Visualization and Analysis of Flow Fields based on Clifford Convolution

Von der Fakultät für Mathematik und Informatik
der Universität Leipzig
angenommene

DISSERTATION

zur Erlangung des akademischen Grades

DOCTOR RERUM NATURALIUM
(Dr. rer. nat.)

im Fachgebiet

INFORMATIK

vorgelegt

von Dipl.-Inform. Julia Ebling

geboren am 27.11.1978 in Mainz

Die Annahme der Dissertation haben empfohlen:

1. Professor Dr. Gerik Scheuermann (Universität Leipzig)
2. Professor Dr. Hans Hagen (TU Kaiserslautern)
3. Professor Dr. Klaus Mueller (Stony Brook University)

Die Verleihung des akademischen Grades erfolgt auf Beschluss des Rates der Fakultät für Mathematik und Informatik vom 17.7.2006 mit dem Gesamtprädikat summa cum laude.

Julia Ebling
Visualization and Analysis of Flow Fields based on Clifford Convolution
Universität Leipzig, Diss., 178 pages, 113 references, 69 figures, 10 tables

Selbstständigkeitserklärung

Hiermit erkläre ich, die vorliegende Dissertation selbstständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialien oder Dienstleistungen als solche gekennzeichnet.

Acknowledgments

This thesis would not have been possible without the enthusiasm and support of Prof. Dr. Scheuermann, who allowed me to work on this fascinating field between flow field visualization and analysis, image processing, and Clifford algebra. I want to thank him for many fruitful discussions, the possibility to present my work at several international conferences, his engagement in the search for interesting problems, applications and cooperations, the possibility to use the visualization tool FAnToM, and for his humor and high spirits.

I want to thank all the members of the FAnToM development team, first at the University of Kaiserslautern, then at the University of Leipzig. This includes Tom Bobach, Christoph Garth, David Gruys, Mario Hlawitschka, Kai Hergenröther, Nikolai Ivlev, Heike Jänisch, Max Langbein, Oliver Paech, Tobias Salzbrun, Michael Schlemmer, Xavier Tricoche and Alexander Wiebel. I further acknowledge the fruitful discussions in the whole computer graphics group at Kaiserslautern and the signal and image processing group at Leipzig. Special thanks go to Prof. Dr. Hagen, because he fascinated me in the field of computer graphics and visualization in the first place. Thanks go also to Prof. Krüger, FB Physik, Kaiserslautern for his help with Clifford algebra.

I was allowed to use some really interesting and relevant datasets. This was a major incentive for the development of useful algorithms as the data sets came mostly from real-world applications. Therefore thanks go to:

- Prof. Kollmann, department of mechanical and aeronautical engineering, University of California at Davis, for producing the swirling jet data set.
- Markus Rütten, DLR Göttingen, for providing the gas furnace chamber, the delta wing, and the ice train data sets.
- Berend Van der Wall, DLR Braunschweig and the complete HART II team, consisting of members from DLR Germany, NASA Langley and US Army AFDD, the French ONERA and the Dutch DNW. They provided 3C-PIV real world high quality data for analysis and practical application of the methodologies.

And last but not least, I want to thank my parents Barbara Schulte-Ebling and Richard-Jürgen Ebling, and my brothers Fabian and Frederik Ebling, for their everlasting assistance and encouragement.

Abstract

Vector fields from flow visualization often contain millions of data values. It is obvious that a direct inspection of the data by the user is tedious. Therefore, an automated approach for the preselection of features is essential for a complete analysis of nontrivial flow fields. This thesis deals with automated detection, analysis, and visualization of flow features in vector fields based on techniques transfered from image processing. This work is build on rotation invariant template matching with Clifford convolution as developed in the diploma thesis of the author. A detailed analysis of the possibilities of this approach is done, and further techniques and algorithms up to a complete segmentation of vector fields are developed in the process.

One of the major contributions thereby is the definition of a Clifford Fourier transform in 2D and 3D, and the proof of a corresponding convolution theorem for the Clifford convolution as well as other major theorems. This Clifford Fourier transform allows a frequency analysis of vector fields and the behavior of vector-valued filters, as well as an acceleration of the convolution computation as a fast transform exists.

The depth and precision of flow field analysis based on template matching and Clifford convolution is studied in detail for a specific application, which are flow fields measured in the wake of a helicopter rotor. Determining the features and their parameters in this data is an important step for a better understanding of the observed flow. Specific techniques dealing with subpixel accuracy and the parameters to be determined are developed on the way. To regard the flow as a superposition of simpler features is a necessity for this application as close vortices influence each other. Convolution is a linear system, so it is suited for this kind of analysis. The suitability of other flow analysis and visualization methods for this task is studied here as well.

The knowledge and techniques developed for this work are brought together in the end to compute and visualize feature based segmentations of flow fields. The resulting visualizations display important structures of the flow and highlight the interesting features. Thus, a major step towards robust and automatic detection, analysis and visualization of flow fields is taken.

Contents

Chapter 1

Introduction

”One of the most interesting aspects of the world is that it can be considered to be made up of patterns.” (Norbert Wiener)

Visualization and analysis of vector fields from flow simulations and measurements is an important step in engineering processes, e.g. during the design phase of airplanes, cars, trains, and combustion chambers. A specific role in these visualizations play features, which are often defined as ”phenomena, structures or objects in a data set, that are of interest for a certain research or engineering problem” [73]. It is not possible to give a list of all features of interest in a flow field as these differ from application to application and small changes of one feature lead to a variety of new features. Nevertheless, most features can be categorized into a few groups like vortices and other swirling flows, shock waves, shear flow and boundary layers, reversed flow, saddle points, separation and attachment lines or surfaces, areas with convergent or divergent behavior, and regions with laminar flow.

Most flow simulations and measurements want to study overall structure and specific features, i.e. patterns of streamlines with conspicuous behavior. Flow visualization intends to help the user to find and analyze features and structures. Direct visualization methods like hedgehogs do not always reveal the features. Streamline based methods may lead to missing features, too, especially without knowing the right starting points. Texture based methods like line integral convolution [13] do a quite good job in 2D, but a convincing solution in 3D is still missing. Topology [18, 44, 79, 81, 94, 96, 97], on the other hand, is directed to the overall structure since not all features are easily connected to it. Furthermore, the presentation of 3D topology produces visibility problems. Another, quite different, approach for the visualization is to use information visualization methods like brushing for interactive exploration of the data sets [22].

A huge amount of data is generated nowadays by flow simulations and measurements. The resulting vector fields often contain millions of data values, but even for small datasets with only thousands of values, direct inspection by the user is tedious and features are missed easily. Therefore, many automated feature detection methods have been developed in the last years (e.g. [24, 42, 57, 73, 77, 90]. Streamlines can then be used in a second step to study the features. Earlier attempts usually try to give an analytic model of a feature and create an algorithm for feature detection from there. Besides the limitations of the models, most approaches have severe robustness problems.

Image processing and computer vision are mature fields and have produced methods for analysis, feature extraction and derivative computation [51, 52]. Convolution based approaches are robust in terms of white noise because of the inherent averaging present in the convolution method. Furthermore, many image processing methods allow precise analysis of accuracy for operations including sampling, interpolation and smoothing. Noise is also suitably modeled and dealt with. Therefore, it is sensible to apply these methods to vector fields.

In her diploma thesis [23], the author has developed Clifford convolution, which is a transfer of convolution to vector fields with the help of Clifford algebra. Clifford convolution has some nice properties as it unifies the convolution of scalar, vector and spinor fields. Furthermore, a rotation invariant template matching method based on geometric properties of this convolution was developed. These algorithms have been fundamental work on which this dissertation thesis is build.

In image processing, the convolution and Fourier transform operators are closely related by the convolution theorem. One major contribution of this thesis is the extension of the Fourier transform to include general elements of Clifford Algebra, which are called multivectors and include scalars and vectors. The resulting convolution and derivative theorems extend those applicable to the Fourier transform on scalar fields while still remaining reasonably simple. The Clifford Fourier transform allows a frequency analysis of vector fields and the behavior of vector-valued filters. In frequency space, vectors are transformed into general multivectors of the Clifford Algebra. Many basic vector-valued patterns such as source, sink, saddle points and potential vortices can be described by a few multivectors in frequency space. Furthermore, the existence of fast algorithms for the computation of the Clifford Fourier transform together with the corresponding convolution theorem allow an acceleration of the computation of the Clifford convolution.

Vector-valued templates are linear and shift invariant filters. The linearity property is also known as the superposition principal, and a well known property in signal and image processing. In flow field visualization, however, this perception has been neglected. In some applications [100], the user is not interested as much in the actual flow behavior as in an approximation of the phenomena us-

ing simpler features models. Then, superposition effects have to be taken into account in order to analyze these features and compute their parameters. This is inherently done by using template matching. One question, which is examined in this thesis, is how other basic flow analysis and visualization techniques cope with superposition phenomena.

One application and test for feature analysis based on Clifford convolution are three-component particle image velocimetry (3-C PIV) measurements within the wake of a helicopter rotor from the HART II test. These PIV-images are quite a challenge as the noise due to the measurement method and the inherent turbulence of the flow can not be distinguished. Furthermore, features are often hidden by a flow-through component, which is influenced by vortices and therefore not easy to determine. Several image processing methods based on convolution, integration, bisection, and the orientation tensor, are investigated in this thesis in order to determine parameters of vortices given by a cut of the vortices with the image plane. Position, size, direction and velocity at the core radius can be determined quite robust and precise with the presented methods, for an accuracy of edge-length as well as subpixel accuracy. The methods are all independent of any mean flow and robust in terms of noise, which is both important for the evaluation of the HART II PIV images. Thus, this application serves also as an example and instruction for flow feature analysis based on template matching.

Finally, the gained knowledge and techniques are brought together in an algorithm for automatic computation of a feature-based segmentation of 2D, 3D, and time dependent vector fields. The template matching allows an unified approach for the detection of different features, and the segmentation is based on the features themselves. Another advantage is that feature models used by engineers can be coded into the templates and thus automatic determination of the model parameters is possible within this framework as well. Several challenges of designing the algorithm like misclassification and superposition effects are presented and discussed. The visualizations of the segmentations clearly depict the structures of the flow data as the features are displayed in conjunction with each other. Thus, even highly turbulent data can be studied easily.

Structure of this Thesis

Chapter 1: In this chapter, **introduction** and overview of the topics discussed in this thesis are given. Furthermore, the structure of this thesis is described and the publications, which arose during the work on this thesis, are enumerated.
Chapter 2: This chapter gives an introduction to **flow field visualization**. Central definitions concerning flow fields and their discretization are given there. Flow visualization is closely connected to feature detection. Therefore direct flow visualization techniques are presented as well as common feature definitions and

feature detection for subsequent visualization.
Chapter 3: **Signal and image processing** fundamentals are discussed in the third chapter. This includes the definitions of signals and systems, especially the class of linear and shift invariant systems, which also play an important role in this thesis. Discretization issues are discussed in this chapter as well as the Fourier transform and Gabor filters. Techniques for feature detection and segmentation of images are presented, including approaches for rotation and scale invariant matching.
Chapter 4: **Clifford algebra** is a geometric algebra. Once understood, it provides an intuitive access to vector fields as well as a unified notation for scalar, vector and spinor data. This thesis is based on Clifford convolution which makes extensive use of the properties of this algebra, therefore an introduction is given in Chapter 4. Differing definitions of Clifford algebras are discussed there as well.
Chapter 5: Several definitions of convolution of vector fields have been developed so far, **Clifford convolution** being one of them. Their advantages and disadvantages are discussed in Chapter 5. Furthermore, two rotation invariant template matching techniques based on two different convolution definitions, based on scalar product and Clifford multiplication of two vectors, are explained.
Chapter 6: There are a few **data sets** which are repeatedly used to demonstrate techniques and results in this thesis. For a more structured approach, all data sets are presented and explained in detail in this chapter. The templates and vortex definitions used in this thesis are introduced there, too.
Chapter 7: Convolution and Fourier transform are closely connected by the convolution theorem. By transferring convolution into Clifford algebra, the question of an appropriate Fourier transform arose. In Chapter 7, a definition of a **Clifford Fourier transform** for Clifford algebra for 2D and 3D is given and major theorems including the convolution theorems are stated and proved. This led also to the definition of **Clifford Gabor filters**, which can be interpreted as a Clifford Fourier transform localized by a Gabor filter. Furthermore, typical vector valued flow patterns are analyzed in frequency domain in this chapter.
Chapter 8: This chapter is a collection of several issues and techniques concerning the application of template matching based on Clifford convolution to flow fields. So far, Clifford convolution was only defined on uniform grids. As many flow fields are defined on **irregular grids** and even **surfaces**, the template matching technique had to be transfered to these kinds of grids. Another topic is the perception of a flow field as a **superposition** of several, possibly simpler flow fields, and the response of basic flow analysis and visualization techniques to this phenomena. The application of vector-valued template matching to **feature analysis** is described in detail for the example of a specific application. All acquired knowledge is then brought together in an algorithm for a feature based **segmentation** of flow fields.

Overview of Publications

This thesis is based on the following publications of the author:

- Julia Ebling and Gerik Scheuermann. Clifford Convolution And Pattern Matching On Vector Fields. IEEE Visualization 2003. [24]
- Mario Hlawitschka and Julia Ebling and Gerik Scheuermann. Convolution and Fourier Transform of Second Order Tensor Fields. IASTED VIIP 2004. [49]
- Julia Ebling and Gerik Scheuermann. Pattern Matching on Vector Fields using Gabor Filter. IASTED VIIP 2004. [25]
- Julia Ebling, Gerik Scheuermann and Berend G. van der Wall. Analysis and Visualization of 3-C PIV Images from HART II using Image Processing Methods. Data Visualization 2005. [30]
- Julia Ebling, Gerik Scheuermann. Clifford Fourier Transform on Vector Fields. IEEE Transactions on Visualization and Computer Graphics, 2005. [27]
- Julia Ebling and Gerik Scheuermann. Clifford Convolution And Pattern Matching On Irregular Grids. Scientific Visualization: The Visual Extraction of Knowledge from Data, 2005. [26]
- Julia Ebling and Gerik Scheuermann. Segmentation of Flow Fields using Pattern Matching. Data Visualization 2006. [28]
- Julia Ebling and Gerik Scheuermann. Template Matching on Vector Fields using Clifford Algebra. IKM 2006. [29]
- Julia Ebling, Alexander Wiebel, Christoph Garth and Gerik Scheuermann. Topology Based Flow Analysis and Superposition Effects. Submitted to Proceedings of TopoInVis2005 [31]

Notation

notation	meaning
p	scalar
$\mathbf{v}$	vector
$\hat{\mathbf{v}}$	normalized vector
s	spinor
$\mathbf{A}$	multivector
T	matrix
t	time value
$\mathbf{x}$	position in $\mathbb{R}^n$
$f()$	continuous function or signal
$f[]$	discrete function or signal
$f\{\}$	system
$\mathscr{F}\{\}$	Fourier transform
$p(\mathbf{x})$	scalar field
$\mathbf{v}(\mathbf{x})$	vector field
$s(\mathbf{x})$	spinor field
$\mathbf{A}(\mathbf{x})$	multivector field
i	imaginary number
e	unit
n	dimension
N	quantity

Chapter 2

Flow Field Visualization

A lot of flow data sets are generated nowadays, either by simulation or by measurement. The application ranges from evaluation of the flight quality of aircraft and helicopters, to evaluating the drag of moving cars, trucks and trains, the mixing of gases and efficiency of combustion chambers, the comfort of closed compartments with regards to the air condition, and many more.

Over the years, many flow visualization methods have been developed. Often they are variants of a few basic techniques: glyphs, particle tracing, visualization of derived quantities, and feature detection with subsequent visualization of the results. In this chapter, an introduction to flow fields and their visualization is given which concentrates on basic approaches. For an extensive overview of flow visualization, the reader is referred to surveys [40, 72, 73].

2.1 Flow Fields

In all practicable applications, vector fields describing flow are given in 2 or 3 dimensions. Sometimes the vector field will be given on an arbitrary curved surface, therefore the definition of vector fields needs to be done on arbitrary manifolds:

Definition 2.1.1 *Let TM be a tangent bundle, that is a collection of all tangents along with their position, and let $T_{\boldsymbol{x}}M$ be the tangent space associated with the point $\boldsymbol{x}$. Let M be a smooth m-manifold with boundary and $N \subset M$ be a smooth n-dimensional sub-manifold with boundary. Let $I \subset \mathbb{R}$ be an open interval of real numbers. Then the map*

$$\boldsymbol{v} : N \times I \rightarrow TM$$

with

$$\boldsymbol{v}(\boldsymbol{x}, t) \in T_{\boldsymbol{x}}M$$

is called **time-dependent vector field**. *If*

$$\boldsymbol{v} : N \times I \to TN \subset TM,$$

then $\boldsymbol{v}$ *is called* **tangential time-dependent vector field**.

In the following, only tangential vector fields will be of interest. Most of the time the vector field will be given in Euclidean space. Then, the definition of tangential vector fields can be simplified:

Definition 2.1.2 *Let* $\Omega \subset \mathbb{R}^n$ *be an Euclidean space and let* $I \subset \mathbb{R}$ *be an open interval of real numbers. Then the map*

$$\begin{aligned} \boldsymbol{v} : \Omega \times I &\to \mathbb{R}^n \\ (\boldsymbol{x}, t) &\mapsto \boldsymbol{v}(\boldsymbol{x}, t) \end{aligned}$$

is a **tangential time-dependent vector field**.

Vector fields which are time-independent are usually called steady. In the following, vector fields will always denote a steady vector fields if not indicated otherwise. Integral curves and vector fields are closely interrelated as vector fields can describe the derivatives of curves, e.g. curves of the path of particles released into a flow:

Definition 2.1.3 *Let* M *be a smooth manifold with boundary, let* TM *be a tangent bundle. Let* $0 \in J \subset I \subset \mathbb{R}$ *be open intervals of real numbers, and let* $\boldsymbol{v} : M \times I \to TM$ *be continuous. Let* $\boldsymbol{x}_0 \in M$. *A* **pathline** *for* $\boldsymbol{v}$ *with* **initial condition** $\boldsymbol{x}_0$ *is a* C^1*-map*

$$\begin{aligned} \alpha : J &\to M \\ \alpha(0) &= \boldsymbol{x}_0 \\ \alpha'(t) &= \boldsymbol{v}(\alpha(t), t) \; \forall t \in J \end{aligned}$$

An C^1*-map with initial condition* x_0 *and*

$$\alpha'(t) = \boldsymbol{v}(\alpha(t), \tau) \; \forall t \in J, \tau \in I$$

is called **streamline**.

Pathlines follow a particle over time, and streamlines are integral curves within a fixed timestep. Another important type of curves are streaklines which are gained by constantly releasing particles into the flow and then taking a snapshot of all particles at once. For steady, time-independent vector fields pathlines, streamlines and streaklines are the same.

In some applications, the magnitude of the velocity is neglected and only the directions are of interest.

Definition 2.1.4 *Let $\boldsymbol{v}$ be a vector field. Then the field $\hat{\boldsymbol{v}}$ given by*

$$\hat{\boldsymbol{v}}(\boldsymbol{x}) \begin{cases} \frac{\boldsymbol{v}(\boldsymbol{x})}{\|\boldsymbol{v}(\boldsymbol{x})\|} & \|\boldsymbol{v}(\boldsymbol{x})\| \neq 0 \\ 0 & else \end{cases}$$

is called the **normalized** *vector field.*

Note that this definition is in contrast to the usual definition of normalizing, where the range of values is projected onto a scaled range.

2.1.1 Streamlines and Velocity

In this subsection, the relation of the velocity magnitude of the flow to the resulting streamlines is analyzed. Streamlines are independent of the velocity magnitude of the flow so far as changing the velocity magnitude corresponds to a reparametrization of the streamline:

Theorem 2.1.5 *Let $D \subset \mathbb{R}^n$ be an open domain and $\boldsymbol{v} : D \to \mathbb{R}^n$ a vector field satisfying the Lipschitz condition. Let $Z := \{\ \boldsymbol{z} \in D \mid \boldsymbol{v}(\boldsymbol{z}) = 0\ \}$ be the set of critical points and $D_\varepsilon := \{\ \boldsymbol{x} \in D \mid \forall \boldsymbol{z} \in Z : |\boldsymbol{x} - \boldsymbol{z}| > \varepsilon\ \}$ the domain without the critical points and their ε-neighborhood. Let $\boldsymbol{w}$ be the vector field $\boldsymbol{v}$ without the critical points and their ε-neighborhood, and $\hat{\boldsymbol{w}}$ the corresponding normalized vector field:*

$$\begin{aligned} \boldsymbol{w} : D_\varepsilon \to \mathbb{R}^d, \qquad & \boldsymbol{x} \mapsto \boldsymbol{v}(\boldsymbol{x}) \\ \hat{\boldsymbol{w}} : D_\varepsilon \to \mathbb{R}^d, \qquad & \boldsymbol{x} \mapsto \frac{\boldsymbol{w}(\boldsymbol{x})}{|\boldsymbol{w}(\boldsymbol{x})|}. \end{aligned}$$

For $\boldsymbol{x}_0 \in D_\varepsilon$ and $I = (0, t_{max})$ let $\alpha : I \to D_\varepsilon$ be the well-defined streamline of $\boldsymbol{w}$ through $\boldsymbol{x}_0$ with maximal length, that is

$$\begin{aligned} \alpha(0) &= \boldsymbol{x}_a \\ \frac{\partial \alpha}{\partial t}(t) &= \boldsymbol{w}(\alpha(t)). \end{aligned}$$

Define the following mapping for reparametrization which makes use of the arc length:

$$l : I \to l(I), \quad l(t) = \int_0^t |\boldsymbol{w}(\alpha(\tau))|\, d\tau.$$

This function is strictly monotonic and therefore invertible. Then the streamline $\tilde{\alpha}$, a reparametrization of α using this mapping, is the well-defined streamline of $\hat{\boldsymbol{w}}$ through starting point $\boldsymbol{x}_0$:

$$\tilde{I} = (l(0), l(t_{max})) = l(I)$$

$$\tilde{\alpha} : \tilde{I} \to D_\varepsilon,\ s \mapsto \alpha(l^{-1}(s))$$

Proof*:*

$$\begin{aligned}
\tilde{\alpha}(0) &= \tilde{\alpha}(l(0)) = \alpha(0) = \boldsymbol{x}_0 \\
\frac{\partial \tilde{\alpha}}{\partial s}(s) &= \frac{\partial \alpha}{\partial s}(l^{-1}(s)) \\
&= \boldsymbol{w}(\alpha(l^{-1}(s)))\frac{1}{\frac{\partial l(t)}{\partial t}} \\
&= \boldsymbol{w}(\alpha(l^{-1}(s)))\frac{1}{|\boldsymbol{w}(\alpha(l^{-1}(s)))|} \\
&= \boldsymbol{w}(\tilde{\alpha}(s))\frac{1}{|\boldsymbol{w}(\tilde{\alpha}(s))|} \\
&= \hat{\boldsymbol{w}}(\tilde{\alpha}(s)).
\end{aligned}$$

This means that the streamlines of $\mathbf{w}$ are the same as the streamlines of $\hat{\mathbf{w}}$, the normalized field.

2.1.2 Grid Types

In practice, flow fields are gained by simulations or measurement. Therefore the data is not continuous but discrete. In this section, the underlying structures of discrete data are defined. First of all, the term simplex is introduced:

Definition 2.1.6 *Let $M \subset \mathbb{R}^n$ be a finite set. Let $\{\boldsymbol{x}_0, .., \boldsymbol{x}_r\} \subset M$ be $r+1$ geometrically independent points. Then*

$$S = \{\boldsymbol{x} \in \mathbb{R}^n | \boldsymbol{x} = \sum_{j=0}^{r} \lambda_1 \boldsymbol{x}_j,\ \sum_{j=0}^{r} \lambda_j = 1,\ \lambda_j \geq 0, j = 0, .., r\}$$

is a **r-simplex over** M **of the vertices** $\{\mathbf{x}_0, .., \mathbf{x}_r\}$.

Note that a 0-simplex is a point, a 1-simplex a line, a 2-simplex a plane, and a 3-simplex a volume. These simplices can be brought together to form cells:

Definition 2.1.7 *Let $M \subset \mathbb{R}^n$ be a finite set. A finite union*

$$C = \bigcup_{j=0}^{r} S_j$$

of q-simplices S_j over P is called a **q-cell over P** *if the following properties are satisfied:*

1. *C is simply connected.*
2. *The intersection of any two q-simplices of C is a k-simplex of C with $k < q$, or empty.*
3. *There exists a q-dimensional affine subspace of $\mathbb{R}^n$ containing the cell C.*

Often used cell types are quads and triangles in 2D, and cubes, tetrahedrons and hexahedrons in 3D, though many more cell types can occur. A grid is a collection of points together with neighborhood information coded in cells:

Definition 2.1.8 *Let $\{C_0,..,C_r\}$ be a set of q-cells with $C_m \cap C_n = S$, S is a k-simplex with $k < q$, $m \neq n$. Then*

$$G = \bigcup_{j=0}^{r} C_j$$

is called **grid**.

Various grid types are distinguished (Figure 2.1):

Definition 2.1.9 *Let $\boldsymbol{x}_j \in \mathbb{R}^n$, $j = 0,..,k$ be the positions at which flow information is given. If no neighborhood information of the data points is given, the data is called* **scattered data**. *When the data points are given as nodes of a grid, and no other information about a structure of the grid can be given, the grid is called* **unstructured grid**. *When*

$$\boldsymbol{x}_j = \boldsymbol{x}(k_1,..,k_n),\ k_m \in \mathbb{N},\ m = 1,..,n,$$

it is called **curvilinear grid**. *A curvilinear grid with*

$$\boldsymbol{x}_j = \boldsymbol{x}(k_1,..,k_n) = \boldsymbol{x}_0 + \begin{pmatrix} k_1\Delta_{1,k_1} \\ \vdots \\ k_n\Delta_{n,k_n} \end{pmatrix}$$

is called **rectilinear grid**, *and if*

$$\boldsymbol{x}_j = \boldsymbol{x}(k_1,..,k_n) = \boldsymbol{x}_0 + \Delta \begin{pmatrix} k_1 \\ \vdots \\ k_n \end{pmatrix},$$

the grid is a **uniform grid**.

Within the cells of a grid, data values are gained by using linear interpolation. Thus, continuous vector fields are approximated.

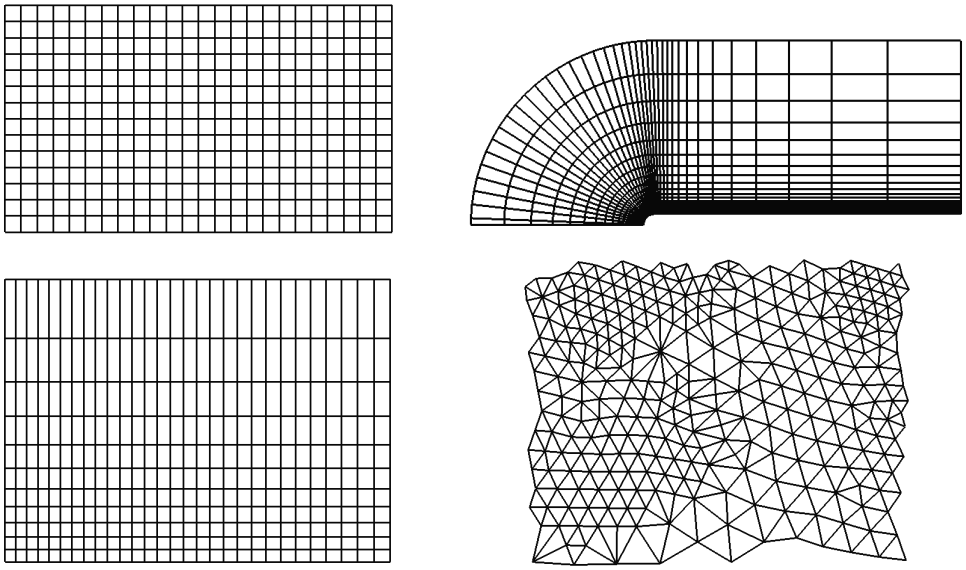

Figure 2.1: **Top left**: A uniform grid. **Top right**: A curvilinear grid. **Bottom left**: A rectilinear grid. **Bottom right**: An unstructured triangle grid.

