Fluid Flow Structure Identification Software

Joseph Badra and Sebastian Lora

The George Washington University

Mechanical and Aerospace Engineering Department

May 5th, 2019

Abstract

This project focused on the creation of a software suite that automates the analysis of a variable fluid flow dataset. This solution was built within the Matlab computing environment and consists of a set of programs that the user can effortlessly incorporate into their workflow. The software suite features variable analysis techniques which can be applied to either scalar or vector field data. the features include filtering, volumetric flow visualization, Proper Orthogonal Decomposition, and structure identification by method of k-means clustering. The software not only ensures effective data analysis, but also provides organized output of processed data in a streamlined and approachable manner. The customer, The George Washington University Thermo-Fluids Laboratory, conducts research in the field of fluid mechanics. The software simplifies the process of analyzing and understanding fluid flow; it is to aid in the analysis of the fluid data to be able to better and more easily characterize formed structures.

1 Introduction

1.1 Scope

Fluid mechanics can be complex and unpredictable when looking at chaotic, turbulent flow. In order to further the understanding of this type of flow there must be adjacent advances in the understanding of the prominent structural patterns constituent of the flow. Specifically, advances in identifying the flow's vortices which exhibit unique and dynamic characteristics.

Vortices and structures in turbulent, chaotic fluid flow are difficult to identify and even more difficult to accurately model. This being said, it is essential to undertake this difficult task when attempting to understand dynamics of the flow. The customer, The George Washington University Thermo-Fluids Laboratory, performs research with this same goal in mind. They run an ongoing experiment involving a buoyant jet discharging freely into linearly stratified environment, see Figure 1 below.

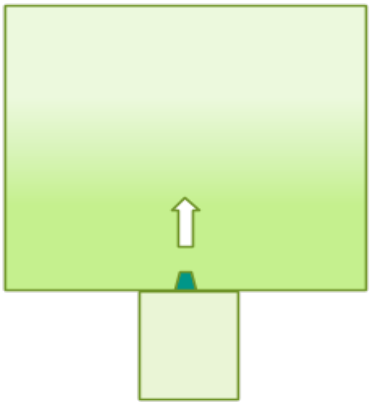

Fig. 1: Stratified Jet Experiment pre-experimental run: The lighter the shade of green, approximately the lower density the fluid in that region is

The solution must be compatible with specified types of datasets in either single or dual matrix configuration. Furthermore, the automated analysis capabilities must include the calculation of swirling strength, Fourier analysis with variable filtering, both linear and nonlinear proper orthogonal decomposition, and a structure identification method. The customer specified

that they do not want the design to be constrained by a computational power limit so long as it can be utilized in a reasonable time frame.

The utilization of datasets created from actual flow was a thoughtful decision. An alternative approach would be to create an ideal simulated flow. Although technologically feasible, the approach would not be practical or realistic. It would require an unreasonable increase in computational power and processing time. Due to this the datasets used throughout the development were sourced from the laboratory's ongoing stratified jet experiment.

1.2 Technical Literature Review

At its core, the software suite simplifies complex patterns and behaviors. More specifically, it simplifies the vortices of a fluid flow presented in the form of a velocity field history. By conducting an extensive review of published literature, different computational method were identified. The literature referenced was either recommended by a mentor or compiled from sources with authority on the subject.

The usefulness of gathered information was measured in a two-step process. First, preliminary experimental testing of potential computational methods was conducted. Once a meaningful amount of results were collected they were validated by method of comparison to known physical properties, known conditions of the stratified jet experiment, and extraneous situational indicators. In the end, it was decided that the most useful information came in the form of coherent structures, dominant structures, and non-dominant yet potentially repeating structures.

A modern approach of extracting useful information from a flow is by applying Proper Orthogonal Decomposition (POD). POD is a computational approach in a similar vein to the other integrated approaches included in the suite. The first, and arguably simplest, step to the

application of POD is the organization of raw flow data. This involves structuring the data in an organized and rigorous manner. It is a crucial step because the output is formatted with the same structure as the input. Without a noted and standardized organization, the resultant data is unusable. The next step is to compute a correlation matrix C relevant to the data. This can be accomplished in a number of ways, such as C=U*U' or C=(V*V' + U*U')/n. [1][2] Furthermore, excerpts of code were found in both [1] and [2], with Chen 2012 proving to be the more useful reference.

Swirling strength was an additional method that was both requested and researched. During literature review it was found that this method would most likely prove to be the least significant of the suite, yet it was included for redundancy and comprehensive design. [5] André 2017 presents the use of swirling strength, it's analysis, and its interpretation.

Fourier analysis can be used to filter data with the purpose of extract more meaningful results. This analysis can be performed on the temporal, spectral, or spatio-temporal domains. This technique allows for common structures with varying magnitudes to be easily identified.[7] Fourier analysis, when used in conjunction with either a low-pass or high-pass filter, allows the user a greater level of control over the level of detail exhibited in the analysis.

