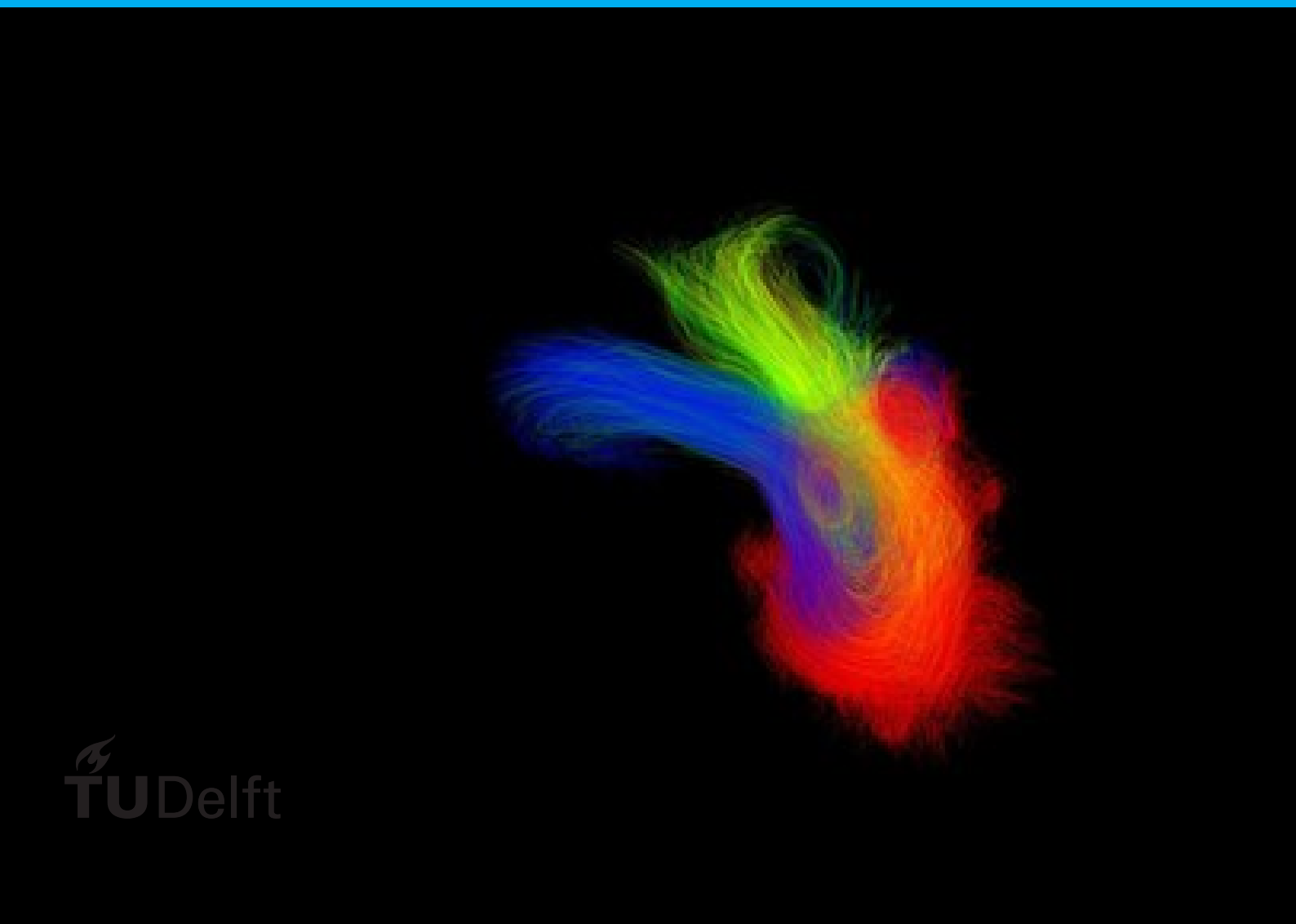
Stochastic Neighbor Embedding for interactive visualization of flow patterns in 4D flow MRI

Master Thesis

Mitchell Martijn de Boer

Advised by
Prof.dr.ir. B.P.F Lelieveldt



TU Delft

Stochastic Neighbor Embedding for interactive visualization of flow patterns in 4D flow MRI

Master Thesis

by

Mitchell Martijn de Boer

in partial fulfilment of the requirements of the degree of
Master of Science in Biomedical Engineering track Medical Physics
at the Delft University of Technology,
to be defended publicly on Monday April 15, 2024 at 10:45 AM.

Student number: 5184193
Thesis advisor: Prof.dr.ir. B.P.F Lelieveldt
Thesis committee: Prof.dr.ir. B.P.F Lelieveldt(chair); Dr. F.M. Vos; Dr.ir R.J. van der Geest

Contents

Abstract

Flow visualization is an important topic in many scientific domains and has been an active field of research for many years. Many different methods of analysis can be used in order to analyze flow, however recently big progress have been reported on the manifold learning algorithms for high-dimensional data. This thesis investigates the use of Stochastic Neighbor Embedding (SNE) methods, t-distributed stochastic Neighbor embedding(t-SNE) and hierarchical stochastic Neighbor embedding (HSNE) for flow analysis. In this thesis the Manivault (Vieth et al., 2024) software platform has been used in order to create an interactive analysis tool for SNE methods used on flow data. This tool consists of a 3D viewer plugin that visualizes the full pathlines and an existing scatterplot plugin that is used in order to interact with the created SNE maps. The experiments and comparisons reported in this thesis aimed to compare the use of t-SNE and HSNE for analysis of flow structures in 4D Flow MRI data. From this, it can be concluded both tSNE and HSNE are useful for interaction with and analysis of 4D Flow MRI data. t-SNE can best be used in order to explore and analyze flow data in search for flow structures, and comparing flow patterns between subjects. HSNE on the other hand gives a better separation between different flow components, however at the expense of a less accurate preservation of vortices and other flow structures.

Keywords: Manifold Learing, Flow visualization, T-SNE, H-SNE, Fluid Flow Analysis, Flow Stuctures

1 Introduction

Flow visualization is an important topic in many scientific domains and has been an active field of research for many years. It can for instance be used in the aerospace sector to study the air flow around airplanes and improve their aerodynamics (Wu et al., 2017), in meteorology to visualize atmospheric flow patterns to predict the weather and in radiology to visualize blood flow patters to improve diagnostic procedures for different heart conditions.(Vasanawala et al., 2015)

The data used to visualize flow is often represented as a vector field, which is a multidimensional Euclidean space, usually consisting of an x, y and z position value, where each point in that space represents a vector encoding the flow velocity components in that specific position. This vector contains a value in the x, y and z direction which represents a velocity, or some kind of other directional vector. These values can also be gathered over multiple time points in order to analyze the changes in flow over that period, or the evolution of a specific flow structure that could for instance appear or disappear over time. The use of these vector fields in flow visualization allows the tracking of particle movement from point A to point B over a period of time.

1.1. Flow visualization methods

The visualization of the fluid flow can be performed using varying flow visualization methods which can be divided into two groups: direct and indirect flow visualization. (Hölt, 2021) Direct flow visualization uses the data points of the vector field directly to visualize the flow, whereas indirect flow visualization first computes a derived flow representation from the data that is subsequently displayed. There are 2 main methods for direct flow visualization. The first method is called an Arrow/Hedgehog plot (Fig. 1.1 left). This plot contains arrows that visualize the direction of the flow in each data point of the vector field. This visualization method gives an overview of the entire fluid flow, but it can become very cluttered due to the amount of arrows that are shown. Other direct visualization methods are smearing techniques (Fig. 1.1 right). These methods combine random noise and the vector field in order to create a pattern that represents the flow inside the vector field. An example of a smearing technique is Line Integral Convolution (LIC). (Cabral and Leedom, 1993) LIC blurs the random noise tangential to the vector field which causes lines to appear which are representative for the fluid flow inside the vector field.

Indirect flow visualization tends to create a line for specific particles in the data which represents the flow based on selected starting points. Examples of indirect flow visualizations are Streamlines, Path lines, Streak lines and Timelines (Figure 1.2). A streamline visualization looks at the flow in a specific time point. It therefore freezes the vector field on that time point and then follows a massless particle that would be released at a specific point in this frozen vector field. The path that this particle takes through the vector field is called the streamline. Path lines are relatively similar to streamlines, however with path lines time is not frozen and a particle is followed over a vector field which can change over time. If the vector field is constant however the path line will be identical to the streamline. A streak line is created when massless particles are released in a vector field at the same point in space but at different time points. A line which connects these time points is called the streak line. Finally a timeline is the reverse of a streak line. A timeline is created when massless particles are realised at the same

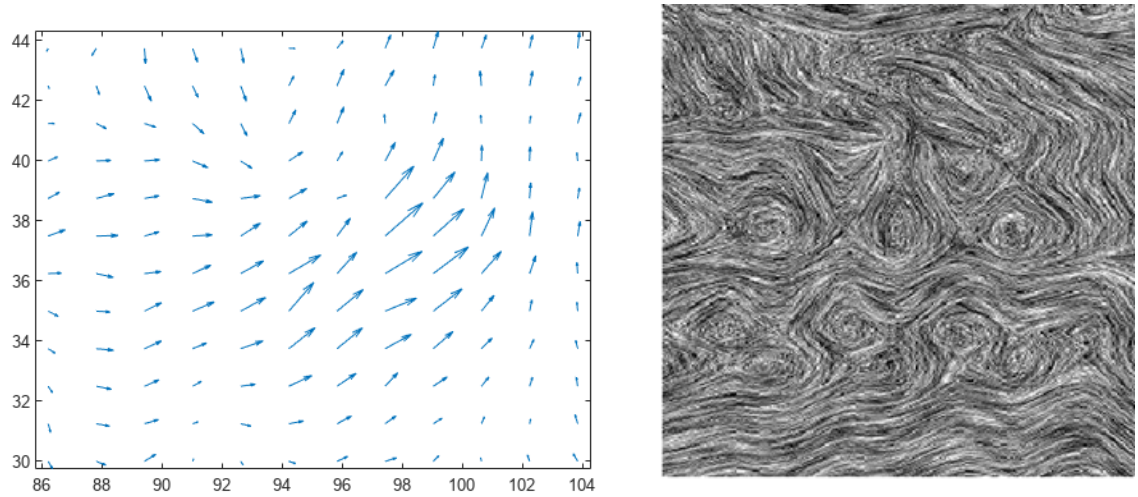


Figure 1.1: This figure shows direct vector field visualization methods. The left image shows an arrow or hedgehog plot and the right image shows a smearing technique called Line Integral Convolution.
("Quiver", 2022;Cabral and Leedom, 1993)

point in time but a different point in space. The line connecting these points is called the timeline.

1.2. Conventional flow analysis methods

The data visualization methods explained in Section 1 are mostly used to create a visual representation of the data, while more quantitative analysis can be performed on these visualizations as well. The aim of quantitative analysis of the flow data over just visualizing the data is that analysis of the data can discover underlying flow structures which are not always obvious in the visualization. In literature many different methods are traditionally used for fluid flow analysis and flow structure discovery, examples of these methods are Principle Component Analysis (PCA) (Jolliffe, 2002), the lambda 2 method Jeong and Hussain, 1995 and the Finite Time Liapunov Exponents (FTLE) (Shadden et al., 2005).

1.2.1. Principle Component Analysis (PCA)

The first conventional flow analysis method that we will discuss is PCA (Jolliffe, 2002), which is also known as Proper Orthogonal Decomposition (POD) in the fluid dynamics field. This is a well known method of dimensionality reduction that can be used in a wide range off applications outside of flow analysis as well. PCA aims to extract the most important information from a high dimensional data set and convert is it into a lower number of dimensions in order make it to easier analyze and visualize the data. This is done by using the covariance matrix of the data to determine the Eigenvectors and Eigenvalues. The Eigenvectors that correspond to low Eigenvalues are discarded and the higher Eigenvalue dimensions are used to create a representation of the data in the dimensions of the remaining Eigenvectors (also known as principal axes). Creating a representation of the data in this way causes data points which have a similar composition, like fluid direction or cell type, to cluster together. The decomposition of the data into multiple Eigenvectors creates the opportunity to dissect for instance a wave pattern into its individual components, as seen in Figure 1.3. Although PCA is a very efficient and simple algorithm, the major drawback is that it performs a linear transformation on the data into the new principal axis. This means that PCA is not able to deal with non-linear structures like vortexes very well.

1.2.2. Lambda 2 method

Second, the Lambda 2 method is a quantitative detection method that uses the fluid dynamics definition of a vortex structure in order to discover vortex structures in flow data rather than depending on visualization techniques. (ElBaz et al., 2013) The aim of the method is to identify vortexes inside a vector field using the data itself and is considered to be the most accepted method in performing this

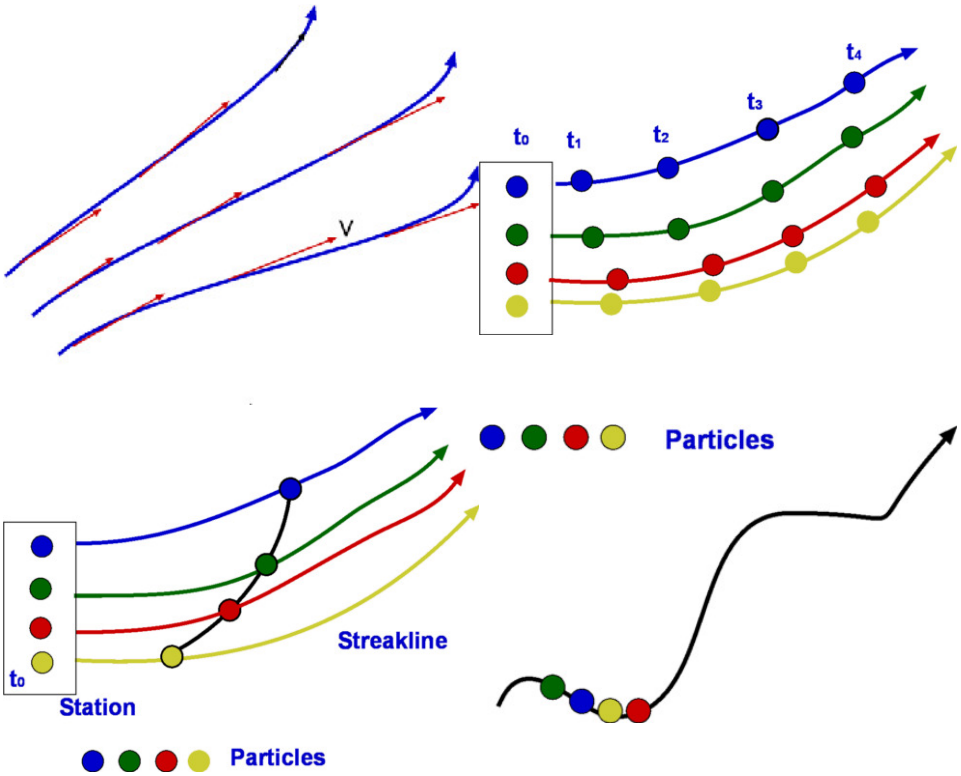


Figure 1.2: Four methods for indirect vector field visualization. The top left image is a representation of a streamline, each point on the streamline is tangent to the vector at that specific point in the vector field. The top right of the image is a representation of a path line, a mass-less particle is followed through the vector field which changes over time. The bottom left image shows a streak line connecting particles released at the same time, and the bottom right image shows a timeline.
("Flow description, Streamline, Pathline, Streakline and timeline", 2005)

task. (Kheradvar and Pedrizzetti, 2012) The lambda 2 method requires the three velocity components, x- y- and z-direction, as input. Using these components the velocity gradient tensor is computed which will be decomposed into its symmetric part, the strain deformation tensor **S**, and its asymmetric part, the spin tensor Ω. This step is followed up by an Eigenvalue analysis on the sum $\mathbf{S}^2 + \Omega^2$. Finally a point in a vector field is part of a vortex structure when at least 2 of its Eigenvalues are below zero i.e. if λ_1, λ_2 and λ_3 are the Eigenvalues and $\lambda_1 \geq \lambda_2 \geq \lambda_3$, then a point is part of the vortex structure if $\lambda_2 < 0$. In practice this cutoff creates issues due to noise interference, therefore a threshold $T_\lambda 2 < 0$ is used. (ElBaz et al., 2013) When the method is applied over the entire vector field the vortex structures can be defined as neighboring points that have been determined to be part of a vortex, this can for instance be done using the Connected Component Analysis. (Haralick and Shapiro, 1992) A major advantage to this method is that visualization of the data is not required before the analysis is performed, making this method faster than other methods that identify vortex structures. However the lambda 2 method requires subjective human input to determine where the cutt-off for something being part of a vortex, $T_\lambda 2 < 0$, needs to be placed.