2.2 Derived Quantities

A vector field has many derived quantities like vorticity, divergence and many more. These are often used for direct visualization of the vector field as well as for feature detection. Therefore, some of the derived values are presented here. The Jacobian or gradient of a vector field is the source of many of these values.

Definition 2.2.1 *Let* $\{\boldsymbol{e}_1,..,\boldsymbol{e}_n\}$ *be a basis of* R^n. *Let* p *be a scalar field. The* **partial derivative** *of* p *with respect to* $\boldsymbol{e}_j$ *is* $\frac{\partial p}{\partial \boldsymbol{e}_j}$, $j = 1,..,n$. *The* **gradient** *of* p *is the vector of its partial derivatives*

$$grad\ p = \nabla p = (\frac{\partial p}{\partial \boldsymbol{e}_1},..,\frac{\partial p}{\partial \boldsymbol{e}_n})^T.$$

The gradient of a vector field $\boldsymbol{v} = (\boldsymbol{v}_1,...\boldsymbol{v}_n)^T$ *is called* **Jacobian** *and is defined as*

$$\nabla \boldsymbol{v} = \begin{bmatrix} \frac{\partial \boldsymbol{v}_1}{\partial \boldsymbol{e}_1} & \cdots & \frac{\partial \boldsymbol{v}_1}{\partial \boldsymbol{e}_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \boldsymbol{v}_n}{\partial \boldsymbol{e}_1} & \cdots & \frac{\partial \boldsymbol{v}_n}{\partial \boldsymbol{e}_n} \end{bmatrix}$$

The vorticity or curl is defined as the local circulation or rotation of a vector field:

Definition 2.2.2 *Let $\boldsymbol{v}$ be a vector field and let $\times$ be the cross product. Then the* **vorticity** *or* **curl** *of $\boldsymbol{v}$ is defined as*

$$curl\ \boldsymbol{v} = \nabla \times \boldsymbol{v}$$

The **divergence** *is defined as*

$$div\ \boldsymbol{v} = \nabla \cdot \boldsymbol{v} = \sum_{j=1}^{n} \frac{\partial \boldsymbol{v}_j}{\partial \boldsymbol{e}_j}$$

Helicity is defined as the cosine between velocity and vorticity:

Definition 2.2.3 *Let $\boldsymbol{v}$ be a vector field and $\boldsymbol{w}$ its vorticity. The* **helicity** *of $\boldsymbol{v}$ at $\boldsymbol{x}$ is given by*

$$\frac{\langle \boldsymbol{v}(\boldsymbol{x})\boldsymbol{w}(\boldsymbol{x})\rangle}{\|\boldsymbol{v}(\boldsymbol{x})\|\|\boldsymbol{w}(\boldsymbol{x}))\|}.$$

Another quantity of 3D vector fields which is often used for vortex detection is λ_2. Before it can be defined, the decomposition of a matrix into its eigenvalues and eigenvectors has to be introduced.

Definition 2.2.4 *Let T be a complex-valued $n \times n$ matrix. An* **eigenvalue** *of T is a scalar $\lambda \in \mathbb{C}$ which solves the equation $M\boldsymbol{x} = \lambda \boldsymbol{x}$. $\boldsymbol{x}$ is the corresponding non-zero* **eigenvector** *of T.*

Note that the eigenvectors and eigenvalues of of a Jacobian indicate the direction of tangential curves of the flow.

Definition 2.2.5 *Let $\boldsymbol{v}$ be a vector field in $\mathbb{R}^3$ and $J = \nabla \boldsymbol{v}$ the velocity gradient tensor. Let $S = \frac{J+J^T}{2}$ be the symmetric part of J and let $\Omega = \frac{J-J^T}{2}$ be the antisymmetric part. S is also called deformation or rate of strain tensor, and Ω is called spin tensor. Let the eigenvalues of $S^2 + \Omega^2$ be ordered by magnitude. Then λ_2 is the middle eigenvalue.*

2.3 Direct Flow Visualization Techniques

A basic example of direct visualization are glyphs serving as local flow probes. Most of the time, arrows are used as glyphs. The resulting visualization is called a hedgehog:

Definition 2.3.1 *Let $v(x) = v(x,\tau)$ be a vector field. A set of local flow probes is called* **hedgehog** *if the local flow probes are arrows with direction $\frac{v(x)}{\|v(x)\|}$ and magnitude proportional to $\|v(x)\|$.*

A disadvantage of glyphs, especially in 3D, is visual clutter. Another possible visualization of vector fields is particle tracing, that is the display of streamlines, pathlines and streaklines. The lines better represent the continuity of the flow. In the following, mostly streamlines are mentioned. Placing the streamlines can be a difficult problem between visually missing features because of clutter and occlusion, and missing features because of not placing an adequate streamline. One solution is to detect the features first and use the results for streamline placing. There also exist some other approaches trying to give a complete overview of the streamlines. One of these methods is line integral convolution:

Definition 2.3.2 *Let $\boldsymbol{v}$ be a vector field in $\mathbb{R}^2$. Let p be a scalar field in $\mathbb{R}^2$ defined over the same region as the vector field. Let the values of p be computed as white noise. Then the image computed by locally smoothing p along streamlines of v is called a* **line integral convolution (LIC)** *image [13].*

Another approach is to compute and visualize the topology of the flow, that is a segmentation based on streamline start and end regions. This works well because streamlines can not cross each other.

Definition 2.3.3 *Let $\boldsymbol{v} : B \subset \mathbb{R}^n \to \mathbb{R}^n$ be a vector field. Let $\boldsymbol{p} \in B$ be a position within the vector field. Let $c_{\boldsymbol{p}}(t)$ be the streamline of $\boldsymbol{v}(\boldsymbol{x})$ through $\boldsymbol{p}$. Then*

$$A(c_{\boldsymbol{p}}) = \left\{\boldsymbol{x} \in B | \exists (t_n)_n = 0^\infty \subset \mathbb{R}, t_n \to -\infty, lim_{n\to\infty} c_{\boldsymbol{p}}(t_n) = x\right\}$$

is the α **limit set** *of $c_{\boldsymbol{p}}$ and*

$$\Omega(c_{\boldsymbol{p}}) = \left\{\boldsymbol{x} \in B | \exists (t_n)_n = 0^\infty \subset \mathbb{R}, t_n \to \infty, lim_{n\to\infty} c_{\boldsymbol{p}}(t_n) = x\right\}$$

is the ω **limit set** *of $c_{\boldsymbol{p}}$. Let $M \subset B$. The sets*

$$B_\alpha(M) = \left\{\boldsymbol{p} \in B | A(c_{\boldsymbol{p}}) = M\right\}$$

and

$$B_\omega(M) = \left\{\boldsymbol{p} \in B | \Omega(c_{\boldsymbol{p}}) = M\right\}$$

are called α **basin** *and ω* **basin** *of M, respectively. A segmentation of B into simply connected components of intersections of α and ω basins is called* **flow topology**.

For 2D linear vector fields, the computation of the flow topology can also be done via critical points and separatrices:

Definition 2.3.4 *Let $\boldsymbol{v}$ be a vector field. A position $\boldsymbol{x}$ with $\|\boldsymbol{v}(\boldsymbol{x})\| = 0$ is called a* **critical point**.

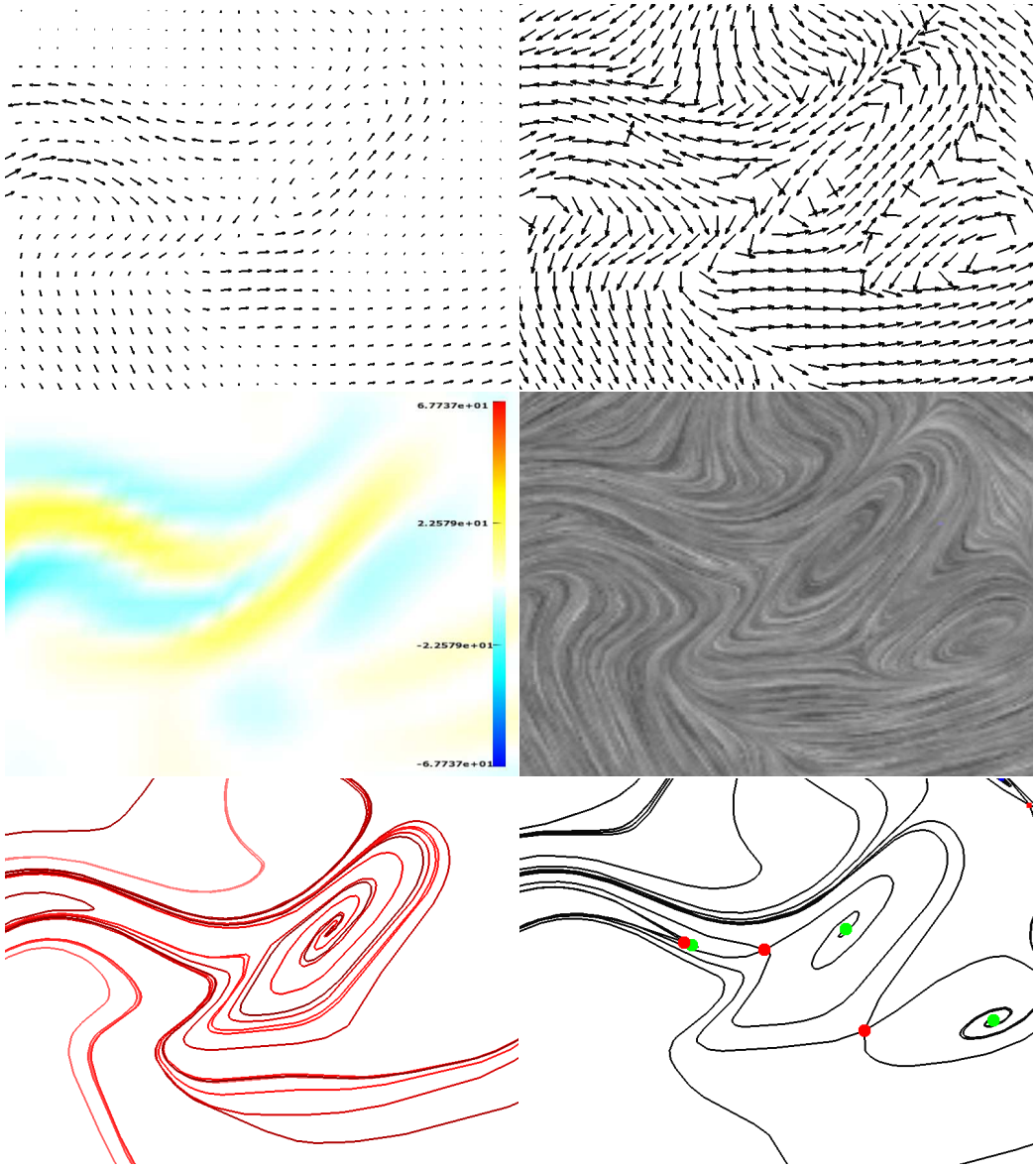

Figure 2.2: Examples of basic visualization techniques for a 2D data set. Shown is a section of the OM06 data set (Section 6.2). **Top left**: Hedgehog. **Top right**: Hedgehog, all arrows drawn with same length. **Middle left**: Color-coding of vorticity. **Middle right**: LIC. **Bottom left**: Streamlines. **Bottom right**: Topology with sources (green), saddles (red) and separatrices (black).

These critical points can be classified according to the eigenvalue decomposition (Definition 2.2.4) of their Jacobian (Definition 2.2.1) into sinks (ω limit sets), sources (α limit sets), and saddles. Starting streamlines at the saddles in direction

of the eigenvectors yields the so called **separatrices** which divide the flow into regions of same flow behavior with respect to start and end region. A **closed orbit** is a streamline c with initial condition $c(0) = \mathbf{x}_0$ and a parameter $\alpha \neq 0$ with $c(\alpha) = \mathbf{x}_0$. The set of all critical points and orbits together with the separatrices produce the topology of the vector field [44].

Directly visualizing derived scalar values of the flow like magnitude of vorticity is easily done in 2D using color-coding. In 3D, there exist two basic possibilities: Volume rendering, that is a 2D projection of the 3D data, and visualization of isosurfaces.

Definition 2.3.5 *Let p be a scalar field, and $\alpha \in \mathbb{R}$. The set of all values $\boldsymbol{x}$ with $p(\boldsymbol{x}) = \alpha$ is called* **isosurface**, *or* **isoline** *in 2D, of p at value α.*

For volume rendering, the choice of the transfer function, which assigns color and opacity to former scalar values, plays an important role for the quality of the images.

2.4 Feature Detection in Flow Fields

A huge amount of data is generated nowadays by flow simulations and measurements. The resulting vector fields often contain millions of data values, but even for small datasets with only thousands of values, direct inspection by the user is tedious and features are missed easily. Therefore, many automated feature detection methods have been developed in the last years to assist the user.

Pagendarm and Walter [67] and Walsum et al. [101] were the first to shape the term ”features” for flow fields. A feature is simply a region of interest in the dataset. It can be extracted using a feature criterion evaluation function. This function can be a logical combination of several scalar thresholds using boolean algebra. Often, and especially on vector-valued flow fields, the feature criterion does not evaluates the data itself but scalar functions defined on the data.

2.4.1 Feature Definitions

Most feature detection methods follow an analytic model and then employ a suitable algorithm, however, when features are not well defined, the limitations of the model introduce problems. One example for a feature definition which can not be defined precisely is one of the most important classes of features in flow fields: swirling flows or vortices. They are often described as a swirling motion around a central region. However, this description is not precise enough for an implementation of a feature detection algorithm. Therefore, over the years, several different

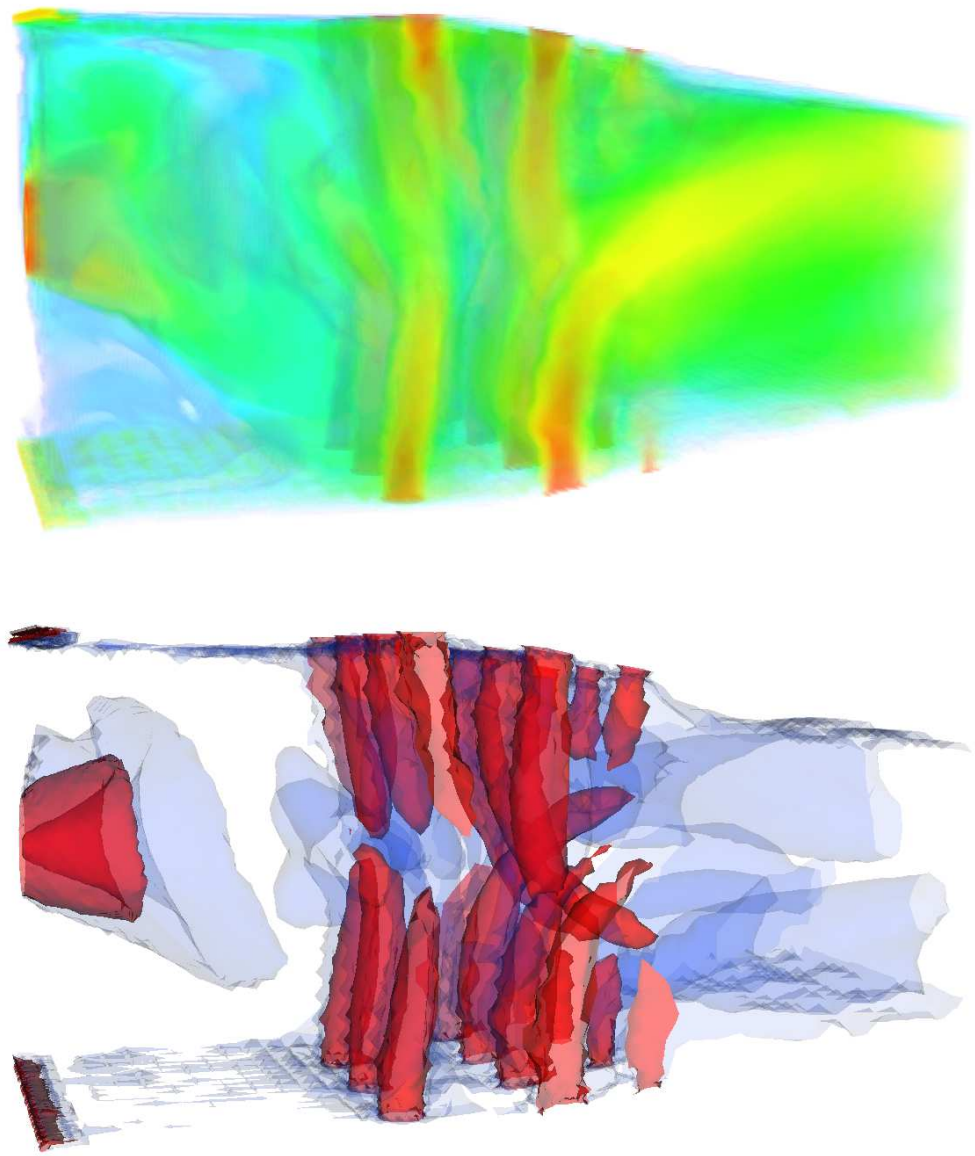

Figure 2.3: Examples of basic visualization techniques for a 3D data set. Shown is the velocity magnitude of a gas furnace chamber (Section 6.3). **Top**: Volume rendering. **Bottom**: Isosurfaces with isovalue 10 (blue) and 20 (red).

algorithms have been proposed which all describe part of the phenomenon ”vortex”. Some examples are the maximum of vorticity (Definition 2.2.2) and helicity (Definition 2.2.3) as well as the isosurface of λ_2 (Definition 2.2.5) of zero (or

of a slightly negative) value [53]. Another algorithm for vortex detection is the method of Sujudi and Haimes [90], which makes use of the classification of local flow similar to flow topology:

Definition 2.4.1 *Let* $\mathbf{v}$ *be a vector field and* $\mathbf{x}$ *a position within the vector field. Let J be the Jacobian (Definition 2.2.1) of the velocity. Compute the eigenvalue decomposition (Definition 2.2.4) of J. When two of the three eigenvalues are complex and the investigated vector is one of the eigenvectors, the vector is classified as belonging to a vortex core according to the* **vortex core detection algorithm of Sujudi and Haimes** *[90].*

The parallel-vectors operator as introduced by Roth [77] can be used for unifying the notations of the different definitions. Other vortex core detection algorithms are based on classifying the streamlines, e.g. using the winding angle which is a measure of the change of direction of streamline segments, or based on labeling the vectors belonging to one cell. With so many different vortex definitions assumed by the different algorithms, it is quite obvious that some kind of verification of the results has to take place which has to be left to the user. However, the user can be aided by highlighting potential vortex areas, and combining the results with direct visualizations of the flow.

Another example for model based feature extraction is due to Kenwright [57], for separation and attachment line detection. These lines show where the flow attaches itself to or separates from a surface. Kenwright gives two algorithms again based on eigenvector analysis of the velocity gradient tensor to extract separation and attachment lines.

Note that for most of the definitions mentioned here, derivatives have to be computed. These are sensitive to the presence of noise, which can make smoothing necessary. This robustness is especially important when dealing with measured as opposed to simulation data.

Image processing and computer vision are mature fields and have produced methods for analysis, feature extraction and derivative computation [51, 52]. Convolution based approaches are robust in terms of white noise because of the inherent averaging present in the convolution method. They also allow an intuitive definition of features. Furthermore, many image processing methods allow precise analysis of accuracy for operations including sampling, interpolation and smoothing. Noise is also suitably modeled and dealt with. Therefore, it is sensible to apply these methods to vector fields. The convolution operation has already been extended to vector fields which has lead to the creation of pattern matching algorithms for vector fields. These algorithms will be discussed in Chapter 5, as soon as the prerequisites have been defined and explained.

2.4.2 Galilean Invariant Features

The notion of Galilean invariance can play an important role for feature detection:

Definition 2.4.2 *The principle stating that the fundamental laws in physics are the same for every inertial frame of reference is called* **Galilean invariance**. *This also implies that lengths and times are unaffected by a change of velocity, e.g. the velocity of the observer. A property or feature is called Galilean invariant if it is unaffected by the speed and direction of a moving observer.*

Features which are not Galilean invariant will thus change with the frame of reference, and may not be detected at all under the wrong circumstances. Examples for Galilean invariant features are vorticity, derivatives and λ_2. Streamlines, however, are not Galilean invariant, which can be seen e.g. in Figure 2.4. Thus, feature detection and visualization based on streamlines is highly dependent on the frame of reference. Different vortices, for example, may appear or disappear when subtracting or adding different constant flows [99].

Regarding for example a Galilean invariant vortex detection, one method could be to compute the average and remove it from the vector field. But the vortices, though having zero average, can add to the average of the whole field as they are often assumed to be spread out infinitely (Section 6.1.1) and only a part of the vortex, not having zero average, might be within the data set. Thus, removing the average will change the results of a latter analysis of the data.

Another approach of Galilean invariant feature detection is to divide the flow field into three fields containing the divergence, rotation and harmonic parts using the Helmholtz-Hodge decomposition theorem [69, 70, 93]:

Definition 2.4.3 *For a vector field $\boldsymbol{v}$, there exists a unique decomposition into two potentials $d, \boldsymbol{r}$ and a vector field $\boldsymbol{h}$ with*

$$\boldsymbol{v} = \nabla d + \nabla \times \boldsymbol{r} + \boldsymbol{h},$$

∇d being normal to the boundary, and $\nabla \times \boldsymbol{r}$ being tangential to the boundary. This decomposition is called **Helmholtz-Hodge decomposition**.

Note that $\mathbf{r}$ is a scalar field in 2D, and a vector-valued field in 3D.

Lemma 2.4.4 *∇d and $\nabla \times \boldsymbol{r}$ are vector fields with the following properties:*

$$curl\ \nabla d = curl\ grad\ d = 0$$

$$div\ \nabla \times \boldsymbol{r} = div\ curl\ \boldsymbol{r} = 0$$

Thus, ∇d is a **curl-free** *and $\nabla \times \boldsymbol{r}$ a* **divergence-free** *vector field. $\boldsymbol{h}$ is a* **harmonic** *vector field as both curl and divergence vanish. Thus*

$$div\ \boldsymbol{v} = div\ \nabla d + div\ \nabla \times \boldsymbol{r} + div\ \boldsymbol{h} = div\ \nabla d$$

$$curl\ \boldsymbol{v} = curl\ \nabla d + curl\ \nabla \times \boldsymbol{r} + curl\ \boldsymbol{h} = curl\ \nabla \times d$$

Features can then be detected as extremal points of the divergence and rotational field, or using feature detection methods in the resulting vector fields. However, it is not quite clear how analysis and visualization methods are effected by this decomposition (Figure 2.4). Furthermore, the choice of the boundary can greatly influence the results.

A better choice of a decomposition to aid in the detection of Galilean invariant features is the so called localized flow [109]. There, the boundary induced flow is removed to get a region-specific or localized flow which is Galilean invariant itself:

Definition 2.4.5 *Let* $\boldsymbol{v} : \Omega \subset \mathbb{R}^n \rightarrow \mathbb{R}^n$ *be a vector field. Then the* **region-specific flow** $\boldsymbol{v}_R$ *is defined by the following properties:*

$$\boldsymbol{v}_R \cdot \boldsymbol{n} = 0 \text{ on } \partial\Omega$$

$$div\, \boldsymbol{v} = div\, \boldsymbol{v}_R$$

$$curl\, \boldsymbol{v} = curl\, \boldsymbol{v}_R$$

This means that the flow through the boundary of the region-specific field is zero. Vorticity and divergence, and thus the local features of the original flow, are preserved in the region-specific flow and can be visualized accordingly (Figure 2.4). The region-specific flow thus is independent of superposed constant and homogenous flows and represents a basis with non-changing topology for fields with different superposed constant or homogenous flows. Computing the boundary-induced flow is a Neumann-Laplace problem and can thus be solved [109]. The region specific flow is then obtained by removing the boundary induced flow from the original vector field.

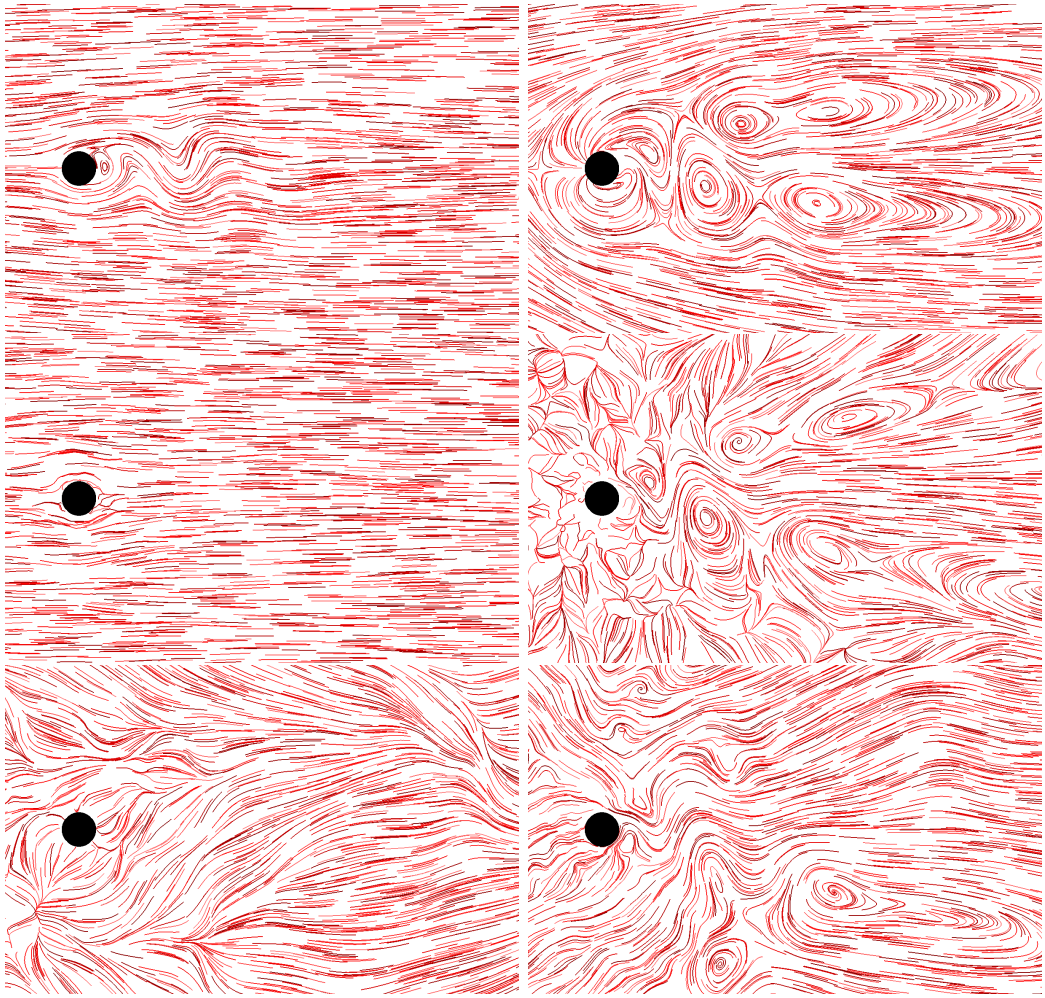

Figure 2.4: Comparison of different fields obtained from a cylinder data set with a Kármán vortex street. **Top left**: Streamlines in the original flow. Only sinuous structures of the lines give hints on the vortices. **Top right**: Four vortices revealed by removing the average flow. **Middle left**: Potential flow induced by the boundary. **Middle right**: Subtracting the potential flow reveals all six vortices. **Bottom left**: Rotational free component as computed by the Helmholtz-Hodge decomposition. **Bottom right**: The divergence free component as computed by the Helmholtz-Hodge decomposition shows two vortices, and one at a rather surprising position. The results of the Helmholtz-Hodge decomposition are difficult to interpret.