A team at the Natl. Research Council–Marine Technology Research in Rome, Italy collaborated on the project and proved to be a valuable resource. Their primary impact to act as reference throughout the duration of the project. They published a paper on the dimensionality reduction applicable to the datasets used by the suite. [8]

1.3 Functional Requirements

Table 1: Functional Requirements

	Functional Requirements
Dataset Compatibility	Two 3D matrices (U & V)
Automation of Analysis Methods	One 3D matrix (Vorticity)
	Swirling strength
	Fourier Analysis (filtering)
	POD (linear/nonlinear)
	Structure Identification (K-Means)
Verification Method	Comparison to moving average

The functional requirements, as established during the project design, are displayed in Table 1. The requirements were developed in collaboration with our advisor and customer. Great care was taken to identify functional requirements that, once met, would ensure the suite provides a comprehensive assortment of features and workflow optimizations. The first set of requirements, dataset compatibility, ensure the software provides compatibility with the most common and relevant flow data structures. The following set of requirements, automation of analysis methods, are a concise yet comprehensive selection of analysis methods. The selected methods are capable of both streamlining typical laboratory workflow and providing quick access to alternative analysis methods which still produce meaningful results. The final requirement, verification method, was selected to act as a baseline evaluation during development and debugging.

2 Team Member Roles

Table 2: Team Member Roles Categorized by Function

Name	Writing Role	Technical Role
Sebastian Lora	Chief Writer	Programming implementation. Project Management Leader. Assisted in subsystem design: - Fourier analysis filtering
Joseph Badra	Editor	Programming Team Leader Graduate and Faculty Liaison

Table 2 gives a brief overview of the some of the major roles taken on by team members during the duration of the engagement. Due to the nature of being such a small team, there was significant overlap across defined roles and functions. Acting as programming implementation, Sebastian Lora oversaw the implementation and integration of key features which facilitated the development and testing of the software. He also took an active role in subsystem design, particularly when developing the strategy for Fourier analysis filtering.

As programming team leader, Joseph was able to leverage his significant wealth of knowledge and expertise in the field of computer science to guide the development and technical design of the software. Additionally, as graduate and faculty liaison, he spearheaded both the proposal of the project and its subsequent realization.

3 Design Description

3.1 Summary of Design

The software capabilities, outlined in Figure 3, provides a comprehensive overview of included features and standard workflow. This being said, the design of the software as a suite of three standalone programs provides a great deal of flexibility to the user. Provided the proper data structure, each program can be independently executed to produce meaningful results. The suite consists of three separate programs: proper orthogonal decomposition, both linear and nonlinear; filtering and vortex identification; and swirling strength computation.

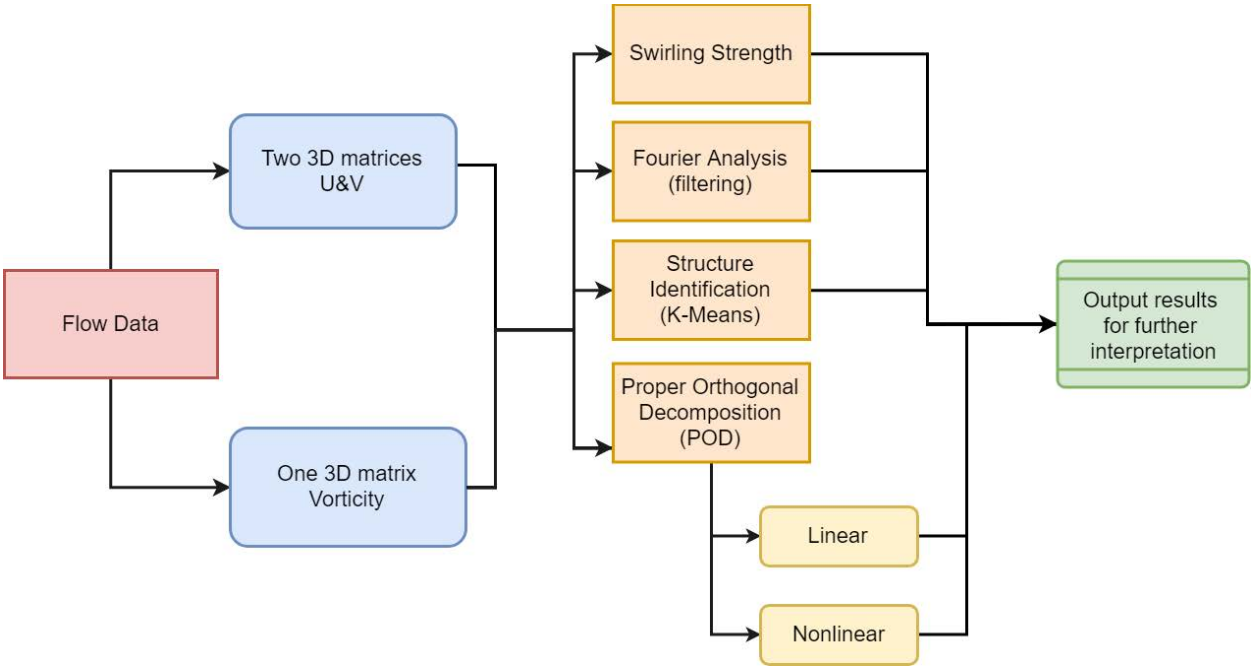

Figure 3: High-level flowchart visualization of design

3.2 Detailed Description

3.2.1 Dataset Compatibility

The software suite is meant to be usable on flow data where the u(x,y,t) and v(x,y,t) matrices represent flow. The intended datasets are either two 3-dimensional matrices (u and v

vectors) or a single 3-dimensional matrix of calculated vorticity data. The specified dataset stem from the origin of the data, the laboratory's stratified jet experiment. The experiment applies particle image velocimetry (PIV) by method of a non-intrusive video capturing setup in collaboration with the DaVis software by LaVision. (McCleney et al. 2016)

The data used is composed of 8000 consecutive frames with the same duration being between each frame. Although the run time of the software is not a restriction given by the customer the data was down-sampled to make certain computations faster during testing. The data was downsampled by a factor of approximately ¼, as shown in Figure 4 below.

