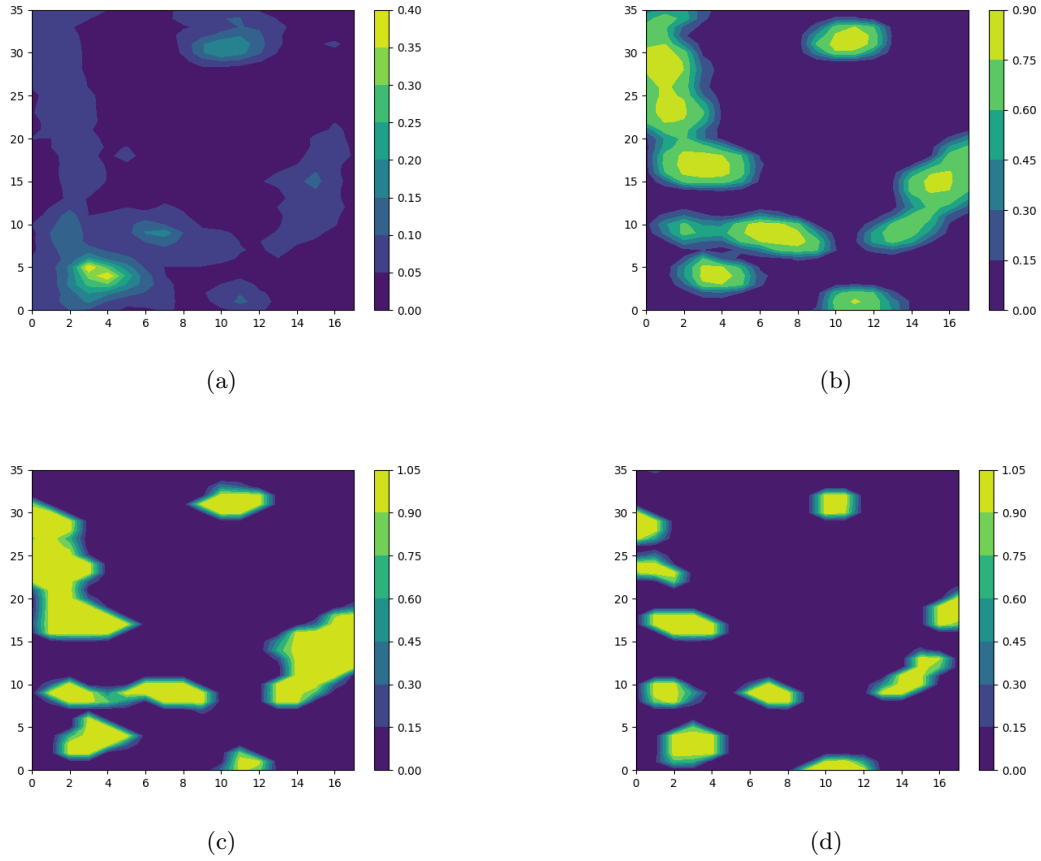
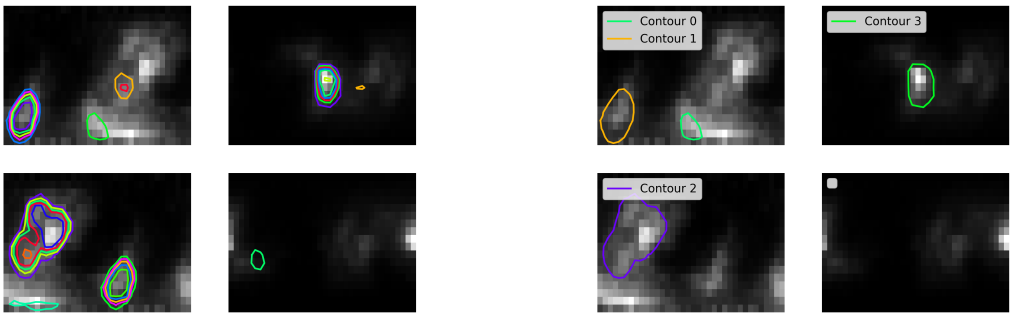


Figure 8.4: (a) Artificial image, (b) U-net prediction with 70% of peak maximum label, (c) U-net prediction with 75% of peak maximum label and (d) U-net prediction with 80% of peak maximum label. 70% of maximum, for example, means values of 70% of the maximum and higher are included in the label

8.3.3 U-net histogram predictions

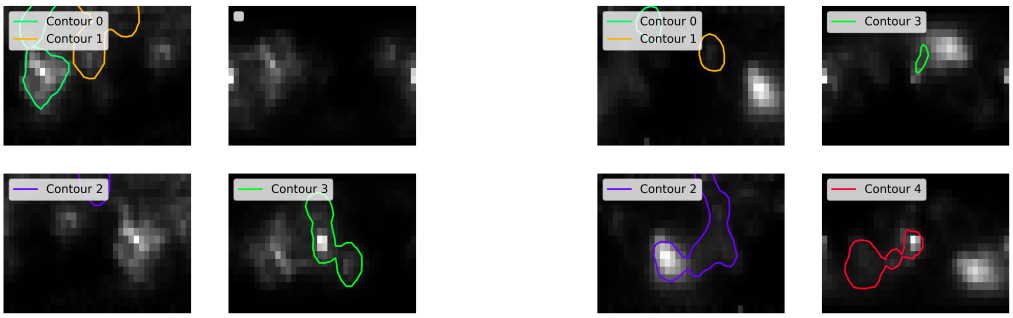
Figure 8.5 shows the before and after difference of selecting contours to find points in. Note that in Subfigure 8.5(a) contours that touch the edge of the image are removed. From the result in Subfigure 8.5(b), it can be observed that all of the significant peaks are captured to the near full extent and that no peak is captured twice. The contour-selecting



(a) (b)

Figure 8.5: Converting a binary image into several structure contours. (a) Direct network output (b) Contour selection. The end result is that per four images there is one contour per peak.

algorithm was further inspected with subsequent cubes and some contours touching boundaries were observed, as seen in Figure 8.5. While unexpected, the negative consequences seem minimal. All the peaks are still captured across the four figures without duplicates. Therefore, no further investigation is initiated at this point in time. Another aspect of



(a) (b)

Figure 8.6: Contours touching boundaries (a) Example 1 (b) Example 2

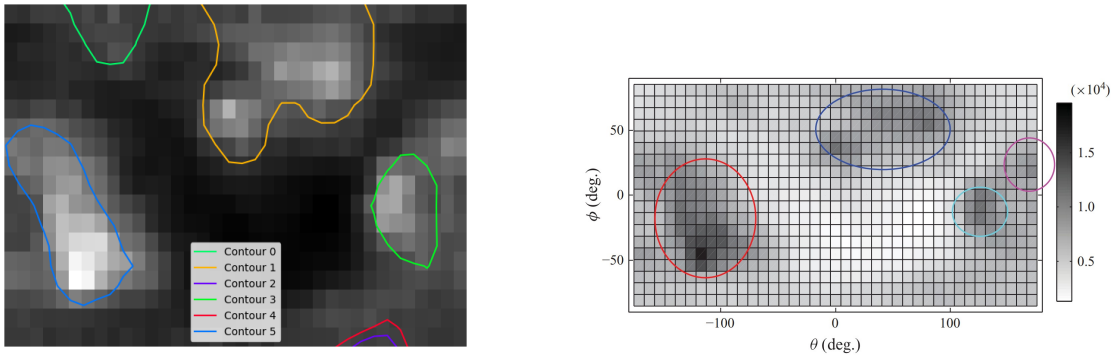
the identification algorithm verified in detail is the effect of downsampling. Without any downsampling, post-processing all the structures in a HIT dataset takes several days on a 12th Gen Intel(R) Core(TM) i9-12900H CPU utilising 10% of the processor. One could modify the settings and code to use more CPU, a GPU or a cluster, however, this would also increase the computational speed, not the cost. To reduce the cost and still get an

initial estimate of the structures, downsampling is performed. Figure 8.7 shows the effect of downsampling with a factor of 2 and 4 in each direction. From visual inspection, one could argue the structures are roughly the same in terms of occupied volume and length scales. The factor 4 structure seems slightly larger and does not have the smaller greenish structure present in Subfigure 8.7(a). Overall though, proceeding with a downsampling factor of 4 seems reasonable, especially considering that the integral length scale is represented by 50 gridpoints in each direction.

Figure 8.7: Effect of downsampling (a) Factor 2, $L = 100$ gridpoints (b) Factor 4, $L = 50$ gridpoints (c) Factor 2 3D (d) Factor 4 3D

8.3.4 Validation with Literature

Comparing Subfigure 8.8(a) with Subfigure 8.8(b) by (Elsinga and Marusic, 2010) which is the same slice shows that the orientations of the structures match. Structure 5 resembles the same peak as the red one in Subfigure 8.8(b), both on the left. Structure 1 resembles the blue one, both at the top centre. Finally, structure 3 resembles the turquoise structures.



(a) (b)

Figure 8.8: Comparison of (a) U-net slice 1 output after processing and (b) Histogram manually created by Elsinga and Marusic (2010)

The six structures found in Figure 8.8 are visualised as isosurfaces in Figure 8.10 through MATLAB. Again, the results are fairly similar. One could argue the bottom structure from structure 1 (Subfigure 8.10(b)) is the blue structure at the centre in Figure 8.9. The top structure from structure 3 (Subfigure 8.10(d)) could be the top red structure and the left structure in structure 5 (Subfigure 8.10(f)) could be the blue structure at the bottom. Structures from 3 and 4 (Subfigure 8.10(d) and Subfigure 8.10(e), respectively) are even similar to the purple and turquoise structures from Figure 8.9 in that they are relatively small and on the edge of the spatial domain.

One remark to make is that neither result is the absolute truth although their being similar shows that both are reasonably accurate. Hence, it can be concluded that U-net can successfully detect large-scale structures and performs equally if not better than manual labour. Note that in order to match the literature structures as closely as possible, these detected structures were not downsampled. Furthermore, only the unshifted images were used for the detection which allowed for easier visual comparisons. The vertically shifted ones distort the shapes of the peaks.

Some tests were also performed on these non-downsampled structures from Figure 8.8 and Figure 8.10 as shown in Figure 8.11. The rationale is that the length scales can be compared to visual inspection from both Figure 8.10 as well as against those found by (Elsinga and

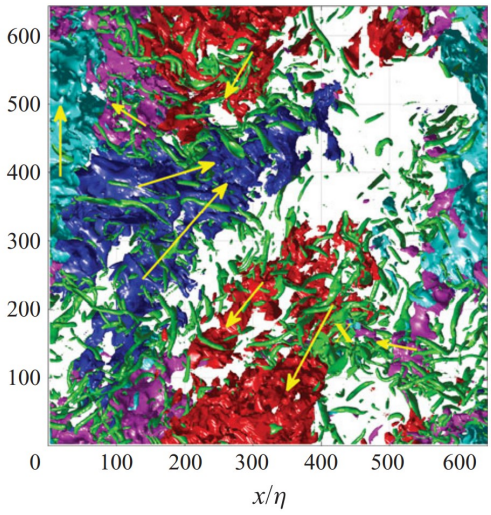


Figure 8.9: Histogram structures manually obtained by Elsinga and Marusic (2010). Note how the colour coding matches Subfigure 8.8(b)

Marusic, 2010). Both length scale methods are non-dimensionalised against the integral length scale L and for Minkowski volume and surface area, a cube volume and area of this scale are taken. Only the statistic of the largest continuous structure of each figure is displayed. The Minkowksi functionals and shapefinders from Subfigure 8.11(a) seem to produce unrealistic length scales. None of the length scales are close to the integral length scale which seems strange. Furthermore, the integral length scale is 80% of the x and y grid length and when looking at Figure 8.10, at least structures 1, 3 and 5 would expected to be of this order. Therefore it is speculated that the structures are not convex and Minkowski functionals and shapefinders are not very accurate in these conditions. The finding was further supported by the fact that the condition $T \leq W \leq L$ does not hold for any of the structures. Another interesting observation is that the higher relative volumes are roughly 0.025 while the higher relative surfaces are roughly 0.25. An order of 10 difference seems quite high and would further suggest the structures having a lot of holes and not being convex.