Chapter 3

Signal and Image Processing

A signal can be defined as transmitted energy that is used to convey information — it is a message. Signal processing therefore is the art of displaying, transforming and manipulating signals and thus the information they carry. It began with analog circuits and the study of continuous signals. Discrete signals became interesting with the invention of digital circuits. The first digital circuits were quite slow, but they were also quite versatile. Thus they were often used for approximating analog circuits. The development of the fast Fourier transform (Section 3.4) was a milestone for the development of digital circuits as it provided an enormous acceleration of the computation as well as an understanding of the importance of time-discrete signals. A lot of new filters which could not be implemented analogous were now available.

Image processing started as a consequent transfer of signal processing, which had mostly been one-dimensional, to two dimensions and thus to images. The signals, however, remained real and complex. In this chapter, signal and image processing are introduced as far as they are later transfered to vector-valued signals — vector fields. Most of the information on signal and image processing presented here can be found in textbooks by e.g. Oppenheim et al [66], Jain [52], and Jähne [51].

3.1 Signals

In the following, continuous signals or functions will be studied as well as discrete sequences. To distinguish these two types of signals, continuous signals are always indicated by parenthesis, that is $f(t) : \mathbb{R} \to \mathbb{C}$ is a continuous signal. Discrete sequences, on the other hand, are notated using brackets, and thus $f[t] : [0,..,N-1] \subset \mathbb{N} \to \mathbb{C}$ denotes a discrete signal. If no constraint is set on the function, the continuous form is chosen. A complex signal can be split up into its

real part and its imaginary part:

Definition 3.1.1 *Let $f(t) : \mathbb{R} \to \mathbb{C}$ be a signal. Let $f_r(t)$ be the* **real part** *and $f_i(t)$ be the* **imaginary part** *of the signal $f(t) = f_r(t) + i f_i(t)$. Let $f^*(t)$ denote the* **complex conjugate** *signal, that is $f^*(t) = f_r(t) - i f_i(t)$.*

Lemma 3.1.2 *Let $f(t) : \mathbb{R} \to \mathbb{C}$ be a signal. Then*

$$\begin{aligned} f_r(t) &= \tfrac{1}{2}(f(t) + f^*(t)) \\ \textit{and} \quad f_i(t) &= \tfrac{1}{2}(f(t) - f^*(t)). \end{aligned}$$

Another split-up of a signal can now be done according to its symmetry properties:

Definition 3.1.3 *Let $f(t) : \mathbb{R} \to \mathbb{C}$ be a signal. If $f(t) = f^*(-t)$, the signal is* **conjugate symmetric** *and denoted with $f_e(t)$. If $f(t) = -f^*(-t)$, the signal is* **conjugate antisymmetric** *and denoted with $f_o(t)$. If $f(t)$ is real and conjugate symmetric, it is called* **even**, *and if $f(t)$ is real and conjugate antisymmetric, is is called* **odd**.

Lemma 3.1.4 *Let $f(t) : \mathbb{R} \to \mathbb{C}$ be a signal. Then $f(t) = f_e(t) + f_o(t)$ with*

$$\begin{aligned} f_e(t) &= \tfrac{1}{2}(f(t) + f^*(-t)) \\ \textit{and} \quad f_o(t) &= \tfrac{1}{2}(f(t) - f^*(-t)). \end{aligned}$$

3.2 LSI Filter

A system is a function that transforms one signal into another. An important class of systems, or filters, are linear, shift invariant filters as a lot of systems can be described by them, and the rest can be approximated quite well. The linearity and shift invariance of these systems lead to many other properties which are advantageous for signal processing.

Definition 3.2.1 *Let $f_1, f_2 : \mathbb{R} \to \mathbb{C}$ be two complex signals, let $\alpha_1, \alpha_2 \in \mathbb{C}$ and let T be a system or filter. T is called* **linear** *if*

$$T\{\alpha_1 f_1 + \alpha_2 f_2\} = \alpha_1 T\{f_1\} + \alpha_2 T\{f_2\}$$

The linearity of a system is also known as the superposition principle as the results of applying the system to two superposed signals equals the superposed results of the single signals.

Definition 3.2.2 *Let $f_1 : \mathbb{R} \to \mathbb{C}$ be a signal and let T be a system. Let $f_2 : \mathbb{R} \to \mathbb{C}$ be a second signal with $f_2(t) = f_1(t - t_0)\ \forall t, t_0 \in \mathbb{R}$. T is called* **time-invariant** *or* **shift-invariant** *if*

$$T\{f_2\}(t) = T\{f_1\}(t - t_0)$$

Definition 3.2.3 *A system which is linear and shift invariant is called a* **linear, shift invariant (LSI) system** *or filter.*

A signal with a special role in signal processing is the impulse δ, which is an impulse with width zero and amplitude ∞. This signal is often used as basis for the space of signals f:

Definition 3.2.4 $\delta : \mathbb{R} \to \mathbb{C}$ *is the signal defined by*

$$\delta(t) = \begin{cases} \infty & t = 0 \\ 0 & else \end{cases}$$

and for any signal $f : \mathbb{R} \to \mathbb{C}$, $t \in \mathbb{R}$

$$f(t) = \int_{\mathbb{R}} f(k)\delta(t - k)dk$$

Definition 3.2.5 *Let T be a LSI filter and let $t \in \mathbb{R}$. Then $h : \mathbb{R} \to \mathbb{C}$ with $h(t) = T\{\delta(t)\}$ is called* **pulse response** *of T.*

Convolution and correlation are two basic and often used operations:

Definition 3.2.6 *Let $f_1, f_2 : \mathbb{R} \to C$ be continuous signals and let $t, k \in \mathbb{R}$.* **Convolution** *of f_1 and f_2 is defined as*

$$(f_1 * f_2)(t) = \int_{\mathbb{R}} f_1(k)\, f_2(t - k)dk.$$

Spatial **correlation** *of f_1 and f_2 is defined as*

$$(f_1 \star f_2)(t) = \int_{\mathbb{R}} f_1(k)\, f_2(t + k)dk.$$

Thus, a correlation is a convolution with a signal that has been reflected at its center:

Theorem 3.2.7 *Let $f_1, f_2 : \mathbb{R} \to \mathbb{C}$ be two signals. Let $\dot{f}_1 : \mathbb{R} \to \mathbb{C}$ be defined by $\dot{f}_1(t) = f_1(-t)$. Then*

$$f_1 \star f_2 = \dot{f}_1 * f_2.$$

Proof*:*

$$\begin{aligned}(f_1 \star f_2)(t) &= \int_{\mathbb{R}} f_1(k) f_2(t+k) dk \\ &= \int_{\mathbb{R}} \dot{f}_1(-k) f_2(t+k) dk \\ &= \int_{\mathbb{R}} \dot{f}_1(k') f_2(t-k') dk' \\ &= (\dot{f}_1 * f_2)(t)\end{aligned}$$

Theorem 3.2.8 *Convolution is a commutative operation:*

$$(f_1 * f_2)(t) = (f_2 * f_1)(t)$$

Proof*:*

$$\begin{aligned}(f_1 * f_2)(t) &= \int_{\mathbb{R}} f_1(k) f_2(t-k) dk. \\ &= \int_{\mathbb{R}} f_1(t-l) f_2(l) dl. \\ &= \int_{\mathbb{R}} f_2(l) f_1(t-l) dl. \\ &= (f_2 * f_1)(t)\end{aligned}$$

One reason for the usefulness of LSI filters is the fact that they can be described by a convolution with their pulse response:

Theorem 3.2.9 *Let $f : \mathbb{R} \to \mathbb{C}$ be a signal and let T be a LSI filter with pulse response h. Then*

$$T\{f\}(t) = \int_{\mathbb{R}} f(k) h(t-k) dk$$

Proof*:*

$$\begin{aligned}T\{f\}(t) &= T\{\int_{\mathbb{R}} f(k)\delta(t-k) dk\} \\ &= \int_{\mathbb{R}} f(k) T\{\delta(t-k)\} dk \\ &= \int_{\mathbb{R}} f(k) h(t-k) dk\end{aligned}$$

Transferring these definitions and theorems to spacial 2D and 3D signals or images instead of time-dependent signals is straightforward. Concerning a transfer to discrete data, the basic definitions and most important differences can be found in Section 3.4.

3.3 Fourier Transform

The description of a signal as a weighted sum of shifted basis signals plays an important role in signal processing. In the last section, impulses were used as basis functions. The domain thus described is also called time domain. In this section, the description as a sum of complex exponentials and sinusoids is introduced. These basis functions are special as the complex exponentials are eigenfunctions of LSI systems and thus the system answer to a sinusoid is a sinusoid with same frequency. The domain defined by the complex exponentials is called frequency or Fourier domain as the Fourier transform is the basis transform from time to frequency domain and back again.

Definition 3.3.1 *For a continuous signal $f : \mathbb{R} \to C$, the* **Fourier transform** *of f is defined as*

$$\mathscr{F}\{f\}(u) = \int_{\mathbb{R}} f(t)e^{(-2\pi itu)}dt$$

provided the integral exists. The **inverse Fourier transform** *is defined as*

$$\mathscr{F}^{-1}\{f\}(u) = \int_{\mathbb{R}} f(t)e^{(2\pi itu)}dt\ .$$

Note that i with $i^2 = -1$ is the imaginary unit of the complex numbers. The Fourier transform is sometimes defined using the kernel $e^{(-itu)}$. Note that this is mainly due to the difference of defining frequency in radians per second (rad/s), or in Herz (Hz) as $1Hz = 2\pi rad/s$. However, the symmetry of the transform is thus destroyed. A discussion of the existence of $\mathscr{F}$ can be found in Bracewell [7]. In general $\mathscr{F}$ exists if

$$\int_{\mathbb{R}} |f(t)|e^{(-2\pi itu)}dt \ < \infty\ .$$

This is not always true even though the integral of $\mathscr{F}$ may exists nonetheless. Note that the complex exponential of the Fourier kernel can also be written as the sum of sine and cosine according to Euler's formula:

Theorem 3.3.2 (Euler's Formula)

$$e^{ix} = cos(x) + isin(x)\ \forall x \in \mathbb{R}$$

Now the property of complex exponentials as eigenfunctions of LSI systems is shown:

Theorem 3.3.3 *Let T be a LSI system with pulse response $h(t)$. Let $f(t) = e^{(2\pi itu)}$ be a signal. Then*

$$T\{f\}(t) = \mathscr{F}\{h\}(u)e^{(2\pi itu)}$$

and $e^{(2\pi itu)}$ is an eigenfunction of T with eigenvalue $\mathscr{F}\{h\}(u)$.
Proof*: Using the commutativity of the convolution yields*

$$\begin{aligned} T\{f\}(t) &= \int_{\mathbb{R}} e^{(2\pi iku)} h(t-k)dk \\ &= \int_{\mathbb{R}} e^{(2\pi i(t-k)u)} h(k)dk \\ &= \int_{\mathbb{R}} e^{(2\pi itu)} e^{(-2\pi iku)} h(k)dk \\ &= e^{(2\pi itu)} \int_{\mathbb{R}} h(k) e^{(-2\pi iku)} dk \\ &= e^{(2\pi itu)} \mathscr{F}\{h\}(u) \end{aligned}$$

In the following, central theorems of the Fourier transform are introduced. First of all, the Fourier transform is linear:

Theorem 3.3.4 (Linearity) *Let $f_1, f_2 : \mathbb{R} \to \mathbb{C}$ be two signals, and let $\alpha_1, \alpha_2 \in \mathbb{R}$. The Fourier transform is linear:*

$$\mathscr{F}\{\alpha_1 f_1 + \alpha_2 f_2\} = \alpha_1 \mathscr{F}\{f_1\} + \alpha_2 \mathscr{F}\{f_2\}$$

Proof*:*

$$\begin{aligned} \mathscr{F}\{\alpha_1 f_1 + \alpha_2 f_2\} &= \int_{\mathbb{R}} (\alpha_1 f_1(t) + \alpha_2 f_2) e^{(-2\pi itu)} dt \\ &= \alpha_1 \int_{\mathbb{R}} f_1(t) e^{(-2\pi itu)} dt + \alpha_2 \int_{\mathbb{R}} f_2(t) e^{(-2\pi itu)} dt \\ &= \alpha_1 \mathscr{F}\{f_1\} + \alpha_2 \mathscr{F}\{f_2\} \end{aligned}$$

Time and frequency shifts of a signal are modeled by a multiplication with a complex exponential:

Theorem 3.3.5 (Time and frequency shifts) *Let $f : \mathbb{R} \to \mathbb{C}$ be a signal and $k \in \mathbb{R}$. Then the Fourier transform of a time-shift is*

$$\mathscr{F}\{f(t-k)\} = e^{(-2\pi ik)} \mathscr{F}\{f(t)\}$$

an the frequency-shift in Fourier domain is given by

$$\mathscr{F}\{e^{(-2\pi ik)} f\}(u) = \mathscr{F}\{f\}(u-k)$$

Proof*:*

$$\begin{aligned}\mathscr{F}\{f(t-k)\}(u) &= \int_{\mathbb{R}} f(t-k)e^{(-2\pi itu)}dt \\ &= \int_{\mathbb{R}} f(l)e^{(-2\pi i(l+k)u)}dl \\ &= e^{(-2\pi ik)}\int_{\mathbb{R}} f(l)e^{(-2\pi ilu)}dl \\ &= e^{(-2\pi ik)}\mathscr{F}\{f(l)\}(u)\end{aligned}$$

The proof for the frequency shift is analogous.

Theorem 3.3.6 (Plancherel's theorem) *Let* $f_1, f_2 : \mathbb{R} \to \mathbb{C}$ *be two signals, and for* $c = a + ib \in \mathbb{C}$ *let* $c^+ = a - ib$ *denote the complex conjugate of c. Then*

$$\int_{\mathbb{R}} f_1(t)\, f_2^+(t)dt = \int_{\mathbb{R}} \mathscr{F}\{f_1\}(u)\mathscr{F}^+\{f_2\}(u)du$$

Parseval's theorem, which follows directly out of Plancherel's theorem, states that the energy of a signal is equal in both time and frequency domain:

Theorem 3.3.7 (Parseval's theorem)

$$E\{f\} = \int_{\mathbb{R}} \|f(t)\|^2 dt = \int_{\mathbb{R}} \|\mathscr{F}\{f\}(u)\|^2 du = E\{\mathscr{F}\{f\}\}$$

The convolution theorem is one of the most important theorems in signal processing. It states that a convolution of two signals in time domain is equal to a multiplication of the signals in frequency domain:

Theorem 3.3.8 (Convolution theorem) *Let* $f_1, f_2 : \mathbb{R} \to \mathbb{C}$ *be two signals. Then*

$$\mathscr{F}\{f_1 * f_2\} = \mathscr{F}\{f_1\}\mathscr{F}\{f_2\}$$

Proof*:*

$$\begin{aligned}\mathscr{F}\{f_1 * f_2\}(u) &= \int_{\mathbb{R}} (f_1 * f_2)(t)e^{(-2\pi itu)}dt \\ &= \int_{\mathbb{R}} (\int_{\mathbb{R}} f_1(k)\, f_2(t-k)dk)e^{(-2\pi itu)}dt \\ &= \int_{\mathbb{R}}\int_{\mathbb{R}} f_1(k)\, f_2(t)e^{(-2\pi iku)}dke^{(-2\pi itu)}dt \\ &= \int_{\mathbb{R}} f_1(k)e^{(-2\pi iku)}dk \int_{\mathbb{R}} f_2(t)e^{(-2\pi itu)}dt \\ &= \mathscr{F}\{f_1\}(u)\mathscr{F}\{f_2\}(u)\end{aligned}$$

Thus, the Fourier transform of the pulse response of a system is often contemplated.

Definition 3.3.9 *Let $h : \mathbb{R} \to \mathbb{C}$ be the pulse response of a system T. The Fourier transform $\mathscr{F}\{h\}$ of h is called* **frequency response**.

The following derivation theorem is only valid for signals f which are bounded with $lim_{t\to\pm\infty} f(t) = 0$. This is the case for all physically significant signals.

Theorem 3.3.10 (Derivative theorem) *Let $f : \mathbb{R} \to \mathbb{C}$ be a signal with $lim_{t\to\pm\infty} f(t) = 0$. Then*

$$\begin{aligned} \mathscr{F}\{f'\}(u) &= 2\pi iu\mathscr{F}\{f\}(u) \\ \text{and} \quad \mathscr{F}\{f^{(n)}\}(u) &= (2\pi iu)^n\mathscr{F}\{f\}(u). \end{aligned}$$

Proof*: Using integration by parts yields*

$$\begin{aligned} \mathscr{F}\{f'\}(u) &= \int_{\mathbb{R}} f'(t)e^{(-2\pi itu)}dt \\ &= [f(t)e^{(-2\pi itu)}]_{-\infty}^{\infty} - 2\pi iu\int_{\mathbb{R}} f(t)e^{(-2\pi itu)}dt \\ &= -2\pi iu\int_{\mathbb{R}} f(t)e^{(-2\pi itu)}dt \\ &= -2\pi iu\mathscr{F}\{f\}(u) \end{aligned}$$

Iteration yields the formula for the n-th derivative.

The Fourier transform also has several symmetry properties:

Theorem 3.3.11 *Let $f(t) : \mathbb{R} \to \mathbb{C}$ be a signal. Then*

$$\begin{aligned} \mathscr{F}\{f^*\}(u) &= \mathscr{F}^*\{f\}(-u) \\ \mathscr{F}\{f_r\}(u) &= \mathscr{F}_e\{f\}(u) \\ \mathscr{F}\{if_i\}(u) &= \mathscr{F}_o\{f\}(u) \\ \mathscr{F}\{f_e\}(u) &= \mathscr{F}_r\{f\}(u) \\ \mathscr{F}\{f_o\}(u) &= j\mathscr{F}_o\{f\}(u) \end{aligned}$$

Let $f(t) : \mathbb{R} \to \mathbb{R}$ be a real signal. Then

$$\begin{aligned} \mathscr{F}\{f\}(u) &= \mathscr{F}^*\{f\}(-u) \\ \mathscr{F}_r\{f\}(u) &= \mathscr{F}_r\{f\}(-u) \\ \mathscr{F}_i\{f\}(u) &= -\mathscr{F}_i\{f\}(-u) \end{aligned}$$

Proof*:*

$$\begin{aligned}\mathscr{F}\{f^*\}(u) &= \int_{\mathbb{R}} f^*(t)e^{(-2\pi itu)}dt \\ &= \int_{\mathbb{R}} (f(t)e^{(2\pi itu)})^* dt \\ &= \mathscr{F}^*\{f\}(-u)\end{aligned}$$

The rest follows directly from this property, the definitions of real, imaginary, conjugate symmetric, conjugate antisymmetric, and the linearity of the Fourier transform.

3.4 Discrete Signals and the FFT

In the former sections, only continuous signals have been regarded. As digital signals and systems play an important role nowadays, they are addressed in this section.

Definition 3.4.1 *A sequence of numbers $f[t]$, $t \in \mathbb{Z}$, $-\infty < t < \infty$, $f[t] \in \mathbb{C}$ is called a* **time-discrete signal***. A time-discrete signal $f[t]$ with $f[t] \in Q \subset \mathbb{C}$ where Q is a finite set of numbers, is called a* **discrete** *signal.*

Let $f(t) : \mathbb{R} \to \mathbb{C}$ be a continuous signal. A time-discrete signal $f[t]$ is usually gained by periodic sampling of $f(t)$: $f[t] = f(tT)$, $T \in \mathbb{R}$. A discrete signal is then gained by quantizing the values of $f[t]$. Both operations together describe an analog-to-digital converter, or digitizer.

Definition 3.4.2 *$\delta : \mathbb{Z} \to \mathbb{C}$ is the signal defined by*

$$\delta[t] = \begin{cases} 1 & t = 0 \\ 0 & else \end{cases}$$

Thus for or any time-discrete signal $f : \mathbb{Z} \to \mathbb{C}$

$$f[t] = \sum_{k=-\infty}^{\infty} f[k]\delta[t-k]$$

The results from Section 3.2 can be transfered directly to time-discrete and discrete signals. The convolution of discrete signals, for example, is defined as follows:

Definition 3.4.3 *Let* $f_1, f_2 : [0,..,N-1] \to C$ *be discrete signals and let* $t,k \in \mathbb{R}$. **Convolution** *of* f_1 *and* f_2 *is defined as*

$$(f_1 * f_2)[t] = \sum_{k=0}^{N-1} f_1[k]\, f_2[t-k].$$

However, the Fourier transform of discrete signals has some special peculiarities.

Definition 3.4.4 *For a time-discrete signal* $f : \mathbb{Z} \to C$*, the* **discrete time Fourier transform (DTFT)** *of* f *is defined as*

$$\mathscr{F}\{f\}(u) = \sum_{t=-\infty}^{\infty} f[t] e^{(-2\pi i t u)}$$

provided the limit of the sequence exists. The **inverse DTFT** *is defined as*

$$\mathscr{F}^{-1}\{f\}[u] = \int_{t=0}^{1} f(t) e^{(2\pi i t u)} dt\ .$$

For a finite time-discrete signal $f : [0,N-1] \subset \mathbb{N} \to \mathbb{C}$*, the* **discrete Fourier transform (DFT)** *is defined as*

$$\mathscr{F}\{f\}[u] = \sum_{t=0}^{N-1} f[t] e^{\left(\frac{-2\pi i t u}{N}\right)}$$

and the inverse transform as

$$\mathscr{F}\{f\}[u] = \frac{1}{N} \sum_{t=0}^{N-1} f[t] e^{\left(\frac{2\pi i t u}{N}\right)}$$

It is important to note that the discrete Fourier transform of a time-discrete signal is always a periodic continuous signal with period 1. Therefore the inverse transform only needs to be evaluated in $[0,1]$.

Theorem 3.4.5 *The DTFT spectrum is periodic with period* 1.
Proof*: Let* $t \in [0,m] \subset \mathbb{N}$.

$$\begin{aligned} e^{(-2\pi i t(u+1))} &= e^{(-2\pi i t u)} e^{(-2\pi i t)} \\ &= e^{(-2\pi i t u)} e^{(-2\pi i)^t} \\ &= e^{(-2\pi i t u)} (1)^t \\ &= e^{(-2\pi i t u)} \end{aligned}$$

When a signal is sampled and quantized, information loss usually occurs. For band-limited signals, a threshold for the sampling frequency can be determined. Above this threshold, a perfect reconstruction of the signal is possible provided the sampling frequency is still known.

Theorem 3.4.6 (Nyquist-Shannon sampling theorem) *Let $f(t) : \mathbb{R} \to \mathbb{C}$ be a band-limited signal with maximal frequency f_{max}. When a time-discrete signal $f[t] : \mathbb{Z} \to \mathbb{C}$ is computed by sampling $f(t)$ with period $T = \frac{1}{f_s}$, and the sampling frequency f_s fulfills*

$$f_s \geq 2 f_{max},$$

then perfect reconstruction of $f(t)$ out of the signal $f[t]$ and the known sampling frequency f_s is possible.

Proof*: The Fourier transform of a band-limited signal with maximal frequency f_{max} has a finite width of $2 f_{max}$. When the sampling frequency is larger than $2 f_{max}$, the periodic repetitions of the Fourier transform do not overlap and the original signal can be reconstructed using an ideal low-pass filter which only lets frequencies in $[-f_{max}, f_{max}]$ pass. Further details can be found in the literature [66].*

A DFT of length $N = 2^m$ can be computed as the sum of two DFTs of length $\frac{N}{2}$:

Theorem 3.4.7 *Let $f : [0,..,N-1] \to \mathbb{C}$ be a discrete signal and let $N = 2^m$. Let $f^{even}, f^{odd} : [0, N/2-1] \to \mathbb{C}$ be the two signals with $f^{even}[t] = f[2t]$ and $f^{odd}[t] = f[2t+1]$. Then*

$$\mathscr{F}\{f\}[u] = \mathscr{F}\{f^{even}\}[u] + e^{\left(\frac{2\pi i u}{N}\right)} \mathscr{F}\{f^{odd}\}[u]$$

and

$$\mathscr{F}^{-1}\{f\}[t] = \mathscr{F}^{-1}\{f^{even}\}[t] + e^{\left(\frac{2\pi i t}{N}\right)} \mathscr{F}^{-1}\{f^{odd}\}[t]$$

Proof*:*

$$\begin{aligned}
\mathscr{F}\{f\}[u] &= \sum_{t=0}^{N-1} f[t] e^{\left(\frac{-2\pi i t u}{N}\right)} \\
&= \sum_{t=0}^{N/2-1} f[2t] e^{\left(\frac{-2\pi i (2t) u}{N}\right)} + \sum_{t=0}^{N/2-1} f[2t+1] e^{\left(\frac{-2\pi i (2t+1) u}{N}\right)} \\
&= \sum_{t=0}^{N/2-1} f^{even}[t] e^{\left(\frac{-2\pi i (2t) u}{N}\right)} + e^{\left(\frac{-2\pi i u}{N}\right)} \sum_{t=0}^{N/2-1} f^{odd}[t] e^{\left(\frac{-2\pi i (2t) u}{N}\right)} \\
&= \sum_{t=0}^{N/2-1} f^{even}[t] e^{\left(\frac{-2\pi i t u}{N/2}\right)} + e^{\left(\frac{-2\pi i u}{N}\right)} \sum_{t=0}^{N/2-1} f^{odd}[t] e^{\left(\frac{-2\pi i t u}{N/2}\right)} \\
&= \mathscr{F}\{f^{even}\}[u] + e^{\left(\frac{2\pi i u}{N}\right)} \mathscr{F}\{f^{odd}\}[u]
\end{aligned}$$

The proof for the inverse Fourier transform is analog.

Definition 3.4.8 *The algorithm for computing the DFT recursively based on Theorem 3.4.7 is called* **fast Fourier transform (FFT)** *[16].*

Note that DTFT and DFT are only defined for (time-)discrete signals with equidistant spacing of the supporting points of the signals here. The FFT is further limited as it needs equidistant spacing of $N = 2^m$ supporting points here. Fast DFT algorithms for other decompositions of N exist, too. Discretizations of the Fourier transform for signals with irregular spacings also exist, and even fast algorithms for their computation. Further information can be found in [74, 88].

Let $f[t] : [0,..,N-1] \to \mathbb{C}$ and $h[t] : [0,..,M-1] \to \mathbb{C}$ be two discrete signals with $N > M$. The FFT reduces the complexity of the computation of the DFT of $f[t]$ from $O(N^2)$ to $O(Nlog(N))$. The computation of a convolution of $f[t]$ and $h[t]$ has complexity $O(NM)$. Computing the convolution in frequency domain via convolution theorem and FFT's reduces the cost to $O(Nlog(N))$. Thus, the computation of the convolution via frequency domain is faster for $M > log(N)$.

3.5 Gabor Filter

Due to the uncertainty principle, a signal can either be optimally localized in time domain or in frequency domain. Examples are the impulses δ which are optimally localized in time domain but spread out infinitely in frequency domain, and the sinusoids which are optimally localized in frequency domain and spread out infinitely in time domain.

Theorem 3.5.1 *Let $f : \mathbb{R} \to \mathbb{C}$ be a signal and $E(f) = \int_{\mathbb{R}} f(t)dt$ its mean value. Let $\Delta t = \sqrt{2\pi E((f - E(f))^2)}$ and $\Delta u = \sqrt{2\pi E((\mathscr{F}\{f\} - E(\mathscr{F}\{f\}))^2)}$. Δt and Δu measure the variances, normalized with $\sqrt{2\pi}$, of the signal f and its Fourier transform $\mathscr{F}\{f\}$. The* **uncertainty principle** *is given by the relation*

$$\Delta t \Delta u \geq \frac{1}{2}.$$

Thus, resolution in frequency domain has to be halved to get double resolution in time domain and vice versa. The Gabor filter is the only function which achieves the equality $\Delta t \Delta u = 1/2$ as it is the modulation product of a complex sinusoid with an impulse of the form of a probability or Gaussian function [35].