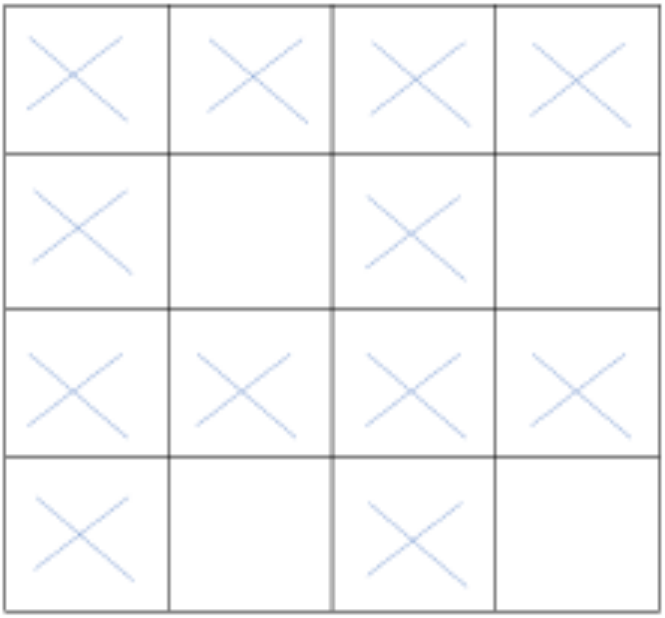

Figure 4: 2D representation of a downsampled data slice; 'X' denotes a removed point

3.2.1.1 Vorticity Conversion

Although the software does not contain a vorticity calculation step, it is still an important part of the software's operation. To convert an existing velocity field to a vorticity field, a basic MATLAB function must be applied. The vector to scalar converter is used with input arguments of curl() or rot() which are the z component of a vorticity field, Figure 5 below shows the symbolic representation. This conversion creates vorticity field, seen in Figure 6 below, in a fast

and verifiably correct manner – an aspect crucial to the design components which utilize vorticity data. The components of vorticity are stored in two arrays in the x and y directions within a greater data structure of vorticity. This mirrors the data structure of the existing velocity field provided by the experiment. The vorticity profile is then plotted by selecting the arrays as their respective axes along an x-y plane.

$$\vec{\omega} = \nabla \times \vec{v}$$

Figure 5: Vorticity, Symbolic Representation

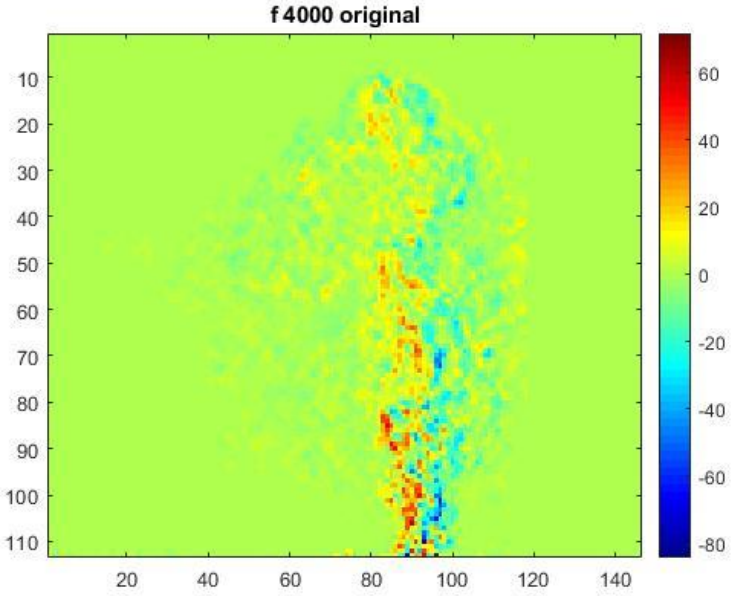

Figure 6: Vorticity of Frame 4000 in s^{-1}.

3.2.2 Swirling Strength

Swirling strength computation, shown symbolically in figure X below, is performed with the intention of obtaining the strength of the swirling motion quantified by the imaginary variable of the eigenvalue. Swirling strength is defined as the imaginary part of the complex eigenvalue pair from a decomposition formula of the velocity gradient tensor. This method simplifies the motion present in the dataset into simple swirling strength values, seen in Figure 8 below. Furthermore, Proper Orthogonal Decomposition can be applied to the resultant dataset. This

method is particularly applicable to the identification of vortices, which is valuable as it means the method can be applied to the raw dataset in a straightforward implementation [André 2017].

$$\boldsymbol{D} = \nabla \boldsymbol{u}$$

$$\boldsymbol{D} \equiv [d_{ij}] = [\boldsymbol{v}_r \;\; \boldsymbol{v}_{cr} \;\; \boldsymbol{v}_{ci}] \begin{bmatrix} \lambda_r & & \\ & \lambda_{cr} & \lambda_{ci} \\ & -\lambda_{ci} & \lambda_{cr} \end{bmatrix} [\boldsymbol{v}_r \;\; \boldsymbol{v}_{cr} \;\; \boldsymbol{v}_{ci}]^{-1}$$