As an alternative to the Minkowski method, PCA is performed and the results are shown in Subfigure 8.11(b). PC1 is the axis along which the variance of the data is highest, PC2 is orthogonal to PC2 and the second highest. PC3 is orthogonal to both and the third highest. These three form a 3D axis system for each structure and along each dimension, the maximum distance between the most outward point is taken as the length scale. The PCA from Subfigure 8.11(b) in general, seems to produce far more realistic results with 1,3 and 5 all having at least one dimension of the integral length scale. The maximum lengths

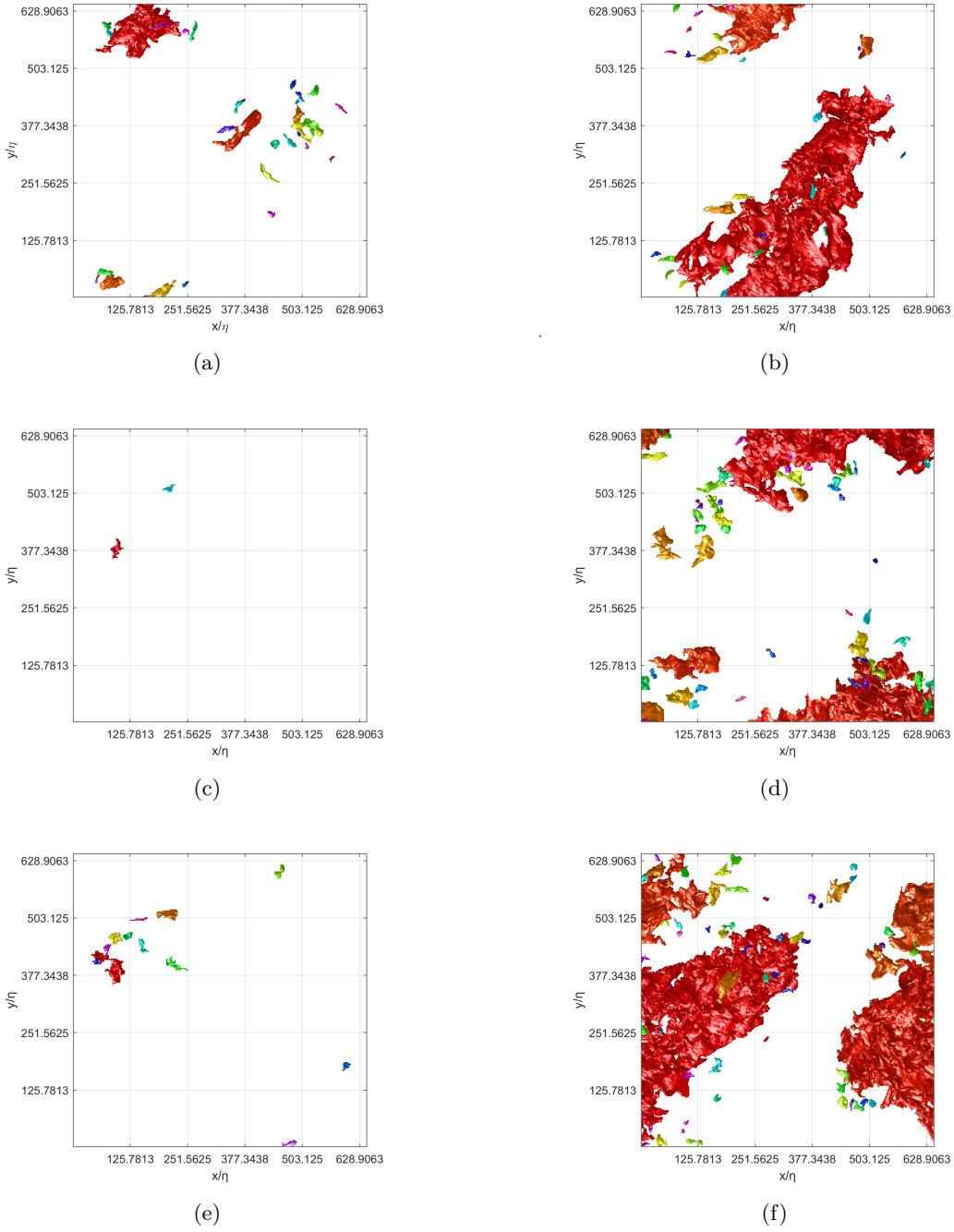


Figure 8.10: Structures from slice 1 (a) Structure 0 (b) Structure 1 (c) Structure 2 (d) Structure 3 (e) Structure 4 (f) Structure 5

are also between 3-5 standard deviations. This suggests that the two furthest points are part of the dataset and not noise points, meaning that the MATLAB code to find continuous structures is working correctly.

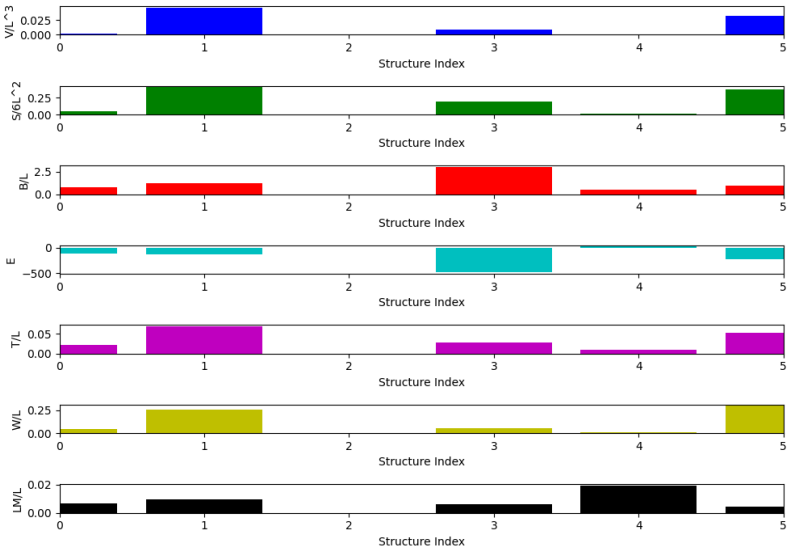
8.3.5 General statistics of detected structures

Both absolute kinetic energy and volume fraction for all of the detected structures are observed in Figure 8.12. The indexing of the structures is kept the same for all figures in subsection 8.3.5. Note that the relative quantities sum up to roughly 80%. These numbers are expected to be slightly inflated as the values were computed directly from the histogram detection, prior to isolating continuous structures of the data. Sub-structures, such as the green one in the left corner of Subfigure 8.7(a) (also note sub-structures in Figure 8.22 and Figure 8.16), are included in the calculation. These sub-structures can be part of other detected structures and hence there could overlap in these statistics. It also makes sense from a physical point of view. The structures have an approximate uniform velocity, meaning that there can be a variance in the direction of the structure. This is also seen in the histograms where the peaks can be quite broad. These peaks being broad means that two different structures in physical space can share the same direction and hence there is overlap. Nevertheless, ignoring the influence of the sub-structures seems a reasonable assumption.

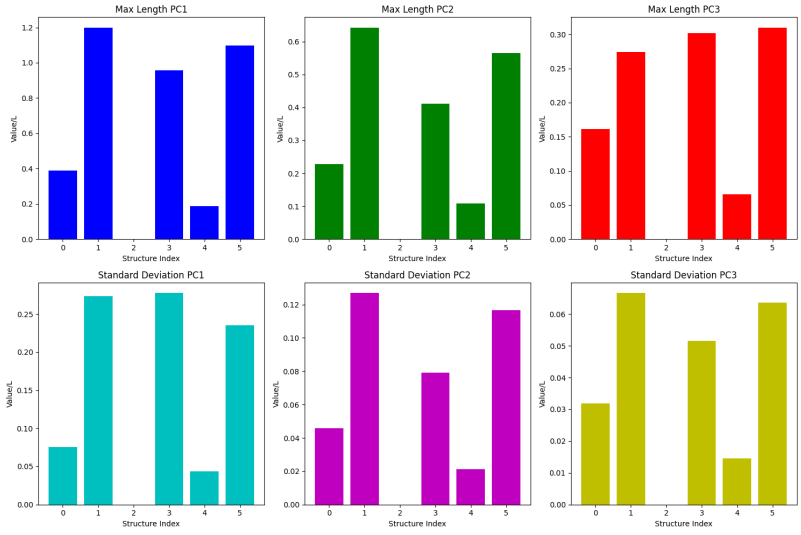
It can be observed that the largest structures have relative kinetic energy and volume fraction above 20% which is significant and seems in line with the Kolmogorov hypothesis. As for the number of detected structures, 9 seems reasonable compared to the 6 from the slice, observed in Figure 8.8. It is expected that large-scale structures of the order of the integral length scale will pass multiple slices as the dimensions of a slice are $1.25L$ by $1.25L$ by $0.5L$. The total volume fraction captured by the 9 structures is 85%.

Minkowski Functionals and shapefinders

From the start, one could expect the Minkowski functionals and shapefinders not to produce accurate results. The downsampling will inevitably create holes as points are removed and therefore the scaling of the surface area, mean breadth and Euler characteristic is non-linear, as mentioned in subsection 8.3.3. Nevertheless, the functionals and shapefinders are plotted in Figure 8.13. One measure that is still worth consideration is the volume. Taking the largest one, structure 5 it can be seen that this has a volume of $0.7L^3$. Taking this as a volume fraction of the entire domain yields 33% which is much larger than that observed in Figure 8.12. Just as with the validation slice Subfigure 8.11(a), the relative surface area is significantly larger than the relative volume although this time by a factor of 4 instead of 10.



(a)



(b)

Figure 8.11: Length scales from Figure 8.10 (a) Minkowski functionals and shapefinders (b) PCA

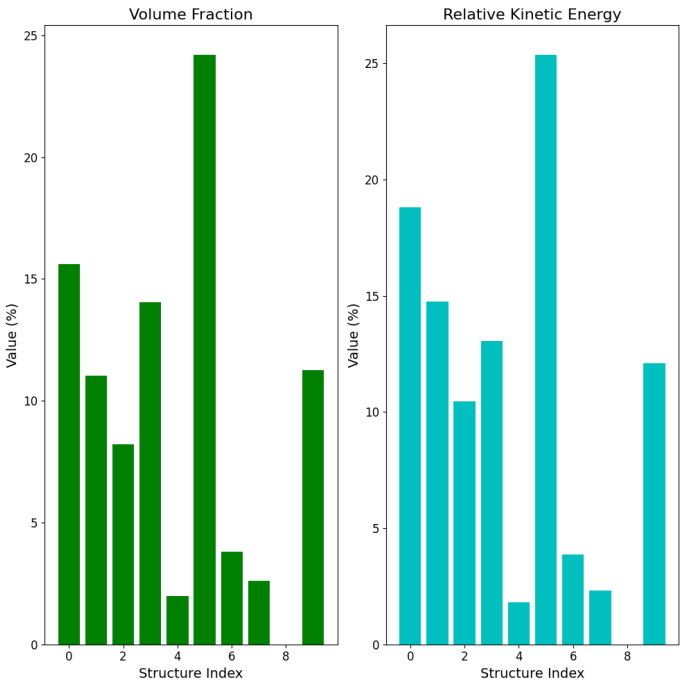


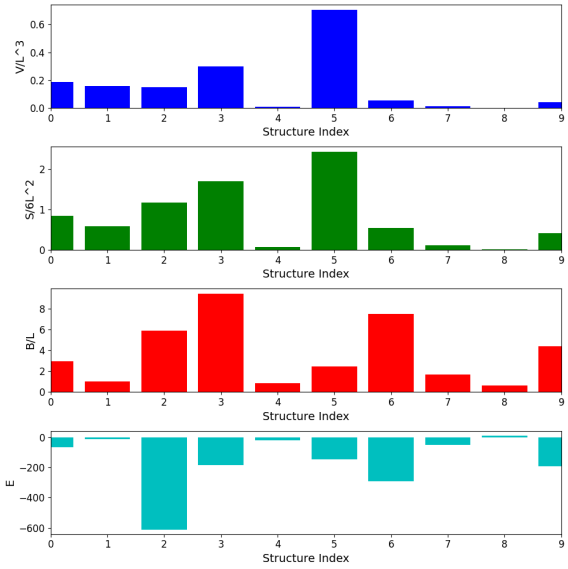
Figure 8.12: General statistics

PCA length scales

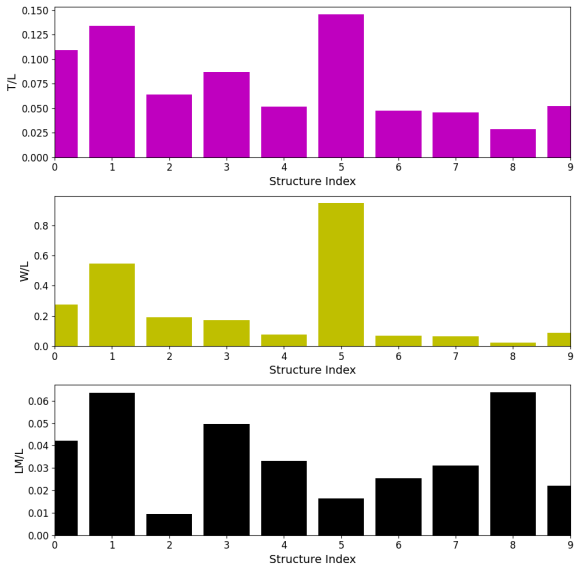
Figure 8.14 shows the results from the PCA. As a sanity check the ratio of maximum distance versus standard deviation for the 10% longest structures is computed. It was found that these distances were between 4 and 6 standard deviations which suggests that the two furthest points are part of the dataset and not noise points. The main observation that can be made from Figure 8.14 is that the distances PC1 of the largest structures are between $1.25L$ and $1.8L$. This is close to the maximum achievable length in the volume being $2.22L$ and of the order of the integral length scale. It can also be observed that most of the structures have all three dimensions larger than L which seems plausible.