Definition 3.5.2 *The* **Gaussian function** *is defined as $g(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{\left(\frac{-t^2}{2\sigma^2}\right)}$*

Definition 3.5.3 *Let $g : \mathbb{R} \to \mathbb{R}$ be a Gaussian filter with variance σ. The pulse response $h : \mathbb{R} \to \mathbb{C}$ of the* **Gabor filter** *is defined as*

$$h(t) = g(t) * e^{(2\pi i t u)}.$$

The frequency response is

$$\mathscr{F}\{h\}(u) = e^{-2\pi\sigma u}.$$

The Gabor transformation thus defined can also be understood as a Fourier transform localized by the Gaussian window function, as special type of a short-time Fourier transform. Note that, in 2D and 3D, sometimes Gaussians with elliptical instead of circular shape are used.

The Gabor filters form a complete non-orthogonal basis. If $f : \mathbb{R} \to \mathbb{C}$ is a finite-dimensional function, it can be expressed by a weighted sum of appropriately shifted Gabor functions:

Definition 3.5.4 *The* **Gabor expansion** *of a function $f : \mathbb{R} \to \mathbb{C}$ is defined as*

$$f(t) = \sum_r \sum_u \beta_{r,u} g(t - t_r) e^{(2\pi i (t - t_r) u)}$$

The sequences of shifts x_r, and modulation frequencies u, have constant spacings X and U satisfying $XU = 1$. Together, they form the **Gabor lattice**.

An example of a Gabor lattice in 2D can be found in Figure 3.1. When the density of positions t_r and frequencies u is equal to or higher than that given by the sampling theorem, the Gabor expansion scheme is exact. Then, $f(t)$ can be exactly reconstructed from the expansion coefficients $\beta_{r,u}$ though it is a complex process as the basis is non-orthogonal. A good and often used approximation [6, 65] is given by:

$$\beta_{r,u} \approx \int f(t) g(t - t_r) e^{(-2\pi i (t - t_r) u} dt.$$

When $f(x)$ is approximately band limited, e.g. $f(x)$ has been sampled, a finite number of expansion coefficients can adequately represent the important frequencies of the function. More important is the localization property, that is a large coefficient $\beta_{r,u}$ corresponds to the dominant frequencies in the spatial vicinity of t_r. The Gabor expansion can be generalized using other windowing functions than the Gaussian, but then the optimal localization in both frequency and spatial domain is lost.

Gabor filters are often used for wavelet transformations. The difference between a Gabor expansion and a Gabor wavelet transform is that the coefficients of the Gabor expansion are computed using an equally sized Gaussian and for the

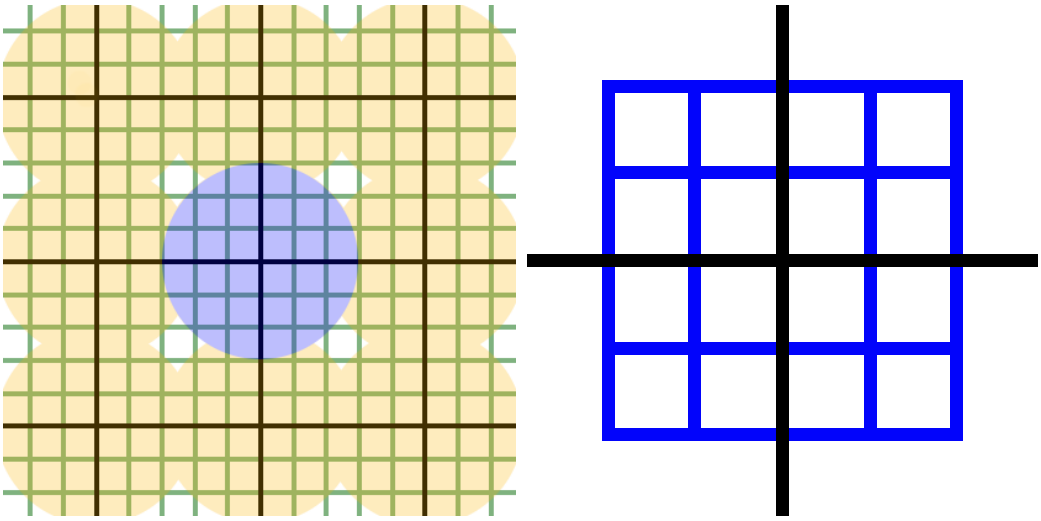

Figure 3.1: The Gabor lattice. **Left**: Original grid (green), coarser spatial resolution of the Gabor lattice (black) and the Gaussians adumbrated by the orange and blue circles. **Right**: At every grid node of the coarser spatial grid of the Gabor lattice (black), a short time FT is computed. The underlying grid of the Gabor coefficients in frequency domain is drawn in blue. The coarse spatial grid and the local grids in frequency domain together form the Gabor lattice.

wavelet transform, the size of the Gaussian is directly proportional to the wavelength of the frequency to be computed. The advantage of the wavelet approach is that the Gaussian is optimally scaled in proportion to the frequency to be described.

An interesting observation is the fact that simple cells in the human visual cortex, that is in the first steps of human visual information processing, can be modeled as Gabor filters [65]. Thus, it is only consequential to use Gabor filters in image processing. Applications of Gabor filters and wavelets so far include frequency-space analysis for resampling [106], segmentation of texture [6, 9, 32], image registration and motion estimation [64] and object recognition in general, e.g. the recognition of faces [39, 60, 64].

3.6 Segmentation of Images

The step from 1D signals with time as variable to 2D signals in spacial domain is straightforward, the signals are evaluated for each dimension.

Definition 3.6.1 *Let* $X_j = [0,..,m_{x_j}] \subset \mathbb{N}$, $G = [0,..,m_g] \subset \mathbb{N}$, $m_{x_j}, m_g \in \mathbb{N}$, $j =$

$1,..,k$. *An element* $\boldsymbol{x} = (x_1,..,x_n) \in X_1 \times .. \times X_k$ *is called a* **pixel** *(or* **voxel** *for* $k = 3$*). A discrete scalar field* $p[\boldsymbol{x}] : X_1 \times .. \times X_k \rightarrow G$ *is called a* **grayscale image**. *A discrete scalar field* $\boldsymbol{p}[\boldsymbol{x}] : X_1 \times .. \times X_k \rightarrow G^{k_c}$ *with* $k_c > 1 \in N$ *is called a* **color image**. *The number* k_c *of color channels is usually 3 or 4.*

Note that there are several conventions for choosing the color space. RGB splits the color into red, green and blue parts. HSV, on the other hand, is based on hue, saturation and brightness value. YUV was developed to add color to grayscale television while keeping downward compatibility and has one luminance channel corresponding to the grayscale image, and two chrominance channels. CMYK is based on cyan, magenta, yellow, and black, and is mostly used for printer. Note that a color from one color space may not have a correspondence in another space, though most colors can be converted. Due to the differences in defining the color space, no unifying approach for extending algorithms for grayscale images to color has been developed.

RGB color images can be converted into grayscale images via the magnitude of their color vector. Otherwise, each color channel can be examined separately, and the results be interpreted according to the color space which was used. It is important to note that though the color channels form vectors, they have not much in common with vectors that describe directions. Therefore algorithms for color images are not necessarily transferable to vector fields from flow visualization. In the following, only 2D and 3D grayscale images are discussed.

3.6.1 Feature Detection and Segmentation

A central issue in image processing and machine vision is feature detection. Segmentation itself is the process of partitioning the image into multiple regions according to some criterion. Both have always been central issues in image processing, therefore a huge amount of different approaches is available nowadays. In this chapter, the most basic ideas are introduced and discussed to give a short overview of the approaches and the challenges they met.

Whenever the amplitude, here grayscale values, sufficiently characterizes the features, amplitude thresholding is useful:

Definition 3.6.2 *Let* $p[\boldsymbol{x}]$ *be a discrete grayscale image.* **Amplitude thresholding** *is the process of segmenting an image based on a given threshold* α *by classifying every pixel* $\boldsymbol{x}$ *according to its value* $p[\boldsymbol{x}]$ *as* $p[\boldsymbol{x}] \leq \alpha$ *or* $p[\boldsymbol{x}] > \alpha$.

The choice of the threshold, however, is often a difficult problem as it greatly influences the result of the segmentation (Figure 3.2). Without further information about the information displayed in the images, the threshold can only be guessed. When the edge between feature and background is symmetric, the median of their

grayscale values is the correct threshold. Here, analysis of the histogram of the gray values is helpful in choosing the threshold. Note that when amplitude values of feature and background overlap, only a partial segmentation is possible, or none at all.

This way, the image can be binarized by thresholding, and each pixel is labeled as belonging to the object or not. Thus, pixels or areas separated from the main object can still belong to the same object. This is often not desired, as an object is inherently characterized by connectivity. Further processing can amend this. The binarized image can be used e.g. for component labeling, where the connectivity of pixels with their neighbors is examined in order to assign the pixel to objects.

Another classical approach for segmentation in image processing is based on edge detection. An edge in a discrete image can be defined as follows:

Definition 3.6.3 *Let $p[\boldsymbol{x}]$ be an image. An pixel $\boldsymbol{x}$ belongs to an* **edge** *if the difference of the grayvalues of neigbouring pixels is large in one direction, and small in the orthogonal directions.*

Thus, edges can be described as extrema of the magnitude of the gradient. This approach leads to the approximation of edge-detection operators by discrete differentials. To improve the performance, the Sobel operator, based on the first derivative, smoothes the image along one direction while computing the derivative into the other. Another description of edges is a zero-crossing of the second derivative. However, not every zero-crossing describes an edge, and many zero-crossings are only due to noise. Only those zero-crossings with a significant extrema on either side are due to edges. The operator computing the second derivative is called Laplace operator. Note that both first and second derivative are sensitive to noise (Figure 3.2). Smoothing of the object, e.g. by using Gaussian filters (Definition 3.5.2), can be necessary. Furthermore, thresholding has to be applied after edge detection to filter the results and discard insignificant information. Then the edges of objects have to be combined to form the boundaries of the objects. This can be done e.g. by contour following or edge linking.

Region growing is computed by dividing the image into regions of same gray levels. Adjacent regions which are similar with respect to the gray value are then merged until neighboring regions are sufficiently different. This can also be computed the other way round by splitting algorithm. These region based approaches, also called clustering algorithms, are less sensitive in terms of noise. However, they can be quite complex. Furthermore, the merging or splitting criteria is not easy to choose and it greatly influences the results.

Pattern matching has been used for segmentation of images as well. There, similarity to templates from a given list is computed at all positions in the image via a correlation (Definition 3.2.6). Positions with a significant local maxima of

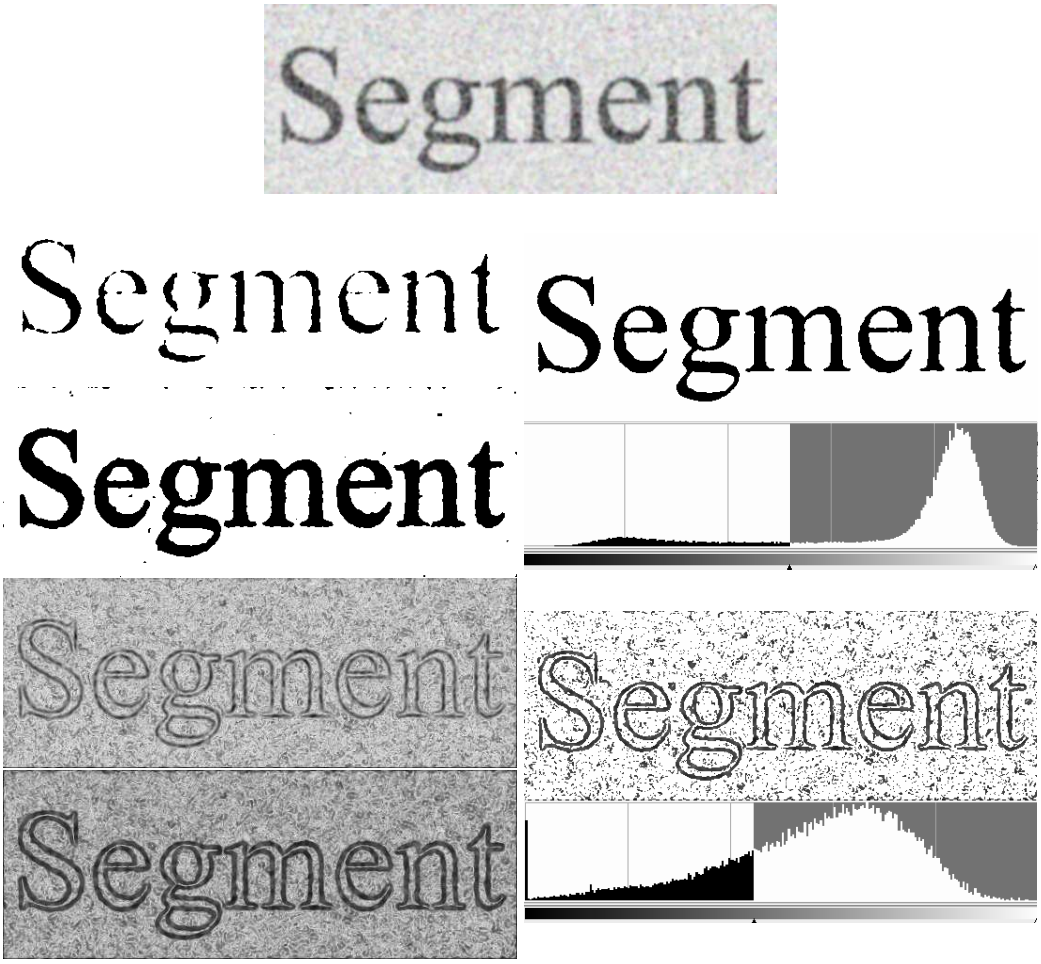

Figure 3.2: Segmentation based on thresholding and edge-detection. The values are distributed between 0 (black) and 255 (white). **Top**: Original image. Binarization based on: **Left row top two images**: Thresholding (value 80 and 180) depends on the chosen threshold. **Right row top**: The median of the two maxima of the color distribution is often chosen as threshold (here value 133). **Left row bottom two images**: Edge detection using the gradient and the Sobel operator. **Right row bottom**: Binarization using thresholding of edge detection results (gradient operator). Edge detection algorithms are sensitive to noise, therefore their results are not convincing for this example.

the similarity values have a high probability of being the center of the feature to be found. The size of the feature is assumed to be the size as coded in the template.

The correlation has to be computed at every point of the grid. But at the border of the image, the correlation would need values outside the image. This problem of boundary values is usually solved using one of the following approaches:

1. Using zero values. Thus, artificial edges at the border are created.
2. Extrapolation. At the simplest case, one can take the values at the boundary. All extrapolations lay too much stress on the border values.
3. Cyclic convolution. The results depend on the chosen display window as most images are not periodic as assumed here.
4. Windowing. The values are gradually reduced to zero near the boundary. Some values at the border are lost but otherwise this is the preferred approach in image processing.

Another problem of template matching using correlation is choosing the right threshold for determining significant similarity. One technique to obtain sharper local maxima of the similarity values are symmetric phase-only matching filters, which are introduced in Section 3.6.2. Other challenges are rotation and scale invariant similarity measures. These can be approximated by using templates with different sizes and directions. However, this is computationally expensive. For scale invariant matching, use of the scale space and multi-grid approaches can be made (Section 3.6.3). For rotation invariant matching, the orientation tensor in Section 3.6.4 can be used to determine the major direction of the structure. Template and structure in the field can thus be aligned for maximal similarity.

3.6.2 Symmetric Phase-Only Matched Filter

A distinct disadvantage of template matching based on the correlation is that the similarity depends more on the energy structure of the image than the spatial structures. Therefore, template matching often provides a poor discrimination between objects of different shapes but similar energy content. Furthermore, the shape of the filter output around the maximum is broad. This can become a problem for the detection of local maxima, especially in the presence of noise. Phase-only matching solves these problems as the phase preserves the object location and is insensitive to the object energy.

Definition 3.6.4 *A filter $h(x) : \mathbb{R}^n \to \mathbb{C}$ is called a* **phase-only matched filter** *if the phase of its pulse response is equal to the spectral phase of the function $f(x) : \mathbb{R}^n \to \mathbb{C}$ to be analyzed.*

A further improvement can be achieved by extracting the phases of both functions, and matching them [14]:

Definition 3.6.5 *Let $f, h : \mathbb{R}^n \to \mathbb{C}$ be two functions. Let $q : \mathbb{R}^n \to \mathbb{C}$ be a filter defined by*

$$q(\boldsymbol{x}) = \mathscr{F}^{-1}\left\{\frac{\mathscr{F}\{f\}}{\|\mathscr{F}\{f\}\|}\frac{\mathscr{F}\{h\}}{\|\mathscr{F}\{h\}\|}\right\}(\boldsymbol{x}).$$

Then $q(\boldsymbol{x})$ is the result of **symmetric phase-only matched filtering** *(SPOMF) of $f(\boldsymbol{x})$ and $h(\boldsymbol{x})$.*

3.6.3 Scale Space

In Section 3.5, the uncertainty principle was introduced which states that a feature can only be localized well in either time (or spatial) domain, or in frequency domain. The idea of Gabor filters was to get both information of the involved frequencies and the spatial resolution. The frequencies contain information about the size of a feature. Thus, a range of frequencies is also called a scale, and dividing the occurring frequencies into different scales is the basis for a scale space, in which the frequencies are subsequently removed according to their scale. This is analogous to a diffusion process.

Definition 3.6.6 *Let $p[\boldsymbol{x}]$ be an image. Let $s, s_1, s_2 \in [0, m_s]$ with $s_1 < s_2$. Then $q[\boldsymbol{x}, s]$ is a* **scale space** *if*

$$\begin{aligned} q[\boldsymbol{x}, 0] &= p[\boldsymbol{x}], \\ \|q[\boldsymbol{x}, s_1\|] &\geq \|q[\boldsymbol{x}, s_2]\|, \end{aligned}$$

and all local extrema in $q[\boldsymbol{x}, s_2]$ are local extrema in $q[\boldsymbol{x}, s_1]$ and of the same type.

Thus, the information in $q[\mathbf{x}, s]$ is reduced with growing s. This is achieved using smoothing filters. As the smoothing process should also be independent of the scale s at which it is started, the Gaussian filter (Definition 3.5.2) is the only LSI filter satisfying both criteria as well as isotropy and homogeneity [63].

A disadvantage of the scale space is the additional scale dimension and thus an extensive increase in computational time and storage space. This leads to multigrid approaches for discrete grids, the best known being the Gaussian pyramid created by subsequently smoothing with the Gaussian filter and reducing the size of the grid as far as possible while still satisfying the Nyquist theorem (Theorem 3.4.6). The Gaussian pyramid can then be used for template matching on each of its grids, and combining the results of different scales results in a scale-invariant similarity measure. Sometimes it it also interesting to investigate the Laplace-Pyramid, which is the opposite of the Gaussian pyramid as the information which is lost between two subsequent grids of the Gaussian pyramid is computed.

3.6.4 Rotation Invariant Matching

Rotation invariant matching is a difficult problem. The only general solution known is to generate many templates with the object of interest rotated into different directions, match with all of them, and use the maximal similarity. This is

obviously computationally expensive. When a feature is rotationally symmetric, it is obvious that the similarity to this feature will be rotation invariant, too. For other features, different approaches have to be found.

Let a feature be describable by a simple function, that is a function that varies only in one direction. Basic examples for such features are edges and lines. If the neighborhood of a point can be modeled by a simple function, as it is the case within such features, local orientation, symmetries and curvature can be determined by a so called orientation tensor [36]. This orientation tensor is based on an outer product of the response of quadrature filters to a function. Quadrature filters are filters with a zero transfer function in one half plane of the frequency domain. The normal of this plane is called the orientation of the quadrature filter. Note that the real part of the quadrature filter corresponds to a line detector and the complex part to an edge detector. Furthermore the magnitude of the filter response is proportional to the square of the angle between the directions of filter and image.

Definition 3.6.7 *Let $p(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ be a scalar field. Let $h_k : \mathbb{R}^n \to \mathbb{R}, k \in \mathbb{N}$ be a quadrature filter with direction $\boldsymbol{n}_k$, and the $\boldsymbol{n}_k$ be evenly distributed over a half-sphere. Let I be the identity tensor, let $\alpha, \beta \in \mathbb{R}$ and $\boldsymbol{x} \in \mathbb{R}^n$. Then the tensor*

$$T_o(\boldsymbol{x}) = \sum_{j=0}^{N-1} \|(p * h_k)(\boldsymbol{x})\|(\alpha \boldsymbol{n}_k \boldsymbol{n}_k^T - \beta I)$$

is called **orientation tensor**. *α, β, N and the template directions $\boldsymbol{n}_k$ depend on the dimension. A possible distribution of the filter directions is [36]:*

2D: $N = 3$, $\alpha = 4/3$, *and* $\beta = 1/3$. *Let* $a = 0.5$ *and* $b = \frac{\sqrt{3}}{2}$. *Then the directions* $\boldsymbol{n}_k$, $k = 1,2,3$, *are* $\boldsymbol{n}_1 = (1,0)^T$, $\boldsymbol{n}_2 = (a,b)^T$ *and* $\boldsymbol{n}_3 = (-a,b)^T$

3D: $N = 6$, $\alpha = 5/4$ *and* $\beta = 1/4$. *Let* $a = 2$, $b = (1+\sqrt{5})$, *and* $c = (10 + 2\sqrt{5})^{-\frac{1}{2}}$. *Then the directions* $\boldsymbol{n}_k$, $k = 1,..,6$, *are*
$\boldsymbol{n}_1 = c(a,0,b)^T$, $\boldsymbol{n}_2 = c(-a,0,b)^T$,
$\boldsymbol{n}_3 = c(b,a,0)^T$, $\boldsymbol{n}_4 = c(b,-a,0)^T$,
$\boldsymbol{n}_5 = c(0,b,a)^T$, $\boldsymbol{n}_6 = c(0,b,-a)^T$

Note that these directions in 2D correspond to the x-axis rotated 0°, 60° and 120°, which are also the axes of a hectahedron. In 3D, the directions correspond to the axes of a icosahedron.

Theorem 3.6.8 *Let $p(\boldsymbol{x}) : \mathbb{R}^3 \to \mathbb{R}$ be a 3D scalar field. Let $h_k : \mathbb{R}^3 \to \mathbb{R}, k \in \mathbb{N}$ be a quadrature filter with direction $\boldsymbol{n}_k$. Let $T_o(\boldsymbol{x}_j)$ be the orientation tensor generated by $p(\boldsymbol{x}_j)$ and h_k at position $\boldsymbol{x}_j$.*

1. *Let the neighborhood of $\boldsymbol{x}_j$ have one unique orientation n_d. Let s be the similarity of the frequencies of field and filter. Then $T_o(\boldsymbol{x}_j)$ has the eigenvalues $\lambda_1 = s^2$, $\lambda_2 = \lambda_3 = 0$ and the eigenvector $e_1 = n$.*

2. *Let the neighborhood of $\boldsymbol{x}_j$ consist of a linear combination of two unique orientations n_{d_1} and n_{d_2}. Let s_1, s_2 be the similarity of the two frequencies of the field and the one frequency of the filter. Then $T_o(\boldsymbol{x}_j)$ has the eigenvalues $\lambda_3 = 0$ and*

$$\lambda_{1,2} = \frac{s_1^2 + s_2^2}{2} \pm \sqrt{\left(\frac{s_1^2 + s_2^2}{2}\right)^2 - \|s_1\|\|s_2\|\|sin^2(\phi)\|}.$$

   If n_{d_1} and n_{d_2} are orthogonal, this simplifies to $\lambda_1 = s_1$, $\lambda_2 = s_2$, and the eigenvectors are $e_1 = n_{d_1}$ and $e_2 = n_{d_2}$.

3. *Let the neighborhood of $\boldsymbol{x}_j$ consist of a linear combination of three unique orientations. Then $T_o(\boldsymbol{x}_j)$ has three non-zero eigenvalues.*

For other dimensions of the scalar field, the properties are analogous. The proof can be found in the literature [36,41]. This orientation tensor can also be combined with arbitrary templates and used for determining a rotation invariant similarity as well as the orientation of the structure in the field as long as the template is describable by a simple function. For other templates, only approximations can be computed and the error may be high. Further discussion will be postponed to Section 5.2, where the orientation tensor is applied to template matching of vector fields.

Chapter 4

Clifford Algebra

Clifford algebra extends the classical description of Euclidean n-space — which is a real n-dimensional vector space with scalar product — to a real algebra. Elements of this algebra are called multivectors and include scalars, vectors, and other elements. The vectors describing flow are usually 2D or 3D, and therefore the main focus in this thesis is on Clifford algebras in these dimensions. Here, a special Clifford algebra which is also called geometric algebra is used because of its useful geometric properties. Using Clifford algebra has the following advantages:

- It provides a unifying notation for scalars, vectors, and spinors. Complex numbers, which play an integral part of many signal processing concepts and methods, can be integrated as well.
- When working with multidimensional signals and including vector fields, many different mathematical approaches for representing the elements are used, and the operations on them are mixed as well. Clifford algebra provides a unification for the operators, too.
- When rotations are described by means of matrix algebra, their use is often not intuitive. Rotation angle and axis of a 3D rotation do not necessarily show up in their corresponding matrix. A more intuitive description of rotations are spinors, which are contained in Clifford algebra. (In 3D, spinors are also called quaternions.)
- The basis elements can be interpreted as points, vectors, planes, and volumes. Operations on these basis elements have geometric correspondences as well, which aids in the understanding of and thinking in Clifford Algebra.
- For the definition of a correlation of vector fields, a multiplication of vectors had to be defined or chosen. Clifford algebra provides a multiplication with

useful geometric properties.

In this chapter, Clifford algebra is explained as far as it is important for the understanding of this thesis. Clifford algebra in 2D and 3D are introduced in Section 4.1 and 4.2 respectively. A short elaboration of Clifford algebra in nD, as well as some Clifford calculus including the definition of integral and derivative, can be found in Section 4.3. As some of the related work is based on other, slightly different definitions of Clifford and other algebras, these algebras and the relationship to the Clifford algebra used in this thesis can be found in Section 4.4. The material of this chapter is mainly taken from [46, 47, 79].

4.1 Clifford Algebra in 2D

The 2D Euclidean vector space E^2 is spanned by two orthonormal vectors e_1 and e_2:

Definition 4.1.1 *Let E^2 be the $\mathbb{R}$-vector space generated by the basis e_1, e_2 with the Euclidean* **inner product**

$$\begin{gathered}\langle\cdot,\cdot\rangle : E^2 \to \mathbb{R} \\ (a,b) \mapsto \langle a,b\rangle = a\cdot b = a_1b_1 + a_2b_2 \\ \text{for } a = a_1e_1 + a_2e_2 \text{ and } b = b_1e_1 + b_2e_2\end{gathered}$$

which is distributive and associative, and identical to the scalar product.

Note that $\langle e_i, e_j\rangle = \delta_{ij}$ which yields the orthogonality. Furthermore, the inner product is grade reducing as it maps from E^2 into $\mathbb{R}$ and thus reduces vectors to a scalar.