1.2.3. Lagrangian Coherent Structures and Finite Time Lyapunov Exponents

Lagrangian Coherent structures are surfaces of trajectories in dynamical systems that have a big influence on other trajectories that are close to the surface over time. The use of this method was proposed by Haller, 2005 and studied by among others Shadden et al., 2005. LCS aims to identify coherent structures based on flow data. This way structures like vortexes can be identified without any subjective input (Haller, 2005) like needed for methods like Lambda 2 (Attiya et al., 2018). The definition of a vortex can be identified with an LCS as a set of fluid particles along which the strain acceleration tensor is indefinite over directions of zero strain. (Haller, 2005) In other words, the vortexes are defined as tubes in which particles do not adhere to directions suggested by the strain eigenvectors. Advantages of the use of LCS are that coherent structures that cannot be identified with Euler based methods

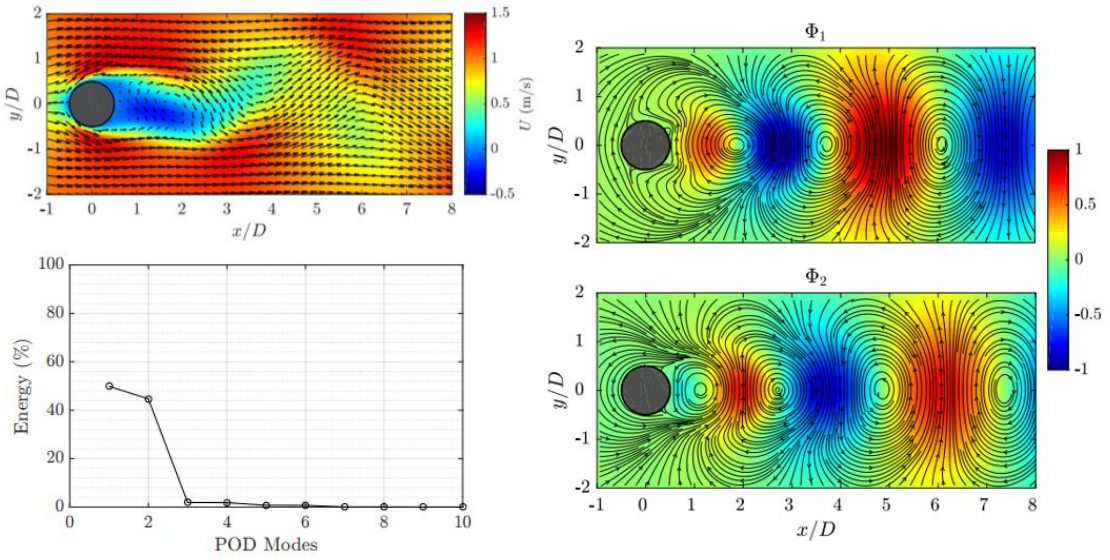


Figure 1.3: This figure demonstrates a POD for a given flow pattern(top left image). This flow pattern is decomposed into its Eigenvectors, or POD modes. The contribution of the 10 POD modes with the highest Eigenvalues, or Energy, can be seen in the bottom left image. Finally the top two POD modes can be seen in the right images. Images obtained from Julien Weiss, A tutorial on proper orthogonal decomposition.

(Julien, 2019)

can be identified using LCS. (Attiya et al., 2018) And no subjective input is necessary to identify these structures. Attiya et al., 2018; Haller, 2005

These LCSs can be detected using methods like Finite Time Lyapunov Exponents (FTLE) (Shadden et al., 2005). FTLE is a function which changes over time and space, therefore it is often used in a field (Figure 1.5) in order to create a visualization. An FTLE field is a scalar field that cauterizes the maximum degree of separation over the trajectory of a specific fluid particle over time (Attiya et al., 2018; “LCS tutorial: The finite-time Lyapunov exponent”, 2005). LCSs can be characterized as ridges in these FTLE fields. (Shadden et al., 2005)

Also the FTLE method has a high capability to detect unsteady separation profiles in chaotic flow. (Shadden et al., 2005) However this method is Computationally very expensive and will therefore be slow. (Attiya et al., 2018)

1.3. Manifold Learning

Other methods that can be interesting to use for the analysis of fluid flow and the recognition of flow structures are methods based on manifold learning. Manifold learning is a machine learning framework that uses raw high dimensional data and is able to extract low dimensional structures that are situated in the high dimensional data. It does this by using a calculation that determines the similarity between data points. Contrary to linear algorithms like PCA, which also aim to do extract structures from high dimensional data, manifold learning is able to deal with non-linear structures. Because manifold learning does not directly create a visualization in the vector field but rather creates a map of data points that have similar qualities, the map is suitable for user interaction. Combined with a visualization method, they may enable interaction with indirect flow visualization primitives (streamlines, path lines). Although pioneering work in flow visualization has been done with older versions of manifold learning by for instance Tauro et al., 2014, who used Isometric feature Mapping(IsoMap) (Tenenbaum et al., 2000) to analyze flow data, many big improvements are made in the field of manifold learning by the creation of Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2018), t-distributed Stochastic Neighbor Embedding (T-SNE) (Van der Maaten and Hinton, 2008) and Hierarchical Stochastic Neighbor Embedding (HSNE) (Pezzotti et al., 2016).

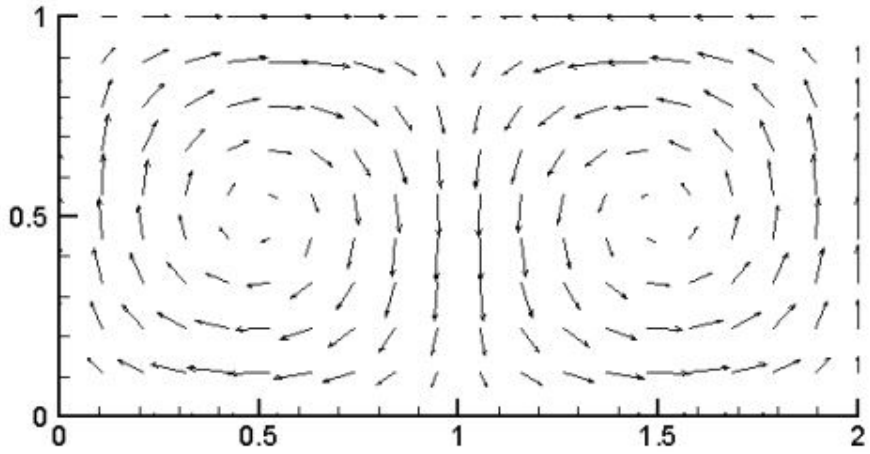


Figure 1.4: This figure shows a vector field that is used to create an FTLE field in figure 1.5. This figure was obtained from. ("LCS tutorial: The finite-time Lyapunov exponent", 2005)

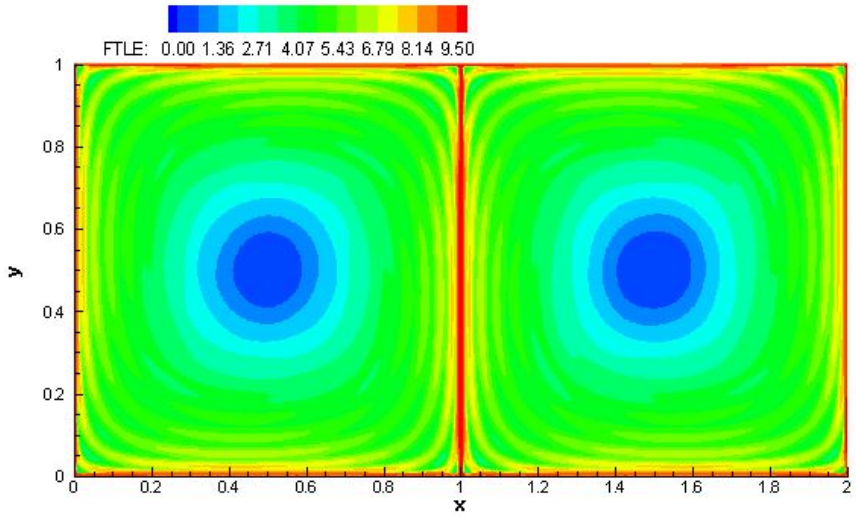


Figure 1.5: This figure shows an FTLE field which was created using the vector field depicted in 1.4. This figure was obtained from.

("LCS tutorial: The finite-time Lyapunov exponent", 2005)

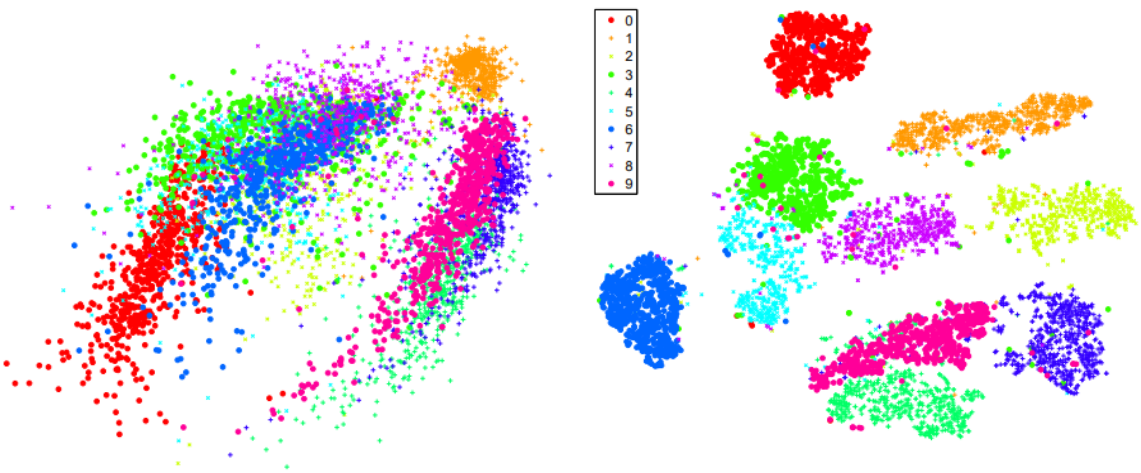


Figure 1.6: This figure shows an embedding created by Isomap(left) and T-SNE(right) on the same data set. As can be seen in the two images T-SNE creates a clearer clusters of the different data groups than IsoMap. When using these methods for flow visualization analysis the groups could be different flow patterns or fluid directions.
(Van der Maaten and Hinton, 2008)

1.3.1. IsoMap

A dated manifold learning technique that has already been used for analyzing flow data by Tauro et al., 2014 is IsoMap (Tenenbaum et al., 2000). IsoMap can directly identify relevant features from data without intermediate processing being necessary. (Tenenbaum et al., 2000) It does this by using geodesic distance, which is defined as the sum of weights along the shortest path between two nodes, in combination with euclidean distance in the Neighbor graph in order to incorporate the similarity between data points. Like PCA, IsoMap is a fairly simple method that does not take a lot of computing power, making analysis quick. Unlike PCA, IsoMap can identify non-linear structures because it does not rely on a linear transformation. This is a clear advantage of IsoMap over PCA. A drawback of this manifold learning technique is that it is very dependent on image quality. (Tauro et al., 2014) It, for instance, does not handle noise, non-uniform background, low contrast or poor resolution very well. Finally, as seen in Figure 1.6 left, IsoMap's readability is not very good.

1.3.2. U-MAP

U-MAP (McInnes et al., 2018) is another Manifold learning method. It uses a fuzzy high-dimensional data graph which accurately represents the high-dimensional data. Based on this fuzzy graph the weights between the edges of the data points are calculated. These weights are then used in order to compress the fuzzy graph into a lower dimensional graph which represent the higher dimensional data. The use of a fuzzy graph is beneficial because it enables the algorithm to compress the entire graph at the same time, rather than transferring points from high dimensional space to low dimensional space one by one. This means that the U-MAP algorithm is more computationally efficient than algorithms which do transfer points one by one. (McInnes et al., 2018; Tran, 2022) Another advantage of this algorithm are that it has a good balance between being able to create a good representation of the overall data due to fuzzy graph that is created based on the high dimensional data, while still being able to create a good representation of the similarity between individual points. McInnes et al., 2018; Tran, 2022 Some drawbacks of U-MAP are that it lacks the readability provided by algorithms like PCA (Jolliffe, 2002), because the dimensions of the U-MAP embedding have no meaning, while the PCA dimensions represent the dimension of the highest variance in the data. (McInnes et al., 2018) And finally U-MAP can experience problems when used on smaller data sets, due to inaccuracies in approximations of the nearest Neighbor algorithms and negative sampling, which are used to increase the computational efficiency. (McInnes et al., 2018)

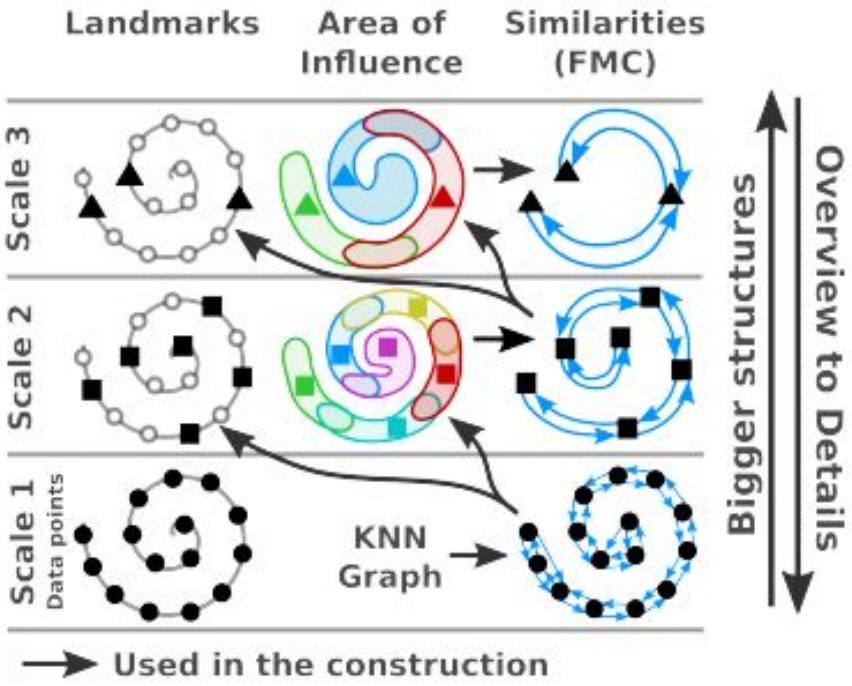


Figure 1.7: This figure shows a visual representation of how the hsne algoritm generates its embedding. (Pezzotti et al., 2016)

1.3.3. T-SNE

A third manifold learning technique is T-SNE (Van der Maaten and Hinton, 2008) which is a variation on Stochastic Neighbor Embedding (SNE) (Hinton and Roweis, 2002). SNE converts high dimensional Euclidean distance between data points into conditional probabilities that represent the similarity between them. (Van der Maaten and Hinton, 2008) It accomplishes this by placing a Gaussian function at point A and looking at the probability density of point B for it being the neighbor of point A. T-SNE differs from SNE in the fact that it does not use a Gaussian function in low dimensional space in order to calculate the similarity, but rather uses a student-t distribution in low dimensional space. The reason why T-SNE uses a t-distribution rather than the Gaussian used by SNE is that SNE suffers from a crowding problem which causes data points to group up in the middle of the visualization. T-SNE has some advantages over methods like IsoMap. One of these advantages is that T-SNE tends to extract clustered groups of similar data, rather than creating a continuous low dimensional manifold. (Figure 1.6) (Van der Maaten and Hinton, 2008) It is also less dependent on image data quality. Some drawbacks of T-SNE are that, first off all, it is computationally expensive, however PCA can be incorporated in the algorithm to increase computation speed and preserve the global structure. (Van der Maaten and Hinton, 2008) Secondly T-SNE uses local properties of the data in order to reduce the dimensional, which becomes a problem with high intrinsic dimensionality and a high variance in the underlying data. This can cause the created representation to be inaccurate. (Van der Maaten and Hinton, 2008) Finally due to the fact that the cost function is not convex like in other algorithms, IsoMap for instance, it is possible for T-SNE to converge on different visualizations each run. This is caused by the initial optimization parameters and the random initial configuration of map points. (Van der Maaten and Hinton, 2008)

1.3.4. H-SNE

H-SNE is a development on T-SNE and tries to bridge the gap between computationally expensive but highly accurate techniques developed by the machine learning community and the computationally cheap but inaccurately techniques that allow for interactivity developed by the visualization community. (Pezzotti et al., 2016) So the aim of the algorithm is to create an accurate low dimensional representation while still allowing for real time interaction with the data. The Method is built up out of multiple

layers, each having its own landmarks and representing more and more detail the farther down the layers you go. (Pezzotti et al., 2016) The algorithm works as follows, first with the high dimensional data as input a k-nearest neighbor graph is created which can be used to build a Finite Markov Chain (FMC) which encodes the similarities between the points. This FMC can then be used in order to compute the landmarks for a higher layer and their area of influence. This area of influence can then be used to build a new FMC which encodes the similarities in the current layer. This process can be repeated until only a few landmarks are left. (Figure 1.7) Advantages of H-SNE over T-SNE are that the separation between manifolds is better. H-SNE is also less likely to converge in a local minimum that would cause the embedding to be sub optimal. And it is also a lot faster than T-SNE. However where T-SNE is able to preserve the global structure somewhat with the use of PCA, this is not possible for H-SNE.