8.3.6 Hole analysis

While for capturing length scales Minkowski can be replaced by PCA, the question of why the unfiltered structures from Figure 8.10 have so many holes still stands. To get a better idea, the velocity of one of the holes, the one in Subfigure 8.15(a), is analysed. Subfigure 8.15(b) shows a 2D histogram of the velocity directions of the structure and has an x at the velocity direction of the hole centre. The result is that the points in the hole are



(a)



(b)

Figure 8.13: Minkowski (a) Functionals (b) Shapefinders

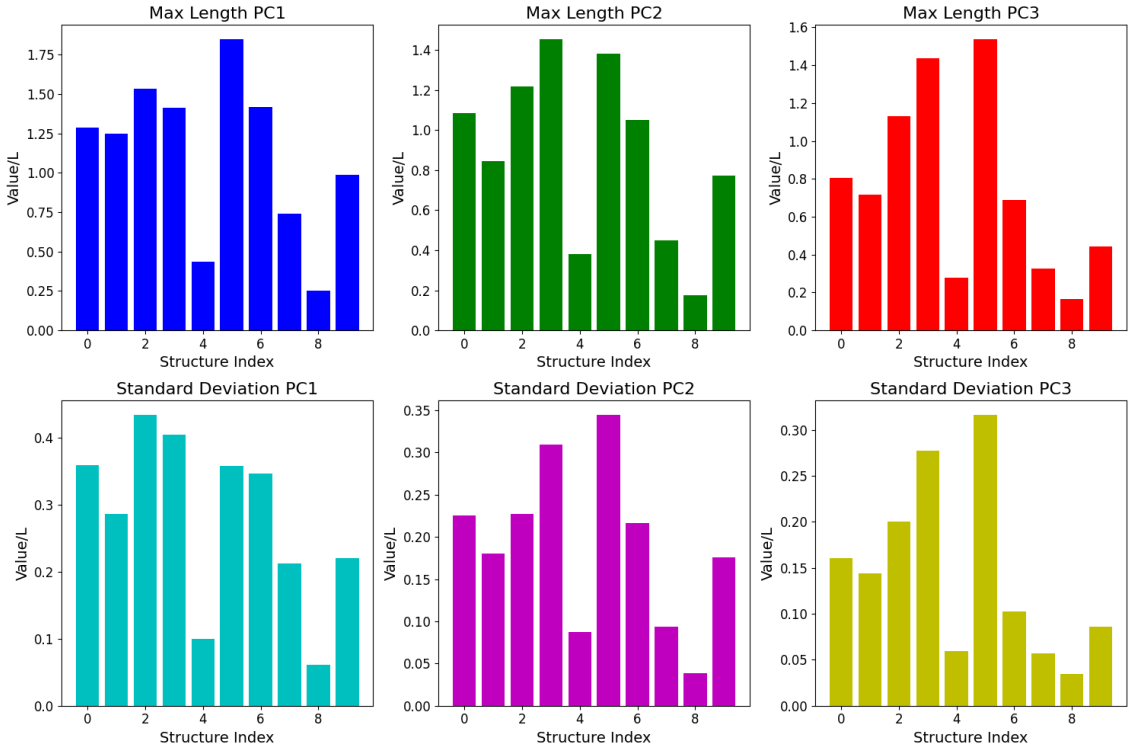
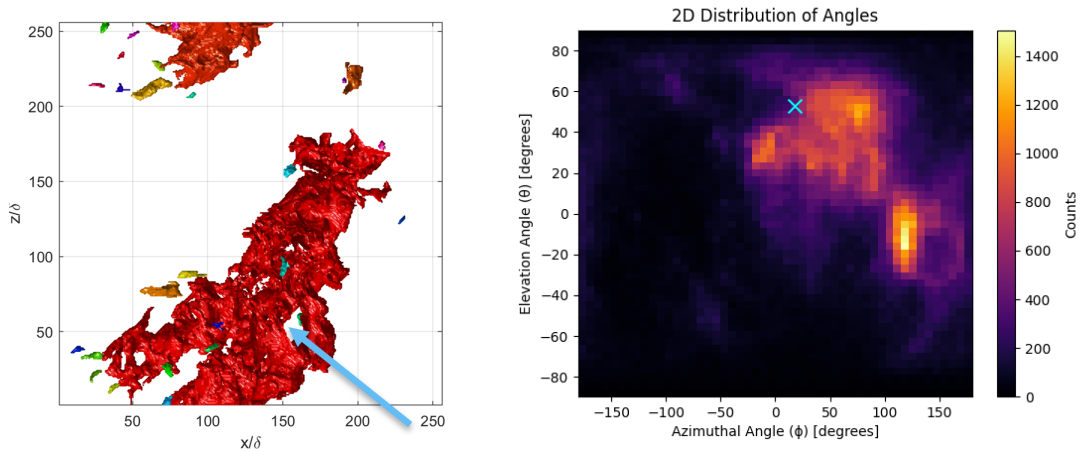


Figure 8.14: PCA length scales

very close to the structure and must not have been far from the predicted contours. The velocity magnitude inside the hole was also checked and very close to that of the structure. Whether this was a mistake of the U-net or an actual topological phenomenon is unclear. In either case, the structure that is detected does look similar to the manually obtained results and is of the order of the integral length scale. Therefore, the lack of knowledge on the holes does not hinder answering the main research questions.



(a) (b)

Figure 8.15: Hole analysis (a) Location of hole in structure (b) Location of hole in histogram

8.4 U-net high Re subvolume

This section shows the main results of the test on the high Reynolds number subvolume. Some detections and structures are also plotted in Appendix B. Comparing these with the structures from the low Reynolds case, one can see that there are far fewer structures per cube and in total for the complete subvolume. This result is expected as the subvolume is smaller than the integral length scale in each direction. The computational cost was not significantly higher than the low Reynolds number case, the Python operations took 5-10 minutes longer (from velocity field to points of a merged structure). Computation times for acquiring Minkowski functionals, shapefinders and PCA lengths were again all negligible. Note that this is a different cube compared to the one used by (Ishihara et al., 2013), shown in Subfigure 3.1(c).

8.4.1 U-net histogram predictions

Similar to the low Reynolds number dataset, the effect of the subsampling is again analysed, as shown in Figure 8.16. Decreasing to a factor of 8 instead of 4 seems acceptable in terms of overall shape and length scales for the blue/purple structure.

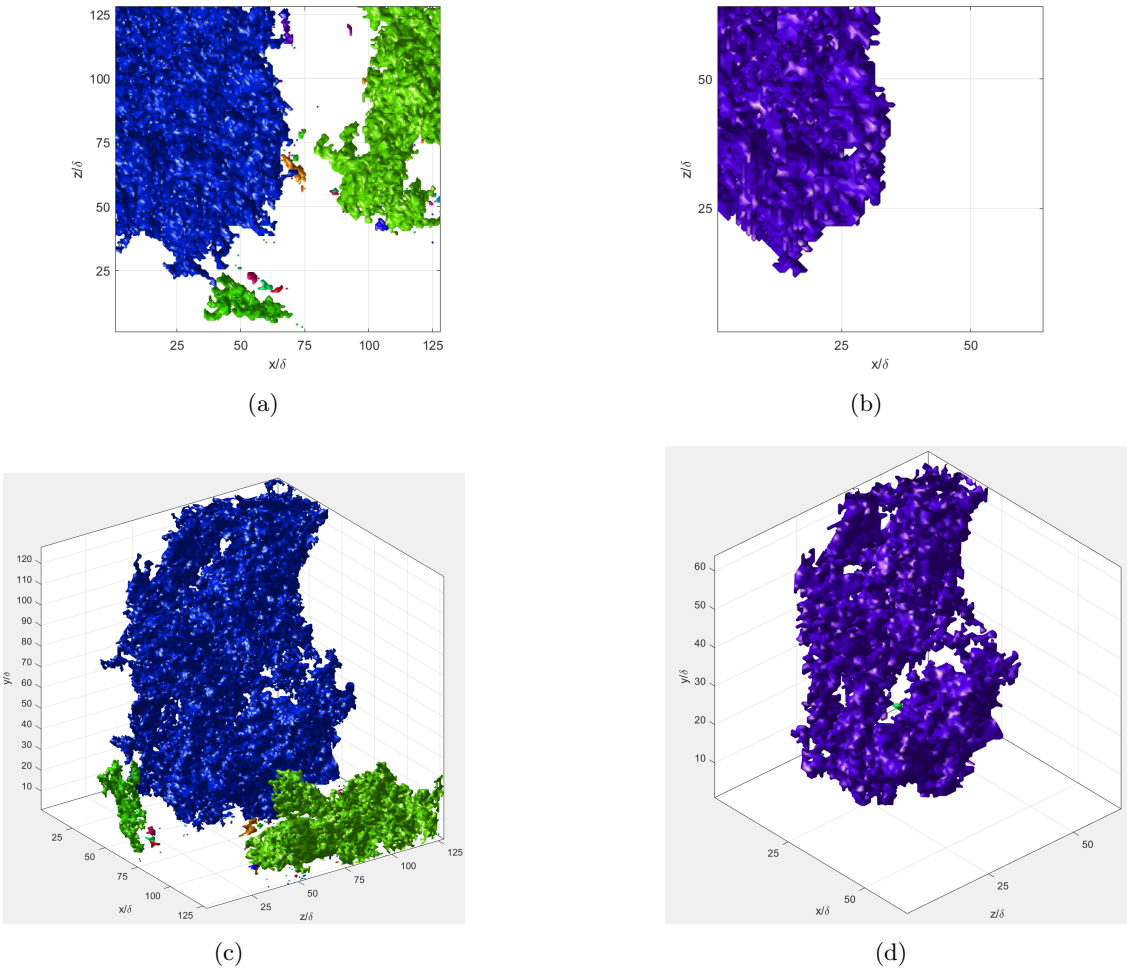


Figure 8.16: Effect of downsampling (a) factor 4, $L = 176$ gridpoints (b) factor 8, $L = 88$ gridpoints] (c) factor 4 3D (d) factor 8 3D. Note that this structure was from an older iteration of the code when merging the structures was not finalised yet, hence it is not present in subsequent statistics. Nevertheless, it shows how the dataset itself is affected by the downsampling.

8.4.2 General statistics of detected structures

Figure 8.17 shows the kinetic energy and volume fraction for the detected large-scale structures. Compared to the low Reynolds number dataset, there are significantly fewer structures and the largest structure contains pretty much all the points and more than half of the energy in the domain. There being only one very big structure makes sense considering that the entire domain is only $0.38L^3$. The higher volume fraction vs energy ratio would suggest more noise points are captured in the detection. Whether this is due to the threshold or the U-net detection is difficult to say. However, comparing the detections from Appendix A and Appendix B there does not seem to be much difference in the AI output which suggests it is the effect of the threshold.