Definition 4.1.2 *Let V_2 be the $\mathbb{R}$-vector space with basis $\{1, \boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_1 \wedge \boldsymbol{e}_2\}$. The bilinear* **outer product** $\wedge$ *is defined as*

$$\begin{gathered}\wedge : V_2 \to V_2, \\ (\boldsymbol{A}, \boldsymbol{B}) \mapsto \boldsymbol{A} \wedge \boldsymbol{B}\end{gathered}$$

given by

$$\begin{gathered}\alpha \wedge \boldsymbol{A} = \alpha\boldsymbol{A} \qquad \forall\, \alpha \in \mathbb{R} \\ \text{and} \quad \boldsymbol{a} \wedge \boldsymbol{b} = -\boldsymbol{b} \wedge \boldsymbol{a} \quad \forall\, \boldsymbol{a}, \boldsymbol{b} \in E^2 \subset V_2.\end{gathered}$$

The outer product is also called wedge or Grassmann product and leads out of E^2. The basis element $\mathbf{e}_1 \wedge \mathbf{e}_2$ can be interpreted as the oriented area spanned by the two unit vectors $\mathbf{e}_1$ and $\mathbf{e}_2$. For arbitrary vectors $\mathbf{a}, \mathbf{b}, \mathbf{c} \in E^2 \subset V_2$ we get $\mathbf{a} \wedge \mathbf{a} = 0$

and $\mathbf{a} \wedge \mathbf{b} \wedge \mathbf{c} = 0$. Note that the wedge product can increase, decrease or maintain the grade of an element $A \in V_2$.

The inner and outer products can be unified to the so called geometric product. With this geometric product, the Clifford algebra G_2 is defined:

Definition 4.1.3 *Let E^2 be as before. The* **Clifford** *or* **geometric algebra** *G_2 is the real 4-dimensional vector space with basis $\{1, \boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_1\boldsymbol{e}_2\}$ and the bilinear, associative* **geometric product**

$$\begin{gathered} G_2 \times G_2 \to G_2, \\ (\boldsymbol{A}, \boldsymbol{B}) \mapsto \boldsymbol{AB} \end{gathered}$$

given by

$$\begin{aligned} 1\boldsymbol{e}_1 &= \boldsymbol{e}_1, \qquad 1\boldsymbol{e}_2 = \boldsymbol{e}_2, \\ \boldsymbol{e}_1\boldsymbol{e}_1 &= 1, \qquad \boldsymbol{e}_2\boldsymbol{e}_2 = 1, \\ \textit{and} \quad \boldsymbol{e}_1\boldsymbol{e}_2 &= -\boldsymbol{e}_2\boldsymbol{e}_1. \end{aligned}$$

The geometric product can also be defined by a multiplication table stating the multiplication rules for each pair of basis elements (Table 4.1).

	1	e_1	e_2	e_1e_2
1	1	e_1	e_2	e_1e_2
e_1	e_1	1	e_1e_2	e_2
e_2	e_2	$-e_1e_2$	1	$-e_1$
e_1e_2	e_1e_2	$-e_2$	e_1	-1

Table 4.1: Multiplication table of the basis elements of G_2. Note that multiplication is not commutative. The left factors are indicated by the first column and the right factors by the first row.

Since Clifford algebra might be considered complicated, here is the derivation of the multiplication of two arbitrary elements $\mathbf{A}, \mathbf{B} \in G_2$. Let

$$\mathbf{A} = a_0 + a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + a_3\mathbf{e}_1\mathbf{e}_2$$

and

$$\mathbf{B} = b_0 + b_1\mathbf{e}_1 + b_2\mathbf{e}_2 + b_3\mathbf{e}_1\mathbf{e}_2.$$

Then the geometric product **AB** can be computed as follows:

$$\begin{aligned}
\mathbf{AB} &= a_0\mathbf{B} + a_1\mathbf{e}_1\mathbf{B} + a_2\mathbf{e}_2\mathbf{B} + a_3\mathbf{e}_1\mathbf{e}_2\mathbf{B} \\
&= a_0b_0 + a_0b_1\mathbf{e}_1 + a_0b_2\mathbf{e}_2 + a_0b_3\mathbf{e}_1\mathbf{e}_2 \\
&\quad + a_1\mathbf{e}_1b_0 + a_1\mathbf{e}_1b_1\mathbf{e}_1 + a_1\mathbf{e}_1b_2\mathbf{e}_2 + a_1\mathbf{e}_1b_3\mathbf{e}_1\mathbf{e}_2 \\
&\quad + a_2\mathbf{e}_2b_0 + a_2\mathbf{e}_2b_1\mathbf{e}_1 + a_2\mathbf{e}_2b_2\mathbf{e}_2 + a_2\mathbf{e}_2b_3\mathbf{e}_1\mathbf{e}_2 \\
&\quad + a_3\mathbf{e}_1\mathbf{e}_2b_0 + a_3\mathbf{e}_1\mathbf{e}_2b_1\mathbf{e}_1 + a_3\mathbf{e}_1\mathbf{e}_2b_2\mathbf{e}_2 + a_3\mathbf{e}_1\mathbf{e}_2b_3\mathbf{e}_1\mathbf{e}_2 \\
&= a_0b_0 + a_1b_1 + a_2b_2 - a_3b_3 \\
&\quad + (a_0b_1 + a_1b_0 - a_2b_3 + a_3b_2)\mathbf{e}_1 \\
&\quad + (a_0b_2 + a_2b_0 + a_1b_3 - a_3b_1)\mathbf{e}_2 \\
&\quad + (a_0b_3 + a_3b_0 + a_1b_2 - a_2b_1)\mathbf{e}_1\mathbf{e}_2
\end{aligned}$$

An arbitrary element $\mathbf{A} = \alpha + a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + \beta\mathbf{e}_1\mathbf{e}_2 \in G_2$ is called multivector and consist of a scalar α, a vector $\mathbf{a} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2$, and a so called bivector $\beta\mathbf{e}_1\mathbf{e}_2$. The bivector basis element $\mathbf{e}_1\mathbf{e}_2$ corresponds to $\mathbf{e}_1 \wedge \mathbf{e}_2 \in V_2$ and can be interpreted as the oriented area spanned by the two unit vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ (Figure 4.1). Note that the orientation of the area is the geometric motivation for using a non-commutative product.

In G_2, the bivector is also called pseudoscalar. It is often denoted by $\boldsymbol{i}$, or $\boldsymbol{i}_2$ when indicating that the pseudoscalar of G_2 is meant. Multiplication of a vector $\mathbf{a}$ with $\boldsymbol{i}$ corresponds to a counterclockwise rotation of the vector by a right angle (Figure 4.1). Note that $\boldsymbol{i}^2 = (\mathbf{e}_1\mathbf{e}_2)^2 = -1$. Thus, the spinors $S_2 = \{\alpha + \boldsymbol{i}\beta | \alpha, \beta \in \mathbb{R}\} \subset G_2$ form a group. Each spinor can be regarded as a representation of a combined rotation and dilation, and $\boldsymbol{i}$ is the generator of the rotations. Furthermore, S_2 is isomorphic to the complex numbers $\mathbb{C}$, which are fundamental for many signal processing concepts and algorithms. $\boldsymbol{i} \in G_2$ corresponds to the imaginary unit $i \in \mathbb{C}$. Thus we also get the following lemma which is later needed for the definition of a Fourier transform of multivectors:

Lemma 4.1.4 *$e^{(\boldsymbol{i}\gamma)} = cos(\gamma) + \boldsymbol{i}sin(\gamma)\ \forall\gamma \in \mathbb{R}$, and thus a spinor.*

Partitioning the basis elements of G_2 into vector and spinor basis elements yields a separation which is closed under multiplication with a bivector or a complete spinor (Table 4.1). It is understandable as the rotation of a vector will again yield a vector, and combining two rotations will result in a new rotation. This separation is also given by the definition of dual pairs:

Definition 4.1.5 *The* **dual** *of A is defined as $\boldsymbol{i}A$. The dual of a scalar is a bivector, and the dual of a vector is another vector. The* **dual pairs** *of the basis elements are*

$$\begin{aligned}
1 &\leftrightarrow \boldsymbol{i} \\
\textit{and}\quad \boldsymbol{e}_1 &\leftrightarrow \boldsymbol{e}_2.
\end{aligned}$$

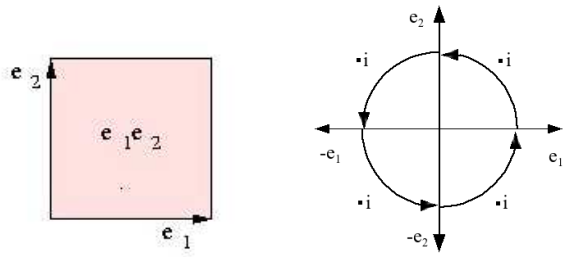

Figure 4.1: **Left**: The two unit vectors and the bivector in the Clifford Algebra G_2. **Right**: Multiplication with $\boldsymbol{i}$ corresponds to a multiplication by a right angle.

Note that the dual pairs are only given as basis elements as e.g. the dual of $\mathbf{e}_1$ is $\boldsymbol{i}\mathbf{e}_1 = \mathbf{e}_2$ and the dual of $\mathbf{e}_2$ is $-\mathbf{e}_1$. Remember that $\boldsymbol{i}$ corresponds to a counter-clockwise rotation by a right angle.

The elements of G_2 can also be sorted by grade (Table 4.2). A multivector $\mathbf{A}$ in G_2 will often be written as the sum of its elements of different grade,

$$\mathbf{A} = \alpha + \mathbf{a} + \boldsymbol{i}\beta$$

with $\alpha, \beta \in \mathbb{R}, \mathbf{a} \in E^2$ and $\boldsymbol{i} = \mathbf{e}_1\mathbf{e}_2$.

name	grade	dimension	basis elements
scalar	0	1	1
vector	1	2	$\mathbf{e}_1, \mathbf{e}_2$
bivector	2	1	$\mathbf{e}_1\mathbf{e}_2$

Table 4.2: The elements of G_2 sorted by grade.

Definition 4.1.6 *The* **grade projectors** $\langle\cdot\rangle_k : G_2 \to G_2$ *are the maps*

$$\langle \boldsymbol{A} \rangle_0 = \alpha,\ \langle \boldsymbol{A} \rangle_1 = \boldsymbol{a},\ \langle \boldsymbol{A} \rangle_2 = \boldsymbol{i}\beta$$

for $\boldsymbol{A} = \alpha + \boldsymbol{a} + \boldsymbol{i}\beta$.

Now we can come back to the relation of the geometric product and inner and outer product. With the canonic vector space isomorphism

$$\Phi : G_2 \to V,$$
$$\Phi(1) = 1, \Phi(\mathbf{e}_1) = e_1, \Phi(\mathbf{e}_2) = \mathbf{e}_2, \Phi(\mathbf{e}_1\mathbf{e}_2) = \mathbf{e}_1 \wedge \mathbf{e}_2,$$

$\wedge$ and $\langle\cdot\rangle$ can be transported into G_2. For two vectors $\mathbf{a},\mathbf{b} \in E^2 \subset G_2$ we get

$$\begin{aligned}\mathbf{ab} &= (a_1\mathbf{e}_1 + a_2\mathbf{e}_2)(b_1\mathbf{e}_1 + b_2\mathbf{e}_2)\\ &= a_1b_1\mathbf{e}_1^2 + a_1b_2\mathbf{e}_1\mathbf{e}_2 + a_2b_1\mathbf{e}_2\mathbf{e}_1 + a_2b_2\mathbf{e}_2^2\\ &= a_1b_1 + a_2b_2 + (a_1b_2 - a_2b_1)\mathbf{e}_1\mathbf{e}_2\\ &= \langle\mathbf{a},\mathbf{b}\rangle + \mathbf{a}\wedge\mathbf{b}.\end{aligned}$$

This motivates the following definitions:

Definition 4.1.7 *Let $\boldsymbol{a},\boldsymbol{b} \in E^2 \subset G_2$. The* **inner product** $\langle\boldsymbol{a},\boldsymbol{b}\rangle$ *is defined by*

$$\langle\boldsymbol{a},\boldsymbol{b}\rangle = \langle\boldsymbol{ab}\rangle_0 = \frac{1}{2}(\boldsymbol{ab}+\boldsymbol{ba}).$$

The **outer product** $\boldsymbol{a}\wedge\boldsymbol{b}$ *is defined by*

$$\boldsymbol{a}\wedge\boldsymbol{b} = \langle\boldsymbol{ab}\rangle_2 = \frac{1}{2}(\boldsymbol{ab}-\boldsymbol{ba}).$$

The inner product can be extended to a scalar product of multivectors. For this, an operation similar to conjugation in $\mathbb{C}$ is defined:

Definition 4.1.8 *Let $\boldsymbol{A} = \alpha + \boldsymbol{a} + \boldsymbol{i}\beta \in G_2$. Then the* **reversion** $\boldsymbol{A}^+$ *is defined by*

$$\boldsymbol{A}^+ = \alpha + \boldsymbol{a} - \boldsymbol{i}\beta$$

Definition 4.1.9 *Let $\boldsymbol{A} = \alpha + \boldsymbol{a} + \boldsymbol{i}\beta, \boldsymbol{B} = \gamma + \boldsymbol{c} + \boldsymbol{i}\delta \in G_2$. The* **scalar product** $\langle\boldsymbol{A},\boldsymbol{B}\rangle$ *is defined by*

$$\langle\boldsymbol{A},\boldsymbol{B}\rangle = \alpha\gamma + \langle\boldsymbol{a},\boldsymbol{c}\rangle + \beta\delta = \langle\boldsymbol{A}\boldsymbol{B}^+\rangle_0 \in \mathbb{R} \subset G_2$$

Definition 4.1.10 *The* **magnitude** *of $\boldsymbol{A} \in G_2$ is*

$$\|\boldsymbol{A}\| = +\sqrt{\langle\boldsymbol{A},\boldsymbol{A}\rangle}$$

For a vector $\mathbf{a} \in G_2$, the magnitude $\|\mathbf{a}\| = +\sqrt{\langle\mathbf{a},\mathbf{a}\rangle}$ is the usual Euclidean length.

The geometric product of two vectors **a** and **b** is a spinor and describes the rotation and dilation which is necessary to transform **a** into **b**. This geometric interpretation of the geometric product, or the inner and outer products, can now be specified:

Lemma 4.1.11 *Let $\boldsymbol{a},\boldsymbol{b} \in E^2 \subset G_2$, and ω be the angle between $\boldsymbol{a}$ and $\boldsymbol{b}$. Then*

$$\begin{aligned}&\langle\boldsymbol{ab}\rangle_0 = \langle\boldsymbol{a},\boldsymbol{b}\rangle = \|\boldsymbol{a}\|\,\|\boldsymbol{b}\|\cos\omega\\ \textit{and}\quad &\langle\boldsymbol{ab}\rangle_2 = \boldsymbol{a}\wedge\boldsymbol{b} = \|\boldsymbol{a}\|\,\|\boldsymbol{b}\|\sin\omega,\end{aligned}$$

Furthermore, every non-zero vector in G_2 has an inverse, and thus division of vectors can be computed in the Clifford algebra G_2:

Lemma 4.1.12 *Let $\boldsymbol{a} \in E^2 \backslash \{0\}$. Then there exist an inverse $\boldsymbol{a}^{-1} = \frac{1}{\|\boldsymbol{a}\|}\boldsymbol{a}$*

Proof*: $\|\boldsymbol{a}| > 0$ for $\boldsymbol{a} \in E^2 \backslash \{0\}$. Therefore $\frac{1}{\|\boldsymbol{a}\|}\boldsymbol{a} \in E^2$ and*

$$\frac{1}{\|\boldsymbol{a}\|}\boldsymbol{a}\boldsymbol{a} = \frac{1}{\|\boldsymbol{a}\|}\langle \boldsymbol{a}, \boldsymbol{a}\rangle + \boldsymbol{a} \wedge \boldsymbol{a} = \frac{1}{\|\boldsymbol{a}\|}\langle \boldsymbol{a}, \boldsymbol{a}\rangle = 1$$

Non-zero spinors also have inverse elements:

Lemma 4.1.13 *Let $\boldsymbol{A} = \alpha + \boldsymbol{i}\beta \in S_2 \subset G_2$. Then there exist an inverse $\boldsymbol{A}^{-1} = \frac{1}{\|\boldsymbol{A}\|}\boldsymbol{A}^+$*

This was expected as spinors describe rotation and dilation, and the inverse of a spinor reverses the rotation and dilation. Note that arbitrary multivectors $\mathbf{A} \in G_2 \backslash \{0\}$ do not always have inverse elements.

4.2 Clifford Algebra in 3D

In the 3-dimensional Euclidean vector space E^3 with basis $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$, the vectors can not only span oriented areas but also volumes. This is reflected in the Clifford or geometric algebra G_3:

Definition 4.2.1 *Let E^3 be the 3-dimensional Euclidean vector space with the basis $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\}$. The* **Clifford** *or* **geometric algebra** *G_3 is the real 8-dimensional vector space with basis $\{1, \boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3, \boldsymbol{e}_1\boldsymbol{e}_2, \boldsymbol{e}_2\boldsymbol{e}_3, \boldsymbol{e}_3\boldsymbol{e}_1, \boldsymbol{e}_1\boldsymbol{e}_2\boldsymbol{e}_3\}$ and the bilinear, associative* **geometric product**

$$\begin{gathered} G_3 \times G_3 \to G_3, \\ (\boldsymbol{A}, \boldsymbol{B}) \mapsto \boldsymbol{A}\boldsymbol{B} \end{gathered}$$

given by

$$\begin{aligned} 1\boldsymbol{e}_j &= \boldsymbol{e}_j, && j = 1,2,3\,, \\ \boldsymbol{e}_j\boldsymbol{e}_j &= 1, && j = 1,2,3\,, \\ \textit{and}\quad \boldsymbol{e}_j\boldsymbol{e}_k &= -\boldsymbol{e}_k\boldsymbol{e}_j, && j,k = 1,2,3,\ j \neq k\,. \end{aligned}$$

An arbitrary multivector $\mathbf{A} \in G_3$ consists of a scalar α, a vector $\mathbf{a} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + a_3\mathbf{e}_3$, a bivector $\mathbf{b} = b_1\mathbf{e}_2\mathbf{e}_3 + b_2\mathbf{e}_3\mathbf{e}_1 + b_3\mathbf{e}_1\mathbf{e}_2$, and a trivector $\beta\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3$. The unit bivectors correspond to three oriented areas orthogonal to each other, and the unit trivector $\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3$ corresponds to the unit volume (Figure 4.2).
In G_3, the trivector $\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3$ is also called pseudoscalar. It is denoted by $\boldsymbol{i}$, or $\boldsymbol{i}_3$ when indicating that the pseudoscalar of G_3 is meant. The trivector $\boldsymbol{i}$ is commutative to every arbitrary multivector $\mathbf{A} \in G_3$ (Table 4.3). The **Hodge-duality** states that the trivector converts vectors into bivectors and vice versa:

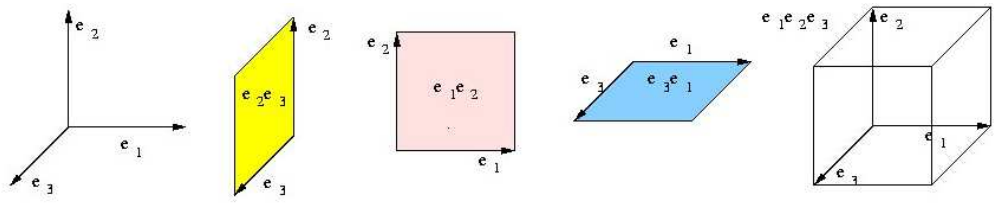

Figure 4.2: **Left**: The three unit vectors in G_3. **Middle**: The three unit bivectors in G_3 representing areas. **Right**: The unit trivector in G_3 representing the unit volume.

Lemma 4.2.2 (Hodge-duality)

$$\begin{aligned} \boldsymbol{e}_1\boldsymbol{e}_2 &= \boldsymbol{e}_1\boldsymbol{e}_2\boldsymbol{e}_3\boldsymbol{e}_3 = \boldsymbol{i}\boldsymbol{e}_3 \\ \boldsymbol{e}_2\boldsymbol{e}_3 &= \boldsymbol{i}\boldsymbol{e}_1 \\ \boldsymbol{e}_3\boldsymbol{e}_1 &= \boldsymbol{i}\boldsymbol{e}_2 \end{aligned}$$

Thus, an arbitrary multivector $\mathbf{A} \in G_3$ can be written as

$$\mathbf{A} = \alpha + \mathbf{a} + \boldsymbol{i}(\mathbf{b} + \beta)$$

with $\alpha, \beta \in \mathbb{R}$, $\mathbf{a}, \mathbf{b} \in E^3$, and $\boldsymbol{i} = \mathbf{e}_1\mathbf{e}_2\mathbf{e}_3$.

Definition 4.2.3 *The* **dual** *of a multivector* $\boldsymbol{A}$ *is defined as* $-\boldsymbol{A}\boldsymbol{i}$*. Thus, the dual of a scalar is a trivector, and the dual of a vector is a bivector analog to the Hodge-duality.*

	1	e_1	e_2	e_3	e_2e_3	e_3e_1	e_1e_2	$e_1e_2e_3$
1	1	e_1	e_2	e_3	e_2e_3	e_3e_1	e_1e_2	$e_1e_2e_3$
e_1	e_1	1	e_1e_2	$-e_3e_1$	$e_1e_2e_3$	$-e_3$	e_2	e_2e_3
e_2	e_2	$-e_1e_2$	1	e_2e_3	e_3	$-e_1e_2e_3$	$-e_1$	e_3e_1
e_3	e_3	e_3e_1	$-e_2e_3$	1	$-e_2$	e_1	$e_1e_2e_3$	e_1e_2
e_2e_3	e_2e_3	$e_1e_2e_3$	$-e_3$	e_2	-1	$-e_1e_2$	e_3e_1	$-e_1$
e_3e_1	e_3e_1	e_3	$e_1e_2e_3$	$-e_1$	e_1e_2	-1	$-e_2e_3$	$-e_2$
e_1e_2	e_1e_2	$-e_2$	e_1	$e_1e_2e_3$	$-e_3e_1$	e_2e_3	-1	$-e_3$
$e_1e_2e_3$	$e_1e_2e_3$	e_2e_3	e_3e_1	e_1e_2	$-e_1$	$-e_2$	$-e_3$	-1

Table 4.3: Multiplication table of the basis elements of G_3. Note that multiplication is not commutative. The left factors are indicated by the first column and the right factors by the first row.

Note that $(\mathbf{e}_1\mathbf{e}_2)^2 = -1, (\mathbf{e}_2\mathbf{e}_3)^2 = -1, (\mathbf{e}_3\mathbf{e}_1)^2 = -1$ and $(\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3)^2 = -1$ (Table 4.3). $S_3 = \{\alpha + \boldsymbol{i}\mathbf{b}\} \subset G_3$ is the group of the quaternions, an extension of the complex numbers which is often used to describe rotations, and which is a special group of spinors.

Now there are several possibilities of choosing an isomorphism to the complex numbers. Often the trivector is chosen for the complex i as already indicated by naming it $\boldsymbol{i}$. $C_3 = \{\alpha + \boldsymbol{i}\beta\}$ is isomorph to the complex numbers $\mathbb{C}$. Any of the sets $\{\alpha + b_j\boldsymbol{i}\mathbf{e}_j\}, j = 1,2,3$ based on the scalar and a unit bivector would also be isomorph to the complex numbers. However, in this thesis the choice of the trivector as imaginary number is advantageous as the resulting complex numbers are commutative to every multivector in G_3 (Table 4.3).

Lemma 4.2.4 *For every scalar* γ

$$e^{(\boldsymbol{i}\gamma)} = cos(\gamma) + \boldsymbol{i}sin(\gamma)$$

and for every multivector $\boldsymbol{A} \in G_3$

$$\boldsymbol{A}e^{(\boldsymbol{i}\gamma)} = e^{(\boldsymbol{i}\gamma)}\boldsymbol{A}.$$

Proof:

$$\begin{aligned} \boldsymbol{A}e^{(\boldsymbol{i}\gamma)} &= \boldsymbol{A}cos(\gamma) + \boldsymbol{A}\boldsymbol{i}sin(\gamma) \\ &= cos(\gamma)\boldsymbol{A} + \boldsymbol{i}sin(\gamma)\boldsymbol{A} \\ &= e^{(\boldsymbol{i}\gamma)}\boldsymbol{A}\,. \end{aligned}$$

Again, inner and outer product of two vectors can be defined. The cross product, which corresponds to the normal vector of the area spanned by two vectors whereas the outer product corresponds to the oriented area itself (Figure 4.3), can also be defined in the Clifford algebra G_3:

Definition 4.2.5 *The* **grade projectors** $\langle\cdot\rangle_k : G_3 \to G_3$ *are the maps*

$$\langle\boldsymbol{A}\rangle_0 = \alpha,\ \langle\boldsymbol{A}\rangle_1 = \boldsymbol{a},\ \langle\boldsymbol{A}\rangle_2 = \boldsymbol{i}\boldsymbol{b}, \langle\boldsymbol{A}\rangle_3 = \boldsymbol{i}\beta$$

for $\boldsymbol{A} = \alpha + \boldsymbol{a} + \boldsymbol{i}(\boldsymbol{b} + \beta)$.

name	grade	dimension	basis elements
scalar	0	1	1
vector	1	3	$\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$
bivector	2	3	$\mathbf{e}_2\mathbf{e}_3, \mathbf{e}_3\mathbf{e}_1, \mathbf{e}_1\mathbf{e}_2$
trivector	3	1	$\mathbf{e}_1\mathbf{e}_2\mathbf{e}_3$

Table 4.4: The elements of G_3 sorted by grade.

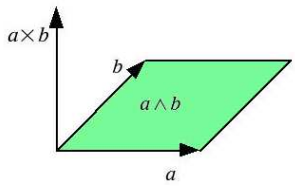

Figure 4.3: The cross product gives the normal vector of the area spanned by the two vectors **a** and **b**, whereas the outer product of **a** and **b** corresponds to the oriented area itself.

Definition 4.2.6 *Let* $\boldsymbol{a},\boldsymbol{b} \in E^3 \subset G_3$. *The* **inner product** $\langle \boldsymbol{a},\boldsymbol{b}\rangle$ *is defined by*

$$\langle \boldsymbol{a},\boldsymbol{b}\rangle = \langle \boldsymbol{ab}\rangle_0 = \frac{1}{2}(\boldsymbol{ab}+\boldsymbol{ba}).$$

The **outer product** $\boldsymbol{a}\wedge\boldsymbol{b}$ *is defined by*

$$\boldsymbol{a}\wedge\boldsymbol{b} = \langle \boldsymbol{ab}\rangle_2 = \frac{1}{2}(\boldsymbol{ab}-\boldsymbol{ba}).$$

The **cross product** $\boldsymbol{a}\times\boldsymbol{b}$ *is defined as*

$$\boldsymbol{a}\times\boldsymbol{b} = -\boldsymbol{i}(\boldsymbol{a}\wedge\boldsymbol{b})$$

Definition 4.2.7 *Let* $\boldsymbol{A} = \alpha+\boldsymbol{a}+\boldsymbol{i}(\boldsymbol{b}+\beta) \in G_3$. *Then the* **reversion** $\boldsymbol{A}^+$ *is defined by*

$$\boldsymbol{A}^+ = \alpha+\boldsymbol{a}-\boldsymbol{i}(\boldsymbol{b}+\beta)$$

Scalar product and magnitude are defined based on this reversion and analogous to G_2. Again every non-zero vector and spinor has an inverse element.