Figure 7: Swirling Strength, Symbolic Procedure

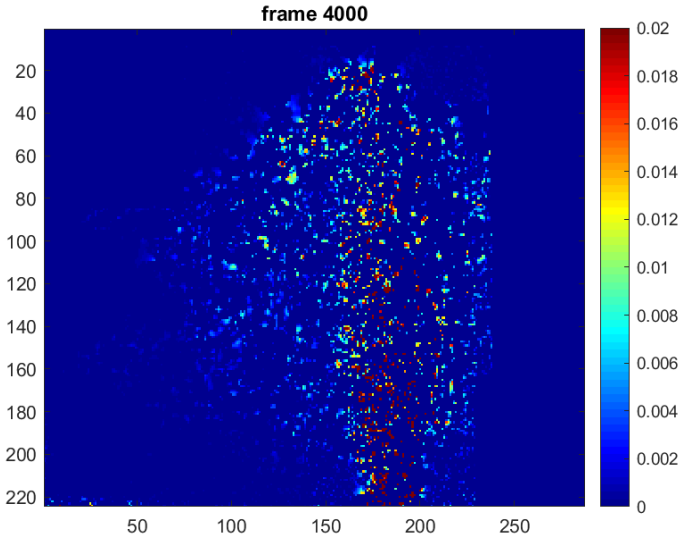

Figure 8: Swirling Strength Visualization

3.2.3 Fourier Analysis Filtering

Fourier analysis filtering is a traditional yet effective method to isolate desired structures, be they dominant or hidden. A fast Fourier transform is applied to the dataset, followed by a user-specified filter. Applying a filter in the Fourier domain is an effective and light weight method of isolating values above or below a certain threshold within the vorticity field. The most

pertinent application is that of a low-pass filter (see Figure X below) as it effectively isolates the major vortices in the field. Examining the variable filter as a key component is useful when trying to understand the process. This simple filter provides a great deal of flexibility in its implementation. Straightforward alterations provide a great deal of control to the user, namely the selection of desired cutoff size and order. The simplicity should not be understated as the functionality provided, relative to its ease of use, means the user does not have to perform intensive micromanagement to yield desired results.

The program preferably uses the vorticity field as an input from which to work, but it is flexible enough to take any vector or scalar field as an input. It applies a Fourier transformation to the vorticity field effectively leaving the field in the Fourier time domain. Before running computational analysis, the user must first specify both the frequency defining the cutoff filter size and whether they want a high-pass or low-pass filter. If the user selects a low pass filter, it effectively filters out the small scales – in this case minor structures and vortices. On the other hand if the user selects a high pass filter it effectively filters out the large scales – in this case the larger vortices. After filtering an inverse Fourier transformation is performed to the vorticity field to bring it back to a regular domain. Figure 9 shows a low pass filter applied to the vorticity at frame 4000.

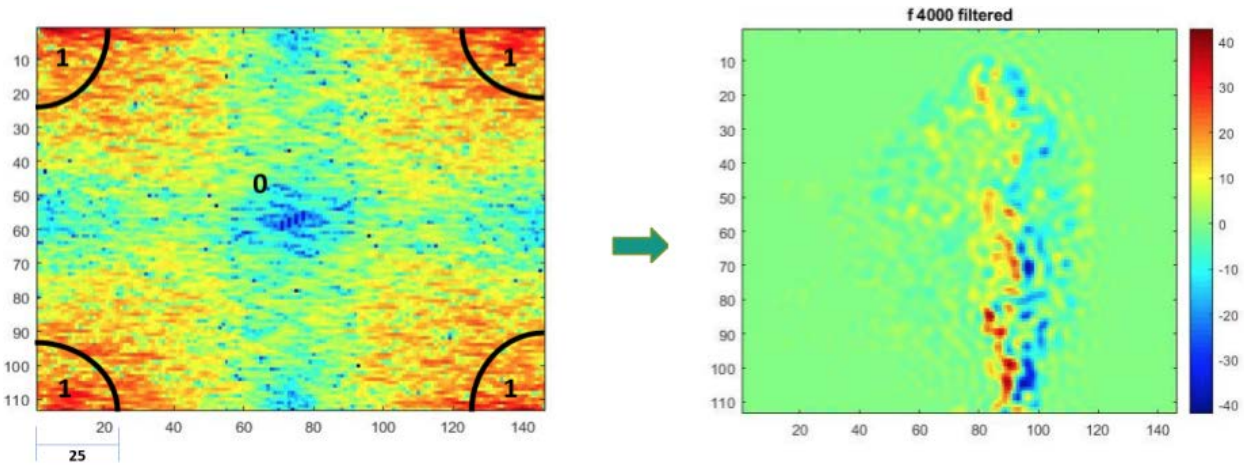

Figure 9: Fourier Analysis Filtering with n=25, where K=1/(n*wavelength).

At this point the important structures are much more visible than before. In Figure 10 above the fast Fourier transform of the same image is shown to illustrate the 25 point cutoff. The dominant structures are further isolated by calculating the absolute value of the field with a cutoff value specified at magnitudes greater than 25 s^{-1}, as seen in Figure X below.