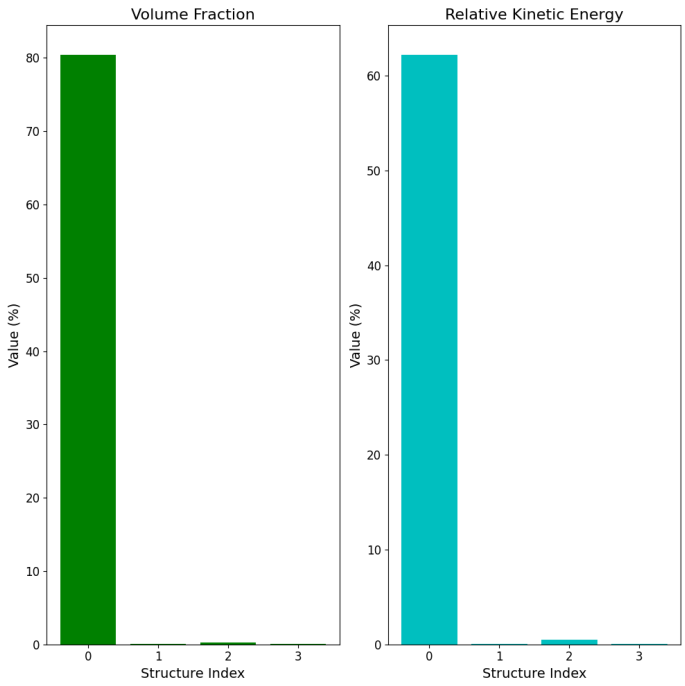
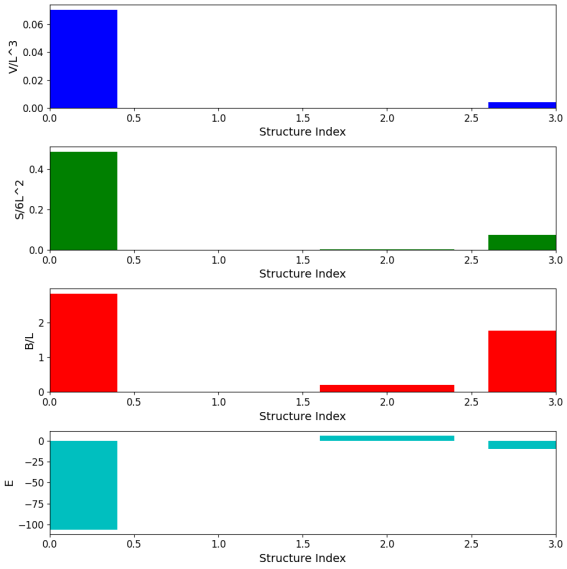


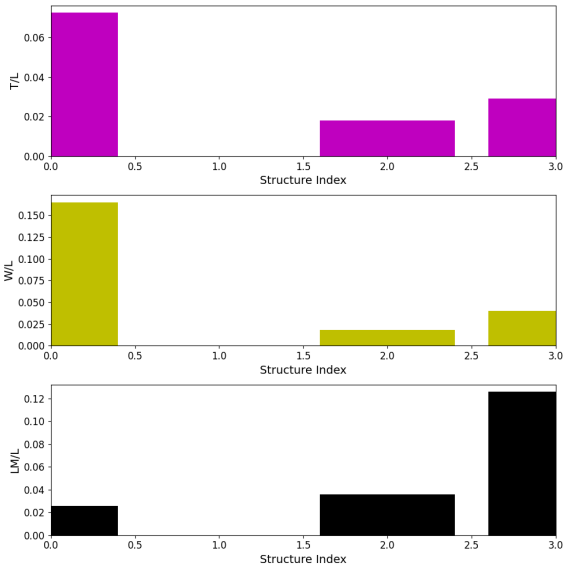
Figure 8.17: General statistics

Minkowski Functionals and shapefinders

Figure 8.18 shows the Minkowski functionals and shapefinders. The highest volume is structure 1 with $0.07L^3$ which equates to 18% of the volume. T The ratio of volume vs surface area is a factor of 7, somewhat similar to the low Reynolds number validation case. This being significantly lower than the 80% found in Figure 8.17 suggests that a large part of the detection is missing.



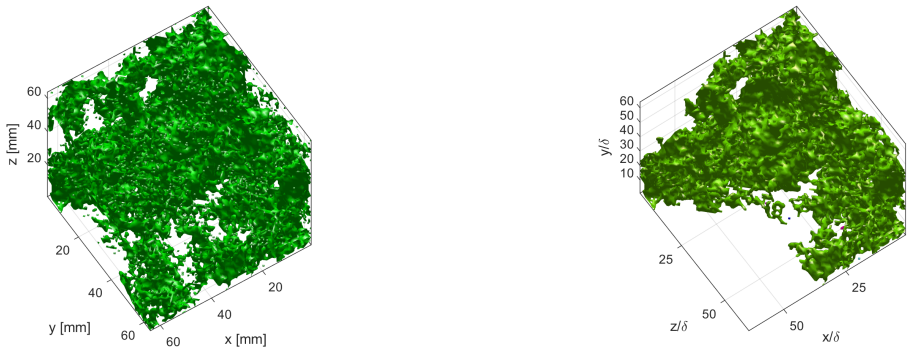
(a)



(b)

Figure 8.18: Minkowski (a) functionals (b) shapefinders

To further investigate what happened, structure 0 is plotted in Figure 8.19 with and without the smaller sub-structures. While at the front there seems to be less structure, it does not seem as if 80% of the entire domain is captured. A possible explanation is that a significant portion of smaller structures got filtered out due to the downsampling, similar to what can be observed in Figure 8.16. The general statistics from Figure 8.12 were computed prior to any downsampling.



(a) (b)

Figure 8.19: Structure 0 (a) Including sub-structures, note that all structures are in green (b) Largest continuous structure

PCA length scales

Figure 8.21 shows the PCA lengths and standard deviations. The maximum lengths of the structures are within 3.5 to 6 standard deviations which is reasonable and suggests the extremes from the continuous structures are not noise points. The most striking observation is that structures 2 and 3 have length scales only a factor 2-3 smaller than structure 0 while structure 0 has a much larger area. Figure 8.20 shows that structure 3 is just a small blob (the purple one), hence the SD is probably skewed due to there being so few points. For structure 0 though, the PCA lengthscales seem realistic when looking Subfigure 8.19(b) and knowing that $L = 88$ gridpoints. Therefore, this PCA method for acquiring length scales should only be used when it is clear the structure is large.

Moreover, on the PCA result, no structures have all three length scales in the order of the integral length scale. This result is expected as the domain itself is not of the integral length scale although it is theoretically possible for a structure of this scale to fit. More likely significant parts of the detected structures were cut off as they were not part of the domain. One could make the argument that the high Reynolds number subvolume dataset is too small compared to its integral length scale to find complete large-scale structures.

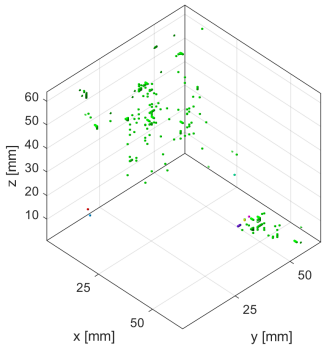


Figure 8.20: Structure 3

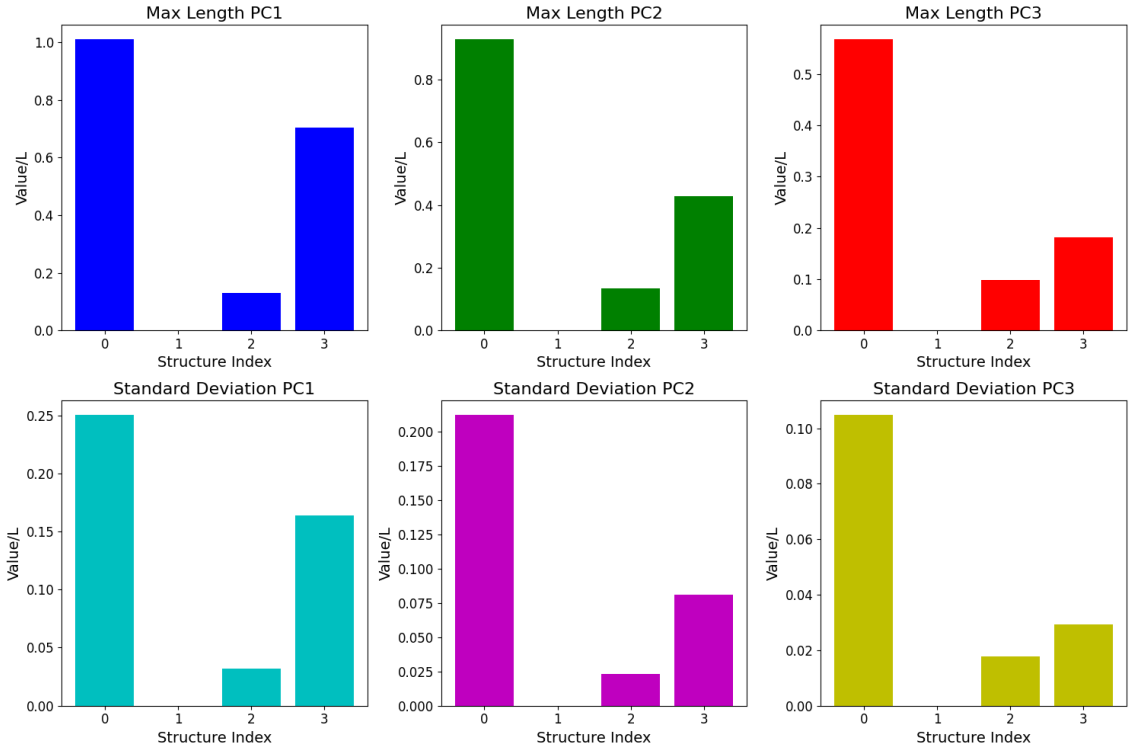


Figure 8.21: PCA length scales

8.5 U-net high Re complete volume

This section discusses the results of the complete high Reynolds number dataset. Note that, unlike the other two datasets, this one was already downsampled by a factor of 8. The first five cube detections and the first five identified large-scale structures are depicted in Appendix C. Computation times were again similar to the other two cases. MATLAB processing took slightly longer than the subvolume as there were more structures found during the detection. The entire MATLAB code was also executed for all the structures using a factor 16. This process took roughly 30 hours. From these computation times, the best practice seems to be to run a low-resolution detection first and get an idea of the most influential structures. Once these are found it is always possible to then run a higher resolution on the largest identified structures.

8.5.1 U-net histogram predictions

Figure 8.22 shows the effect of using a factor 32 downsampling versus a factor 64 downsampling. The 16 and 64 factors include factor 8 from the original downsampling mentioned in subsection 7.8.3. While it is clear a lot of information is lost, the general shape is still preserved. Some volume may have been lost however, this is deemed acceptable.

8.5.2 General statistics of detected structures

Figure 8.23 shows the volume fractions and kinetic energy for the detected large-scale structures. The highest structures seem to capture roughly 14% of the kinetic energy which is significantly lower than for the other two cases. This could be explained by the fact that there are simply more structures present in the volume, 22 versus 9 and 3, respectively. In total, roughly 87% of the total gridpoints is captured, which is comparable to the other two cases. Unlike the other two cases, each structure seems to have relatively more kinetic energy than volume, suggesting fewer noise points captured.

Minkowski Functionals and shapefinders

Figure 8.24 shows the Minkowski functionals and shapefinders for the high Reynolds number full grid. Structure 16 has the highest volume, $13L^3$, which is 8.6% of the total volume which seems close to the 9.6% from Figure 8.23. Note how volume, surface area and mean breadth are all several orders of L which suggests the structures are much larger than the integral length scale.