2

Research Questions

Chapter 1 summarizes recent developments in dimensionality reduction and manifold learning for high-dimensional datasets in general. These innovations have enabled analysis of much bigger datasets than the conventional dimensionality reduction methods and are able to deal with non-linear structures better. Therefore manifold learning techniques such as SNE methods may be an attractive alternative for flow structure analysis and comparison across subjects. Pioneering work in the field of manifold learning for flow structure analysis has already been performed by Tauro et al., 2014 using IsoMap to analyze raw video data of fluid flow. However, due to the many improvements made over the past few years in the field of manifold learning techniques, by the creation of more accurate and efficient Neighbor preservation methods in methods like T-SNE and H-SNE, the use of manifold learning for the use case of fluid flow should be revisited. The overall aim of this MSc thesis is to investigate whether Stochastic Neighbor Embedding methods can be used for interactive 4D flow structure analysis and comparison between subjects. In this pilot study, the following research questions will be specifically addressed:

- In order to use the SNE methods a data feature needs to be selected that will be used in order to generate the SNE analysis. What data feature should be used in order to create the SNE analysis?
- The flow structure and dynamics in intra-cardiac blood flow follows a relatively stable pattern across subjects; To what extent can SNE methods be used to provide an overview of the dynamic flow structure of 4D flow MRI.
- Can SNE methods be deployed to enable user-friendly interaction with complex 4D flow patterns?
- Can SNE methods be deployed to isolate and compare similar flow structures and events across multiple subjects?
- With the generation of pathlines, some of the particles follow an invalid path, this is usually identified during the generation of these pathlines. Is it possible to deploy SNE methods for data quality control by identification and curation of invalid pathlines?
- Finally, because HSNE uses landmarks in order to create its embedding, whether or not the HSNE landmarks can be used in order to create a simplified, but accurate, representation of the entire flow system by reducing the amount of flow lines will be investigated.

3

Methods

3.1. Software Platform

To investigate the effectiveness of t-SNE and HSNE for fluid flow visualization and analysis, a plugin based software platform named ManiVault was used. (Vieth et al., 2024) In ManiVault, (developed by the LUMC LKEB research group and the Computer Graphics and Visualization group at TU Delft), interactive, linked analysis of high-dimensional data can be performed. For the analysis of 4D flow patterns in MRI, a dedicated plugin was developed to address the research questions in Chapter 3. The benefit of using ManiVault over other visualization programs, like Paraview (Ahrens et al., 2005), is that analysis (SNE computation), 3D visualization and user interaction, in embeddings and with the data, can be done very efficiently, making the exploration of the data analysis easier.

3.2. Plugin development for 4D flow analysis

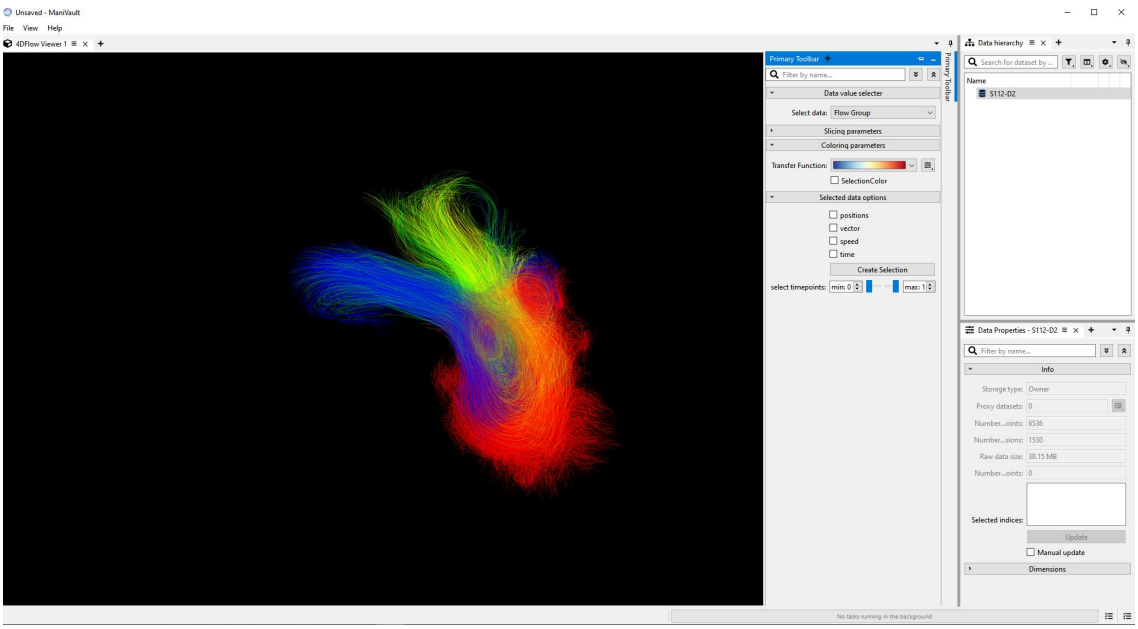


Figure 3.1: Screenshot of the 4D flow analysis plugin created in ManiVault. The right panel shows the data hierarchy, whereas the left panel shows the 3D view of the path line data, colored by flow type. Mouse dragging on the 3D view rotates the view, enabling 3D inspection of the flow structure.

ManiVault already contains several plugins that can be readily used to answer the research questions set out in Chapter 2, in particular the Scatter Plot view, and the t-SNE and HSNE analysis plugins. However, ManiVault did not yet provide facilities for data importing and 3D viewing and exploration, as

required for 3D flow analysis. Therefore, two tailor-made plugins were created for the purpose of 4D flow pattern analysis: 1) a data reader plugin for VTK files that enables loading path line data from VTK files, and 2) a viewer plugin that uses the VTK tool set (Schroeder et al., 2006) and openGL in order to create a 3-dimensional representation of the data. The view was created from a fork of a 3D viewer for high dimensional data that was created as part of my internship at LKEB, a copy of the internship report is included after the appendix. The 4D flow plugin uses datasets where each data point is a vector of 3D point coordinates along a path line for different moments in time. These path lines are then reconstructed in the plugin and transferred to the viewer. Doing this results in a 3D representation of the paths of particles over time. (Figure 3.1)

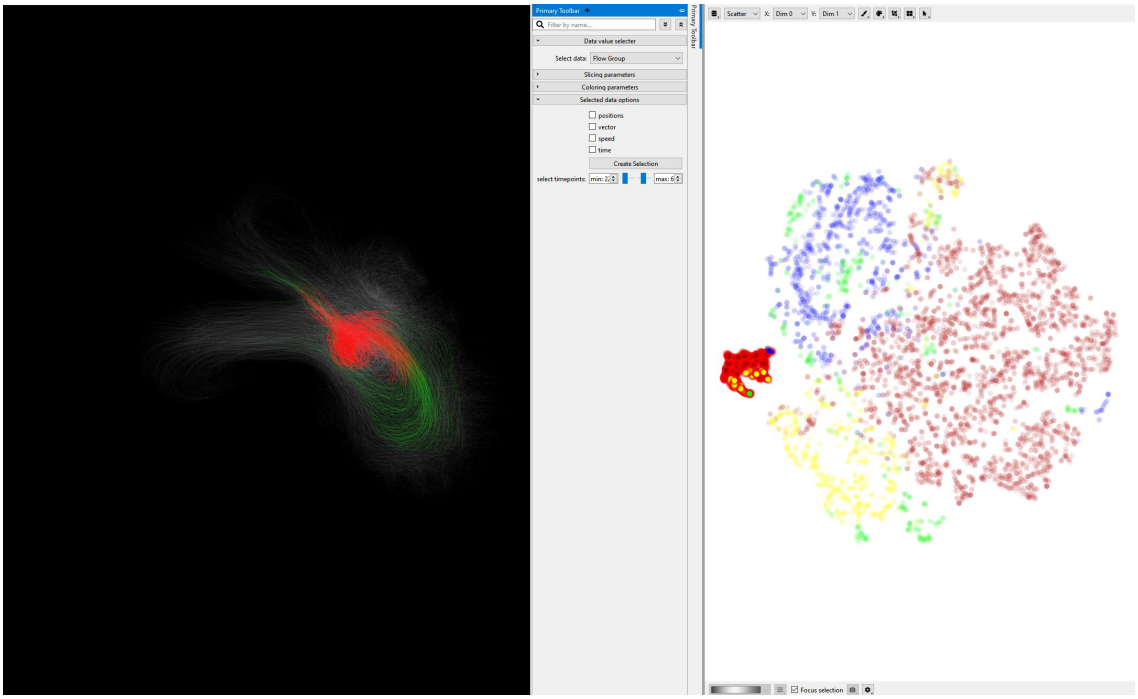


Figure 3.2: Screenshot of the feature created to more specifically select specific flow structures. In this case using the t-SNE analysis depicted on the right a vortex was identified and selected. With the slider selection time-points were then selected in which that time vortex actually appears in this path line. The selected time-points are shown in red, the rest of the path line is shown in green.

To actually use the 3D viewer for 4D flow analysis, extra functionality had to be added. One of these features was the ability to make sure that selection of data in a scatter plot is also represented in the 3D viewer. This is important in order to visualize clusters created by the manifold learning techniques. An example of this can be seen in Figure 4.1. By default the selected data is visualized in red and the not selected path lines are more gray and transparent. However an option has been implemented in order to maintain the coloration of the selected data. Further more a method has been created to isolate time point intervals on a path line to specifically select a specific flow structure. This feature is demonstrated in Figure 3.2. This could be used to more specifically cluster data for manifold learning based on only a specific flow feature, however this has not been used in this thesis.

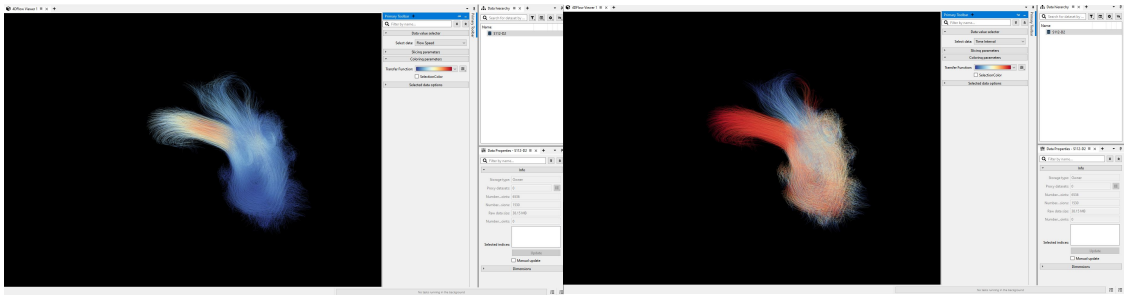


Figure 3.3: This figure shows examples of different data used for path line coloration. In the left image the scalar flow velocity has been used while the right image uses the time point as coloration.

The data coloration in the 3D viewer by default uses the flow component data of the specific path line, where the direct flow has been depicted with green, the retained inflow with yellow, the delayed outflow with blue and the residual flow with red.(Figure 3.1) The coloration however can be changed to represent other data features on points along the path line. (Figure 3.3) The color map can also be changed by changing the color map to other supplied color maps, or by creating a custom color map.

3.3. Data Sets

The data used to test t-SNE and HSNE for data visualization was 4D flow MRI data of the heart generated using cardiovascular magnetic resonance four-dimensional flow assessment. The datasets were collected from a larger study conducted at the University of Leeds looking into the impact of a myocardial infarction in the haemodynamics of the left ventricle. (Garg et al., 2018) For that study forty-eight patients were scanned as well as 20 healthy control subjects. The data used in this thesis is obtained from 11 post-infarct patients. The data consists of path lines, generated by tracing particles for which the seeding was initialized within the LV cavity at end-diastole and tracing is performed in forward and backward direction in time until end-systole. (Calkoen et al., 2015;Roos et al., 2023) The created path lines were then saved per individual flow component (residual flow, retained inflow, delayed ejection and direct flow). Finally there are also particles that follow invalid paths, these particle path lines are not used for the visualization. Each data set consisted of 30 time points with each time point in the line having multiple spatial components indicating the direction of the flow in that specific time point, velocity magnitude of the particle, the time point, the index of the point and a number representing the flow component that it consists of. Furthermore velocity vectors for each time point in a flow line were calculated and added to the dataset. For 2 datasets, this process could not be completed, and these datasets were excluded in the data curation.

The way the data was eventually loaded into ManiVault is by representing each full path line as a single data point with dimensions corresponding to individual velocity components for each time point and their spatial coordinates, therefore all the data points have 1530 dimensions, one for each data component per time point. For the generation of the t-SNE and HSNE embedding only the velocity vector components were used.

3.4. Parameter settings

To generate the t-SNE and HSNE embeddings, the ManiVault default parameters were used.(Table 3.1 and 3.2) This thesis does not explore into experimentation with these established parameters further; For more detail on the effect of the tSNE parameters , see the article: The art of using t-SNE for single-cell transcriptomics. (Kobak and Berens, 2019)

Table 3.1: Parameters used for t-SNE.

Iterations	Perplexity	Exaggeration	Exponential decay	Trees	Checks
1000	30	250	70	4	1024

Table 3.2: Parameters used for HSNE.

# hierarchy scales	random seed	# walks	Walk threshold	Walk length	AOI
2	-1	15	1.5	15	100

3.5. Experiments

To evaluate the research questions several experiments were performed. First, to evaluate whether t-SNE embeddings generate clusters that correspond to specific flow structures or clusters formed with very similar flow lines an experiment was performed where the flow lines of a single dataset were embedded with t-SNE. This embedding was visualized in a scatter plot with a linked 3D visualization of the raw data next to it. This embedding was colored by flow component, and then explored by selecting several clusters and visually inspecting the 3D visualization to see what kind of flow structure it corresponds with.

The data that is used contains a velocity magnitude and a velocity vector that is reconstructed during the loading of the data. The difference between these two is that the velocity magnitude only contains the magnitude of the velocity while the vector uses the directional components of the velocity as well. In order to evaluate which of these velocity representations should be used, an experiment was performed to assess the difference between the t-SNE map created by using the velocity magnitude vs the t-SNE map created by using the velocity vector.

To investigate whether t-SNE analysis can be used to compare flow patterns between different datasets, rather than just within one dataset, two additional experiments were performed. The first of these experiments was to create a t-SNE embedding for all the available datasets and comparing the created embeddings in order to evaluate the topological similarity between them. The second experiment was designed to select a single cluster corresponding to a specific flow structure in a single dataset, and then trying to identify the cluster corresponding to that same flow structure in different datasets.

The outcome of these t-SNE experiments raised a novel question: Is there a relationship between the velocity magnitude of the path lines at each time point and the clustering and cluster localization in the t-SNE map? To study this, an additional experiment was performed where the t-SNE map was colored with velocity magnitude at the selected time point, instead of flow component. Every time point is then captured and placed into a video loop in order to create a time-loop over the t-SNE map that shows at what time points each path line has a high velocity.

To investigate whether HSNE can be used for flow analysis and to asses the advantages or disadvantages of HSNE over t-SNE the same three experiments as performed for tSNE were repeated with HSNE embeddings.

Another experiment was performed where the landmarks chosen by the HSNE algorithm were visualized in order to evaluate if HSNE landmarks create an accurate representation of the entire flow system.

Finally, some path lines in the input data are classified as invalid. This means that the recorded path that was taken by the particle is not physically possible. It would therefore be interesting to see if these invalid path lines can be identified as a single cluster in the t-SNE or HSNE maps: this would enable an intuitive way of quality control for pruning invalid particle traces. In order to investigate this a visualization was made of a t-SNE map and an HSNE map, exploring the embedding structure for localization of invalid particle traces.

4

Results

The experiments outlined in the previous chapter were performed on datasets with both t-SNE and HSNE and then visualized with the 3D pathline visualization and the vortex visualization side by side. Figures 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7 and 4.8 show the results of the t-SNE analysis and the Figures 4.9, 4.10, 4.11, 4.12 and 4.13 show the results ofthe HSNE analyzes. Furthermore in the appendix are additional images that depict the same experiments on additional datasets which are not highlighted in the thesis main text.

4.1. t-SNE results

4.1.1. t-SNE for flow analysis

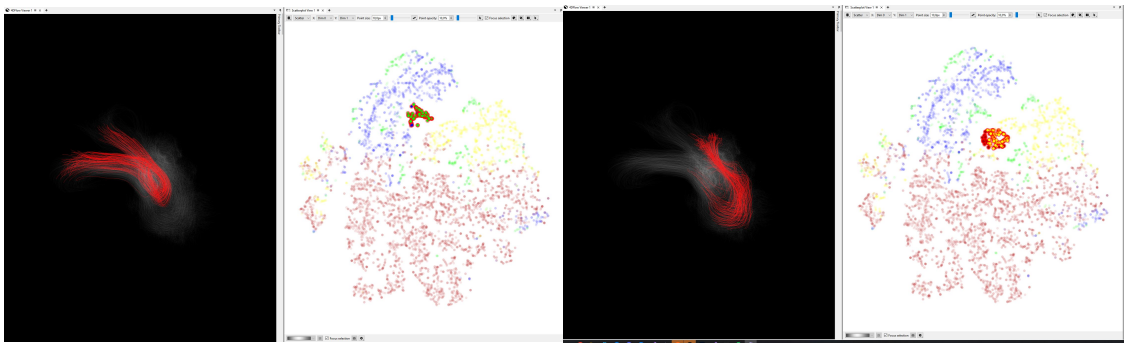


Figure 4.1: [t-SNE] Exploration of dataset 3 with the use of t-SNE analysis. Multiple different clusters are selected and their corresponding flow structure is shown in the 3D viewer.