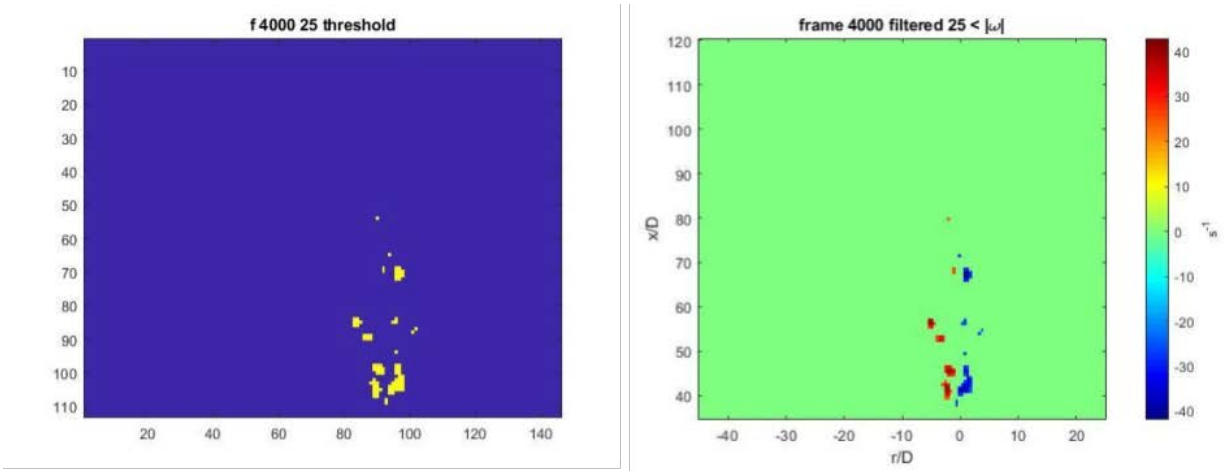

Figure 10: Implementing Threshold where: x < -25 s^{-1} and x > 25 s^{-1}

3.2.4.1 Proper Orthogonal Decomposition

To perform Proper Orthogonal Decomposition the data must first be organized into a standard format. 3-dimensional flow data, as seen in Figure 3, is structured as an m×n matrix U where n is the total number of frames of flow data that exist while m is the number of data points existing in each frame. The correlation matrix $C=U^TU$ is calculated as an n×n matrix. The eigenvalue problem of $Cx=\lambda x$ must then be solved, where λ is a matrix containing n eigenvalues corresponding n eigenvectors in x. The eigenvalues and corresponding eigenvectors are sorted in descending order since a higher eigenvalue correlates to a coherence in the flow.

Nonlinear Proper Orthogonal Decomposition is the application of k-means clustering on the flow data before standard POD. The data can be clustered into two clusters. The data

collection is known to span an amount of time before and after the jet is fired to ensure the entirety of the flow is captured. The flow is separated into 2 clusters to ensure the analysis is only conducted on data collected during the duration of the flow. The data is clustered into 2 sections, flow and no-flow. This strict window begins with the start of the jet and ends when the fluid begins to settle. (Appendix LPCA)

3.2.5 Identification of Significant Structures (k-means clustering)

Clusters, when referring to data and signal analysis, are collections of data points grouped by similar characteristics. K-means clustering requires the user input a certain number of clusters, denoted k, which will then dictate the aggregation of similar data points together into clusters. The specified k is the number of centroids that will define each cluster. Each data point is sorted into its nearest cluster centroid which results in the effective classification of all the similar data points into k number of groups. When applied to the fluid analysis software, the number of clusters becomes dependent on the number of existing vortices. The resultant output provides new intuition towards the identification of patterns and significant structures which are present in the data, but not necessarily obvious upon first glance.

It is considered an iterative algorithm because it alternates between assigning similar data points to k number of clusters and calculating centroids based on the most recent sorting of data points. The assignment of data points to centroids is done by calculating the squared Euclidean distance. On the other hand, the assignment of new centroids is done by calculating the averages, or means, of the previously clustered data points. The cycle repeats until the results of the iterations become relatively constant. That is, the data points stop changing clusters. Due to this, the algorithm optimizes the data points yet does not find the absolute optimum. In application, this means that different runs of the algorithm using the same dataset can produce differ.

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

Figure 11: k-means clustering calculation

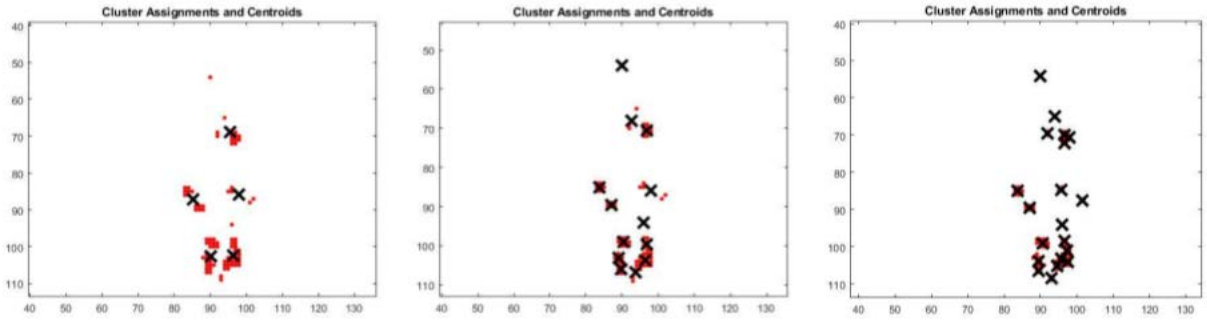

Figure 12.a: k-means clustering where k= 5, 13, and 20

4 Evaluation and Testing

4.1 Summary of Evaluation & Testing

After the software is created implemented, verification of the correctness of the software is done. In programming, verification is done by making sure the output of a program is reliable and consistent (if applicable). In this case since the program's computations are done a lot using built-in functions from Matlab, the credibility is guaranteed. In the case of k-means clustering, despite the k-means built-in function being a part of the Matlab software, the output will not always be exactly the same due to the nature of the algorithm which initially randomly assigns centroids to the data and, through a given number of iterations, adjusts the position of the centroids.