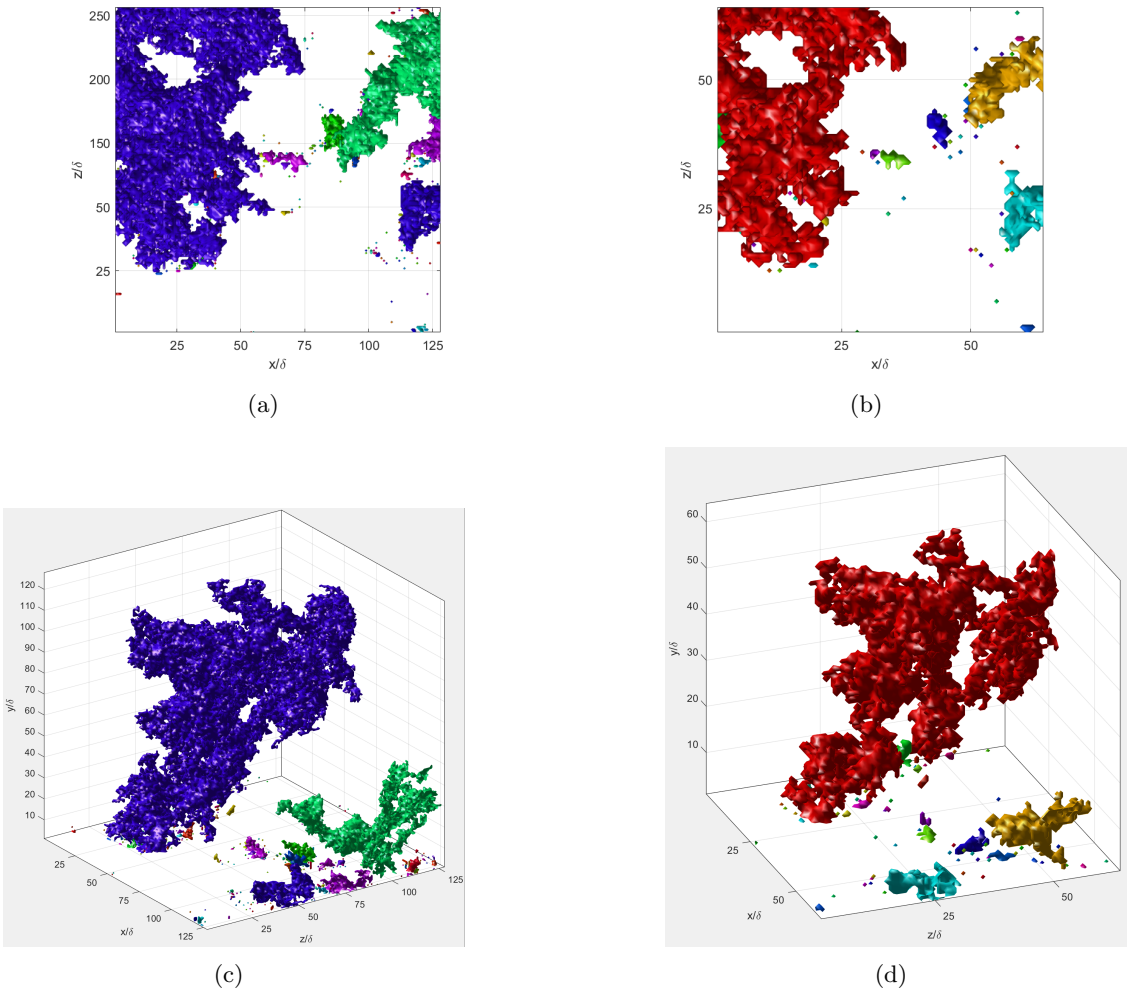


Figure 8.22: Effect of downsampling (a) factor 32, $L = 24$ gridpoints (b) factor 64, $L = 12$ gridpoints] (c) factor 32 3D (d) factor 64 3D.

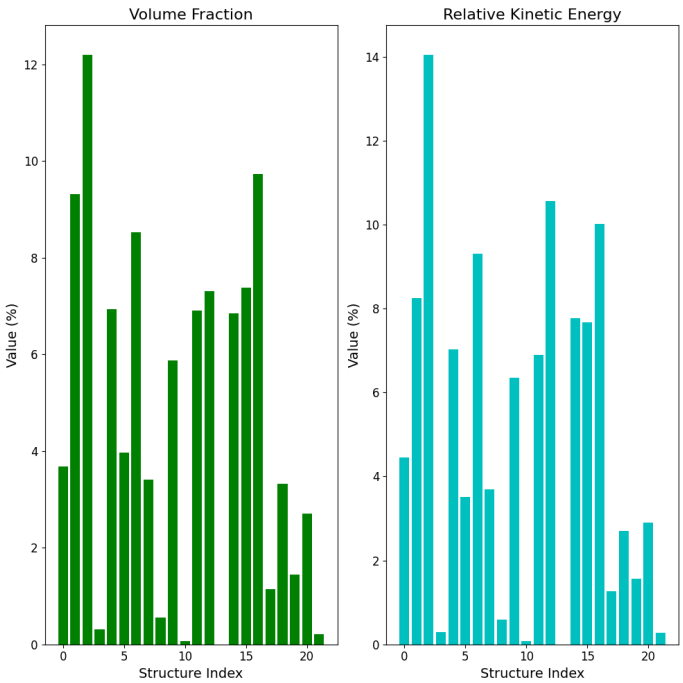
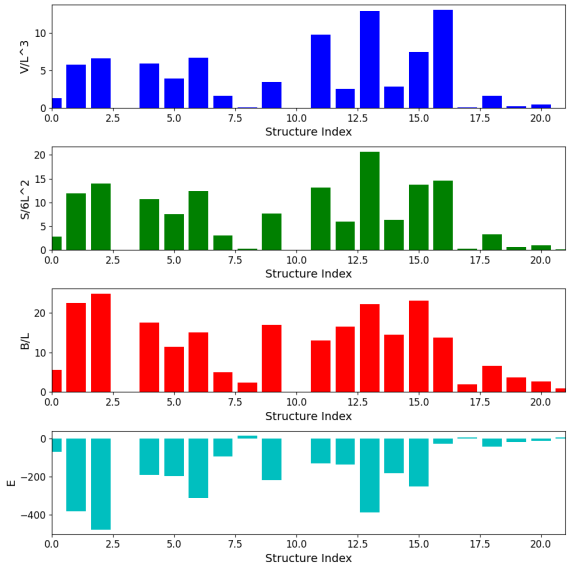


Figure 8.23: General statistics

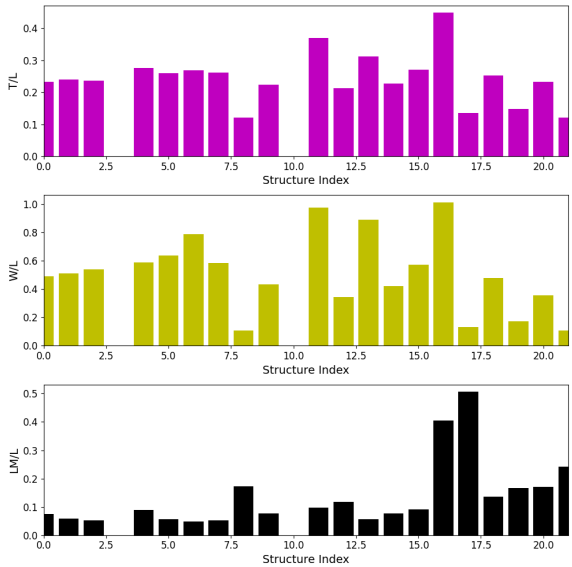
PCA length scales

Figure 8.25 shows the maximum distances and standard deviations for PC1-PC3 of the high Reynolds number full grid. The distances of the highest structures again fall between 4 and 6 standard deviations, further showing the reliability of the continuous structure detection. Comparing the lengths to the integral length scale it appears the structures are several times larger and even have standard deviations of this order, the largest one has a PC1 of $6.4L$, for example. This means there are bigger large-scale structures in this dataset than in the other two.

Comparing the detected structures with those found by Ishihara in the same dataset (although Ishihara used a single subvolume in (Ishihara et al., 2013)) there are a few distinct differences. The largest structures from Ishihara have a length of order L and a thickness of order λ. Figure 8.25 shows, however, that the structures detected by U-net have at least three dimensions of order L. This is expected as the structures from (Ishihara et al., 2013) are considered shear layers while the structures in the thesis are large-scale structures. One could hypothesise that the shear layers or thin vortex sheets lie in between two large-scale structures detected by the U-net.



(a)



(b)

Figure 8.24: Minkowski (a) functionals (b) shapefinders

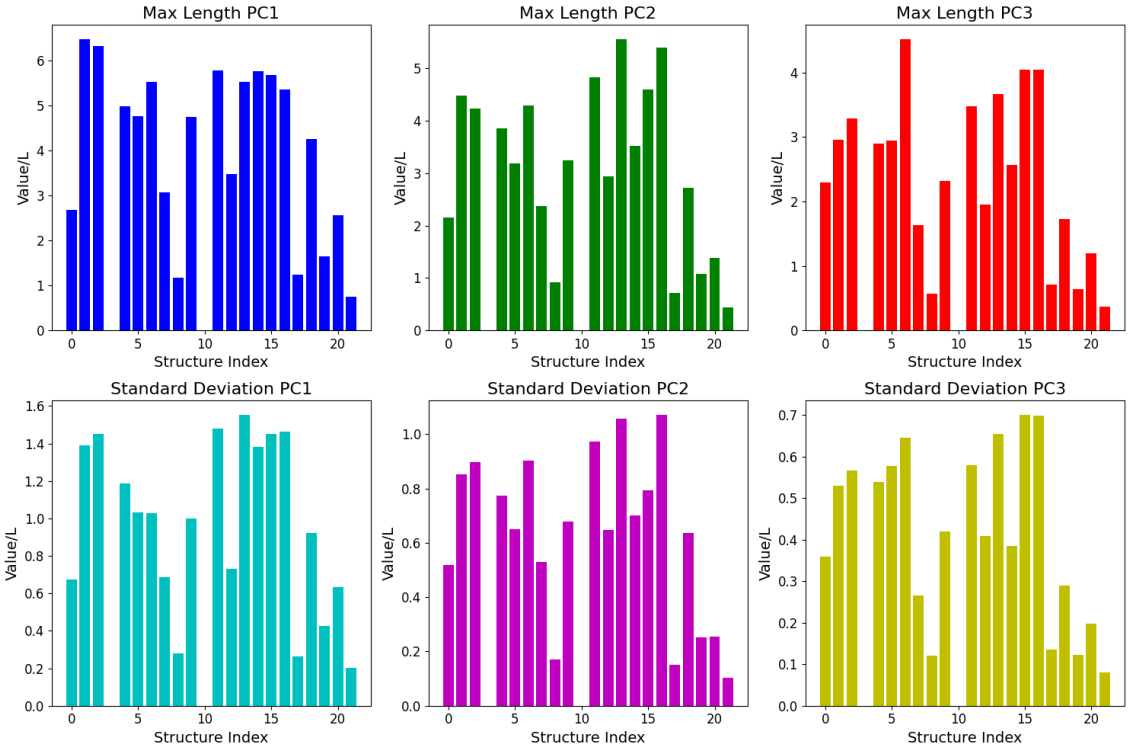


Figure 8.25: PCA length scales

8.6 Effects of downsampling

An interesting observation is how the Minkowski ratio between relative volume and relative surface area changes per downsampling. For the validation case, there is a factor 10 (surface area/volume) difference between the two. The low Re and high Re subvolume, which were reduced from 256^3 and 512^3 to 64^3 grids show factors of 4 and 5, respectively. The high Re dataset, downsampled from 4096^3 to 64^3 shows this factor being roughly 1.5. As the downsampling is increased, the ratio decreases suggesting the relative surface area decreases. This is unexpected as downsampling removes points at the edge of a structure which decreases the surface area but also removes inner points, creating holes which increase the surface area. In either case, it shows that downsampling has significant consequences for the surface topology. For general shape finding and volume occupation, this is not an issue, the length scales from PCA seem realistic upon visually inspecting them, for the biggest large-scale structures. However, if one wants to investigate large-scale structures and shear layer interactions, downsampling may not be ideal. A method to acquire structures at high-fidelity is to first visualise the coarse structures, find the most significant ones and then perform another high-fidelity visualisation for just these structures.

Another observation is that the downsampling seems to affect smaller structures much more. The green sub-structures in Subfigure 8.16(d) completely disappear for example. For further analysis, this can have consequences, such as making PCA no longer realistic, as seen in Figure 8.20 and Figure 8.21 with structure 3. In the grand scheme of the research, this limitation seems acceptable. For now, the largest large-scale structures seem to have the most influence on the flow (contain the most energy and take up the most space), making them the most interesting to dive into. For these structures, PCA seems to work fine as the derived length scales look similar to what one can estimate through visual observation.

8.7 Effects with Reynolds number

Some remarks can be made about how the detected structures change when increasing the Re from $Re_\lambda = 170$ to $Re_\lambda = 1131$. Increasing the Reynolds number does seem to increase the number of large-scale structures per dataset by a factor of 2. Note, however, that the $Re_\lambda = 1131$ has a larger computational domain, by a factor of 4.2 in each direction (when taking dividing ratios of domain/integral length scale of the two datasets) so there are relatively fewer structures present. It is also interesting how the structures from the large Re dataset are $5 - 7L$ while those from the low Re are $1 - 2L$, meaning the large Re structures are a lot larger. It seems as if at lower Re, the structures are split apart into multiple smaller ones. The exact reason for the structures being larger and fewer compared to L at high Re is, however, at this point not clear. A possible explanation could be that

at a higher Re less viscous forces are present, hence less energy dissipation and this would give the structures more opportunity to grow. At $Re_\lambda = 170$ the flow is not fully turbulent yet and the scales are closer together which could explain the potentially stronger influence of viscous dissipation than at $Re_\lambda = 1131$.

Chapter 9

Conclusion and avenues for future work

This chapter summarises the main findings and explores new avenues for future work. The three main drivers of effectiveness for the algorithm discussed below are ease of use, computational cost and quality of the results.

Going back to the main research and sub-questions from chapter 4 and chapter 6, several conclusions can be drawn. At a relatively early stage, it became clear that U-net was the most suitable detection algorithm. At this point, there is insufficient information on large-scale structures to supply DBSCAN with a suitable minimum number of points and radius. YOLOv7 could theoretically work, however, modifying the architecture and training it from scratch is far beyond the scope of a master's thesis in turbulence.