Lemma 4.2.8 *Clifford multiplication of two vectors* $\boldsymbol{a},\boldsymbol{b} \in E^3 \subset G_3$ *results in*

$$\boldsymbol{ab} = \langle \boldsymbol{a},\boldsymbol{b}\rangle + \boldsymbol{a}\wedge\boldsymbol{b},$$

Furthermore, $\boldsymbol{ab}$ *is a spinor or quaternion and has the properties*

$$\begin{aligned} \langle \boldsymbol{ab}\rangle_0 = \langle \boldsymbol{a},\boldsymbol{b}\rangle &= \|\boldsymbol{a}\|\,\|\boldsymbol{b}\|\cos\omega \\ \textit{and}\quad \|\langle \boldsymbol{ab}\rangle_2\| = \|\boldsymbol{a}\wedge\boldsymbol{b}\| &= \|\boldsymbol{a}\|\,\|\boldsymbol{b}\|\sin\omega, \end{aligned}$$

where ω *is the angle between* **a** *and* **b** *and the bivector* $\langle \boldsymbol{ab}\rangle_2$ *corresponds to the area spanned by* **a** *and* **b**, *and thus the plane where the angle is measured.*

4.3 Clifford Algebra and Calculus in nD

Now that Clifford algebra and its elements have been introduced and analyzed in 2D and 3D, a brief extension to arbitrary dimension is given. Some Clifford calculus including the definitions of integral and derivative is situated in this chapter, too.

Definition 4.3.1 *For the n-dimensional Euclidean vector space E^n with basis $\{\boldsymbol{e}_1,..,\boldsymbol{e}_n\}$, the 2^n-dimensional* **Clifford** *or* **geometric algebra** *G_n is defined as a direct sum $V_0 \oplus V_1 \oplus ... \oplus V_n$ with the basis elements from the following table*

space	*name*	*grad*	*dimension*	*basiselements*
V_0	*scalars*	0	$\binom{n}{0} = 1$	1
V_1	*vectors*	1	$\binom{n}{1} = n$	$\boldsymbol{e}_1,..,\boldsymbol{e}_n$
V_2	*bivectors*	2	$\binom{n}{2}$	$\boldsymbol{e}_j\boldsymbol{e}_k, j<k$
V_3	*trivectors*	3	$\binom{n}{3}$	$\boldsymbol{e}_j\boldsymbol{e}_k\boldsymbol{e}_l, j<k<l$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
V_k	$k-vectors$	k	$\binom{n}{k}$	$\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_2}...\boldsymbol{e}_{j_k}, j_1<...<j_k \in 1,..,n$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
V_n	$n-vectors$	n	$\binom{n}{n} = 1$	$\boldsymbol{e}_1..\boldsymbol{e}_n = \boldsymbol{i}_n$

together with the associative, bilinear **geometric product**

$$\begin{array}{c} G_n \times G_n \to G_n, \\ (\boldsymbol{A},\boldsymbol{B}) \mapsto \boldsymbol{AB} \end{array}$$

given by

$$\begin{array}{rrcll} & 1\boldsymbol{e}_j & = & \boldsymbol{e}_j, & j \in \{1,..,n\}, \\ & \boldsymbol{e}_j\boldsymbol{e}_j & = & 1, & j \in \{1,..,n\}, \\ and & \boldsymbol{e}_j\boldsymbol{e}_k & = & -\boldsymbol{e}_k\boldsymbol{e}_j, & j,k \in \{1,..,n\}, j \neq k\,. \end{array}$$

Definition 4.3.2 *An arbitrary element of G_n is called* **multivector**. *An element $A \in V_k$ which can be written as $A = \alpha\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_2}...\boldsymbol{e}_{j_k}$, $\alpha \in \mathbb{R}$ is called* **k-blade**.

Definition 4.3.3 *The canonic projection $\langle\cdot\rangle_k : G_n \to G_n$, $\boldsymbol{A} \mapsto \langle\boldsymbol{A}\rangle_k$ is called* **k-projector**.

The squared pseudoscalar $\boldsymbol{i}_n^2$ is not always -1, e.g. $\boldsymbol{i}_4^2 = 1$. The formula for calculating the square of an arbitrary pseudoscalar is $\boldsymbol{i}_n^2 = (-1)^{\frac{1}{2}n(n-1)}$. Thus the set $\{\alpha + \boldsymbol{i}_n\beta\}$ is not always isomorph to the complex numbers. The definition of the reversion has to be adjusted as well:

Definition 4.3.4 *Let $\boldsymbol{A} \in G_n$ be a multivector. Then the* **reversion** $\boldsymbol{A}^+$ *is defined by*

$$\langle \boldsymbol{A}^+ \rangle_k = (-1)^{\frac{1}{2}k(k-1)} \langle \boldsymbol{A} \rangle_k$$

Inner and outer product of two vectors, and scalar product and magnitude of multivectors are defined analog to G_2. Now let **F** be a multivector-valued function (field) of a vector variable **x** defined on some region of the Euclidean space E^n. (If the function is only scalar or vector valued, it will be called scalar or vector field, respectively.)

Definition 4.3.5 *The* **Riemann integral** *of a multivector-valued function $\boldsymbol{F}$ is defined as*

$$\int_{E^n} \boldsymbol{F}(\boldsymbol{x}) |d\boldsymbol{x}| = \lim_{\substack{|\Delta x_j| \to 0 \\ k \to \infty}} \sum_{i=1}^{k} \boldsymbol{F}(x_j \boldsymbol{e}_j) |\Delta x_j| \, .$$

The quantity $|d\mathbf{x}|$ is used to make the integral grade preserving since $d\mathbf{x}$ is a vector within Clifford algebra. Thus, the integral can be discretized into sums using quadrature formulas.

Definition 4.3.6 *The* **directional derivative** *of $\boldsymbol{F}$ in direction* **r** *is*

$$\boldsymbol{F}_{\boldsymbol{r}}(\boldsymbol{x}) = \lim_{h \to 0} \frac{[\boldsymbol{F}(\boldsymbol{x} + h\boldsymbol{r}) - \boldsymbol{F}(\boldsymbol{x})]}{h}$$

with $h \in \mathbb{R}$.

Definition 4.3.7 *The* **vector derivative** ∇ *is defined as*

$$\nabla = \sum_{j=1}^{n} \boldsymbol{e}_j \frac{\partial}{\partial x_j}$$

Note that ∇ is vector valued, and computation of the derivative can now be done using the geometric product:

Definition 4.3.8 *The (complete)* **derivative** *of $\boldsymbol{F}$ from the left is*

$$\nabla \boldsymbol{F}(\boldsymbol{x}) = \sum_{j=1}^{n} \boldsymbol{e}_j \boldsymbol{F}_{\boldsymbol{e}_j}(\boldsymbol{x}),$$

where $\boldsymbol{F}_{\boldsymbol{e}_j}(\boldsymbol{x})$ are the directional derivatives. The derivative from the right is

$$\boldsymbol{F}(\boldsymbol{x})\nabla = \sum_{j=1}^{n} \boldsymbol{F}_{\boldsymbol{e}_j}(\boldsymbol{x})\boldsymbol{e}_j \ .$$

In contrast to the Jacobian (Definition 2.2.1), which is a matrix, the derivative here is spinor-valued. In flow analysis and visualization, curl and divergence (Definition 2.2.2) are often used. They can also be computed within Clifford algebra:

Definition 4.3.9 Curl *and* **divergence** *of a 2D or 3D vector-valued function $\boldsymbol{f}$ can be computed as*

$$\begin{aligned} & \mathit{curl}\, \boldsymbol{f} = \nabla \wedge \boldsymbol{f} = \frac{(\nabla \boldsymbol{f} - \boldsymbol{f}\nabla)}{2} \\ \mathit{and} \qquad & \mathit{div}\, \boldsymbol{f} = \langle \nabla, \boldsymbol{f} \rangle = \frac{(\nabla \boldsymbol{f} + \boldsymbol{f}\nabla)}{2}. \end{aligned}$$

This curl operator gives the bivector describing the plane of strongest rotation whereas the classical curl operator results in the corresponding normal vector.

4.4 Other Clifford Algebras

In some of the related work concerning the definition of a Fourier transform within Clifford algebra, different definitions of Clifford algebra are used [8, 9, 34]. Let $\mathbf{e}_1, \ldots \mathbf{e}_n$ be an orthogonal basis of E^n. The used Clifford algebras mostly differ in the definition of the square of the unit vectors which is $\mathbf{e}_j^2 = \pm 1, \ j = 1, .., n$. These definitions can be integrated into one algebra:

Definition 4.4.1 *Let $(p,q), p+q=n$ be a non-degenerate quadratic form of signature for E^n, and $\boldsymbol{e}_1, .., \boldsymbol{e}_n$ be an orthogonal basis of E^n. The basis elements of the Clifford algebra $E_{p,q}$ are constructed analog to G_n, and the associative, bilinear multiplication is defined by the rules*

$$\begin{aligned} 1\boldsymbol{e}_j &= \boldsymbol{e}_j, && j = 1, .., n \ , \\ \boldsymbol{e}_j\boldsymbol{e}_j &= 1, && j = 1, .., p \ , \\ \boldsymbol{e}_j\boldsymbol{e}_j &= -1, && j = p+1, .., m \ , \\ \mathit{and} \quad \boldsymbol{e}_j\boldsymbol{e}_k &= -\boldsymbol{e}_k\boldsymbol{e}_j, && j,k = 1, .., n, \ j \neq k \ . \end{aligned}$$

The Clifford algebra G_n which is used in this thesis corresponds to $E_{n,0}$, and another often used Clifford algebra is $E_{0,n}$.

A big disadvantage of Clifford algebras when ignoring the geometric interpretation is the non-commutativity of the multiplication. Subsequent problems have led to the definition and use of hypercomplex algebras where multiplication is commutative, e.g. the algebra HCA_n [9]:

Definition 4.4.2 *Let $\boldsymbol{e}_1,..,\boldsymbol{e}_n$ be n symbols obeying*

$$\begin{aligned} 1\boldsymbol{e}_j &= \boldsymbol{e}_j, & j=1,..,n\ , \\ \boldsymbol{e}_j\boldsymbol{e}_j &= -1, & j=1,..,p\ , \\ \textit{and}\quad \boldsymbol{e}_j\boldsymbol{e}_k &= \boldsymbol{e}_k\boldsymbol{e}_j, & j,k=1,..,n,\ j\neq k\ . \end{aligned}$$

Creation of the 2^n basis elements out of $\boldsymbol{e}_1,..,\boldsymbol{e}_n$ is analog to G_n. These basis elements, and the product defined by the multiplication rules above, define the hypercomplex algebra HCA_n.

Chapter 5

Template Matching of Vector Fields

The use of scalar LSI filters on vector fields is unproblematic for the most part, as convolution can be defined using the multiplication of a vector with a scalar:

Definition 5.0.3 *Let $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ be a vector field and $h : \mathbb{R}^n \to \mathbb{R}$ be a scalar-valued filter. Then* **convolution** *of $\boldsymbol{f}$ and h is defined as:*

$$(h * \boldsymbol{f})(\boldsymbol{x}) = \int_{\mathbb{R}^n} h(\boldsymbol{x'})\boldsymbol{f}(\boldsymbol{x} - \boldsymbol{x'})d\boldsymbol{x'}.$$

The definition of a convolution of two vector fields, however, is more challenging. In this chapter, several approaches to this problem are introduced and discussed.

Note that for template matching, the correlation is of interest and not the convolution. However, every correlation can be computed by a convolution with a suitably adjusted filter or template. The use of convolution instead of the correlation is due to the convolution theorem (Theorem 3.3.8) and the analysis and acceleration thus possible. Therefore, mostly the convolution is regarded in the following sections. Keep in mind that the correlation can always be defined analogously.

The similarity measure should be independent of the direction of the structure within the vector field and the template. Otherwise, one has to rotate the template many times and compute the similarities for all the rotated templates. In a last step, it would be necessary to compute the maximum similarity and take the corresponding direction as the direction of the structure. Thus, defining convolution and correlation for vector-valued fields is only the beginning, and a rotation invariant template matching algorithm based on these definitions has to follow. In this chapter, several approaches for both tasks are introduced and discussed in detail as they will provide a starting point for this thesis. Especially the Clifford convolution (Section 5.4), which was developed in the master thesis of the author [23] along with a rotation invariant template matching algorithm (Section 5.5), will be important later on.

5.1 Separate Matching of Component Fields

The first idea regarding image processing on vector fields is to simply treat a vector field as several scalar fields. Thus, convolution and Fourier transformation of the separated scalar components of the vector can be computed. However, a vector represents more information than its separated components provide. Furthermore, the scalar fields of the components are not independent and do not provide insight into the vector as a whole. Granlund and Knutson [36] have investigated this approach in 2D for optical flow fields as well as vector fields describing the local orientation of textures. The latter fields were computed using the orientation tensor (Definition 3.6.7). Sudden changes in the feature vector descriptor fields describe texture borders which can thus be extracted.

5.2 Template Matching using the Inner Product

Another approach of transferring convolution to vector fields makes use of the generalized inner product of pertinent vectors. This approach is first described by Heiberg et al. [41, 42].

Definition 5.2.1 *Let $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ be a vector field and $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^n$ be a filter. Then the* **vector convolution** $*_v$ *is defined as:*

$$(\boldsymbol{h} *_v \boldsymbol{f})(\boldsymbol{x}) = \int_{\mathbb{R}^n} \langle \boldsymbol{h}(\boldsymbol{x'}), \boldsymbol{f}(\boldsymbol{x} - \boldsymbol{x'}) \rangle d\boldsymbol{x'},$$

The inner product of normalized vectors provides an approximation to the cosine of the angle between the direction of patterns present in the vector field and the direction of the filters. Thus, the vector convolution of a template and the field results in a similarity measure which is approximately proportional to the cosine of the angle of the structures in field and template. Due to the difference of rotating one vector in contrast to rotating the complete template, the similarity may be much smaller, an example can be found in Figure 5.1.

The disadvantage of this approach is that a unifying notation including the convolution of scalar fields, or a scalar and a vector field, is not possible within this framework. Furthermore, Heiberg et al. [41, 42] do not formulate or use a Fourier transform in their method. However, the use of the scalar Fourier kernel for a Fourier transform is possible. This will transform each coordinate of the vector field separately as in Section 5.1.

Definition 5.2.2 *For a continuous signal $\boldsymbol{f} : \mathbb{R}^n \to C^n$, the* **vector Fourier transform** *of $\boldsymbol{f}$ is defined as*

$$\mathscr{F}\{\boldsymbol{f}\}(\boldsymbol{u}) = \int_{\mathbb{R}^n} \boldsymbol{f}(\boldsymbol{x}) e^{(-2\pi i \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} d\boldsymbol{x}$$

Due to the separate transformation of the coordinates, most of the theorems from Section 3.3 can be proven analogously. Because of the bilinearity of the inner product, the convolution theorem (Theorem 3.3.8) can also be proved.

Theorem 5.2.3 *Let $\boldsymbol{f}_1, \boldsymbol{f}_2 : \mathbb{R}^n \to \mathbb{C}^n$ be two signals. Then*

$$\mathscr{F}\{\boldsymbol{f}_1 *_v \boldsymbol{f}_2\} = \langle \mathscr{F}\{\boldsymbol{f}_1\}, \mathscr{F}\{\boldsymbol{f}_2\} \rangle$$

Proof*:*

$$\begin{aligned}
\mathscr{F}\{\boldsymbol{f}_1 * \boldsymbol{f}_2\}\boldsymbol{u} &= \int_{\mathbb{R}} (\boldsymbol{f}_1 * \boldsymbol{f}_2)(\boldsymbol{x}) e^{(-2\pi i \langle \boldsymbol{x}\boldsymbol{u} \rangle)} d\boldsymbol{x} \\
&= \int_{\mathbb{R}} (\int_{\mathbb{R}} \langle \boldsymbol{f}_1(\boldsymbol{k}), \boldsymbol{f}_2(\boldsymbol{x}-\boldsymbol{k}) \rangle d\boldsymbol{k}) e^{(-2\pi i \langle \boldsymbol{x}\boldsymbol{u} \rangle)} d\boldsymbol{x} \\
&= \int_{\mathbb{R}} \langle \boldsymbol{f}_1(\boldsymbol{k}), \int_{\mathbb{R}} \boldsymbol{f}_2(\boldsymbol{x}-\boldsymbol{k}) e^{(-2\pi i \langle \boldsymbol{x}\boldsymbol{u} \rangle)} d\boldsymbol{x} \rangle d\boldsymbol{k} \\
&= \int_{\mathbb{R}} \langle \boldsymbol{f}_1(\boldsymbol{k}), \mathscr{F}\{\boldsymbol{f}_2\}(\boldsymbol{u}) e^{(-2\pi i \langle \boldsymbol{k}\boldsymbol{u} \rangle)} \rangle d\boldsymbol{k} \\
&= \langle \int_{\mathbb{R}} \boldsymbol{f}_1(\boldsymbol{k}) e^{(-2\pi i \langle \boldsymbol{k}\boldsymbol{u} \rangle)} d\boldsymbol{k}, \mathscr{F}\{\boldsymbol{f}_2\}(\boldsymbol{u}) \rangle \\
&= \langle \mathscr{F}\{\boldsymbol{f}_1\}(\boldsymbol{u}), \mathscr{F}\{\boldsymbol{f}_2\}(\boldsymbol{u}) \rangle
\end{aligned}$$

Again, note that the result of the inner product of two vectors, and thus of the vector convolution too, is a scalar and not a vector. Thus, the vector domain in which the computation was done has to be left. Therefore, a unifying approach for these computations would be preferable as a mathematical basis for the analysis and application of these methods.

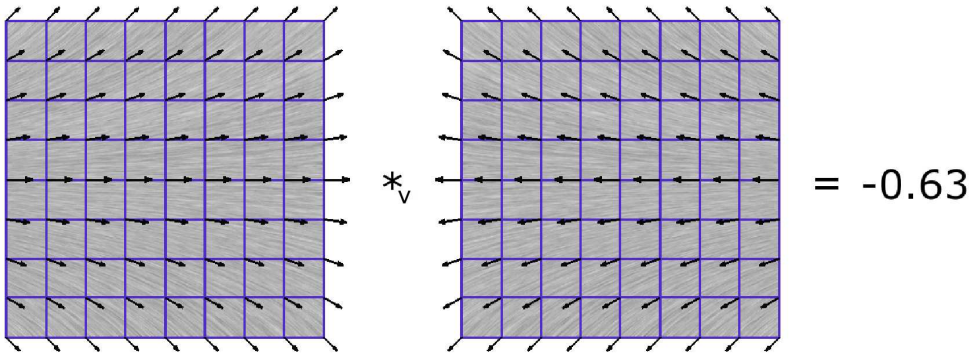

Figure 5.1: Rotation of the template leads to similarities much smaller than the expected value, here -0.63 instead of -1 for a correlation though the rotation angle of the vector field is 180°.

5.3 Rotation Invariant Matching with the Orientation Tensor

Heiberg et al. [41, 42] also propose a rotation invariant template matching algorithm based on the orientation tensor (Definition 3.6.7) to evaluate the filter responses of a few rotated templates to the vector field. This algorithm is introduced in this section in detail, an overview can be found in Table 5.1.

First of all, the vector field is normalized. Note that template matching of the normalized vector field corresponds to matching of the streamlines as they are independent of the velocity (Theorem 2.1.5). Then the templates are defined. Heiberg et al. [41, 42] state that their algorithm works only for axis-symmetric patterns. Let $\mathbf{h}_b$ denote one axis-symmetric, vector-valued template. The size of the patterns should be limited, therefore the templates are multiplied with a rotational symmetric weighting function:

Definition 5.3.1 *Let $r, r_{max} \in \mathbb{R}$. Define the rotational symmetric weighting function w with respect to the radius r as*

$$w : \mathbb{R} \to \mathbb{R}$$

$$r \mapsto \begin{cases} 1 & \|r\| < r_{max} \\ e^{-\frac{\|r\| - r_{max}}{\sigma^2}} & \|r\| \geq r_{max} \end{cases}$$

r_{max} controls the size of the patterns and σ influences the drop-off of the values. Note that for small templates, omitting this step makes no significant difference in the resulting similarity values.

The magnitude of the response of the original quadrature filter used in the orientation tensor is proportional to the square of the angle between the directions of filter and function. This is the motivation for scaling the magnitude of the vectors in the template again, this time setting the energy of the template to one. Thus, the templates used for the matching are

$$\mathbf{h}(\mathbf{x}) = \gamma w(r) \mathbf{h}_b(\mathbf{x})$$

with

$$E\{\mathbf{h}\} = \int_{\mathbb{R}^n} \langle \mathbf{h}(\mathbf{x}), \mathbf{h}(\mathbf{x}) \rangle d\mathbf{x} = 1$$

Each of the templates is now rotated to yield the directional responses necessary for the orientation tensor (Definition 3.6.7). In 2D, three filter directions $\mathbf{n}_k$, which correspond to their symmetry axis, are necessary for the computation of the orientation tensor. A possible filter distribution $\mathbf{h}_k$ can be computed by taking the template itself as well as rotated copies with rotation angles of 60° and 120°. In

3D, six filters $\mathbf{h}_k$ evenly distributed over a half-sphere are necessary, the formula for the filter directions $\mathbf{n}_k$ can be found in Section 3.6.4.

Now, the orientation tensor T_o of the template similarities can be computed at every position of the vector field $\mathbf{v}$:

$$T_o(\mathbf{x}) = \sum_{j=0}^{N-1} \|(\mathbf{v} * \mathbf{h}_k)(\mathbf{x})\|(\alpha \mathbf{n}_k \mathbf{n}_k^T - \beta I)$$

Then, an eigenvalue decomposition of T_o is computed. When template and local structure in the field are equal up to a rotation, the largest eigenvalue should be one, and the corresponding eigenvector gives the direction where the template has to be rotated. Otherwise, the largest eigenvalue gives the similarity of the two structures, and the other eigenvalues can be non-zero.

As the algorithm is based on convolution, it is robust in terms of noise. Note that the smoothing effect of the convolution grows with the size of the templates. One disadvantage of the algorithm is that it works only for axis-symmetric patterns. This in itself is not a big disadvantage in flow visualization as features are usually abstracted, and the basic patterns are quite simple. However, even for axis-symmetric patterns, the algorithm does not always result in satisfying similarity values. This is due to the more complex nature of vector fields in comparison to scalar fields. A pattern like the one in Figure 5.1 is axis-symmetric, however, rotating it disadvantageously, especially in a direction not covered by the half sphere of the filter directions, yields unsatisfying results. For this example, the rotation invariant similarity computed using the orientation tensor was 43%. Note that the similarities can be much smaller than that, e.g. a similar template where the divergent part was doubled and the template than rotated 180° will result in a rotation invariant similarity of 0.2%, which is no significant similarity at all. The error is largest in the hemisphere not covered by the filter directions, so the best way to redeem this is to compute the algorithm a second time for the filter directions in the other hemisphere.

Another solution, and an acceleration when matching with a lot of different templates, is to break down the templates into simpler patterns, match these and combine the resulting similarities. The pattern of Figure 5.1, e.g. was computed by adding a constant flow to a pure divergence from a line (Figure 5.2). However, the directions with maximal similarity have to be taken into account when combining the single similarities. The convolution is linear, so the the results of the single convolutions can be combined quite well. But the orientation tensor, and thus also the similarity value computed via the orientation tensor, is not linear (Theorem 3.6.8). Thus the rotation invariant similarities can not be combined quite as simple. Because of the square of the filter responses in the definition of the orientation tensor, this algorithm does not distinguish patterns which are gained

1. Normalize the vector field
2. Choose the template to be matched
3. Limit the size of the patterns in the template with a rotational symmetric weighting function
4. Scale the template until its energy is one
5. Rotate the template into each of the needed directions
6. For each grid node of the field:
   (a) Compute vector correlation with all templates
   (b) Compute the orientation tensor with the similarity values
   (c) Compute eigenvalue decomposition of the orientation tensor
   (d) The largest eigenvalue gives the rotation invariant similarity

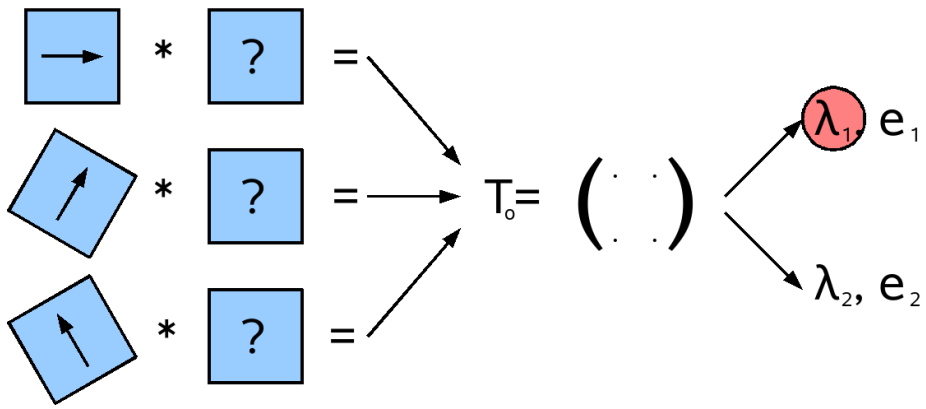

Table 5.1: Outline of the rotation invariant matching algorithm using the orientation tensor as proposed by Heiberg et al. [41, 42], and schematic of the 2D case.

by multiplying every vector in the template with -1. Thus, diverging and converging patterns can not be distinguished, as well as pure left-handed and right-handed rotations. Examples of these patterns can be found in Section 6.1.

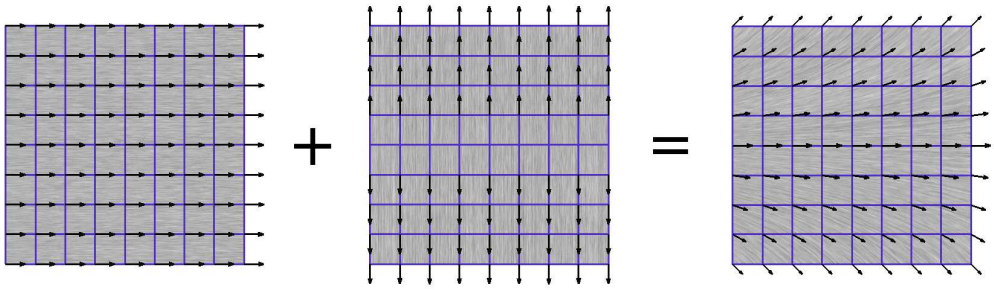

Figure 5.2: Computing a pattern by adding a constant flow and a divergence from a line.

5.4 Clifford Convolution

As Clifford algebra provides a unifying notation for the multiplication of vectors and scalars, as well as valuable geometric properties of the result of the multiplication of two vectors, a convolution of vector fields based on Clifford algebra was defined [23, 24].

Definition 5.4.1 *Let $\boldsymbol{F}$ be a multivector-valued field and $\boldsymbol{H}$ a multivector-valued filter. Then* **Clifford convolution** *based on the geometric product is defined as*

$$(\boldsymbol{H} *_l \boldsymbol{F})(\boldsymbol{x}) = \int_{\mathbb{R}^n} \boldsymbol{H}(\boldsymbol{x'})\boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})|d\boldsymbol{x'}|$$

and analog

$$(\boldsymbol{F} *_r \boldsymbol{H})(\boldsymbol{x}) = \int_{\mathbb{R}^n} \boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})\boldsymbol{H}(\boldsymbol{x'})|d\boldsymbol{x'}|$$

Since Clifford multiplication is not commutative, application of the filter from the left and from the right is distinguished. Clifford convolution is an extension of the convolution of scalar fields, and of the convolution of a scalar with a vector. However, it is also an extension of the vector convolution defined by Heiberg et al. [41, 42] as

$$(\mathbf{h} *_v \mathbf{f})(\mathbf{x}) = \int_{\mathbb{R}^n} \langle \mathbf{h}(\mathbf{x'}), \mathbf{f}(\mathbf{x}-\mathbf{x'})\rangle d\mathbf{x'}$$

and thus

$$(\mathbf{h} *_s \mathbf{f})(\mathbf{x}) = \langle (\mathbf{h} *_l \mathbf{f})\rangle_0 = \langle (\mathbf{f} *_r \mathbf{h})\rangle_0$$

for vector fields $\mathbf{h}, \mathbf{f}$.