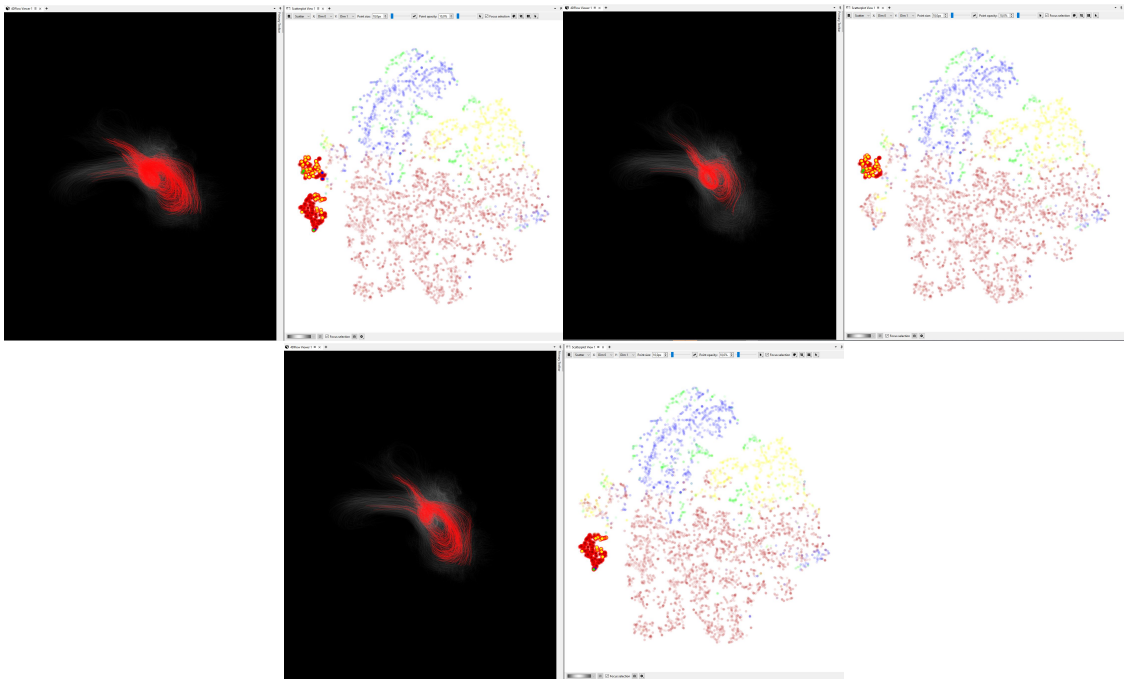


Figure 4.2: [t-SNE] Further exploration of a specific vortex structure (top left image) found in dataset 3. This cluster is itself clustered where one of its clusters mainly consists of residual volume flow lines(bottom) and the other cluster mainly contains retained inflow (right).

Figures 4.1 and 4.2 show different clusters selected in the same dataset. These clusters all show different flow features in the 3D visualization. Figure 4.1(left) shows a direct flow inside the heart, Figure 4.1(right) shows an example of retained inflow while in Figure 4.2 a vortex is found. This vortex is separated into 2 different adjacent clusters in the t-SNE map. One of the clusters consists of just the residual volume inside the vortex while the other cluster also contains retained inflow.

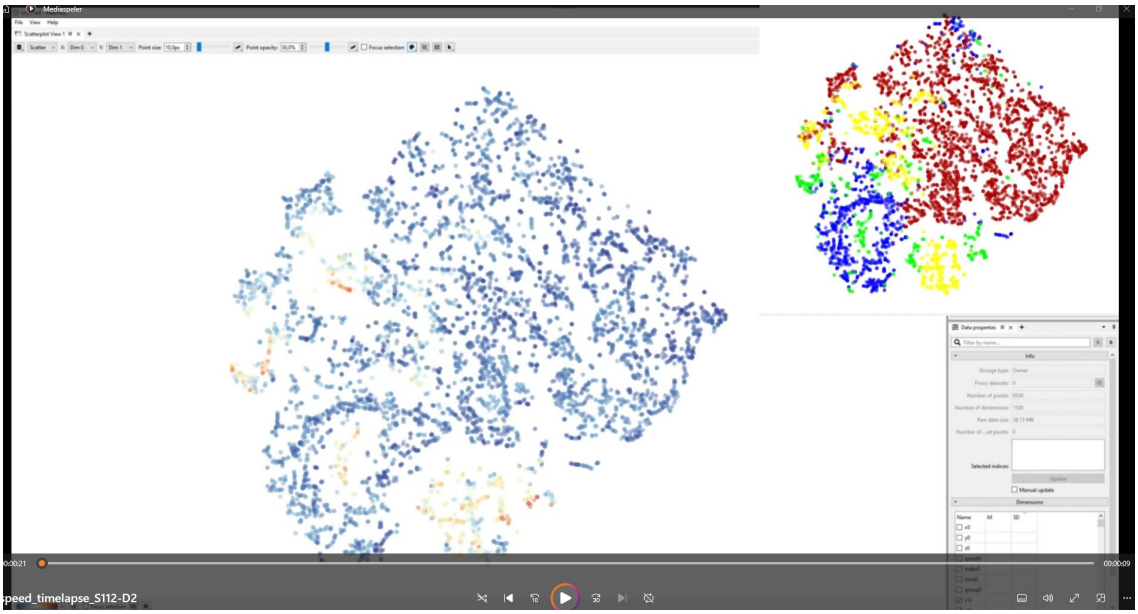


Figure 4.3: [t-SNE] Example of one of the time points with velocity magnitude as coloration. This is a screen-capture of a cine-loop created in order to investigate the relation of velocity magnitude on different time points. The cine-loop was created with dataset 3 and can be accessed using the following link: https://www.dropbox.com/scl/fi/cvfuchp4v4z8npw1fmotm/velocity_timelapse_S112-D2.mp4?rlkey=u98ucfl8k72nbnpvl2z0xjand&dl=0.

Figure 4.3 shows a screen-capture of a cine-loop created to analyze the velocity for of the path lines

in the t-SNE map at specific time points. A link to the cine-loop itself is posted in the description. The cine-loop shows a time lapse of all the 30 time-points and the corresponding velocity magnitude in the t-SNE map. The cine-loop shows that all the path lines within clusters reach high velocities at similar time points, but also that the timing of velocities varies across the map, and per cluster.

4.1.2. t-SNE multi data comparison

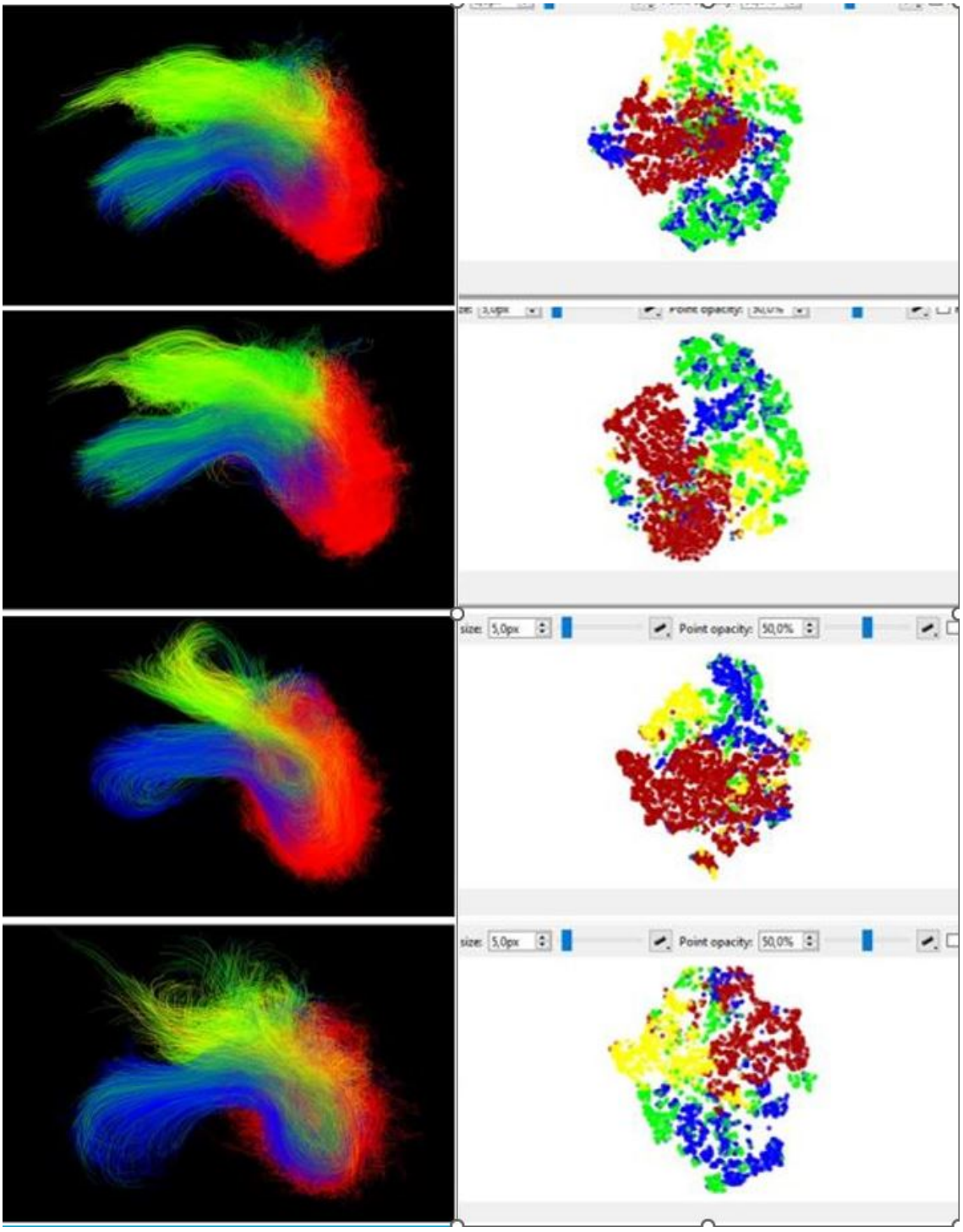


Figure 4.4: [t-SNE] Comparative analysis of 4 tSNE analyzes on the data. This Figure is used to show the topologically similar organization of flow components across different data sets. The datasets shown in this Figure are datasets 1-4.

In the t-SNE experiment of Figure 4.4 it can be seen that each t-SNE map reveals clustering of the different flow components. In these figures it shows that mainly the red flow component cluster, which is the residual volume, stands out from the rest while the other flow components, retained inflow, delayed ejection flow and direct flow, seem to overlap more in the embeddings.

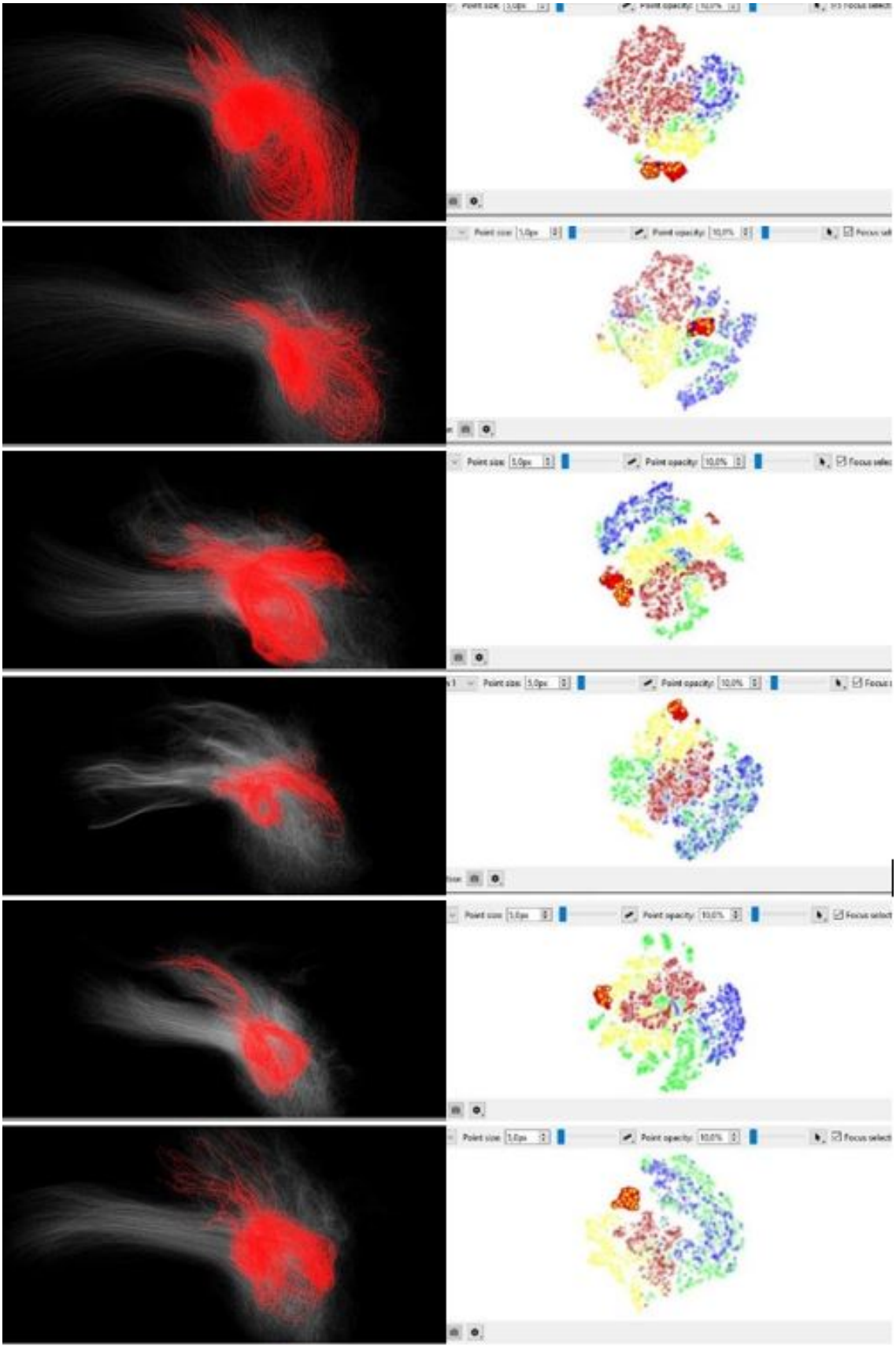


Figure 4.5: [t-SNE] Analysis of a selected cluster that is present in multiple datasets reveals a similar vortex in the 3d space in each dataset. The datasets used in this Figure are datasets 3,4,7,8,5 and 9 are used in that order.

Figure 4.6: [t-SNE]Showing the same vortex as previous figure but in this figure it is located in the direct/delayed outflow part of the image rather than in an overlap of the retained inflow and residual volume. This Figure uses dataset 2.

Figures 4.5 and 4.6 all show the same or similar vortex in the 3D space and the corresponding cluster in the t-SNE map. In the cases of Figure 4.5 all these clusters are located in a similar area of the t-SNE map, except for the second image, and they are all located at an overlap area between the residual inflow and residual volume. In Figure 4.6 however this same vortex is located at an overlap between the direct flow and delayed outflow rather than retained inflow and residual volume.

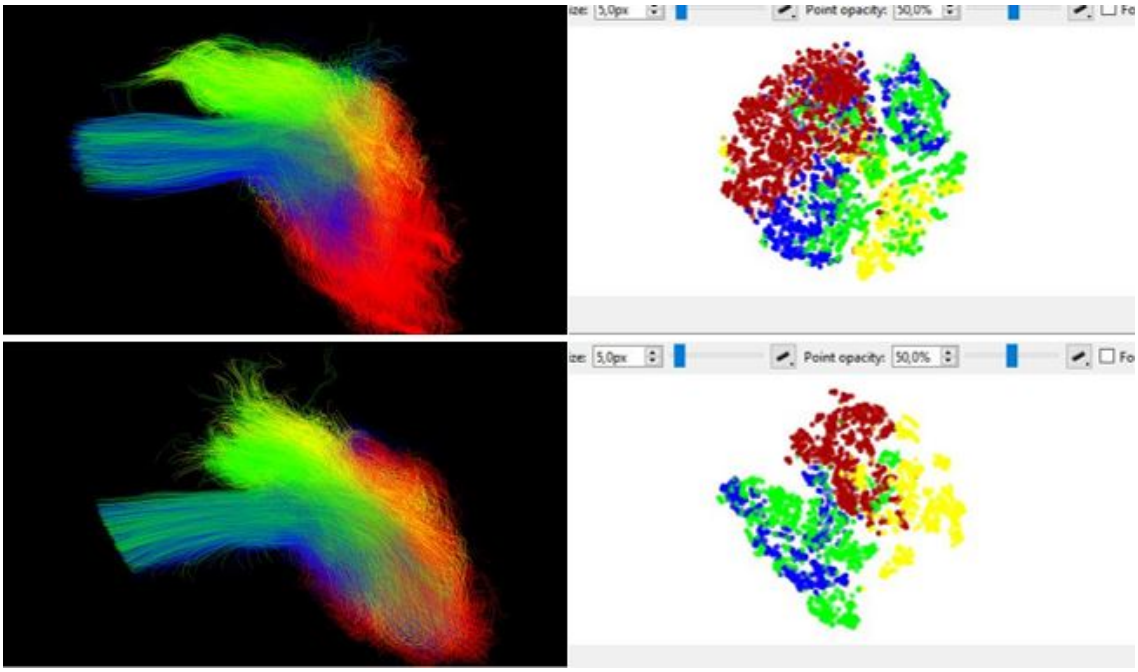


Figure 4.7: [t-SNE]Examples of two datasets that do not contain the same vortex that is found as individual clusters in previous images. Datasets 1 and 6 are in this Figure.