The three-layer deep U-net type network as seen in Figure 7.5 seems to work well with the artificial data. The detection is, however, not fully automated as there are still some user-controlled variables: the radius of the labels around the artificial peaks, the number of cubes created, the downsampling factors and the thresholds for when structures are considered identical. All of these are, however, scalar quantities and the same settings have produced acceptable results on three different datasets, suggesting these settings are fairly robust (or can be tuned to the dataset size in the case of the downsampling factors). Compared to manually setting a threshold in a histogram, this new U-net detection algorithm seems far less prone to precision errors and requires significantly less manual effort. Note furthermore, that the downsampling factors and cube slicing are just as necessary for the manual thresholding method. The human eye will struggle just as much with a denser histogram as a neural network and the point cloud to structure conversion is the same for both methods.

In terms of computational cost, the entire algorithm can simply be run on laptop hardware,

using both the GPU and CPU alongside a Windows operating system. The main driver of computational cost is the isolation of continuous structures within the detected point clouds, caused by the use of MATLAB "for" loops which are notoriously inefficient. The problem is partially remedied through the use of downsampling. Alternatively, one could seek or create a more efficient algorithm. Training and deploying the neural network take several minutes and milliseconds, respectively for the 36x18 images used throughout the thesis. Increasing the number of cubes also increases computation times as more structures are detected and the merging process takes longer.

Evaluating the quality of the detections is inherently difficult as there is no clear-cut definition of a large-scale structure. Nevertheless, there are observable key characteristics derived from hypotheses and previous findings from the literature. The detected structures in this research seem to align with the Kolmogorov hypothesis (Kolmogorov, 1941) in that they are very significant in size, carry most of the energy in the flow and have a relatively uniform velocity. The detected structures also seem relatively robust in terms of shape and volume to downsampling. Compared to the recent literature, the structures seem quite similar to those found by (Elsinga and Marusic, 2010) in the same dataset in terms of shape. They are quite different to those found by (Ishihara et al., 2013) which is expected since these are shear layer structures. It could, however, be likely that the shear layer structures are at the edges of two of the detected large-scale structures.

Now that there is a reliable and efficient method to detect large-scale structures, more data on them can start being gathered. For further research, it would be interesting to evaluate the same or surrounding cube(s) used by (Ishihara et al., 2013), find the large-scale structures in them and see if these align with the shear layers. One would expect the large-scale structures to govern the large-scale straining motions which affect the shear layers. As a further step large-scale structures can be tracked in time and this may give a better understanding of how shear layers are developed. Other turbulence phenomena such as extreme dissipation would potentially also benefit from improved knowledge of large-scale structures and their influence on the general flow.

Another area of further research is to evaluate the Reynolds number scaling of large-scale structures. There were stark differences between the low Reynolds number case and the high Reynolds number full domain case. The high case had a larger quantity of large-scale structures and the structures had significantly larger dimensions. Note, however, that the low Reynolds number was at $Re_\lambda = 175$ which is below 250 and thus the flow is most likely not fully turbulent. Evaluating the $Re_\lambda = 732$ or $Re_\lambda = 675$ datasets from (Ishihara et al., 2007) and comparing results from the $Re_\lambda = 1131$ dataset could give a better picture of large-scale structure scaling with Reynolds number in fully turbulent flows. A deeper dive can also be done in the holes in the detected structures. Are these a physical phenomenon or a result of the detection that cuts parts of the structure off? If they are a result of the detection, one may seek to increase the label radius and/or have the artificial data match

the real histograms to an even closer degree. It would then also be worthwhile to see if the Minkowski functionals and shapefinders produce more realistic results. Nevertheless, with or without holes, the structures are off the integral length scale and occupy significant volumes. Furthermore, (Elsinga and Marusic, 2010) also seems to find non-convex structures.

Regarding the improvement of the current method, one could seek to account for periodicity in the 3D velocity domain. HIT datasets typically have periodic boundaries which means structures can leave at one edge and reappear at the other. Creating an algorithm that can check if two structures at opposite edges are actually continuous is a very difficult task and is also open to a lot of interpretation. The easiest solution would be to have the same cube at the edges so that the edge structures are 'glued' together. This would, however, increase the computational cost by at least five times. Furthermore, it would not work if a structure covers the entire span of the domain. For the time being, these edge continuous structures are treated as individual structures.

To conclude, a new method was developed which is automated, therefore less prone to precision error and computationally efficient. Detections were done on two independent (and one dependent) datasets where the neural network successfully drew contours around the peaks in the 2D histograms of the velocity orientations. The structures that were derived from these match those with literature and seem to behave as predicted by the Kolmogorov hypothesis (Kolmogorov, 1941). The next research step will be to deploy the method to gather more data on large-scale structures and develop universal statistics.

Bibliography

Arvai, K. (2021). kneed: Knee-point detection in python.

Azad, R., Aghdam, E. K., Rauland, A., Jia, Y., Avval, A. H., Bozorgpour, A., Karimijafarbigloo, S., Cohen, J. P., Adeli, E., and Merhof, D. (2022). Medical image segmentation review: The success of u-net.

Bedre, R. (2023). Dbscan clustering algorithm in python (with example dataset). Blog post. Accessed on: April 10, 2023.

Blonigan, P. J., Farazmand, M., and Sapsis, T. P. (2019). Are extreme dissipation events predictable in turbulent fluid flows? *Physical Review Fluids*, 4(4):044606.

Bouchard, Y. (2019). Tesla's deep learning at scale: Using billions of miles to train neural networks. Blog post. Accessed on: April 2, 2023.

Calinski, T. and Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27.

Cardona, A., Saalfeld, S., Preibisch, S., Schmid, B., Cheng, A., Pulokas, J., Tomancak, P., and Hartenstein, V. (2013). A benchmark for the comparison of 3d electron microscopy image segmentation algorithms. *Medical Image Analysis*, 17(8):930–943.

Cheng, C. (2022). Principal component analysis (pca) explained visually with zero math. Towards Data Science. Accessed: 13-11-2023.

Chi, C., Thévenin, D., Smits, A. J., Wolligandt, S., and Theisel, H. (2017). Identification and analysis of very-large-scale turbulent motions using multiscale proper orthogonal decomposition. *Physics of Fluids*, 29(3):035104.

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

Dou, Z., Pecenak, Z. K., Cao, L., Woodward, S. H., Liang, Z., and Meng, H. (2016). Piv measurement of high-reynolds-number homogeneous and isotropic turbulence in an enclosed flow apparatus with fan agitation. *Measurement Science and Technology*, 27(3):035305.

Dwivedi, P. (2020). Yolov5 compared to faster r-cnn. who wins? Blog post.

Elsinga, G. (2023). Detection of large-scale turbulent motions. Laboratory of Aero and Hydro Dynamics, Delft University of Technology.

Elsinga, G. E., Ishihara, T., Goudar, M. V., da Silva, C. B., and Hunt, J. C. R. (2017). The scaling of straining motions in homogeneous isotropic turbulence. *Journal of Fluid Mechanics (2017), vol. 829, pp. 31–64.*

Elsinga, G. E., Ishihara, T., and Hunt, J. C. R. (2020). Extreme dissipation and intermittency in turbulence at very high reynolds numbers. *The Royal Society Volume 476, Issu 2243.*

Elsinga, G. E. and Marusic, I. (2010). Universal aspects of small-scale motions in turbulence. *Journal of Fluid Mechanics (2010), vol. 662, pp. 514–539.*

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 226–231.

Fiscaletti, D., Attili, A., Bisetti, F., and Elsinga, G. E. (2016). Scale interactions in a mixing layer – the role of the large-scale gradients. *Journal of Fluid Mechanics , Volume 791, pp. 154 - 173.*

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2016). Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):142–158.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks.

Gupta, A. (2020). Understanding semantic segmentation with unet. https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47. Accessed: 2023-05-29.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hickel, S. and Hulshoff, S. (2020). CFD 3 – Large Eddy Simulation. Lecture slides from course CFD 3: Large Eddy Simulation - AE4139.

Horiguchi, M., Hayashi, T., Adachi, A., and Onogi, S. (2012). Large-scale turbulence structures and their contributions to the momentum flux and turbulence in the near-neutral atmospheric boundary layer observed from a 213-m tall meteorological tower. *Boundary-Layer Meteorology*, 144:179–198.

Hunt, J. C. R., Wray, A., and Moin, P. (1988). Eddies, streams, and convergence zones in turbulent flows. Ctr-s88, Center for Turbulence Research.

Ishihara, T., Kaneda, Y., and Hunt, J. C. (2013). Thin shear layers in high reynolds number turbulence—dns results. *Flow, turbulence and combustion*, 91:895–929.

Ishihara, T., Kaneda, Y., Yokokawa, M., Itakura, K., and Uno, A. (2007). Small-scale statistics in high-resolution direct numerical simulation of turbulence: Reynolds number dependence of one-point velocity gradient statistics. *Journal of Fluid Mechanics*, 592:335–366.

Jeong, J. and Hussain, F. (1995). On the identification of a vortex. *Journal of Fluid Mechanics*, 285:69–94.

Jordan, P. and Colonius, T. (2012). Wave packets and turbulent jet noise. *Annu. Rev. Fluid Mech. 2013. 45:173–95.*

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Kolmogorov, A. N. (1941). The local structure of turbulence in incompressible viscous fluid for very large reynolds numbers. *Doklady Akademii Nauk SSSR*, 30(4):299–303.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C., Bottou, L., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Kumar, S. and Srivastava, V. (2021). Performance comparison of yolo models. https://learnopencv.com/performance-comparison-of-yolo-models/.

Legland, D. (2023). imminkowski. MATLAB Central File Exchange. Retrieved August 19, 2023.

Leung, T., N., S., and Davidson, P. A. (2012). Geometry and interaction of structures in homogeneous isotropic turbulence. *Journal of Fluid Mechanics*, 710:453–481.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P. (2015). Microsoft coco: Common objects in context.

Liu, C., sheng Gao, Y., rui Dong, X., qian Wang, Y., ming Liu, J., ning Zhang, Y., shu Cai, X., and Gui, N. (2019). Third generation of vortex identification methods: Omega and liutex/rortex based systems. *Journal of Hydrodynamics.*

Liu, C. and Wang, Y. (2020). *Liutex and Third Generation of Vortex Defnition and Identifcation.* Springer, Arlington, TX, USA, first edition.

Malgoezar, A. M., Snellen, M., Merino-Martinez, R., Simons, D. G., and Sijtsma, P. (2017). On the use of global optimization methods for acoustic source mapping. *The Journal of the Acoustical Society of America*, 141(1):453–464.

Marusic, I. and Heuer, W. D. (2007). Reynolds number invariance of the structure inclination angle in wall turbulence. *Physical review letters*, 99(11):114504.

Miller, S. A. and Shen, W. (2018). Radiation from large-scale structures within a turbulent shear layer. *International Journal of Aeroacoustics 2018, Vol. 17(4–5) 541–570.*

Moisy, F. and Jiménez, J. (2004). Geometry and clustering of intense structures in isotropic turbulence. *Journal of Fluid Mechanics.*

Mowlavi, S., Serra, M., Maiorino, E., and Mahadevan, L. (2022). Detecting lagrangian coherent structures from sparse and noisy trajectory data. *Journal of Fluid Mechanics*, 948.

Nazir, M., Shakil, S., and Khurshid, K. (2021). Role of deep learning in brain tumor detection and classification (2015 to 2020): A review. *Computerized Medical Imaging and Graphics, Volume 91, July 2021, 101940.*

Nieuwstadt, F. T., Boersma, B. J., and Westerweel, J. (2016). *Turbulence Introduction to Theory and Applications of Turbulent Flows.* Springer, Delft, The Netherlands, first edition.

Odena, A., Dumoulin, V., and Olah, C. (2016). Deconvolution and checkerboard artifacts. *Distill.*

Ouellette, N. T. (2021). Extending the reach of lagrangian analysis in turbulence. *Journal Fluid Mechanics*, 924:F1.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Pope, S. B. (2000). *Turbulent Flows.* Cambridge University Press, Oxford, UK, first edition.

Pope, S. B. and Chen, J. H. (1982). The evolution of surfaces in turbulence. *International Journal of Engineering Science*, 20:1141–1156.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection.

Ren, S., He, K., Girshick, R., and Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation.

Saltzman, A. J., Lowe, K. T., and Ng, W. F. (2021). Finite control volume and scalability efects in velocimetry for application to aeroacoustics. *Experiments in fluids*.