To verify that the output of the combination of commands, or the software as a whole produces the correct results given the following analyses one can simply observe the flow itself to view the recurring patterns and structures. Figure 12 shows a snapshot of the flow at a time, where structures on the top of the vortex plume are also seen in the k=2 cluster label of LPCA in Figure 13. Vortices on the top of the plume seem to be visible, or come to the viewer's attention, after looking at the results from POD (also LPCA). When looking at around y/D=80 it is visible how the flow settles along that level in the tank. This is visible in Figure 13 as well as Figure 12. Proving that k-means in POD also works, is the fact that in Figure 14, with a cluster label of 1, meaning this part corresponds to the no-flow section of the velocity field history, the magnitude of the velocity is much smaller than in the cluster label 2 in Figure 13. Also, in cluster 1 the flow doesn't seem to be shaping in any way especially as the mode number increases, there is almost no correlation between the created velocity field and the actual flow; the

first more might look like it may mean something, however, this is due to the origin of this data. This data was captured using PIV, and cameras used for this were shut off a few seconds after the flow is seen to have ended in order to ensure the entire flow is captured in the acquired images. After the jet stops, the section is considered no-flow, and therefore the particles still moving from the aftermath of the jet are captured and create the jet like structure in the first mode of Figure 14.

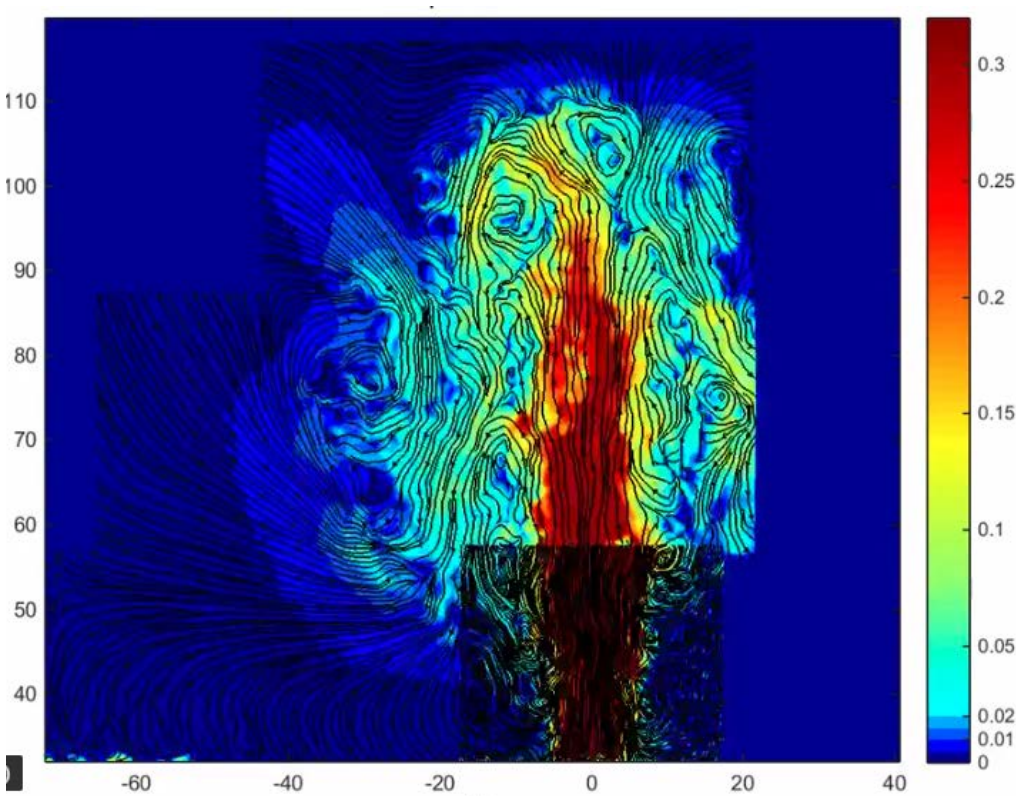

Figure 12.b: Visualization of raw data (Moving Average of Flow)