Contrary to Figures 4.5 and 4.6, in Figure 4.7 the clusters corresponding to the vortex found in the previous datasets could not be found in these datasets.

4.1.3. Vector velocity t-SNE vs velocity magnitude t-SNE

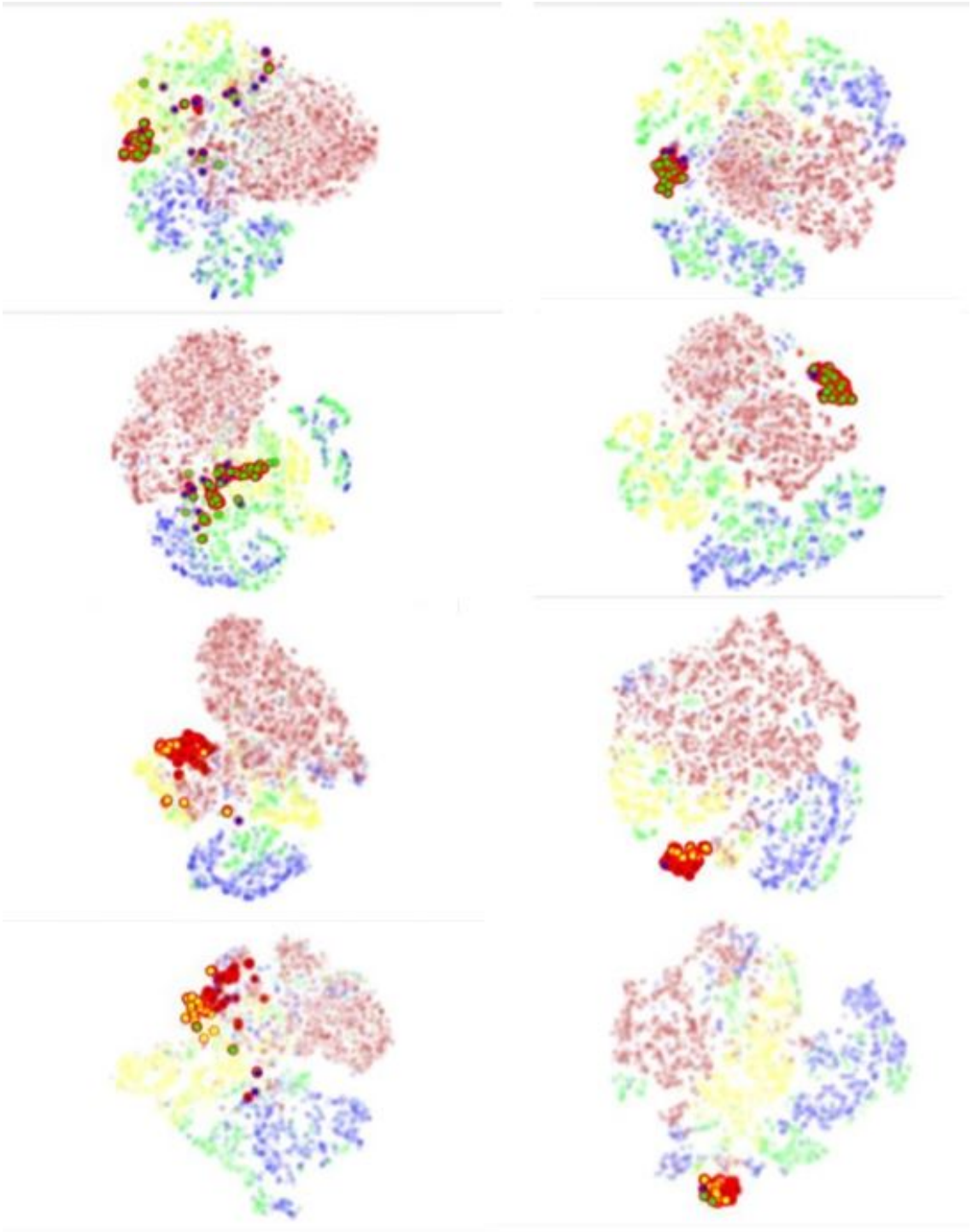


Figure 4.8: [t-SNE] Comparison between t-SNE maps created by using velocity magnitude (left) and velocity vector components (right). Results are shown for datasets 1-4.

Figure 4.8 shows a comparison between the t-SNE maps created by either using the velocity magnitude, or the vector of velocity components as data. Though in both cases, the t-SNE map structure groups flow components in a similar manner, the flow the vector component t-SNE map seem to reveal more compact clusters than the velocity magnitude t-SNE map, which forms less compact clusters.

4.2. HSNE

4.2.1. HSNE for flow analysis

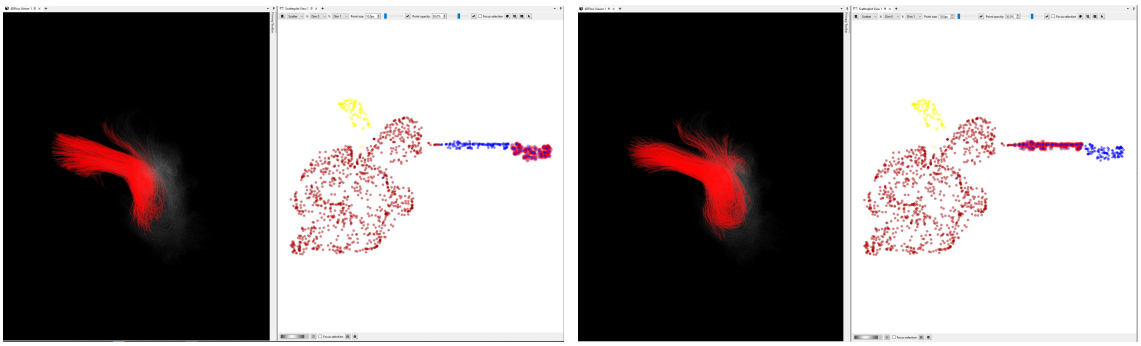


Figure 4.9: [HSNE] Exploration of dataset 3 with the use of HSNE analysis. Two different clusters of the delayed outflow are selected and their corresponding flow structure is shown in the 3D viewer.

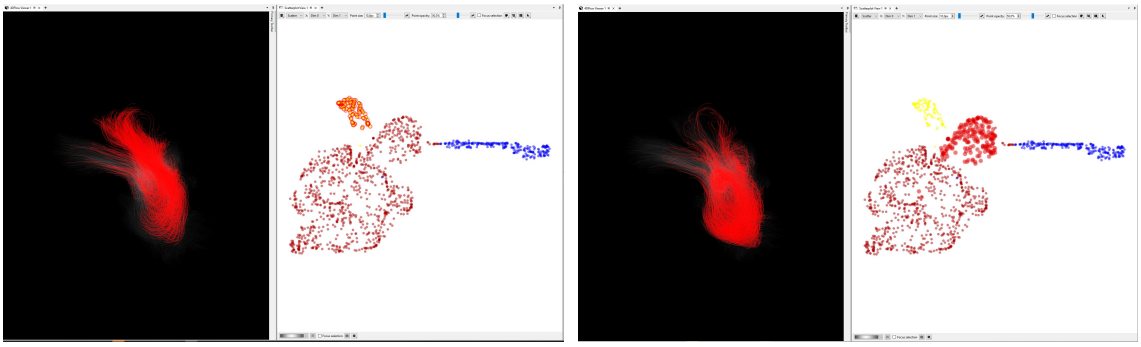


Figure 4.10: [HSNE] Exploration of dataset 3 with the use of HSNE analysis. A cluster containing the retained inflow(left) and a cluster which is separated from the rest of the residual volume(right) are selected.

Figures 4.9 and 4.10 show an experiment looking at different cluster in the HSNE embedding of a single dataset. In Figure 4.9 it can be seen that the delayed outflow cluster is separated into 2 clusters, in the Figure 4.9(left) cluster flow lines that are mainly located in the aorta are selected, while the Figure 4.9(right) shows flow lines that start in the pulmonary veins and and in the aorta. Figure 4.10(left) shows a cluster that is the entirety of the retained inflow and Figure 4.10(right) is a cluster that shows a vortex in the residual volume.

4.2.2. HSNE multi data comparison

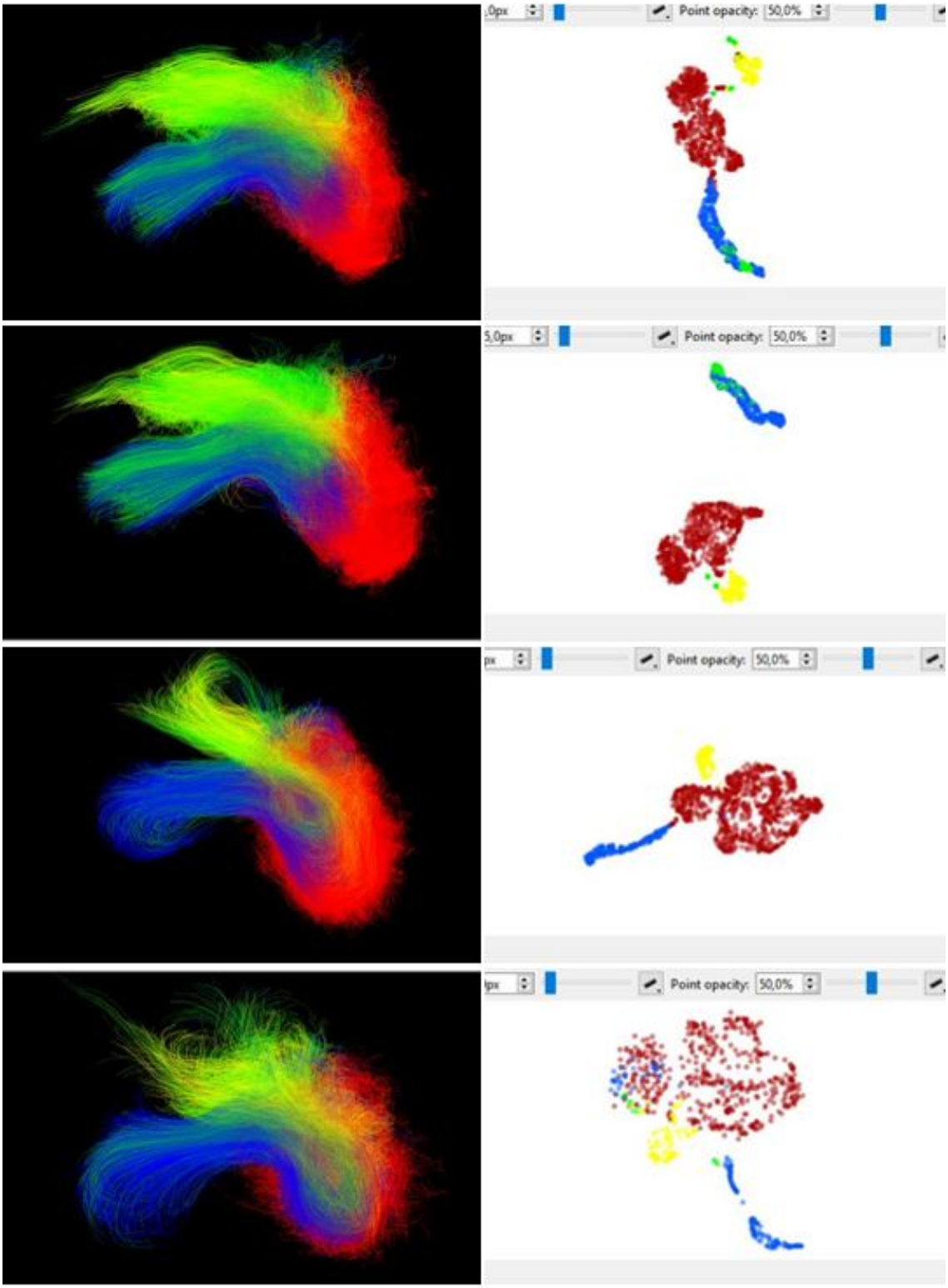


Figure 4.11: HSNE analysis comparing different data sets, showing the topological similarity between different data sets. The datasets shown in this Figure are datasets 1-4.

Figure 4.11 shows an experiment looking at the topological similarity for HSNE embeddings between different datasets. In this experiment the clustering of the residual volume, retained inflow and delayed

ejection are separated very clearly, while the direct flow seems to be less abundant or absent in the HSNE embedding.

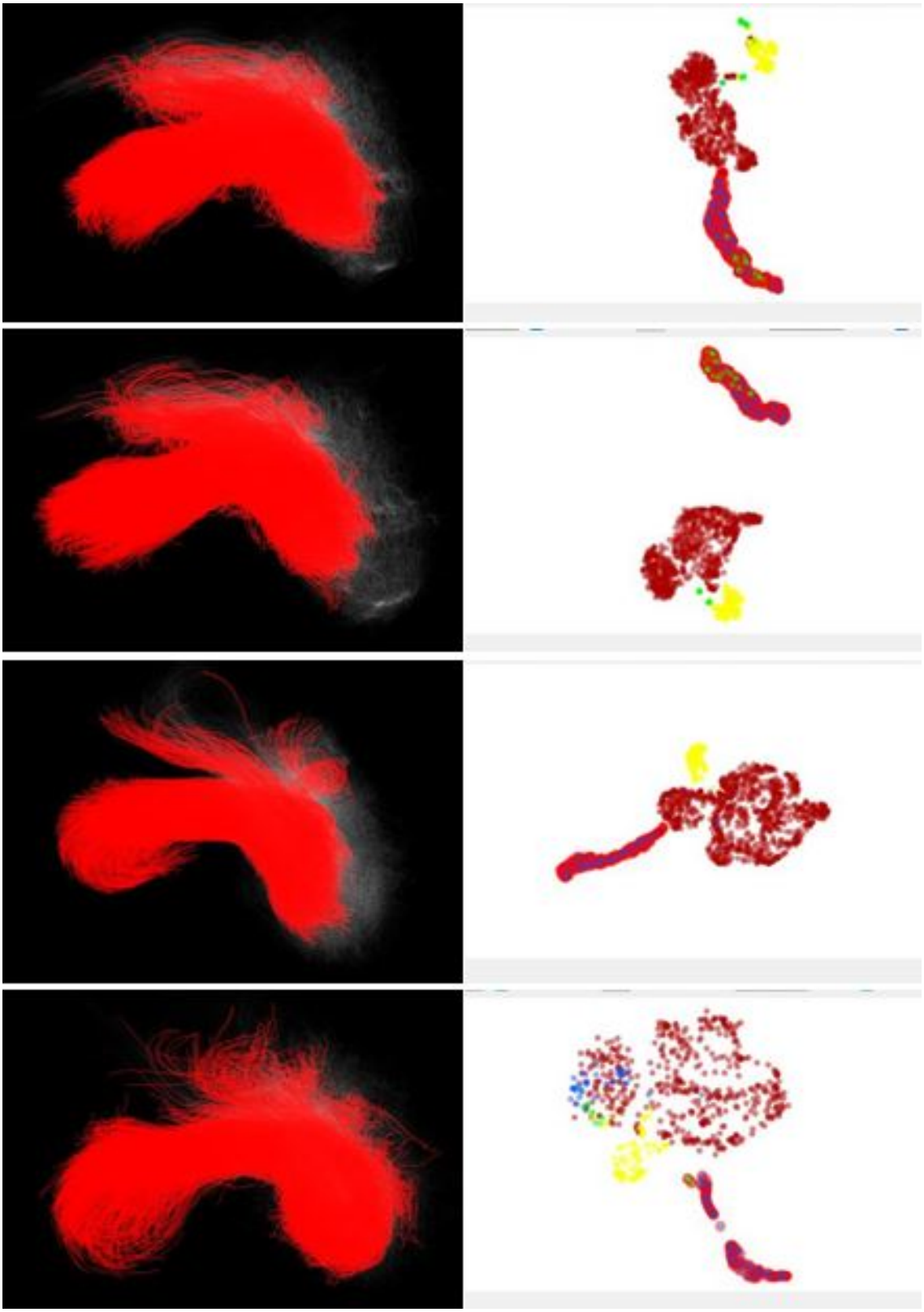


Figure 4.12: [HSNE] Example of a comparison of a selected cluster that is present in multiple dataset which represents a similar vortex in the 3d space. Datasets used are 1-4.