Sandberg, R. D. (2015). Compressible-flow dns with application to airfoil noise. *Flow, Turbulence and Combustion volume 95, pages211–229*.

Schubert, E., Sander, J., Ester, M., Kriegel, H.-P., and Xu, X. (2017). Dbscan revisited, revisited: Why and how you should (still) use dbscan. *ACM Trans. Database Syst. 42, 3, Article 19*.

Siggia, E. D. (1981). Numerical study of small-scale intermittency in three-dimensional turbulence. *Journal of Fluid Mechanics*.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.

Skalski, P. (2022). How to train yolov7 instance segmentation on a custom dataset.

Suss, A., Mary, I., Garrec, T. L., and Marié, S. (2023). Comprehensive comparison between the lattice boltzmann and navier–stokes methods for aerodynamic and aeroacoustic applications. *Journal Pre-proof, to appear in Computers and Fluids*, 257:105881.

Tabor, M. and Klapper, I. (1994). Stretching and alignment in chaotic and turbulent flows. *Chaos, Solitons & Fractals*, 4:1031–1055.

Wang, C.-Y., Bochkovskiy, A., and Liao, H.-Y. M. (2022). Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *ArXiv*, abs/2207.02696.

Wei, Z., Zhang, J., Jia, R., and Gao, J. (2022). An improved method for coherent structure identification based on mutual k-nearest neighbors. *Journal of Turbulence*, pages 1–20.

Yang, R., Zhang, X., Reiter, P., Lohse, D., Shishkina, O., and Linkmann, M. (2022). Data-driven identification of the spatiotemporal structure of turbulent flows by streaming dynamic mode decomposition. *GAMM-Mitteilungen*, 45(1):e202200003.

Ye, B., Ding, C., and He, X. (2005). Discovering clusters in gene expression data: A case study. *Journal of Bioinformatics and Computational Biology*, 3(2):519–542.

Zhou, J., Adrian, R. J., Balachandar, S., and Kendall, T. M. (1999). On the identification of a vortex. *Journal of Fluid Mechanics*, 385:199–214.

Appendix A

Some detections and structures from the low Reynolds number dataset

This appendix shows detections of the first five cubes and non-zero structures in Figure A.1 and Figure A.2. the labelling of both is consistent with that found in chapter 8.

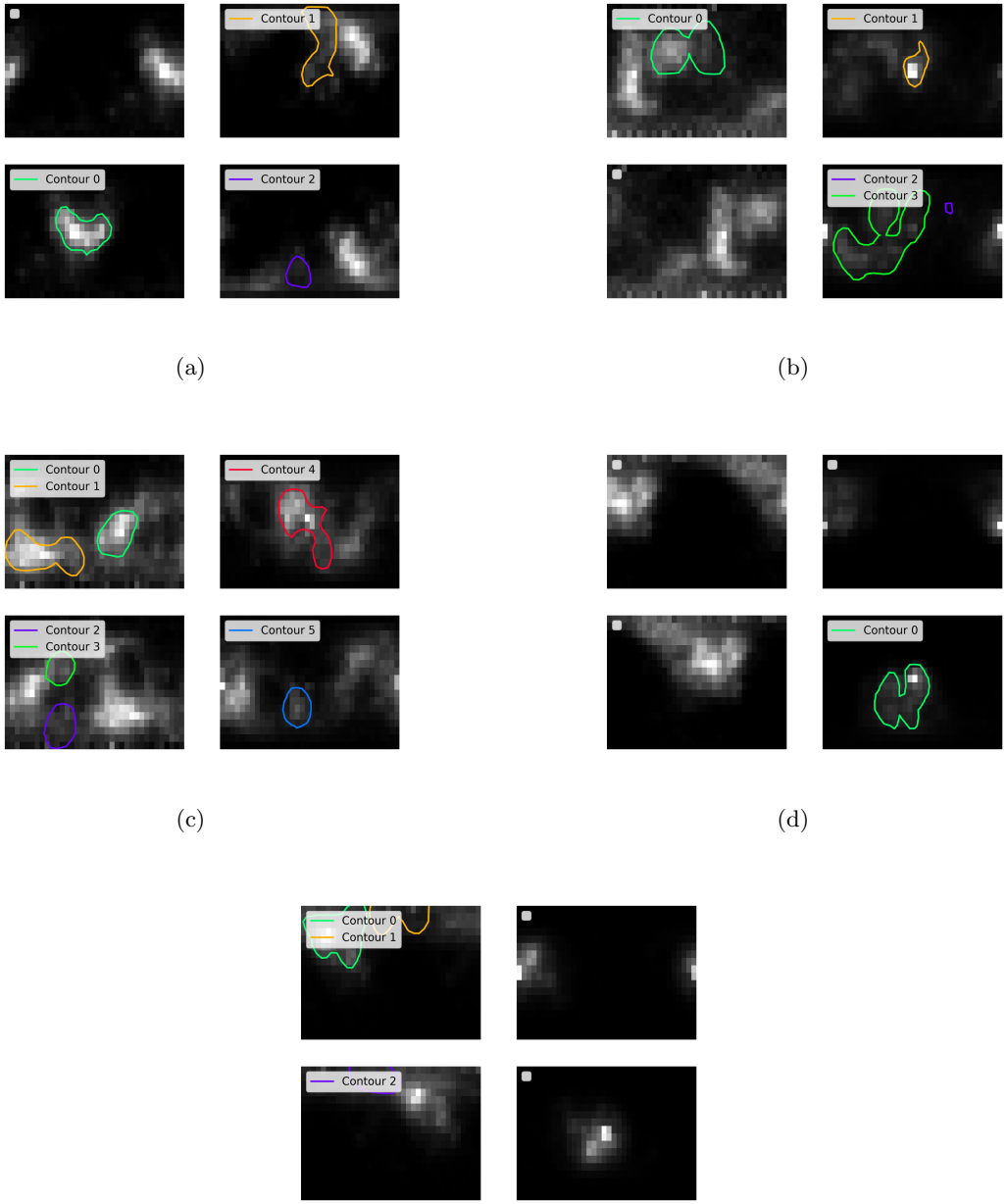


Figure A.1: Detections from the first 5 cubes. (a) Cube 0 (b) Cube 1 (c) Cube 2 (d) Cube 3 (e) Cube 4

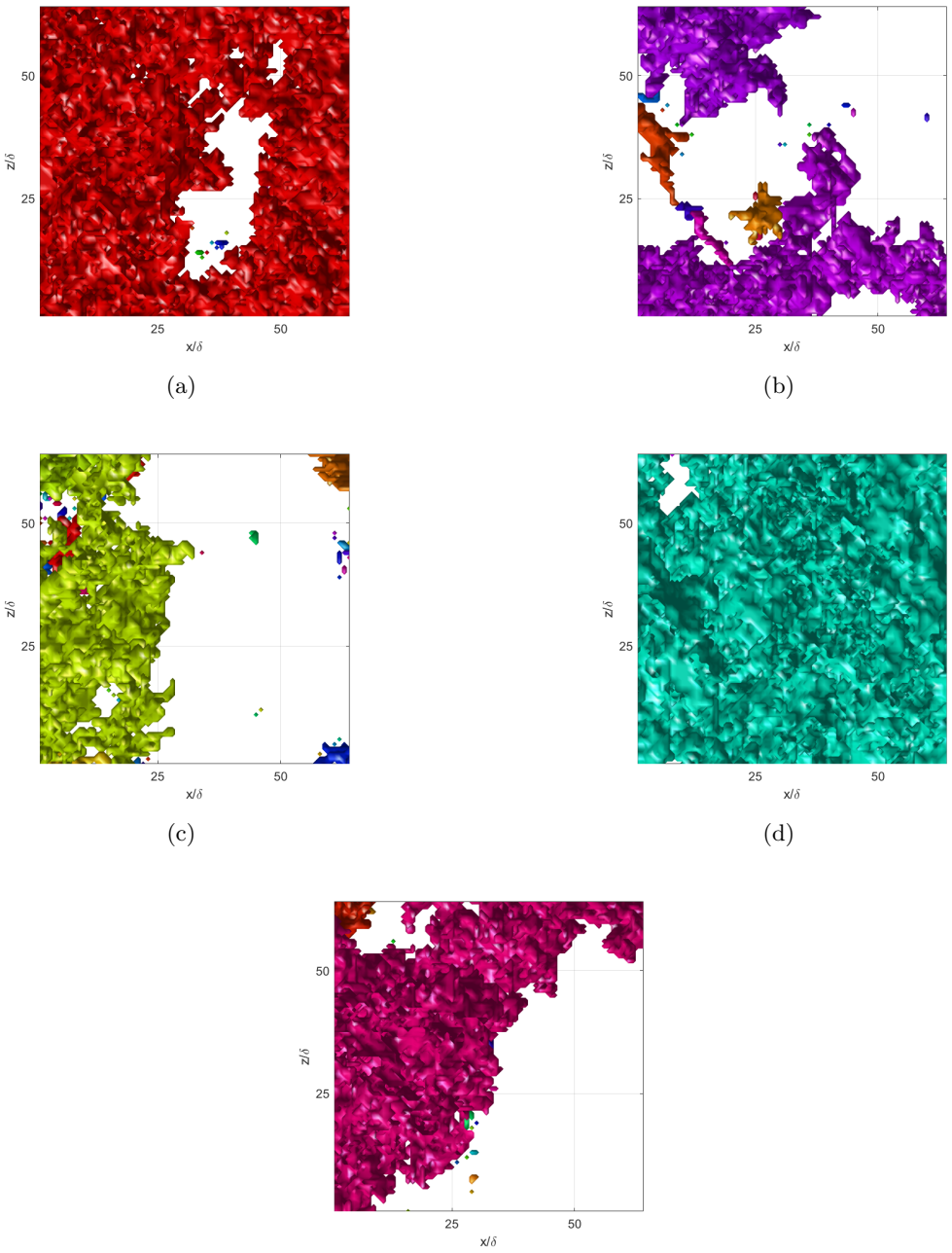


Figure A.2: First five structures (a) Structure 0 (b) Structure 1 (c) Structure 2 (d) Structure 3 (e) Structure 4

Appendix B

Some detections and structures from the high Reynolds number subgrid dataset

This appendix shows detections of the first five cubes and non-zero structures in Figure B.1 and Figure B.2. the labelling of both is consistent with that found in chapter 8.

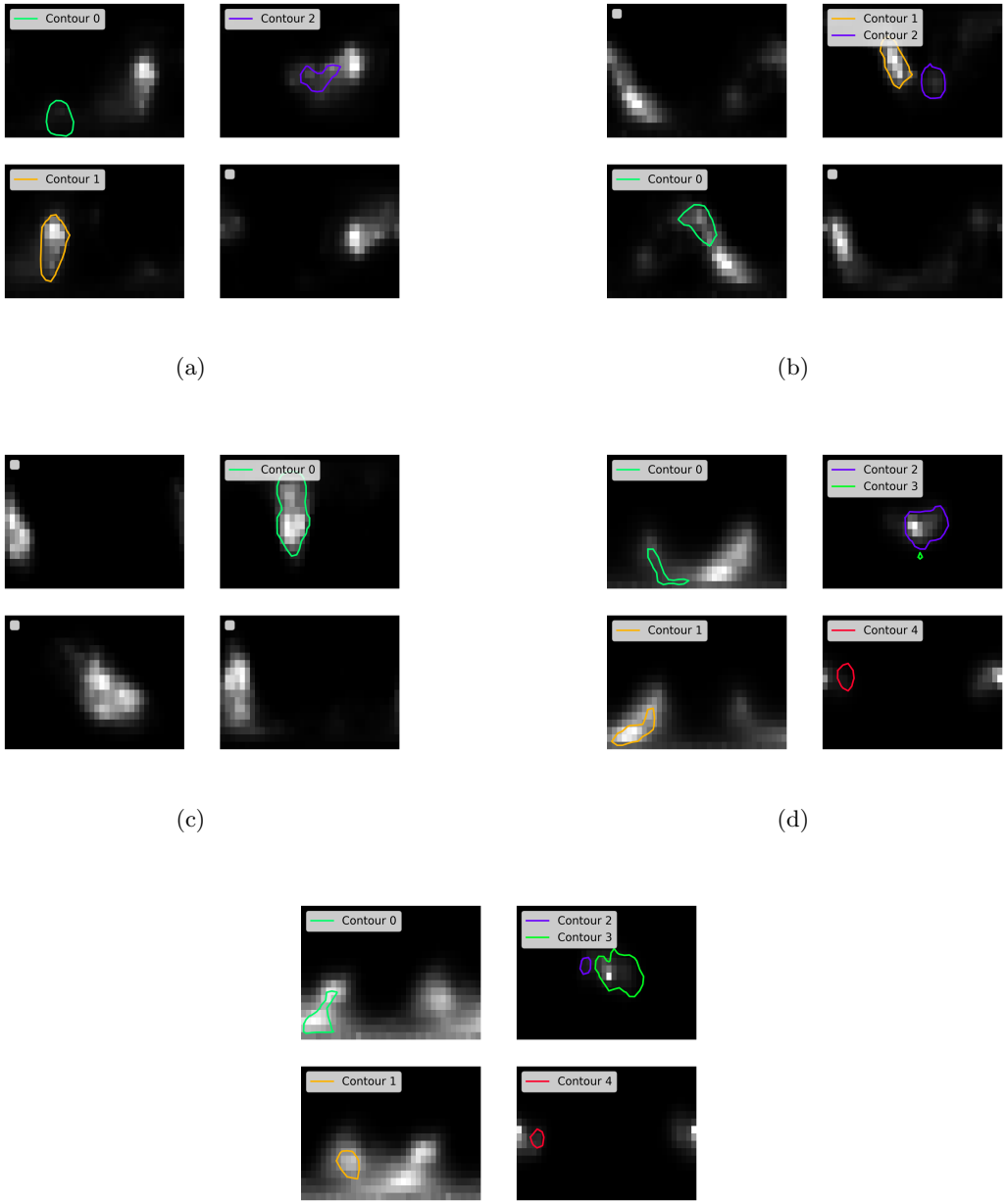


Figure B.1: Detections from the first 5 cubes. (a) Cube 0 (b) Cube 1 (c) Cube 2 (d) Cube 3 (e) Cube 4