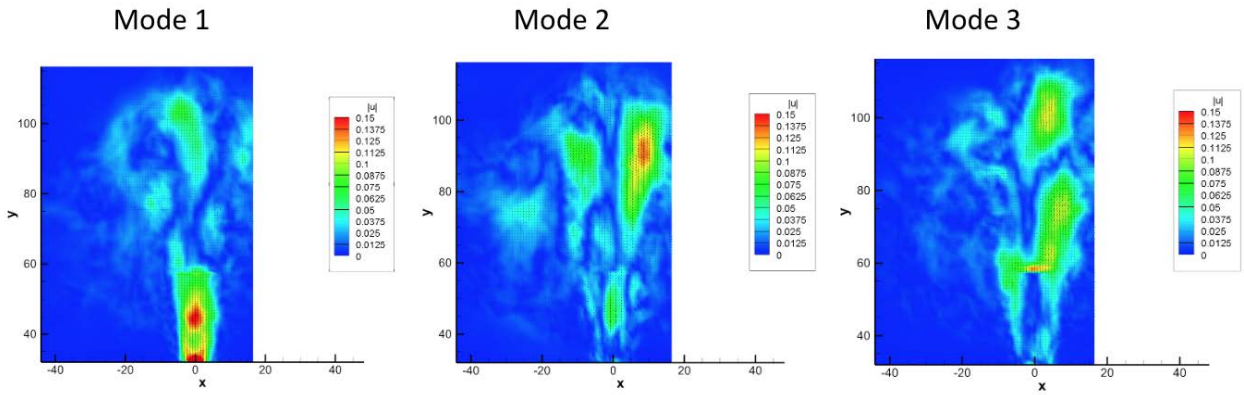

Figure 13: First 3 modes of Local Principal Component Analysis: cluster 2

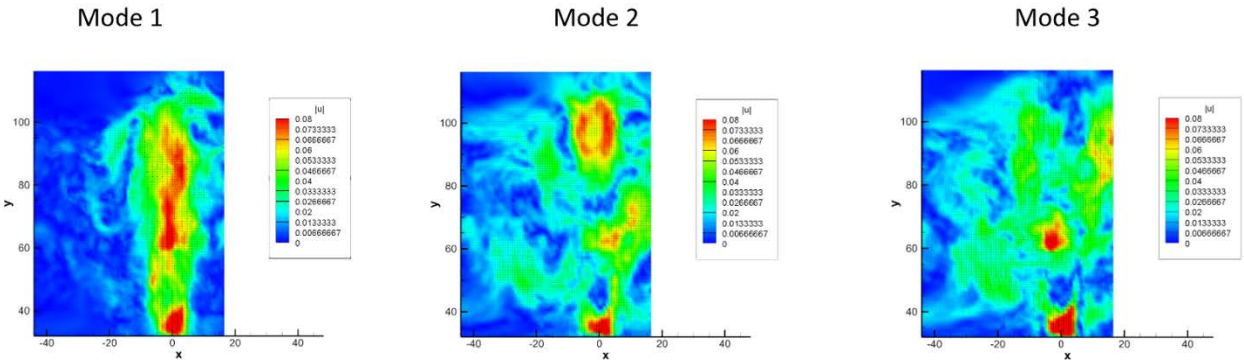

Figure 14: First 3 modes of Local Principal Component Analysis: cluster label 1

5 Summary and Recommendations

The table below provides a visual outline of the technical requirements identified during the primary design phase of the software.

Table: Functional Requirement Status

	Functional Requirements	Status
Dataset Compatibility	Two 3D matrices (U & V)	✓
Automation of Analysis Methods	One 3D matrix (Vorticity)	✓
	Swirling strength	✓
	Fourier Analysis (filtering)	✓
	POD (linear/nonlinear)	✓
	Structure Identification (K-Means)	✓
Verification Method	Comparison to moving average	✓

6 References and Figures

Adrian, R. J., Christensen, K. T., & Liu, Z.-C. (2000). Analysis and interpretation of instantaneous turbulent velocity fields. *Experiments in Fluids*, *29*(3), 275–290.

Adrian, R. J., Meinhart, C. D., & Tomkins, C. D. (2000). Vortex organization in the outer region of the turbulent boundary layer. *Journal of Fluid Mechanics*, *422*, 1–54.

André, M. A., & Bardet, P. M. (2017a). Free surface over a horizontal shear layer: vorticity generation and air entrainment mechanisms. *Journal of Fluid Mechanics*, *813*, 1007–1044.

André, M. A., & Bardet, P. M. (2017b). Free surface over a horizontal shear layer: vorticity generation and air entrainment mechanisms. *Journal of Fluid Mechanics*, *813*, 1007–1044.

Chen, H., Reuss, D. L., Hung, D. L. S., & Sick, V. (2013). A practical guide for using proper orthogonal decomposition in engine research. *International Journal of Engine Research*, *14*(4), 307–319.

Diez, M., Campana, E. F., & Stern, F. (2015). Design-space dimensionality reduction in shape optimization by Karhunen–Loève expansion. *Computer Methods in Applied Mechanics and Engineering*, *283*, 1525–1544.

Farhat, C., & Amsallem, D. (n.d.). CME 345: MODEL REDUCTION - Proper Orthogonal Decomposition (POD). *CME 345 - Stanford University*. Retrieved from https://web.stanford.edu/group/frg/course_work/CME345/CA-CME345-Ch4.pdf

McCleney, A. B. (2016, January 1). Turbulence Statistics of a Buoyant Jet in a Stratified Environment. *ProQuest Dissertations And Theses; Thesis (Ph.D.)--The George Washington University, 2016.; Publication Number: AAT 10001631; ISBN: 9781339415635; Source: Dissertation Abstracts International, Volume: 77-06(E), Section: B.; 249 P.* Retrieved from https://ui.adsabs.harvard.edu/abs/2016PhDT.....5M

Meyer, K. E., Pedersen, J. M., Özcan, O., Haider, S., & Larsen, P. S. (2008). Identify flow structures with proper orthogonal decomposition (POD). *Departement of Mechanical Engineering Technical University of Danmark (DTU)*.