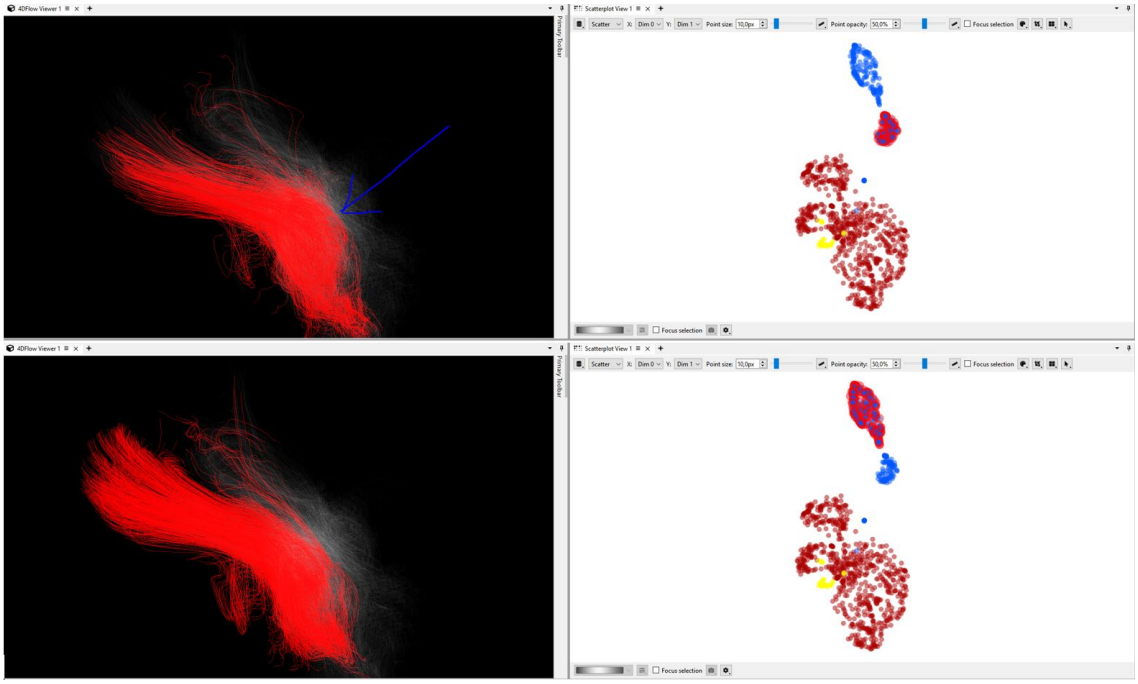


Figure 4.13: [HSNE] Example of different clusters inside the delayed outflow cluster. The blue arrow indicates the difference in flow structure between the clusters. Dataset 7.

Figure 4.12 shows the same cluster in multiple HSNE maps which all contain both the delayed outflow in the direct flow. In the first 2 cases this corresponds to the same flow structure in the 3D representation. While in the other 2 cases the flow structures look different from the first 2. The HSNE map in those datasets also looks different from the first 2 examples. In Figure 4.12 it can be seen that the cluster for the delayed outflow (blue cluster) is often separated into two different clusters, an example of this is shown in Figure 4.13. In this instance the flow lines inside the bottom cluster are all part of a vortex indicated by the blue arrow, while the top cluster does not go through this cluster. Although this cluster separation is present in multiple datasets, it is not always due to a vortex.

4.2.3. HSNE as summary of flow structures

Figure 4.14 shows which flow lines are chosen as landmarks by the HSNE algorithm. Datasets 1,2,4,6 and 8 have a combination of all flow components while for the other datasets either the retained inflow or the direct flow are severely underrepresented. Another observation that can be done based on this image is that specific flow structures seem to be missing in the summary.

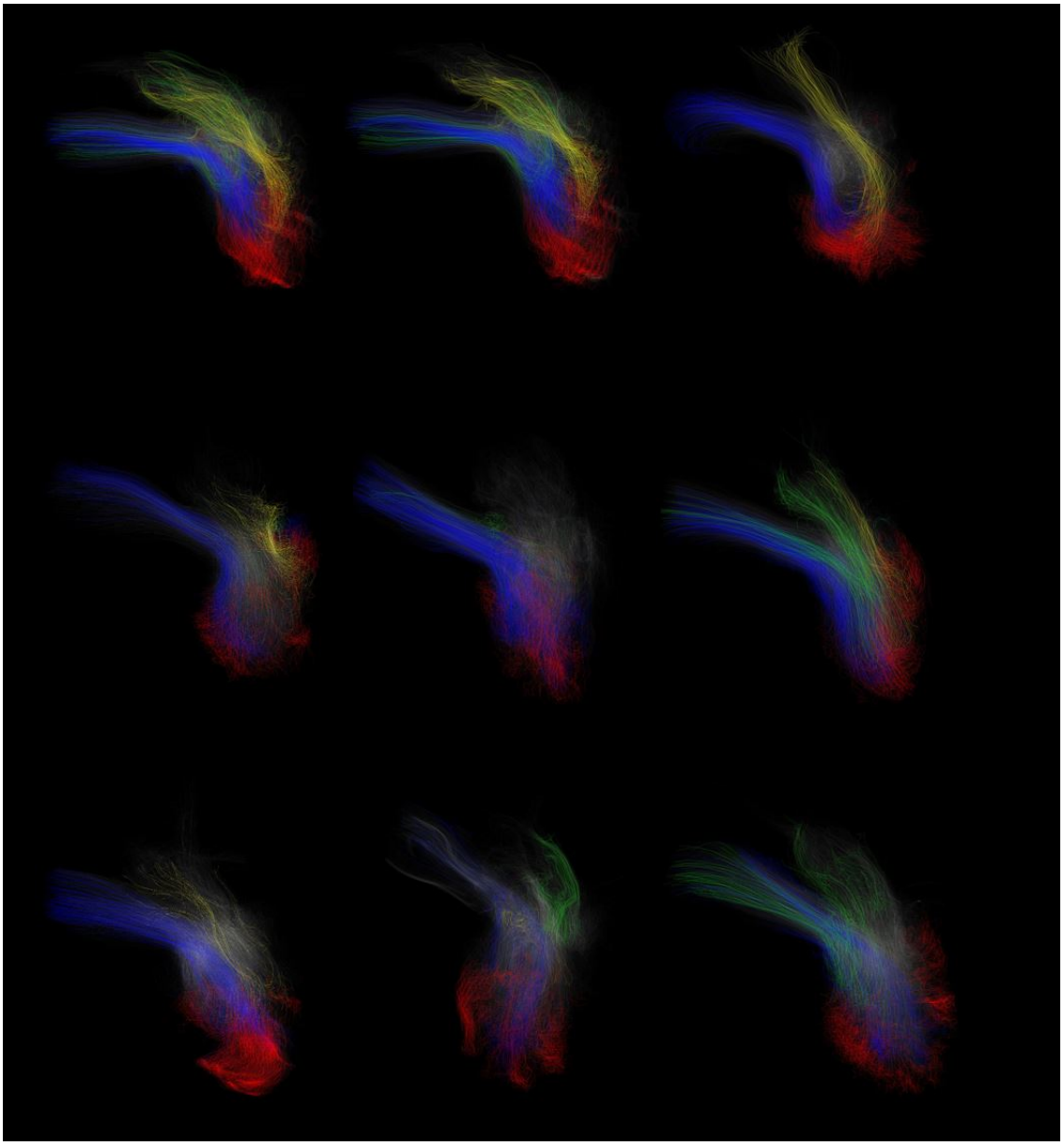


Figure 4.14: [HSNE] Comparison of 9 datasets represented by only landmarks selected by the HSNE algorithm for all datasets in order.

4.3. Invalid particles

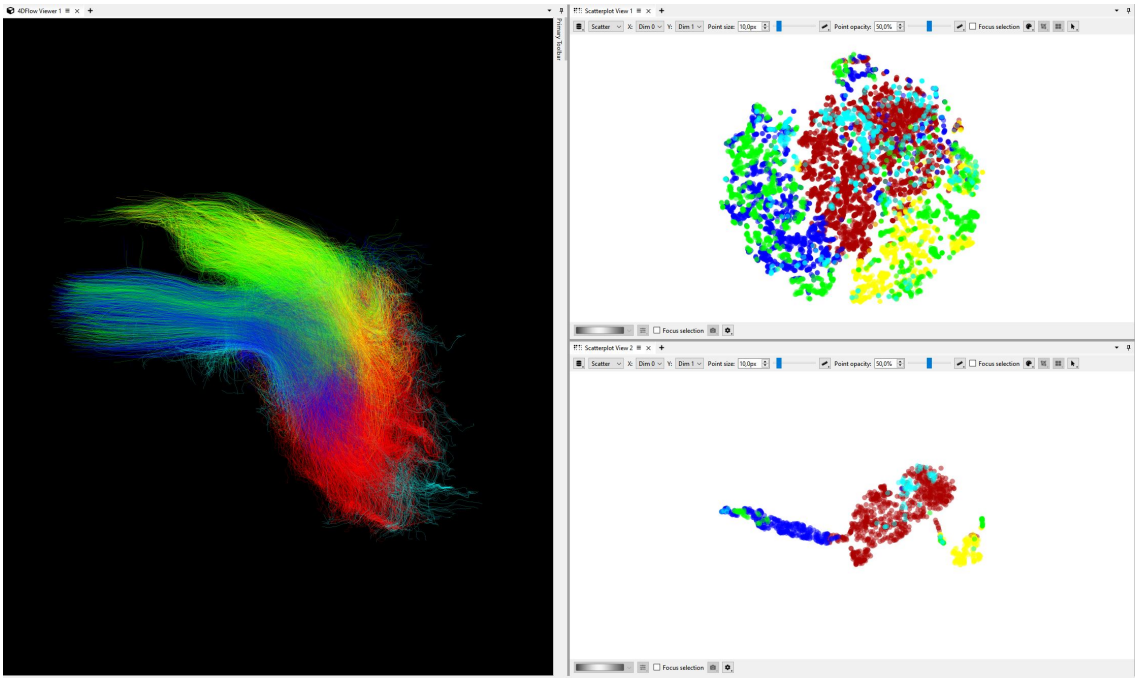


Figure 4.15: [t-SNE and HSNE] Visualization of invalid particles colored cyan on top of normal SNE maps. This figure is made using dataset 1.

Figure 4.15 shows the addition of the invalid particles in a t-SNE and HSNE analysis. The invalid particles are colored cyan.

5

Discussion

5.1. t-SNE

5.1.1. t-SNE for flow analysis

From Figure 4.4, it appears that tSNE maps are loosely sorted by flow components, where path lines of the same flow component mostly cluster together. Exception to this is the direct flow (green component), which appears more distributed over the tSNE map. This is likely caused by the high amount of similarity between large parts of the direct flow and parts if the retained inflow or delayed outflow, the similarity lies in the way the path lines enter the heart for the direct flow and retained inflow and in the way that the path lines exit the heart for direct flow and delayed outflow. Another problem with the separation is that there are multiple individual path lines that are clustered inside one flow component while actually being classified as another flow component. This unclear separation can be explained by the fact that flow components are created using a hard cutoff point where a path line is considered to be part of a specific flow component, therefore very similar path lines can be created but because their starting or ending point is just past the threshold they are considered to be a different flow component. On top of that some of the path lines linger around the mitral valve, causing path lines to appear in clusters near the residual flow while being classified as a different flow component.

When several clusters are selected in the t-SNE embedding of a single dataset, (Figures 4.1 and 4.2) it can be seen that the t-SNE algorithm does create clusters that correspond to specific flow structures, as seen in Figure 4.2. This Figure shows a selection of a cluster of path lines that all converge in a single vortex. An interesting observation with this specific vortex is that this vortex consists of 2 clusters, where the difference in the path lines of the different clusters seems to be only the end point of these path lines. However, when color-coding the velocity magnitude over time on the tSNE map, it becomes clear that the actual difference lies in the moment that the path line actually enters the vortex. Looking at the velocity indication in the cine-loop (Figure 4.3) it shows that there is a difference in the time-point that the velocity in these clusters increases. This indicates that within larger flow structures, the t-SNE algorithm can differentiate specific subparts of flow structures from each other. This also shows that creating a cine-loop of the t-SNE map colored by velocity magnitude can give additional insights in the flow structure.

The created cine-loop (Figure 4.3) shows the evolution of the velocity magnitude over time in the t-SNE map. With this visualization it becomes clear that the t-SNE clusters that are created largely correspond to when path lines reach similar velocities. This cine-loop creates an additional visualization which can facilitate the flow dataset exploration in search of specific flow structures because the path lines in these flow structures will reach similar velocities at similar points in time.

5.1.2. t-SNE comparison between datasets

The first experiment comparing multiple datasets (Figure 4.4) shows that there is topological similarity between the outcomes of the computation of t-SNE on different datasets. The Residual Volume is always mostly separated from the other flow components. The retained inflow and delayed outflow are also not clustered with each other. The direct flow path lines are mostly intermingled with the retained inflow and delayed outflow path lines. This is likely because these path lines are very similar in either

starting or ending position and direction. Due to this t-SNE probably clusters these path lines together because of their similar flow pattern, while their assigned labels may differ.

The second comparison experiment (Figures 4.5, 4.6 and 4.7) shows the corresponding cluster found for a specific feature in the 3D space for multiple datasets. The clusters are all located inside a specific area in the t-SNE map and are composed of both path lines of retained inflow and residual volume, except for the dataset shown in Figure 4.6 where it is composed of direct flow and delayed outflow. The shape of the vortex in the 3D representation also appears to be different so it could be possible that the vortex is just different, causing the t-SNE map also to be different, or simply that the correct cluster was not selected. In some of the datasets this specific vortex was not found in the t-SNE map, this is shown in Figure 4.7. A possible explanation for this is a difference in the actual flow of the blood in the heart for this specific patient is different and therefore does not contain this specific vortex.

These two experiments show that there is a similarity in the created t-SNE embeddings and that in most cases clusters for a specific flow structure in one dataset are also present in another dataset. However because some exceptions were found in the available datasets more research needs to be done on datasets where the same flow structure is known to be present in multiple datasets, so that it can be confirmed that the cluster is created in all cases.

5.1.3. velocity magnitude vs vector velocity t-SNE

The velocity magnitude t-SNE vs vector velocity t-SNE experiment shown in figure 4.8 indicates that while clusters found in the vector velocity t-SNE are also positioned close together in the velocity magnitude t-SNE, they cluster less strongly and in some instances are more scattered. This shows that the addition of velocity direction in the t-SNE does indeed provide additional data for the t-SNE algorithm to create clusters of similar path lines. In the remainder of the experiments, we have therefore used vector-velocity to compute the tSNE and HSNE maps.

5.2. HSNE

5.2.1. HSNE for flow analysis

Figures 4.9 and 4.10 were created to see if the HSNE algorithm could also be used to analyze flow data. Contrary to the results of t-SNE, clusters corresponding to specific flow structures could not be found in the embedding. What is clear from this however is that the clustering that is created by HSNE corresponds more closely to the flow components, demonstrating a separation of the flow components. Although the direct flow was absent in this specific dataset embedding, it is present for some other datasets.(Figure 4.11) In these datasets the direct flow is clustered together with the retained inflow and delayed outflow. Therefore HSNE could be used to differentiate between residual flow, delayed outflow and retained inflow, however direct flow is not easily identified in the HSNE embeddings. When drilling into the delayed outflow and retained inflow clusters, the direct flow does appear and therefore the HSNE method does not see a significant difference between these pathlines to give them their own landmark. A similar issue was observed with t-SNE.

5.2.2. HSNE comparison between datasets

The first comparison experiment, as shown in Figure 4.11, was focused on the topology of the embeddings. As discussed in the previous paragraph the clustering of flow components is very pronounced. The embeddings are all very similar to each other. In all cases the residual volume, retained inflow and delayed outflow are represented by pronounced separate clusters and the direct flow path lines are either missing or clustered together with the retained inflow and delayed outflow.

The second comparison experiment was to find a specific flow structure in a single dataset and find their the corresponding clusters of that flow structure in other datasets.(Figure 4.12) Because the flow structures are not clustered very well using HSNE all the selected clusters are just the delayed outflow combined with some of the direct flow path lines. What was found however is that the delayed outflow is often clustered into two separate clusters or a line.(Figure 4.13) The path lines that are clustered in these two cases are slightly different however actual flow structures were not identified.

5.2.3. HSNE landmarks as summary

The experiment performed in order to evaluate if HSNE landmarks create an accurate representation of the entire flow system (Figure 4.14) also yielded inconclusive results. For most datasets the landmarks

are nicely distributed throughout the entire flow field, however in the case of dataset 5 it seems that the entirety of the atrium does not contain any landmarks. For the other datasets this is not the case, however they do show that the difference between the direct flow and both the delayed outflow and retained inflow is not very clear. This is likely because they have very similar components of the path lines. For the delayed outflow, the terminus of the path line is very similar to the direct flow and for the retained inflow the starting part of the flow line is very similar to the direct flow. Another notable observation from the created flow summaries is that specific flow structures, like vortexes that are present in most of the datasets near the heart valves, are not represented.

5.3. Invalid particles

From the experiment shown in Figure 4.15 it can be seen that the invalid path lines, indicated in cyan, do not cluster together into neat individual clusters. In the t-SNE map the invalid particles are spread out over a large area, but mainly near the residual flow. This is likely because the invalid path lines are mostly similar in shape to the path lines that are considered to be residual flow, they mostly do not enter or exit the through the paths that other path lines follow and are therefore similar to the residual flow. Examining the velocity magnitude cine-loop, they also appear to have a low velocity magnitude, explaining their relative position in the t-SNE maps.

For the HSNE map the invalid particles do cluster together more, in this case they cluster together in 2 parts, the residual flow and in the delayed outflow. Even though the invalid particles cluster together, they do not create separate clusters and can not be easily distinguished from other clusters.

With this experiment in mind the SNE algorithms seem less suited for finding invalid path lines.

5.4. t-SNE vs HSNE

Table 5.1: Comparison between t-SNE and HSNE strengths and weaknesses .

t-SNE vs HSNE		
Aspect	t-SNE	HSNE
Separation of clusters	Medium	Strong
Clustering of flow components	Medium	Strong
Clustering of flow structures	Strong	Weak
Ability to explore flow data	Strong	Medium
Comparison between datasets	Medium	Strong

From the experiments performed for this thesis it can be concluded that both t-SNE and HSNE can be used for flow analysis, however in different ways.

t-SNE has the main use case that the embedding can be fully explored and flow structures cluster together, and that there is an overall ordering emerging based on timing of velocities in flow structures. Therefore exploration of flow structures can be done in an exploitative way in a single dataset. For multiple datasets these clusters look similar making it possible to compare different datasets with each other, however this takes a bit of time. A possible improvement on this could be to perform manifold alignment, however this was outside the scope of this thesis.