The spatial Clifford correlation is defined analogous to the Clifford convolution. It is a Clifford convolution with a filter whose positions have been reflected about its center (Theorem 3.2.7). Another idea for an operator based on the convolution could be to mirror not only the positions of the filters but the multivectors as

well. This needs special treatment for the different grades of the multivector since the scalar part stays the same and the vector part has to be negated. Furthermore, the relation to the convolution becomes more difficult, and it is not motivated from the well-established signal processing theory. Thus, this approach was not pursued further.

The result of the vector convolution based on the inner product of two vectors is valuable as a directed similarity value, therefore it is also used within the Clifford algebra framework. However, the Clifford convolution provides more information about the geometric relation of template and vector field. Remember that in Clifford algebra, the geometric product of two vectors $\mathbf{a},\mathbf{b}$, with angle ω between $\mathbf{a}$ and $\mathbf{b}$, results in the spinor

$$\begin{aligned}\mathbf{ab} &= \langle \mathbf{a},\mathbf{b}\rangle + \mathbf{a}\wedge\mathbf{b} \\ &= \langle \mathbf{a},\mathbf{b}\rangle + \boldsymbol{i}(\mathbf{a}\times\mathbf{b}).\end{aligned}$$

(Lemma 4.1.11 and 4.2.8) and, with respect to the dimension, the following properties are true:

1. (2D)
$$\begin{aligned}\langle \mathbf{ab}\rangle_0 &= \langle \mathbf{a},\mathbf{b}\rangle = \|\mathbf{a}\|\,\|\mathbf{b}\|\cos\omega \\ \text{and}\quad \langle \mathbf{ab}\rangle_2 &= \mathbf{a}\wedge\mathbf{b} = \|\mathbf{a}\|\,\|\mathbf{b}\|\sin\omega\end{aligned}$$

2. (3D)
$$\begin{aligned}\langle \mathbf{ab}\rangle_0 &= \langle \mathbf{a},\mathbf{b}\rangle = \|\mathbf{a}\|\,\|\mathbf{b}\|\cos\omega \\ \text{and}\quad \|\langle \mathbf{ab}\rangle_2\| &= \|\mathbf{a}\wedge\mathbf{b}\| = \|\mathbf{a}\|\,\|\mathbf{b}\|\sin\omega,\end{aligned}$$

Clifford multiplication can be regarded as a correlation of a point in the vector field with a 1×1 template. Thus, Clifford correlation with larger templates is an averaging of the geometric relations of the single vectors. The direction of a structure in the field at position $\mathbf{x}$ can thus be computed out of correlation with a suitable template when the direction of the template is known:

Theorem 5.4.2 *Let m=2,3. Let $\boldsymbol{f},\boldsymbol{h} : \mathbb{R}^m \subset G_m \to \mathbb{R}$ be two vector fields. Let $\boldsymbol{x}\in\mathbb{R}^m$ denote a position in the vector field. Let ω_x denote the angle between the template $\boldsymbol{h}$ and the structure in the field $\boldsymbol{f}$ at point $\boldsymbol{x}$. Then correlation of the two fields at $\boldsymbol{x}$ results in:*

1. (2D)
$$\begin{aligned}\langle(\boldsymbol{h}\star\boldsymbol{f})(\boldsymbol{x})\rangle_0 &\approx \gamma\cos\omega_{\boldsymbol{x}} \\ \langle(\boldsymbol{h}\star\boldsymbol{f})(\boldsymbol{x})\rangle_2 &\approx \gamma\sin\omega_{\boldsymbol{x}}\end{aligned}$$

2. (3D)

$$\begin{aligned} \langle (\boldsymbol{h} \star \boldsymbol{f})(\boldsymbol{x}) \rangle_0 &\approx \gamma \cos \omega_{\boldsymbol{X}} \\ \| \langle (\boldsymbol{h} \star \boldsymbol{f})(\boldsymbol{x}) \rangle_2 \| &\approx \gamma \sin \omega_{\boldsymbol{X}} \end{aligned}$$

$\langle (\boldsymbol{h} \star \boldsymbol{f})(\boldsymbol{x}) \rangle_2$ *is the normal vector of the plane of* $\omega_{\boldsymbol{X}}$

Note that γ *is given by the magnitudes of the velocities of field and template and can be determined directly.*

Due to annihilation effects (Figure 5.3), Clifford correlation of two equal but rotated patterns can result in a wrong estimation of the rotation angle between the two patterns. However, this effect can be avoided, and the geometric properties of the geometric product are used for rotation invariant template matching (Section 5.5).

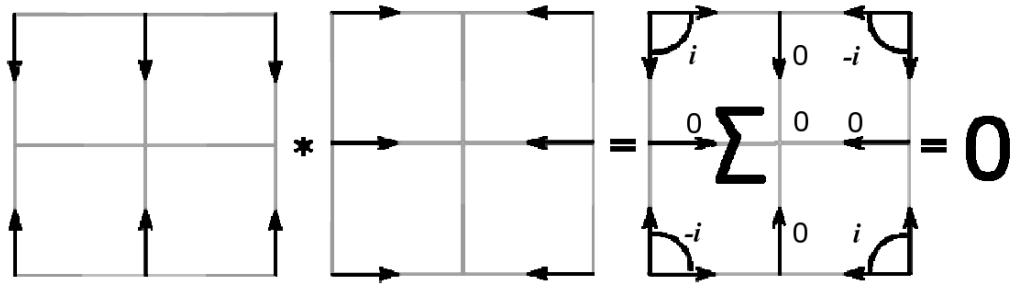

Figure 5.3: Correlation of the template (**left**) and a rotated copy (**middle**) results in a zero multivector as the approximations for the angles annihilate each other (**right**).

5.4.1 Vector Derivation using Convolution

In image processing, it is well known that the derivative operation is a convolution. The vector derivative ∂ as described in Section 4.3 can be discretized using many different approaches. One example are central differences. This is discussed now in relation to convolution and correlation, to show that it corresponds to a convolution with a vector valued template. Discretizing the derivative using central differences yields

$$\partial \mathbf{f} = \sum_{j=1}^{d} \mathbf{e}_j \mathbf{f}_{\mathbf{e}_j}(\mathbf{x}) = \sum_{j=1}^{d} \mathbf{e}_j \frac{\mathbf{f}(\mathbf{x} + s\mathbf{e}_j) - \mathbf{f}(\mathbf{x} - s\mathbf{e}_j)}{2h}.$$

When $\mathbf{f}$ is defined on a uniform 2D grid, the values of the grid nodes are often written as $\mathbf{f}(x) = \mathbf{f}_{m,n}$, and $h = 1$. Thus

$$\partial \mathbf{f} = \frac{\mathbf{e}_1 \mathbf{f}_{m+1,n} - \mathbf{e}_1 \mathbf{f}_{m-1,n} + \mathbf{e}_2 \mathbf{f}_{m,n+1} - \mathbf{e}_2 \mathbf{f}_{m,n-1}}{2h}.$$

Now the templates for the derivative operation using central differences can be computed. They are shown in Fig. 5.4 for convolution in 2D and 3D and correlation in 2D. Curl and divergence can also be extracted out of the results of the computation of the derivative using Clifford convolution (Definition 4.3.9):

$$div\,\mathbf{f} = \langle(\nabla * \mathbf{f})\rangle_0$$

$$curl\,\mathbf{f} = \langle(\nabla * \mathbf{f})\rangle_2$$

Thus, the divergence is the scalar part and the curl the bivector part of the result of the derivative computation. The connection between derivative and divergence becomes clear when looking at the central difference derivative templates as they depict divergence of local flow for correlation and convolution, respectively (Figure 5.4). The connection between curl and derivative or divergence can also be understood when interpreting a unit bivector as a rotation of 90°, and applying this transformation to every vector of a divergence pattern in one plane (Figure 5.5). Note that in 3D, the bivector is vector-valued and gives the rotation axis.

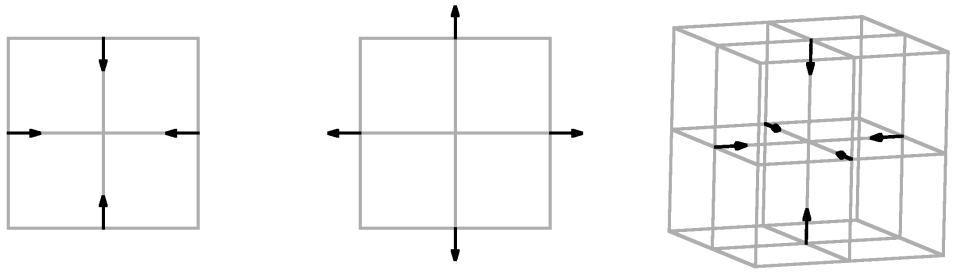

Figure 5.4: Central difference derivative templates. **Left**: Template for 2D convolution. **Middle**: Template for 2D correlation. **Right**: Template for 3D convolution.

5.5 Template Matching with Clifford Convolution

As discussed in Section 5.4, the angle between the directions of the template and the structure in the field can be computed by a Clifford correlation. A basic idea for a rotation invariant matching algorithm would thus be to rotate the template into this direction, and compute one vector correlation for the similarity.

As annihilation effects the approximation of the direction, it is not enough to compute only one Clifford correlation for stable results. Thus, additional templates with different directions have to be used [23, 24]. A possible template distribution, which results in a stable, rotation invariant matching algorithm, is:

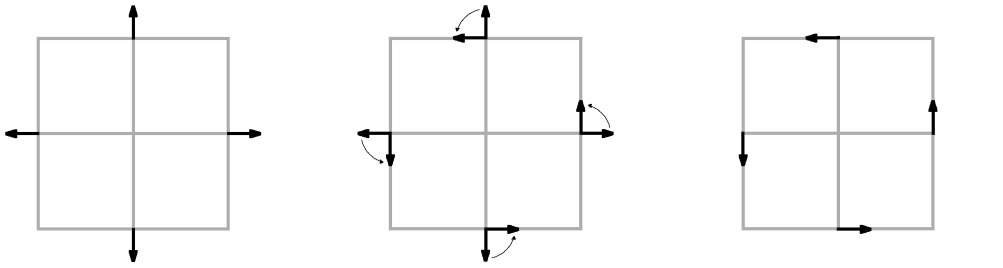

Figure 5.5: Rotating every vector of a 2D divergent pattern about the unit bivector $\mathbf{i}_2$, or 90°, results in a rotational pattern. **Left**: 2D divergent pattern. **Middle**: Rotating every vector. **Right**: 2D rotational pattern.

1. 2D: $a = 0.5$ and $b = \frac{\sqrt{3}}{2}$

   $\mathbf{n}_1 = (1,0)^T$

   $\mathbf{n}_2 = (-a,b)^T$, that is $\mathbf{n}_1$ rotated 120° counterclockwise.

   $\mathbf{n}_2 = (-a,-b)^T$, that is $\mathbf{n}_1$ rotated 240° counterclockwise.

2. 3D: The directions of the principal axes of the coordinate system are used:

   $\mathbf{n}_1 = (1,0,0)^T$, $\mathbf{n}_2 = (-1,0,0)^T$,

   $\mathbf{n}_3 = (0,1,0)^T$, $\mathbf{n}_4 = (0,-1,0)^T$,

   $\mathbf{n}_5 = (0,0,1)^T$, $\mathbf{n}_6 = (0,0,-1)^T$,

The algorithm also works with other directions and other numbers of directions. As usual, one can trade precision for computational speed. The templates are rotated in the desired direction using Clifford algebra and linear interpolation.

For an easier computation of the angles unhindered by the magnitudes of the vectors, both vector fields are normalized. Then, correlation with each rotated template is computed. Now, the approximations $\mathbf{n'}_k(\mathbf{x})$ of the direction of the structure in the field at $\mathbf{x}$ given by the correlations with the different templates are computed. Out of these directions, a single direction $\mathbf{n'}(\mathbf{x})$ has to be determined. In 2D, the template response with the smallest angle to the structure can be used directly. In 3D, the direction is computed analog to the computation of a center of gravity. Only the directions $\mathbf{n'}_k$ calculated out of the template responses $(\mathbf{h} \star \mathbf{f})(\mathbf{x})$ with scalar part $\langle (\mathbf{h} \star \mathbf{f})(\mathbf{x}) \rangle_0 \geq 0$ are used, as they point into the right hemisphere. These directions are weighted with the scalar part of their filter response. The resulting vector is normalized and gives the direction $\mathbf{n'}$ of the structure. For patterns which are rotational symmetric for all rotation directions, this approach

can fail as all filter responses will result in the same similarity value. This has to be considered in the implementation, simple queries concerning the similarity values can detect this case. For these structures, any direction is correct, so the original direction of the template can be chosen.

In a last step, the template is rotated into the computed direction, and another vector correlation is computed as the final similarity value. Note that for this last vector correlation, field and template need not be normalized. This is an advantage in comparison to the approach using the result of the orientation tensor directly. For some applications, it is even necessary that the field is not normalized for the matching. This will be discussed extensively in Section 8.2 and 8.3. A summery of the complete algorithm for rotation invariant matching can be found in Table 5.2.

5.5.1 Acceleration

Besides the obvious acceleration of this algorithm via parallelization, there is another computation that can speed up this algorithm. The rotation of the template at every node of the grid is computational expensive. Therefore, the directions of the template for the final scalar correlation are discretized and all rotated templates are computed only once. The template with the direction closest to the direction of the structure is taken for the scalar correlation at this point. The direction of the template and the γ_{n_j} are computed, and saved with the rotated template. In 2D, the template can be chosen by the angle between the direction and the vector $(1,0)^T$.

In 3D, it is not easy to distribute the directions evenly over the sphere. A subdivision algorithm on the sphere starting with an octahedron gives an approximation. The octahedron is used to support the search for the nearest direction later on. Each triangle of the octahedron is divided into 4 new triangles and the vertices are normalized. With three subdivision steps, one gets 258 directions (Table 5.3). Then, there are 45 directions in each octant. For the computation of the nearest template, the right octant has to be identified. Then the scalar product of the direction of the structure and all directions in the octant is computed. The template with the direction resulting in the biggest scalar product is chosen. As there are only a few points in each octant, a more complex search pattern is not necessary.

Note that the algorithm for rotation invariant matching is computationally still more expensive than the approach with the inner product and the orientation tensor. The amount of correlations is only one more, but the computational cost of the geometric product of two vectors is higher than that of the inner product of the vectors. Furthermore, the computation of the inner product can be sped up using a FFT and the convolution theorem for the vector convolution. For Clifford algebra, a Fourier transform was developed in this thesis and is discussed in Section 7.

1. Convert the vectors from field and template into multivectors
2. Rotate template to get the filter set
3. For each grid node of the field:
   (a) Compute Clifford correlation with all templates and the normalized field
   (b) Compute the direction n' of the structure
   (c) Rotate the template into the direction n'
   (d) Compute a scalar correlation of the rotated template and the field for the similarity value

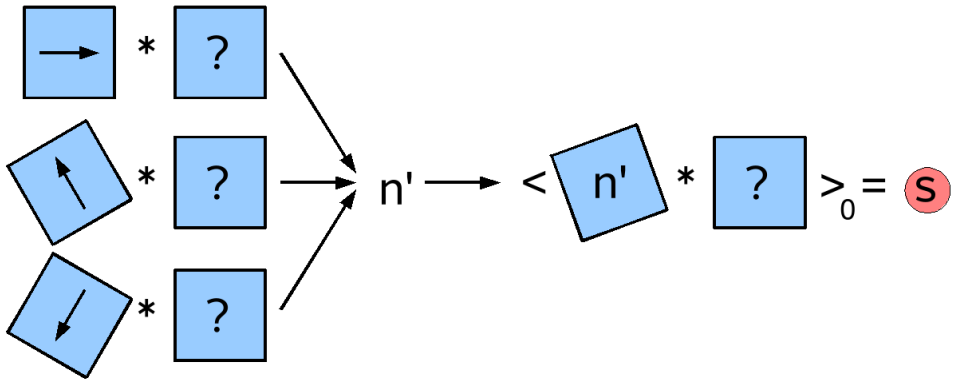

Table 5.2: Outline of the rotational invariant pattern matching using Clifford convolution, and schematic for the 2D case.

step	# points	# triangles
0	6	8
1	18	32
2	66	128
3	258	512

Table 5.3: The number of points, or directions, gained by subdividing an octahedron on a sphere.

Chapter 6

Data Sets and Templates

6.1 Vector-Valued Templates

Many interesting features in flow fields can be described quite intuitively by vector-valued templates. Basic examples of these features are rotation and swirl, converging and diverging patterns, shear flow and saddle points (Figures 6.1 and 6.2). Note that a feature model is often a simplification and abstraction of the real flow. Furthermore, as already mentioned in Section 5.3, complicated patterns can often be modeled as a superposition of simpler features (Figure 5.2). A lot of 3D patterns are just repetitions of 2D patterns along a line. Note that a pure convergence to, or divergence from a point, as well as the rotation pattern in 2D, are rotationally symmetric. This means that the similarity gained by one vector correlation with them is already a rotation invariant measure. Another important issue in flow analysis and visualization is the concept of Galilean invariance, which is the principle stating that the fundamental laws of physics are the same in all inertial frames of reference. That means that a moving observer will see the same as a stationary observer, or one with another velocity. For feature detection, Galilean invariance of a feature definition can be important e.g. when a large flow-through component hides features. This will be investigated further in Section 8.2.

Galilean invariance of the similarity of a template and an arbitrary data set can be determined quite easily:

Theorem 6.1.1 *Let $\boldsymbol{v}[\boldsymbol{x}]$ denote the velocities of a discrete template. Let N be the number of positions $\boldsymbol{x}_j$ in the template. Let the similarity s be obtained by a vector correlation of this template and an arbitrary data set. Then s is Galilean invariant if, and only if, the average velocity*

$$\boldsymbol{v}_a = \frac{1}{N} \sum_{j=0}^{N-1} \boldsymbol{v}[\boldsymbol{x}_j].$$

satisfies $\mathbf{v}_a = 0$.

Note that all templates in Figures 6.1 and 6.2 satisfy $\mathbf{v}_a = 0$ and are thus Galilean invariant.

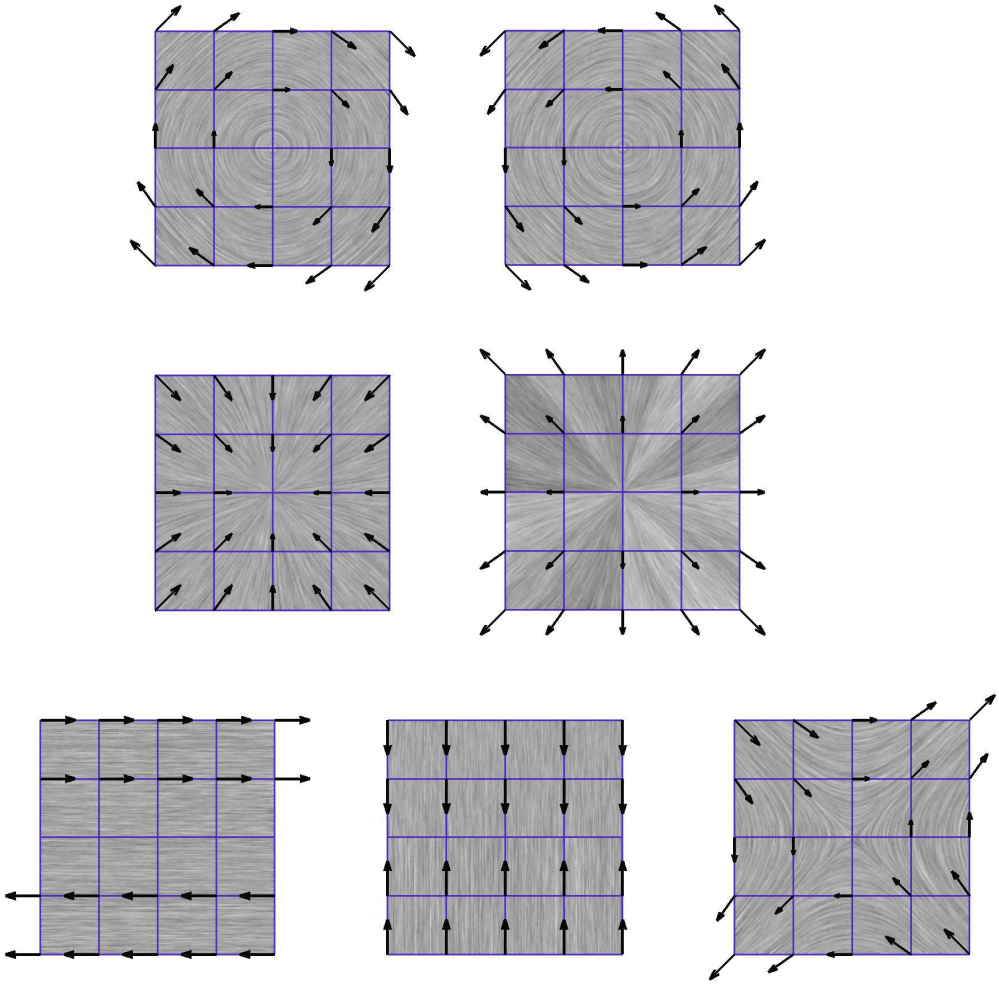

Figure 6.1: A basic 2d template set. Grid (blue), LIC, and hedgehogs (black). **Top**: Clockwise and counter-clockwise rotation. **Middle**: Convergence and divergence. **Bottom left**: Shear flow. **Bottom middle**: Convergence line. **Bottom right**: Saddle. Note that clockwise and counterclockwise patterns can be converted into each other by negating the vectors, as well as divergence and convergence patterns.

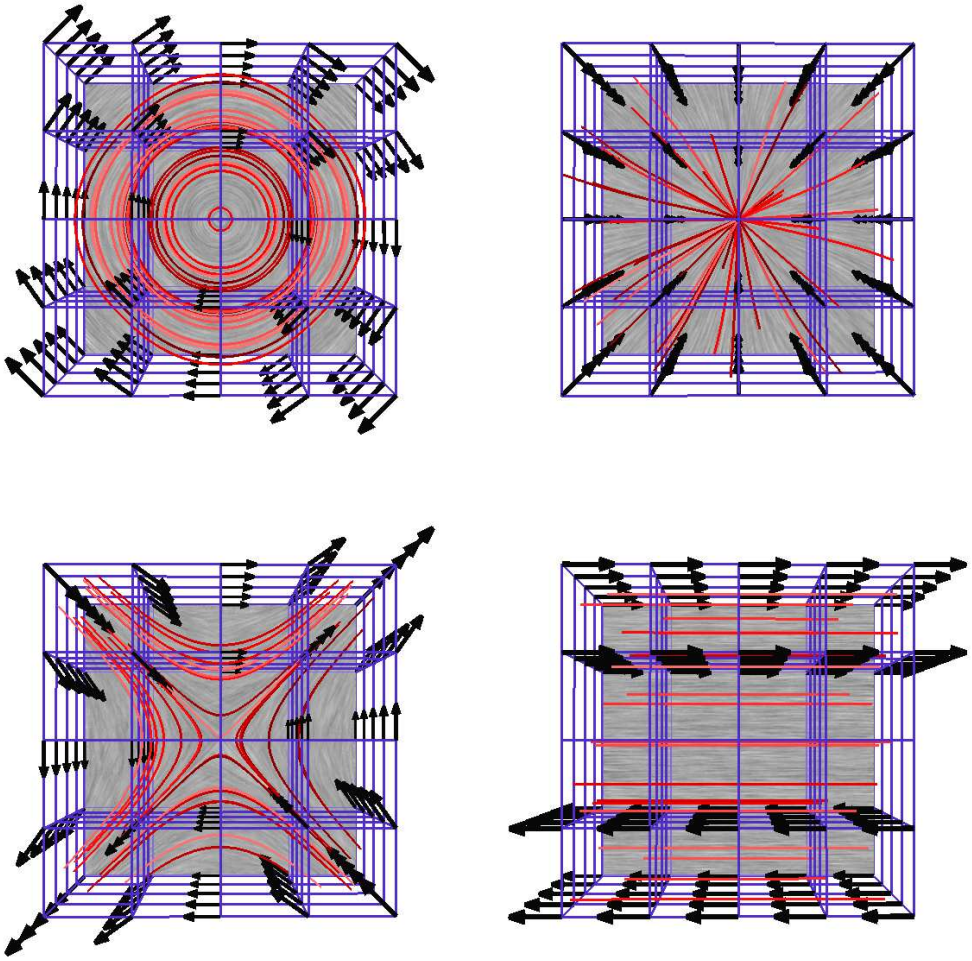

Figure 6.2: A basic 3d template set. Grid (blue), LIC of the rear surface, hedgehogs (black) and some streamlines (red). **Top left**: Rotation. **Top right**: Convergence. **Bottom left**: Saddle. **Bottom right**: Shear flow. Note that counterclockwise and divergence patterns can be found with these templates as well.

6.1.1 Vortex Models

To detect vortices with the use of a correlation, a vector-valued template describing the vortex can be used. A basic approach for a rotational vector-valued template is shown in Figures 6.1 and 6.2. An improved approach is to scale the magnitude of the vectors of the rotational pattern according to some vortex model like Rankine [76], Scully [84] or Vatistas [102]. These vortex models are explained in this section. Note that when matching with a normalized template, the magnitude information coded in these vortex models is lost, and a pure rotational pattern remains (Figure 6.3). Furthermore, note that all of these vortex models are rotationally symmetric and Galilean invariant.

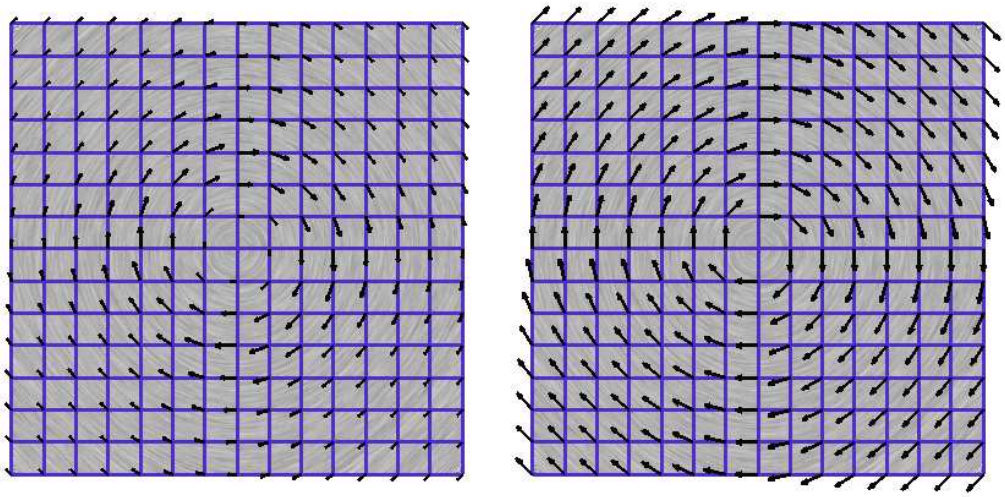

Figure 6.3: **Left**: A Rankine vortex with $r_c = 2$. **Right**: A normalized rotational pattern.