Wenguo, W., Weicheng, F., Guangxuan, L., & Jun, Q. (2001). An improved cross-correlation method for (digital) particle image velocimetry. *Acta Mechanica Sinica*, *17*(4), 332–339.

Zhou, J., Adrian, R. J., Balachandar, S., & Kendall, T. M. (1999). Mechanisms for generating coherent packets of hairpin vortices in channel flow. *Journal of Fluid Mechanics*, *387*, 353–396.

7 Appendix

Appendix: Additional data on POD (PCA)

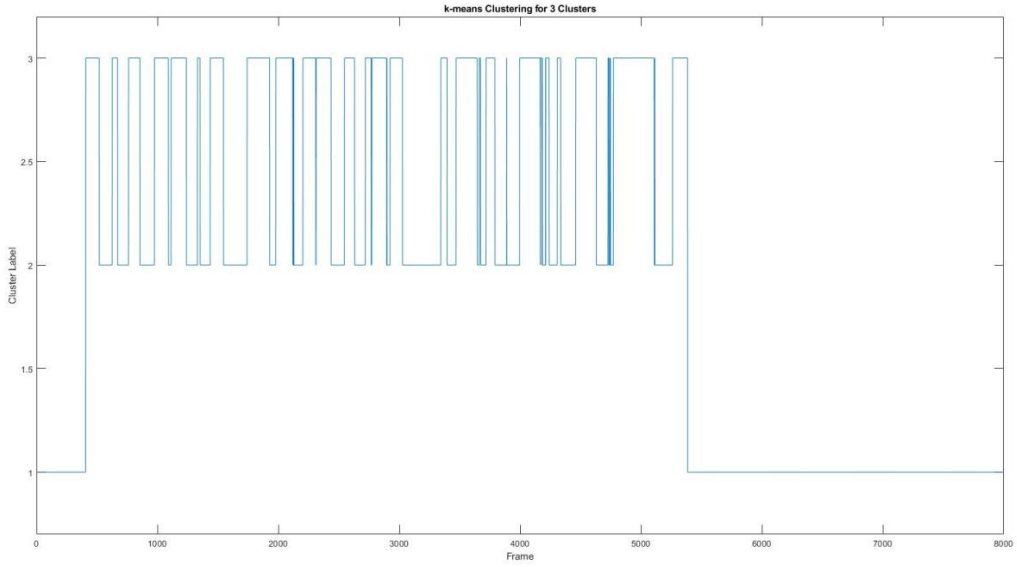

Figure 15: k-means clustering label separation with cluster label vs. frame (time), k= 3

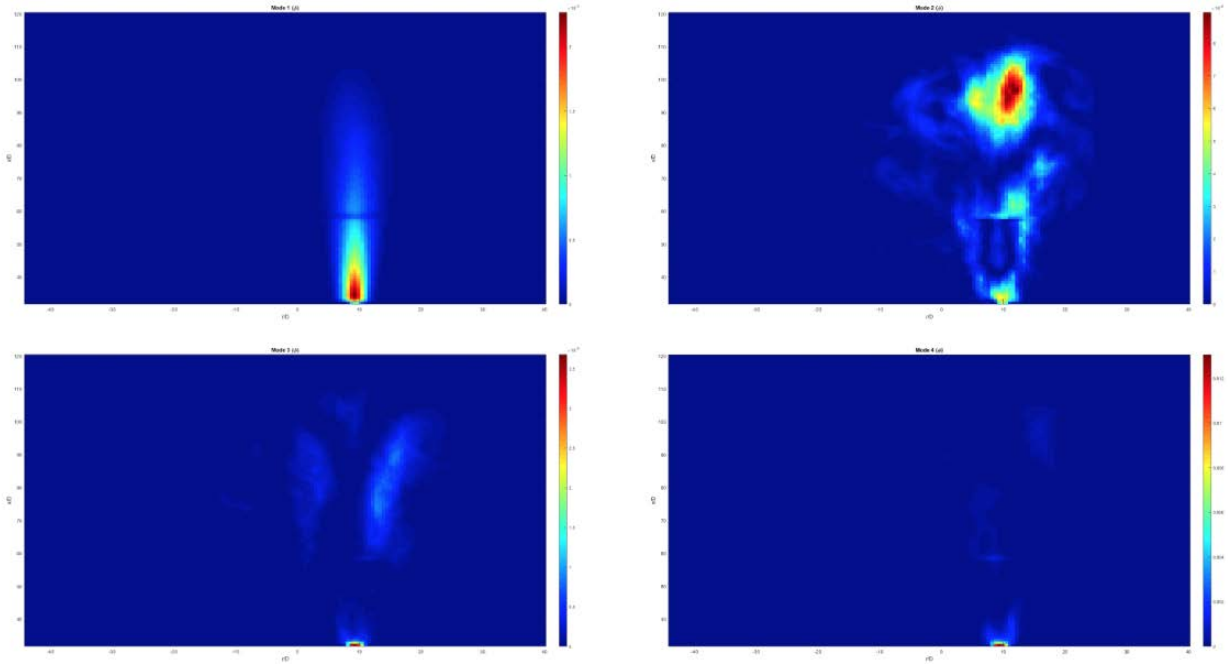

Figure 16: Local Principal Component Analysis, k= 1 results