HSNE on the other hand provides a summary embedding of representative path lines in the data using density-based sub-sampling, while t-SNE embeds all individual path lines, HSNE is more effective in distinguishing flow components from each other and could be useful in this regard, although the algorithm does not correctly identify a difference between direct flow and the other components and more research has to be done to see if there is a solution for this.

5.5. Comparison to other methods

It is difficult to compare the performed methods with other state of the art fluid flow analysis methods because those methods use vector field values while with the t-SNE and HSNE methods path lines were used. This means that a direct comparison between the methods with the used datasets is not possible. A good way to validate this method in the future is to, work with vector fields and generate

the path lines that were used for the t-SNE and HSNE methods. This way the methods can be directly compared to other methods.

An inconvenience with this analysis method is that the t-SNE and HSNE maps are differently oriented every time they are created, because of the fact that the t-SNE and HSNE algorithms use a random map initialization. However the individual clusters do take on the same shape. This can be mitigated in ManiVault by fixating the random seed at different t-SNE runs, and by initialing the t-SNE computation using PCA projections.

6

Conclusion

The experiments performed in this thesis show that SNE methods can be used to explore fluid flow of individual flow datasets. This can be done by selecting individual clusters in the t-SNE map and visually inspecting the visualization of the cluster in the 3D viewer. While this manual exploration of the data can be performed with the use of t-SNE it is rather difficult to find specific flow structures using a static t-SNE map, however with the use of a cine-loop showing the velocity magnitude the data exploration can be simplified. For HSNE with the settings used in this thesis the main use is to separate the flow components.

One of the reasons that SNE methods can be used to explore 4D flow data is that the t-SNE map clusters flow structures based on flow direction and velocity. This causes flow structures to cluster together due to their similar speed at similar points in time. HSNE on the other hand does not cluster in the same way, this method does not cluster the flow structures, however it creates clusters that separates the flow components better than t-SNE.

Another argument for using SNE methods in 4D flow data is that use of the newly created plugin in ManiVault which creates a 3D image representing 4D path lines interaction between the raw data and a visualization can be established. By interacting with the results of SNE analysis a 3D representation of the 4D path lines is shown in real time. This allows for easier exploration of the 4D flow data. Another development that was explored to increase the user friendliness of the methods was the creation of a cine-loop showing the evolution of the velocity magnitude over the t-SNE map.

Other uses for SNE methods for 4D flow analysis were investigated, like the use of HSNE for the creation of a summary of the 4D flow or the identification of invalid path lines with the SNE methods. However in the case of the 4D flow summary, the results were inconclusive and the utility of the SNE methods for the identification of invalid path lines (and with that data curation and quality control) could not be convincingly assessed.

Taken together, the results of the experiments reported in this thesis indicate that the use of SNE methods is an interesting development in the field of 4D flow analysis. By conducting more research into this topic like exploring the use of manifold alignment, as well as the other improvements recommended in the previous chapter, t-SNE could become a novel method which allows for highly interactive 4D flow data exploration.

A

Appendix

A.1. Dataset names

The original names of the used datasets are VTK-data, S110-D2, S111-D2,S112-D2, S114-D2, S115-D2, S117-D2, S118-D2, S120-D2, S121-D2 and S122-D2. Of these datasets 2 were unable to be loaded in, theses datasets are S110-D2 and S122-D2. For simplicity the used datasets are renamed in the thesis to datasets 1-9.(Table A.1)

Table A.1: Conversion original data names to thesis data names.

vtk-data	**S111-D2**	**S112-D2**	**S114-D2**	**S115-D2**
1	2	3	4	5
S117-D2	**S118-D2**	**S120-D2**	**S121-D2**	
6	7	8	9	

A.2. Additional experiment results

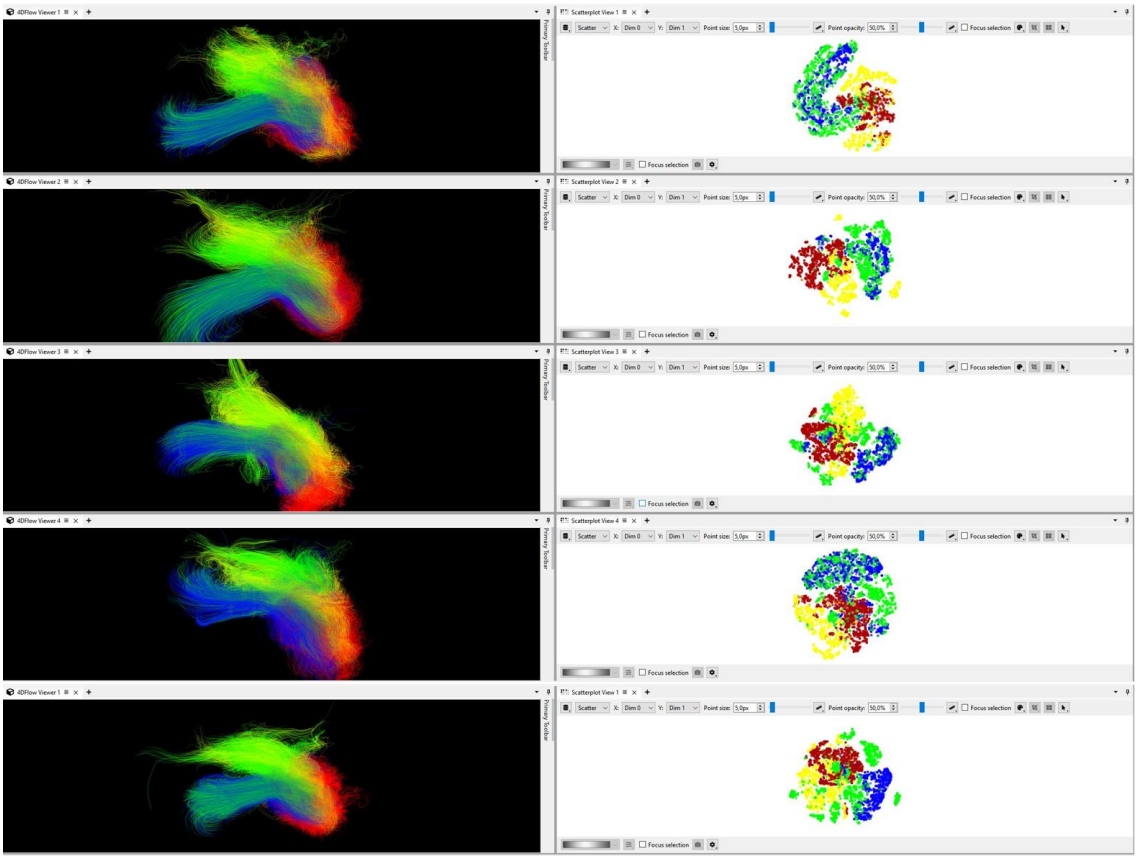


Figure A.1: [t-SNE]This figure shows additional results of the base t-SNE analysis of the dataset in order to show the topological similarity between the t-SNE results of different datasets. This figure shows datasets 5-9.

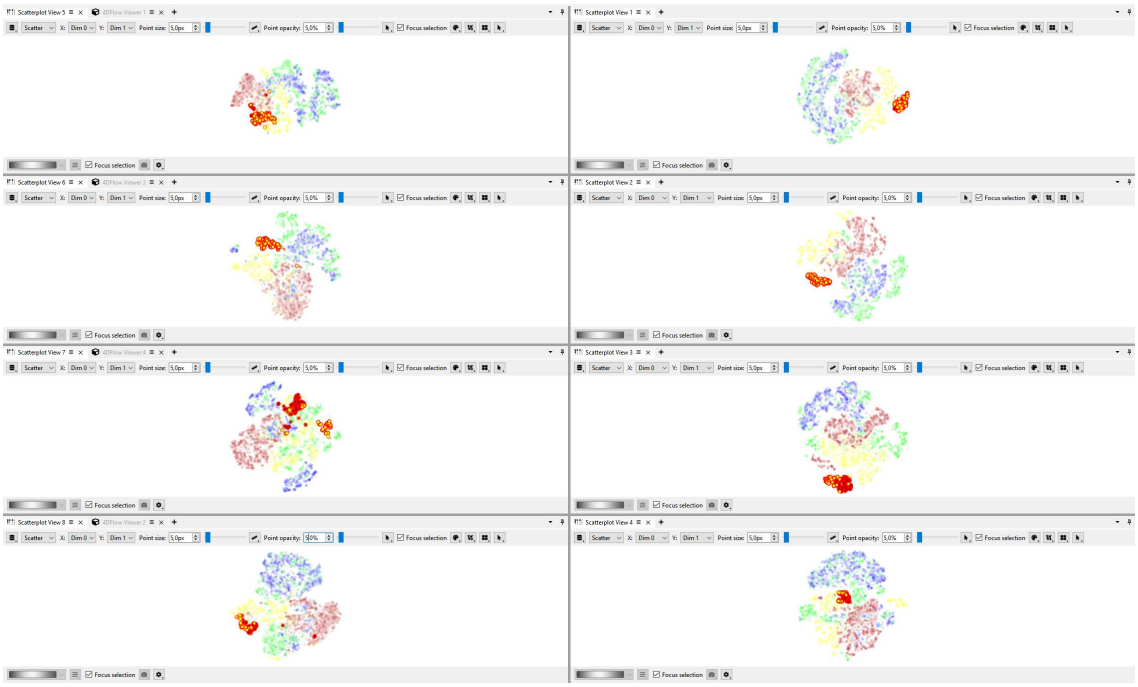


Figure A.2: [t-SNE]This Figure shows additional results of the comparison between vector velocity t-SNE maps and velocity magnitude t-SNE maps. This Figure contains datasets 5-8.

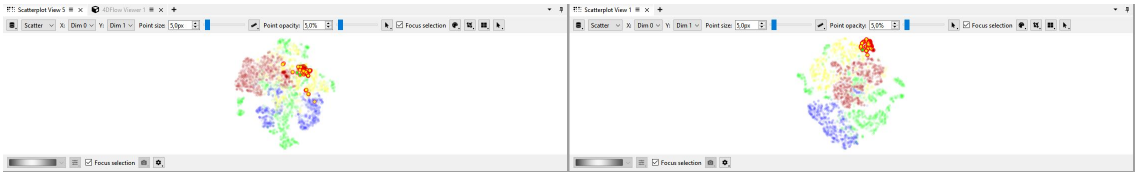


Figure A.3: [t-SNE]This Figure shows additional results of the comparison between vector velocity t-SNE maps and velocity magnitude t-SNE maps. This Figure contains dataset 9.

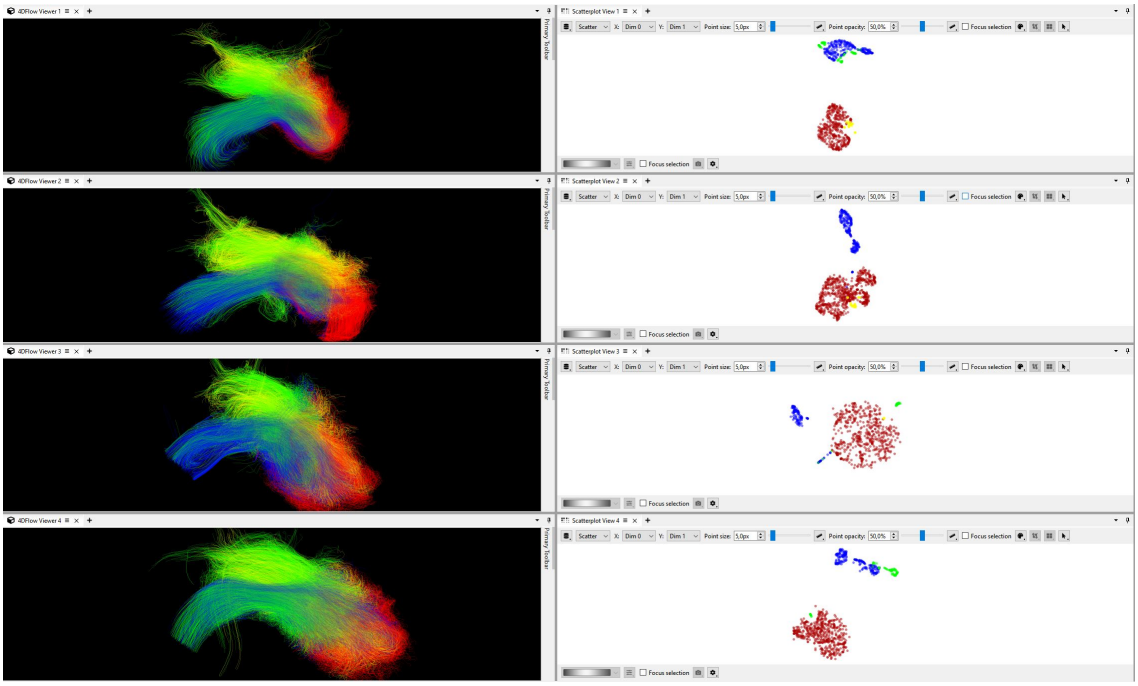


Figure A.4: [HSNE]This figure shows additional results of the base HSNE analysis of the dataset in order to show the topological similarity between the HSNE results of different datasets. This figure shows datasets 6-9.

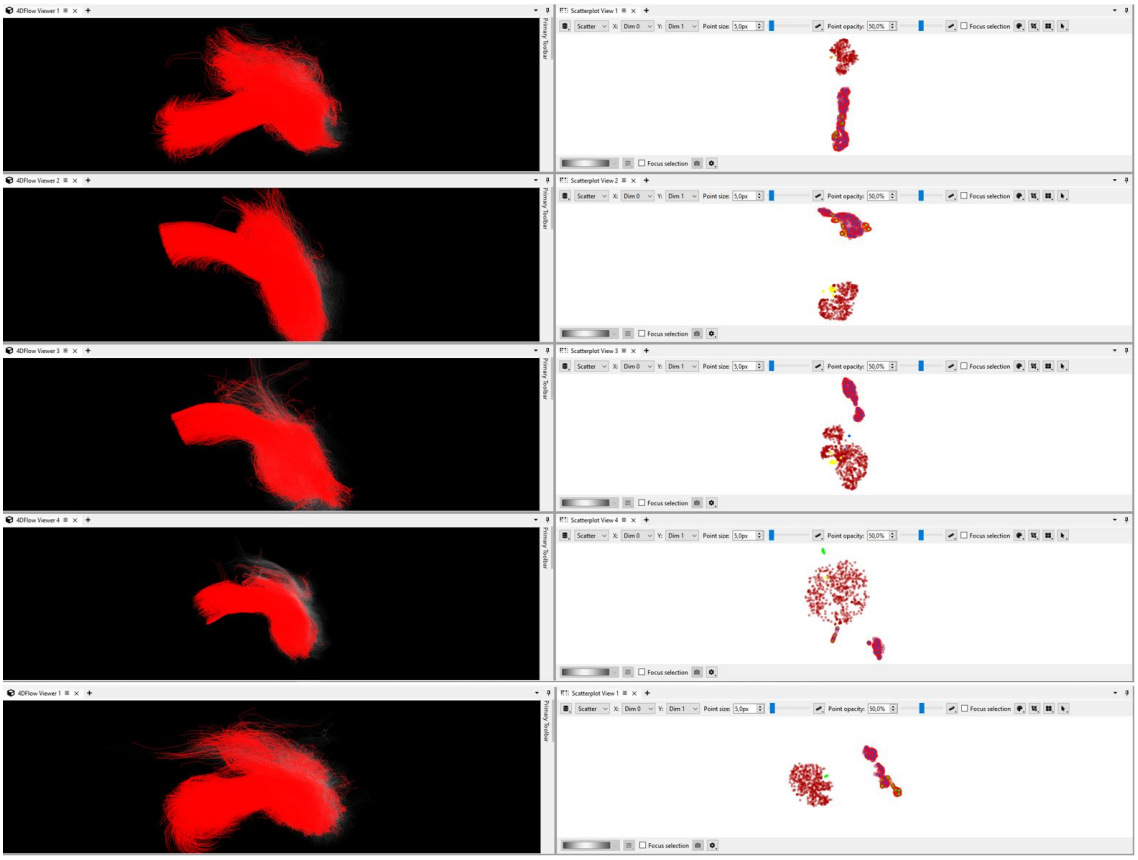


Figure A.5: [HSNE]This Figure shows additional results of finding similar clusters in different datasets. This Figure contains datasets 5-9.

B

Internship report

Abstract

LUMC's LKEB department is working on a High Dimensional Plugin System called HDPS in order to create a single program that contains different kinds of high dimensional data analysis and visualisation methods. For my master internship I got to help develop this program by adding a 3 dimensional data viewer to the system.

B.1. High Dimensional Plugin System

At the LUMC's LKEB department a group of developers is working on the High Dimensional Plugin System, or HDPS, which integrates different visualization and analysis methods focused on high dimensional cell data. The aim of this program is to integrate many different systems that have been developed over the years at the department into one software package and to make it accessible for additional functionality to be implemented with the use of additional plugins. The advantage of having all these analysis and visualization tools in one program is that they are able to be viewed together and can interact with each other, giving the possibility for further data exploration than each of these methods would give on their own.