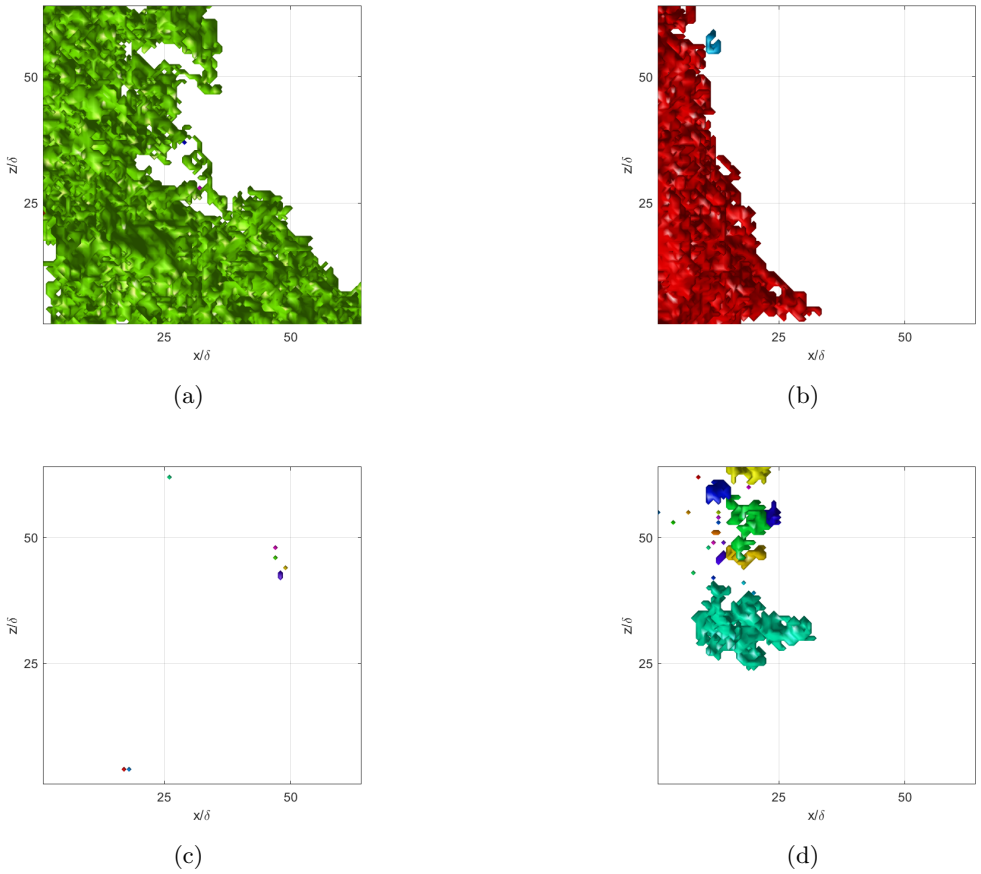


Figure B.2: First five structures (a) Structure 0 (b) Structure 1 (c) Structure 2 (d) Structure 3

Appendix C

Some detections and structures from the high Reynolds full downsampled dataset

This appendix shows detections of the first five cubes and non-zero structures in Figure C.1 and Figure C.2. the labelling of both is consistent with that found in chapter 8.

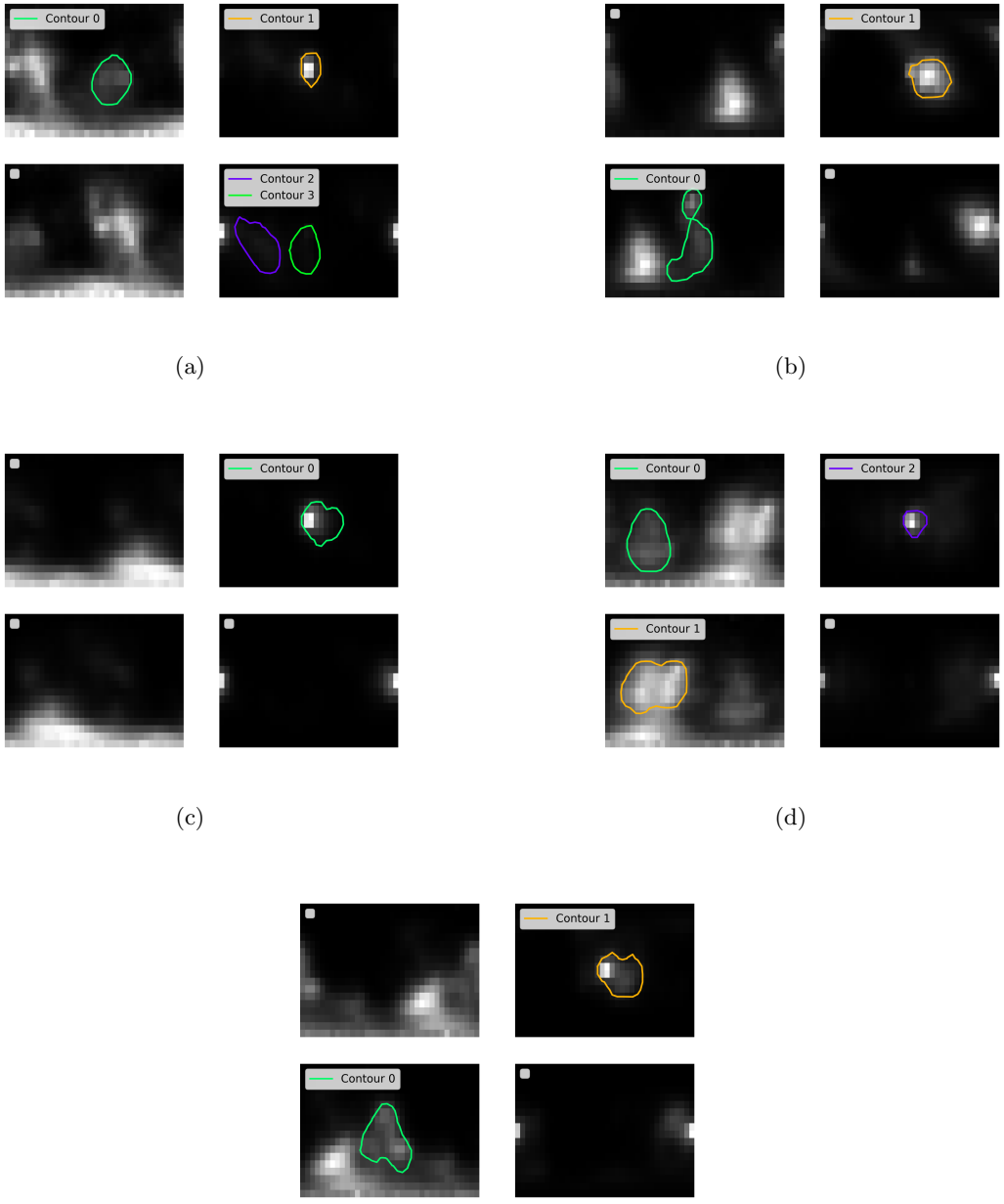


Figure C.1: Detections from the first 5 cubes. (a) Cube 0 (b) Cube 1 (c) Cube 2 (d) Cube 3 (e) Cube 4

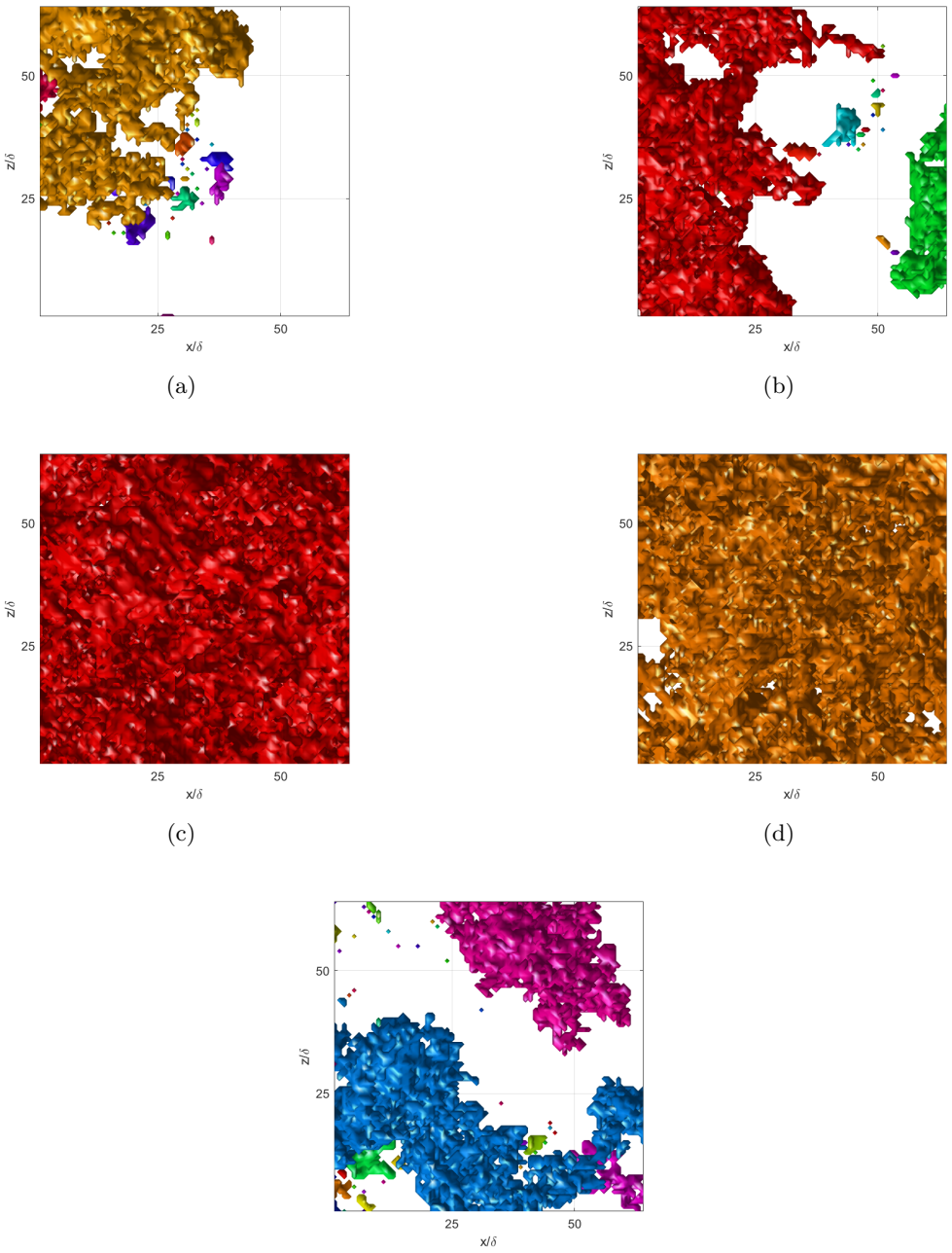


Figure C.2: First five structures (a) Structure 0 (b) Structure 1 (c) Structure 2 (d) Structure 3 (e) Structure 4