Previously the visualisation of 2D data was only supported by HDPS. However 3D visualisation of the data is important in order to gain insight in the spatio-molecular structure of the data.

Therefore the aim of the my internship is to develop a C++ plugin in order to import high-dimensional 3D data files in the HDPS plugin system, and to use this data in order to develop a viewing plugin that enables interactive selection and linked visualization of data points in low-dimensional data embedding in a voxel rendering framework. The interaction is useful to give extra spatial information to data analysis methods such as TSNE (Van der Maaten and Hinton, 2008) and HSNE (Pezzotti et al., 2016). These two methods are algorithms that sort data points based on how similar they are based on multidimensional data. Interaction between these algorithms and the 3 dimensional viewer could enable easy visualisation of specific tissue structures inside the data that are selected in the embedding created by HSNE (Pezzotti et al., 2016) or TSNE (Van der Maaten and Hinton, 2008).

B.2. My experience at LUMC

My experience at the LUMC was pretty good. For daily questions I could contact Thomas Kroes and if I had questions about the data or features I could contact Boudewijn Lelieveldt. What was a bit challenging is that the entire internship was online, this was not to be helped but would have been nicer if it was not online. This caused the internship to feel more like a course project than an internship. The only offline meetup I had was the LKEB department outing to the beach where I could finally meet the people I had been working with for over a month.

While working on the project the most difficult weeks were the first weeks, this was due to my unfamiliarity with the HDPS system and were spend with me trying to get a simple plugin made that displayed a sphere. Once this was made I had a starting point on which I could build the viewer plugin.

B.3. Volume Loader Plugin

In order to create a visualization plugin for high dimensional volume data in the HDPS system. A plugin that loads in this data was first created. This plugin takes the given volume data and turns it into the points datatype typically used in HDPS.

B.3.1. Volume Data

The high dimensional volume data sets used for the plugin have the file type .hdvol. This file type was developed by LKEB a few years before for their previous visualization architecture the High Dimensional Inspector (HDI). The contents of this data first contains meta data about the amount of points inside the data set, the amount of features that the data-set has and the spatial size of the data set. This is then followed up by the spatial coordinates of a point in the data set and it's feature data.

The .hdvol data set does not describe all the data points inside the volume. The number of points described in the meta data corresponds to the amount of points that are present in the data set. The points that are not present in the data set are points that are not of interest for the visualization, namely these points are the background of the image.

B.3.2. Volume Loading

While the exclusion of the irrelevant data points is useful for data analysis methods, it makes the visualization of the data a bit more difficult. In order to remedy this problem the data was loaded into a 4 dimensional vector with the dimensions corresponding to the 3 spatial coordinates and the 4th dimension being the features. The data points inside the vector that do not correspond to a point in the data set keep their initialized value of 0. These points are, after all the points in the data-set are read out , changed to be 1 lower than the lowest value in the data set in the current feature. This is done in order to create a simple way of making those points transparent in the Volume Viewer Plugin.

While a 4D vector which can be easily reduced to multiple 3D arrays is ideal for volume visualization the HDPS system does, as of now, not yet have a dedicated way of storing this, instead it uses a 1D vector of points data. Therefore in order to create this 1D vector the 4D vector is read out into the 1D vector and this is passed to the HDPS system.

In order to give the Volume rendering the meta data needed for the creation of an image data object the spatial dimensions are added to the data as properties.

B.4. Volume Viewer Plugin

The volume viewer plugin is the visualization plugin for the data loaded in using the volume loader plugin. It visualizes the data in 3 dimensional space with the use of VTK (Schroeder et al., 2006).

B.4.1. Volume Rendering

In order to render the data first a render window needs to be created where the to be rendered object can be viewed in. Because the HDPS system uses QT Qt Group, 2024 as a software development platform a render window that is compatible with QT needs to be used. The render window used for this purpose is the VTK Generic OpenGL render window which can be used inside a QVTKOpenGlWidget.

The data that is to be visualized now gets loaded into the renderer and from the dimension metadata an empty image data object is made which is filled with scalar values corresponding to the points data. This data is mapped using a smart volume mapper in order to create a 3D object of a cube with values attached to each point. However this is not yet ready for rendering because the mapper does not yet contain any information about what it needs to do with the specific values inside the cube.

B.4.2. Color and Opacity Mapping

In order to make the renderer create a visualization out of the 3D object a color mapper needs to be added which looks at the data values inside the objects and turns it into a color for the image. The color maps used for the volume viewer plugin are the default color maps that are available in the HDPS system. The background color, as defined in the volume loader plugin, is set to black.

In order to remove the background from the image the entire image value range is set to be opaque with the use of a opacity function which sets the opacity of the background values to 0 , making them transparent, and the foreground values to 1, making them opaque. This creates a 3D object containing

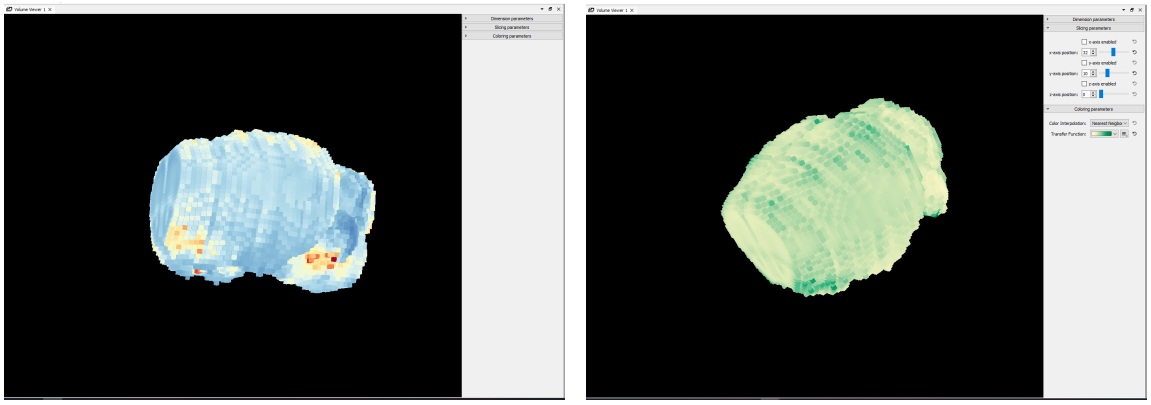


Figure B.1: These images depict the visualization of a mouse brain using 2 different color maps in the Volume Viewer Plugin of the HDPS system.

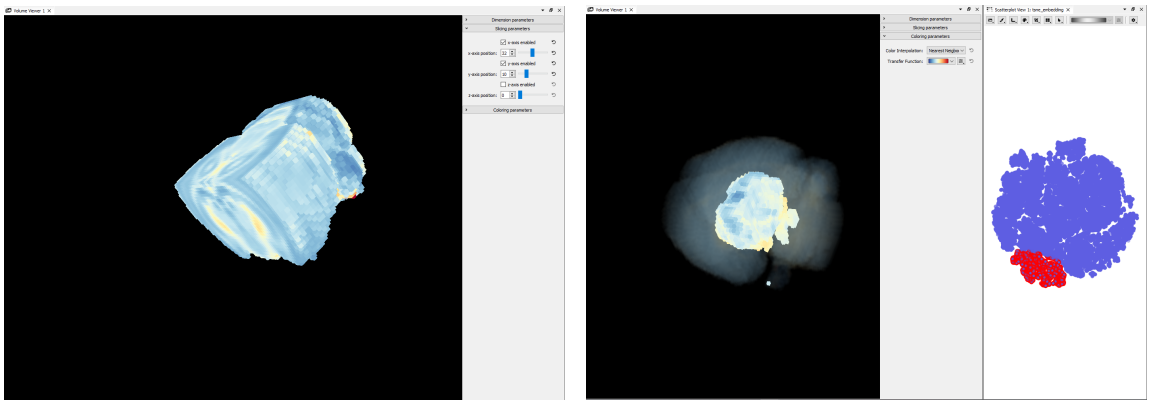


Figure B.2: These images depict the visualization of a mouse brain in the Volume Viewer Plugin of the HDPS system. The left image displays the slicing functionality of the plugin. The right image displays the visualization of data that is selected in a scatter plot.

only the relevant data. (Figure B.1)

B.4.3. Slicing Options

In order to not just see information on the surface of the object, but also inside the object, a slicing option was added. With this slicing options parts of the object can be removed based on spatial coordinates. This creates a intersection of the object where the data in that intersection is visible. This is done with the use of VTK planes which are added to the volume mapper in order to create the object with the desired intersection. The object can be sliced in 3 different directions. (Figure B.2 Left)

The color map is interpolated using the nearest neighbor interpolation, because of this the intersection is not very smooth. In order to create a smoother intersection an option has been added to change the color map interpolation to linear or cubic instead of nearest neighbor. (Figure B.3)

B.4.4. Selection Interaction

Another feature added to the volume viewer plugin is the possibility to select points inside a scatter plot plugin and have the ability to view these data points in relation to the rest of the data in 3D space. This is done by adding a new instance of the data to the renderer containing only the data selected in the scatter plot and changing the opacity of the full data set to be semi transparent. (Figure B.2 Right)

B.5. Conclusion

As a result of the project good working plug-ins were created for both loading in and displaying .hdvol data in 3D. The volume viewer also has several options and interaction capabilities which can be used to explore data structures and its surroundings inside the data.

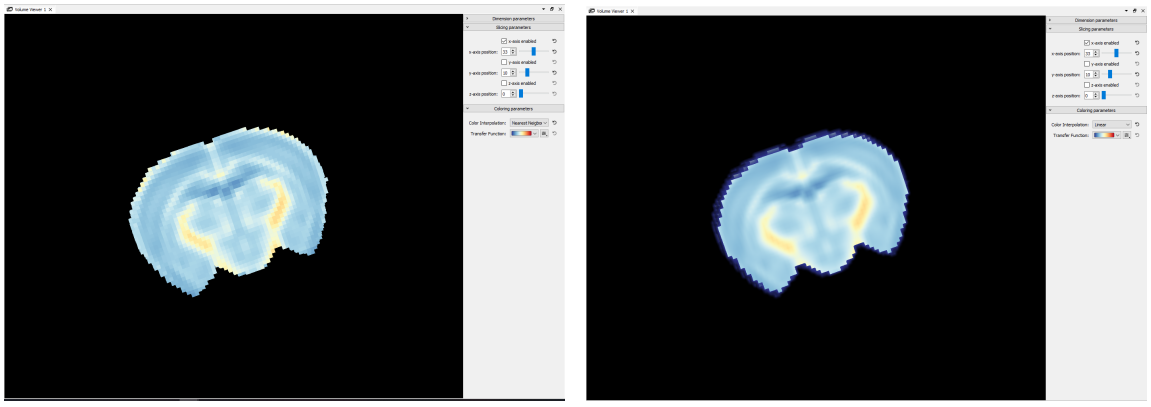


Figure B.3: These images depict the visualization of a mouse brain in the Volume Viewer Plugin of the HDPS system. The left image displays the slicing functionality of the plugin with the Nearest Neighbor interpolation . The right image displays the slicing functionality of the plugin with the Linear interpolation.

B.6. Discussion

Al thought the volume viewer has several options for the visualization of the data there are still a few things that could be added to improve the viewer.

One thing that could be added is a plane viewer. This plane viewer could be used to select a specific section inside the 3D data set that can then be sliced using the planes. This would improve on the slicing options that are currently in the viewer.

Another improvement that could be made is implementing a lighting source and shadows in the visualization. This would improve the readability of the 3D object. This was not done due to time constraints and color mapping options having priority.

A third improvement that could be made is to investigate and eliminate the problem that is created due to the linear interpolation. Currently it creates a colored shell around the data object due to the transition from a data point that has a value in the object and the background.

Bibliography

Ahrens, J., Geveci, B., Law, C., Hansen, C., & Johnson, C. (2005). 36-paraview: An end-user tool for large-data visualization. *The visualization handbook*, *717*, 50038–1.

Attiya, B., Liu, I.-H., Altimemy, M., Daskiran, C., & Oztekin, A. (2018). Vortex identification in turbulent flows past plates using lagrangian method. *Canadian Journal of Physics*, *97*. https://doi.org/10.1139/cjp-2018-0625

Cabral, B., & Leedom, L. C. (1993). Imaging vector fields using line integral convolution. *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, 263–270. https://doi.org/10.1145/166117.166151

Calkoen, E. E., de Koning, P. J., Blom, N. A., Kroft, L. J., de Roos, A., Wolterbeek, R., Roest, A. A., & Westenberg, J. J. (2015). Disturbed intracardiac flow organization after atrioventricular septal defect correction as assessed with 4d flow magnetic resonance imaging and quantitative particle tracing. *Investigative radiology*, *50*(12), 850–857.

ElBaz, M. S., Lelieveldt, B. P., Westenberg, J. J., & Geest, R. J. (2013). Automatic extraction of the 3d left ventricular diastolic transmitral vortex ring from 3d whole-heart phase contrast mri using laplace-beltrami signatures. *International Workshop on Statistical Atlases and Computational Models of the Heart*, 204–211.

Flow description, streamline, pathline, streakline and timeline. (2005). http://www-mdp.eng.cam.ac.uk/web/library/enginfo/aerothermal_dvd_only/aero/fprops/cvanalysis/node8.html

Garg, P., Crandon, S., Swoboda, P. P., Fent, G. J., Foley, J. R., Chew, P. G., Brown, L. A., Vijayan, S., Hassell, M. E., Nijveldt, R., et al. (2018). Left ventricular blood flow kinetic energy after myocardial infarction-insights from 4d flow cardiovascular magnetic resonance. *Journal of Cardiovascular Magnetic Resonance*, *20*(1), 61.

Haller, G. (2005). An objective definition of a vortex. *Journal of fluid mechanics*, *525*, 1–26.

Haralick, R. M., & Shapiro, L. G. (1992). *Computer and robot vision* (Vol. 1). Addison-wesley Reading.

Hinton, G. E., & Roweis, S. (2002). Stochastic neighbor embedding. *Advances in neural information processing systems*, *15*.

Hölt, T. (2021, February). Lecture of course data visualization tu delft.

Jeong, J., & Hussain, F. (1995). On the identification of a vortex. *Journal of fluid mechanics*, *285*, 69–94.

Jolliffe, I. T. (2002). Springer series in statistics. *Principal component analysis*, *29*, 15.

Julien, W. (2019). A tutorial on the proper orthogonal decomposition. *AIAA Aviation Forum*. https://doi.org/https://doi.org/10.14279/depositonce-8512

Kheradvar, A., & Pedrizzetti, G. (2012). Vortex formation in the heart. In *Vortex formation in the cardiovascular system* (pp. 45–79). Springer.

Kobak, D., & Berens, P. (2019). The art of using t-sne for single-cell transcriptomics. *Nature communications*, *10*(1), 5416.

Lcs tutorial: The finite-time lyapunov exponent. (2005). https://shaddenlab.berkeley.edu/uploads/LCS-tutorial/FTLE-derivation.html

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Pezzotti, N., Höllt, T., Lelieveldt, B., Eisemann, E., & Vilanova, A. (2016). Hierarchical stochastic neighbor embedding. *Computer Graphics Forum*, *35*(3), 21–30.

Qt Group, T. Q. C. (2024). Qt framework [https://www.qt.io/].

Quiver. (2022). mathworks.com/help/matlabef/quiver.html

Roos, P. R., Rijnberg, F. M., Westenberg, J. J., & Lamb, H. J. (2023). Particle tracing based on 4d flow magnetic resonance imaging: A systematic review into methods, applications, and current developments. *Journal of Magnetic Resonance Imaging*, *57*(5), 1320–1339.

Schroeder, W., Martin, K., & Lorensen, B. (2006). *The visualization toolkit (4th ed.)* Kitware.

Shadden, S. C., Lekien, F., & Marsden, J. E. (2005). Definition and properties of lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, *212*(3-4), 271–304.

Tauro, F., Grimaldi, S., & Porfiri, M. (2014). Unraveling flow patterns through nonlinear manifold learning. *PloS one*, *9*(3), e91131.

Tenenbaum, J. B., Silva, V. d., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, *290*(5500), 2319–2323.

Tran, M.-H. (2022, January). Comparing umap vs t-sne in single-cell rna-seq data visualization, simply explained. https://blog.bioturing.com/2022/01/14/umap-vs-t-sne-single-cell-rna-seq-data-visualization/

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, *9*(11).

Vasanawala, S. S., Hanneman, K., Alley, M. T., & Hsiao, A. (2015). Congenital heart disease assessment with 4d flow mri. *Journal of Magnetic Resonance Imaging*, *42*(4), 870–886. https://doi.org/https://doi.org/10.1002/jmri.24856

Vieth, A., Kroes, T., Thijssen, J., van Lew, B., Eggermont, J., Basu, S., Eisemann, E., Vilanova, A., Höllt, T., & Lelieveldt, B. (2024). Manivault: A flexible and extensible visual analytics framework for high-dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, *30*(1), 175–185. https://doi.org/10.1109/TVCG.2023.3326582

Wu, J., Wang, J., Xiao, H., & Ling, J. (2017). Visualization of high dimensional turbulence simulation data using t-sne. *19th AIAA Non-Deterministic Approaches Conference*, 1770.