One of the often-used vortex models is the Rankine vortex [76]. It assumes a solid-body rotation within the vortex core and an exponential drop-off outside:

Definition 6.1.2 *Let $\boldsymbol{f}(\boldsymbol{x})$ denote a pure, circular rotation around $\boldsymbol{x}_0$. Let $r = \|\boldsymbol{x} - \boldsymbol{x}_0\|$ denote the radial distance of a position $\boldsymbol{x}$ to the rotation center $\boldsymbol{x}_0$. Let r_c be the vortex core radius and let v_c be the magnitude of the circumferential velocity at r_c. Then the* **Rankine vortex** *[76] is given by the magnitude of the velocity $v(r) = \|\boldsymbol{f}(r)\|$ with*

$$v(r) = \begin{cases} v_c r & r \leq r_c \\ \frac{v_c}{r} & r \geq r_c \end{cases} .$$

Note that the first derivative of the function $v(r)$ is discontinuous in r_c, which is not true for a real vortex. The Rankine vortex is a special case of the Vatistas

vortex [102], where an additional parameter N can be used to control the transition between the solid body rotation and the exponential drop-off (Figure 6.4):

Definition 6.1.3 *Let $\boldsymbol{f}(\boldsymbol{x})$ denote a pure, circular rotation around $\boldsymbol{x}_0$. Let $r = \|\boldsymbol{x}-\boldsymbol{x}_0\|$ denote the radial distance of a position $\boldsymbol{x}$ to the rotation center $\boldsymbol{x}_0$. Let r_c be the vortex core radius and let v_c be the magnitude of the circumferential velocity at r_c. Let $N \in \mathbb{R}$. Then the* **Vatistas vortex** *[102] is given by the magnitude of the velocity $v(r) = \|\boldsymbol{f}(r)\|$ with*

$$v(r) = 2^{1/N} r_c^2 \frac{v_c r}{(r_c^{2N} + r^{2N})^{1/N}}.$$

Note that $2^{1/N} r_c^2$ is a scaling factor to obtain $v(r_c) = v_c$. For $N = 1$, the Vatistas vortex equals another vortex model called the Scully vortex [84], and for $N = \infty$, the Rankine model [76] is obtained.

For an arbitrary vortex within a data set, the engineers want to determine certain parameters of these vortex models to quantify the flow. Important parameters are not only the vortex core center $\mathbf{x}_0$, the vortex core radius r_c, the circumferential velocity at the vortex core radius v_c, but also the circumferential velocity distribution, the overall vorticity within the vortex core, and, in 3D, the maximal axial velocity and the axial velocity distribution. Some of these parameters can be coded directly into the templates, and thus be determined using template matching as will be shown in Section 8.3.

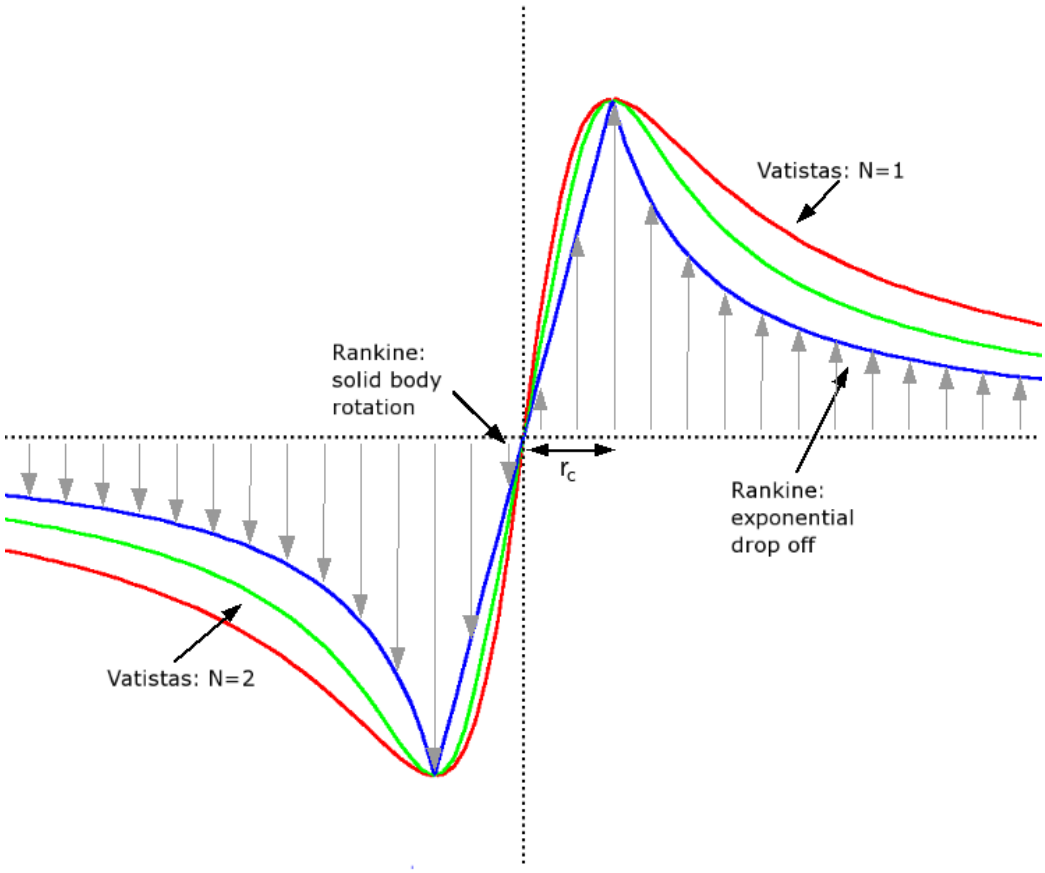

Figure 6.4: A cut through a vortex. Circumferential velocity distribution (grey arrows) as given by the Vatistas vortex model for $N = 1$ (red), $N = 2$ (green), and by the Rankine vortex as a special case of the Vatistas vortex with $N = \infty$ (blue).

6.2 Swirling Jets Entering Fluid at Rest

In this section, several CFD simulations describing a vortex breakdown are introduced. Vortex breakdown can be found in flows ranging from tornados, wing tip vortices, pipe flows to swirling jets. Here, the turbulent swirling jets each enter a fluid at rest. The simulation considers a cylinder, and a planar cut along the axis of the cylinder is used as a domain. The domain is discretized by a 124×101 respectively a 251×159 rectilinear grid with smaller rectangles towards the axis of the cylinder for the OM06 and OM08 data sets (Figure 6.5 and6.6). Since a lot of small and large scale vortices are present in the flow, a discrete numerical simulation (DNS) using a higher order finite difference scheme is used to solve the incompressible Navier-Stokes equations.

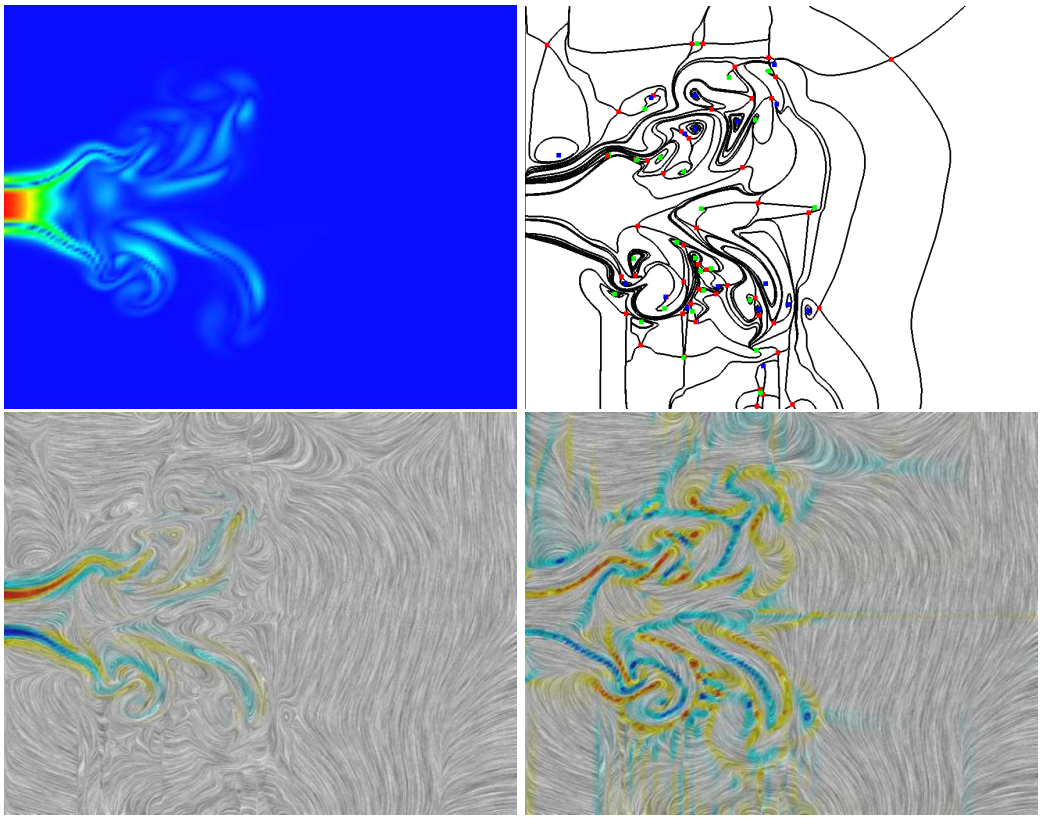

Figure 6.5: **OM06**: A swirling jet entering a fluid at rest. **Top left**: Velocity magnitude from zero (blue) to 7.4 (red). **Top right**: Topology. **Bottom**: LIC and color coding of vorticity from high negative values (clockwise rotation, blue) to high positive values (counter-clockwise rotation, red). **Left**: The velocity of the vectors is high when the swirling jet enters the fluid at rest. **Right**: Normalizing the data set and then computing vorticity reveals more of the flow structures.

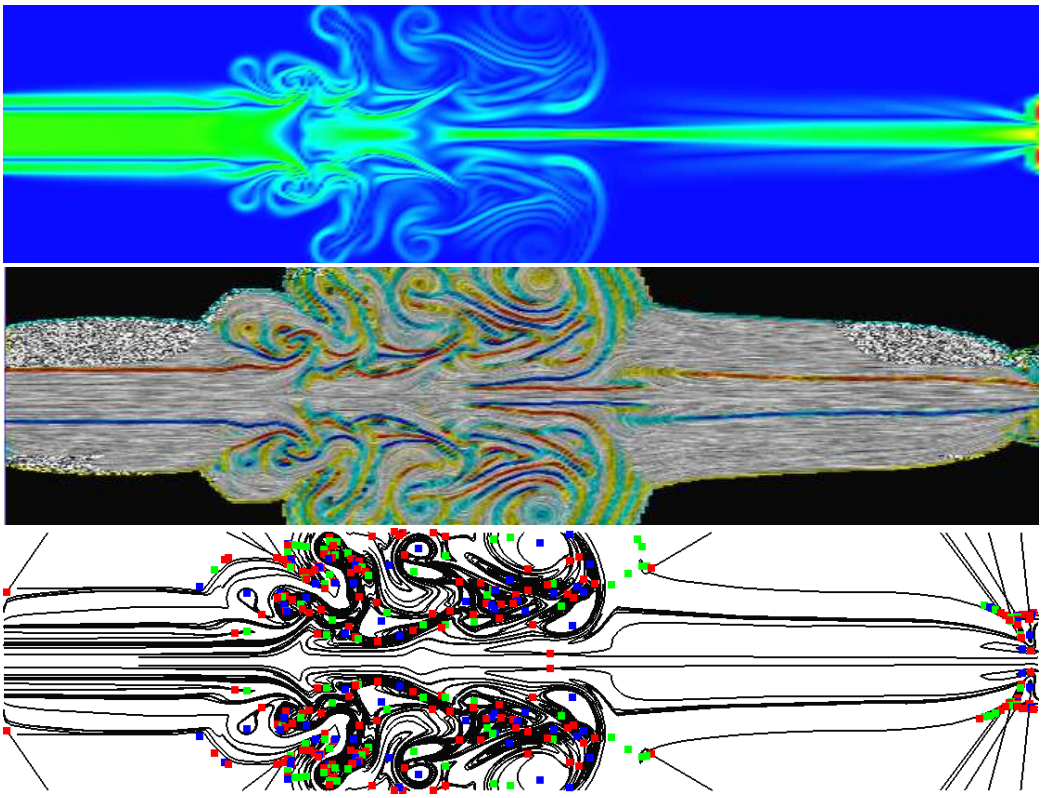

Figure 6.6: **OM08**: A swirling jet entering a fluid at rest. **Top**: Color-coding of the velocity using a logarithmic scale from zero (blue) to 25,7 (red). **Middle**: LIC and color coding of vorticity from high negative values (clockwise rotation, blue) to high positive values (counter-clockwise rotation, red). The black areas of the LIC display areas where the velocity is zero. Again the data set was normalized to reveal more of the flow structures. **Bottom**: Topology of the data set.

For computational issues, resampling of the OM06 data set to a uniform 128×128 rectilinear grid was done. This data set is referred to as OM06(2^k). Note that the magnitude of the inflow dominates the data set and a lot of the features present in the data can only be found when using streamline-based approaches like LIC and topology. For these data sets, the shear flow generated by the inflow of the swirling jet is the most prominent feature. The vortex breakdown results in some vortices and many layers of flow with different directions, again divided by shear flow. The resulting structures are quite complex.

Another example of a swirling jet from CFD simulations are 125 timesteps of an unsteady vector field. Again, the simulation considers a cylinder, and a planar cut along the axis of the cylinder is used as a domain resulting in a 141×251 structured grid (Figure 6.7). This time-dependent data set is called VTOP. Again,

the swirling jet enters a fluid at rest. This time, only one half of the symmetry plane is used. In the different timesteps of this flow the generation, detaching, and general path of vortices can be studied.

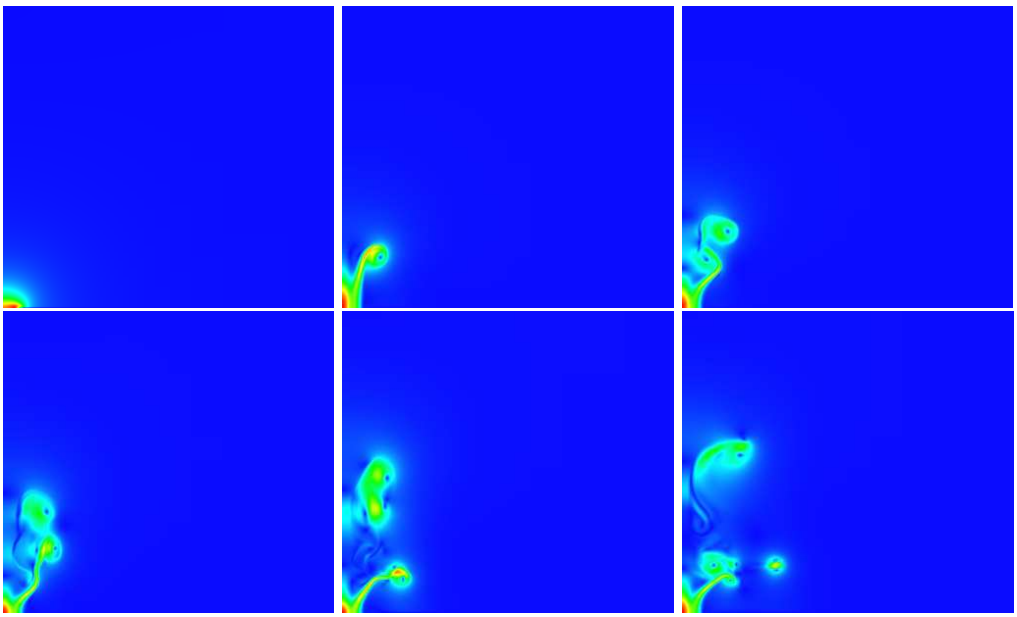

Figure 6.7: **VTOP**: A swirling jet entering a fluid at rest. Here: 6 out of 125 time steps. From top left to bottom right: Timestep 1, 25, 50, 75, 100 and 125. Color-coding of the velocity from zero (blue) to maximal value within the corresponding time step (red). Only the left half of the data set is displayed as the right half has only zero values.

6.3 Gas Furnace Chamber

An interesting flow data set is a gas furnace chamber as it is used for heating a house. The simulation solves compressible Navier-Stokes equations using a turbulent model applied on an irregular grid consisting of 174341 tetrahedra with 32440 vertices. For template matching, the data was also resampled onto a uniform grid with dimensions $126 \times 65 \times 57$. In Figure 6.8, the swirling gas enters the chamber in the center of the left face while the air enters from 9 openings on the top and 9 openings on the bottom, so that the combustion takes place in the center area of the chamber. The products of the combustion leave the chamber on the right. The flow is highly turbulent and exhibits a lot of different scale vortices. This is desirable, as the combustion will be more efficient the longer gas and air mix.

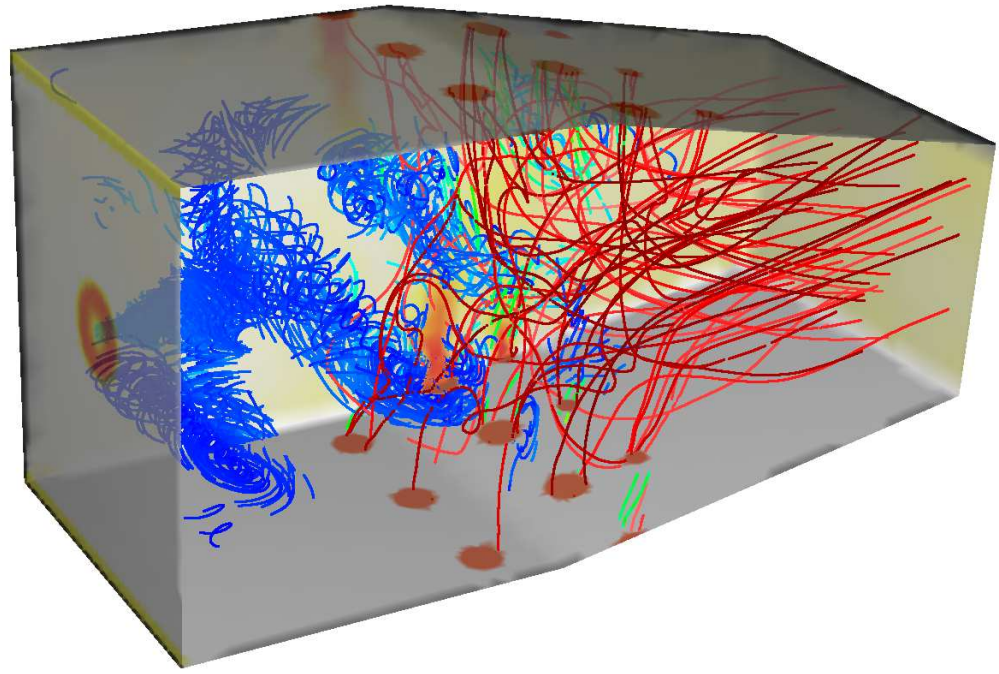

Figure 6.8: A gas furnace chamber. Color-coding of the velocity of the boundary to show the inflow areas. Streamlines (red) started at the top and bottom gas inflow. The gas leaves the chamber in the rear. Streamlines (blue) seeded by the results of matching with a $5 \times 5 \times 5$ rotational template (threshold 0.5) to display vortices in the front half of the chamber.

6.4 Deltawing

The delta wing is a vortex break down study. Ideally, two strong, tubular vortices form above the wing along with secondary and tertiary vortices, each with opposite rotation direction to the former vortex. The two well formed vortices on top of the delta wing are necessary for the generation of the lift, and thus the ability of the airplane to fly. However, in certain circumstances like low velocities and a high angle of attack, the vortices can burst. This drastically reduces the lift which leads to dangerous flight conditions. The pressure, which is exerted on the wing by the vortex burst, can be so high it can lead to structural damage and destroy the wing. Therefore these vortices and their breakdown are studied (Figures 6.9 and 6.10). This data set is a steady simulation with an angle of attack of 25 degrees. The grid consists of 1.8 million unstructured points forming 6.3 million cells. For demonstration purposes and acceleration of the computation of the Clifford convolution, resampling of the area around the wing to a uniform grid of dimensions $66 \times 47 \times 24$ was done (Figure 6.10, top left). The new grid has the lower left corner at (0,-0.25,-0.1) and the upper right corner at (0.7, 0.25, 0.15).

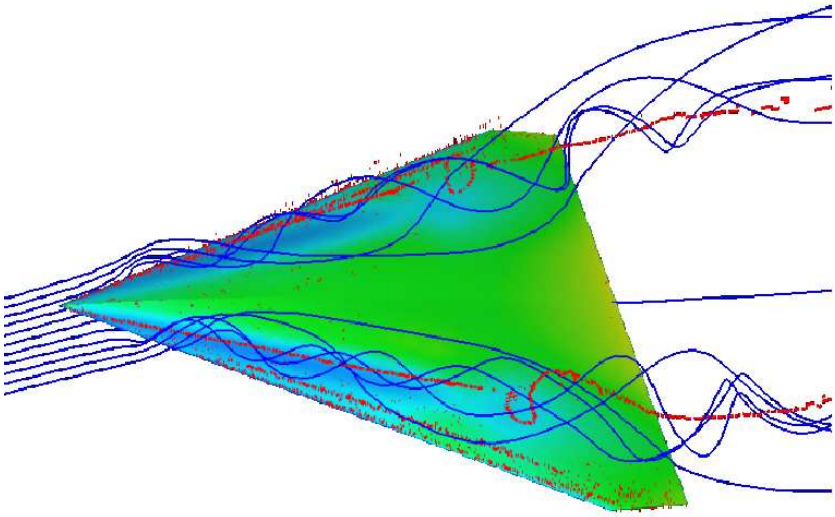

Figure 6.9: The 3D delta wing data set. This is a study of vortex break down. The pressure on the surface is color coded. The results of the algorithm of Sujudi-Haimes basically depict two vortices, though secondary and tertiary vortices can be seen as well. Some streamlines are drawn to enhance the understanding of the flow.

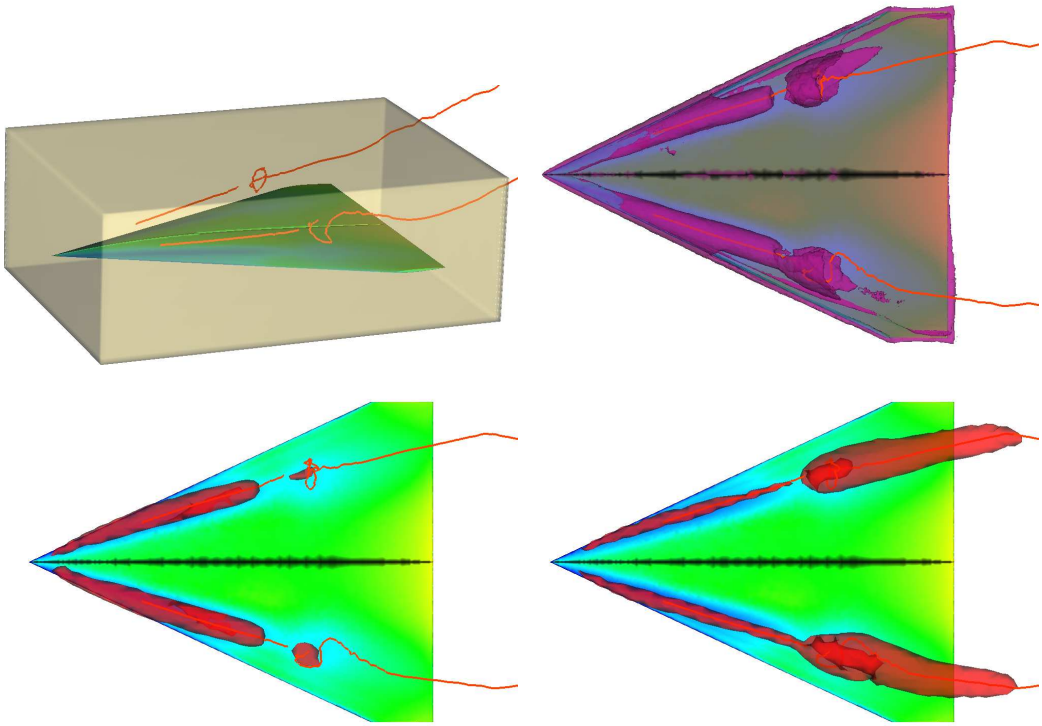

Figure 6.10: The 3D delta wing data set. The pressure on the surface is color coded. The results of the algorithm of Sujudi-Haimes were filtered by segment length and depict the two major vortices. **Top left**: The boundary of the resampled grid. Note that the wing itself vanishes in the resampled data set. As the vortical structures above the wing are of interest, this is no disadvantage here. **Top right**: Isosurface (value: 5000) of the vorticity of original data set. Thus, the shear flow at the boundary of the wing is displayed as well. **Bottom left**: Isosurface (value: 30) of the similarity of the resampled data set to a $3 \times 3 \times 3$ rotational template. **Bottom right**: The data set was normalized before the template matching, enhancing the visualization of the vortical structures in the rear of the wing. Isosurface (value: 0.4) of the similarity.

The shear stress vector field defined on the wing is studied as well. Therefore the surface of the delta wing was extracted, and flattened to get a planar 2D data set (Figure 6.11). This flattened delta wing is defined on an irregular grid with 25800 grid points and 49898 cells. In this data set, the separation and attachment lines are of interest (Figure 6.4) [57, 95]. They are a result of the vortex systems above the wing, as these press the flow onto the surface and lift it up again.

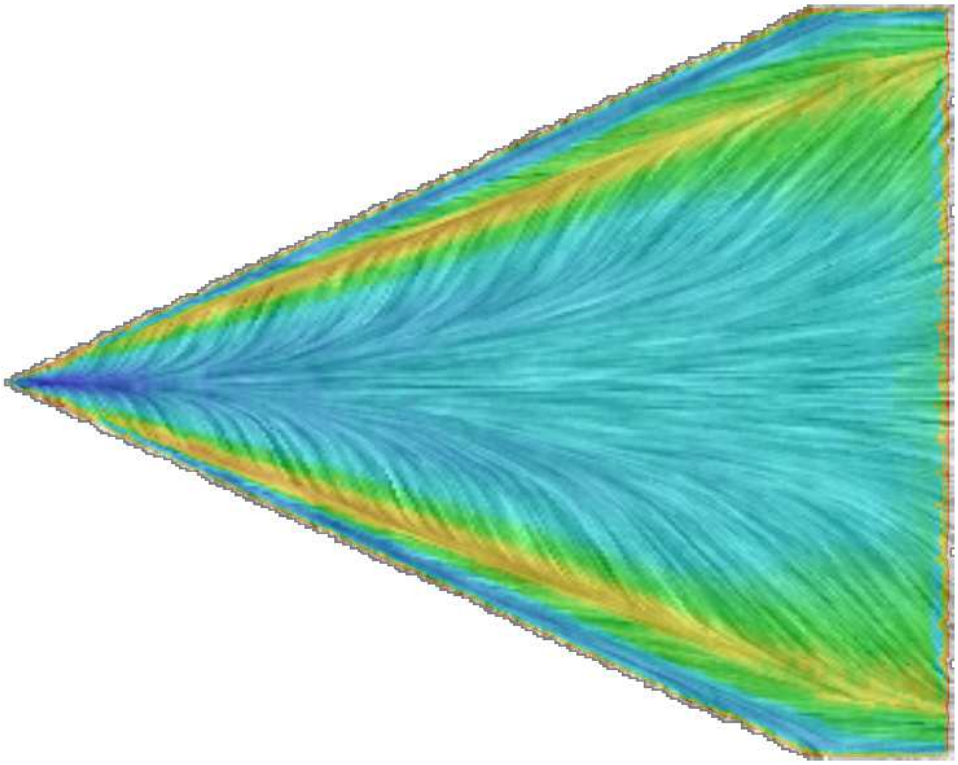

Figure 6.11: The flattened surface of the delta wing with the wall shear stress vectors. LIC and results of pattern matching with a 5×5 divergence template. Adaptive color coding of the results. Red corresponds to high similarity and convergence, and blue to high negative similarity values and divergence. The separation and attachment behavior of the flow is clearly depicted

6.5 ICE

This data set describes the flow around an ICE train (Figure 6.12) moving with a speed of 250 km/h. The wind comes directly from one side but due to the speed of the train, the angle of attack is 15 degrees. The wind hits the ICE train front and left, and then rolls up on the right side of the train forming several vortices. These vortices are closely connected to the air flow attaching to and separating from the surface of the train.

A section plane through three of these vortices with dimensions 51×51 was computed (Figure 6.13), and used as an example of a simple flow with only a few features. The front wagon of the ICE train was extracted and analyzed separately (Figure 6.14). It consists of 26532 positions and 52758 triangular cells. It is used to demonstrate template matching on arbitrary surfaces (Section 8.1.2).

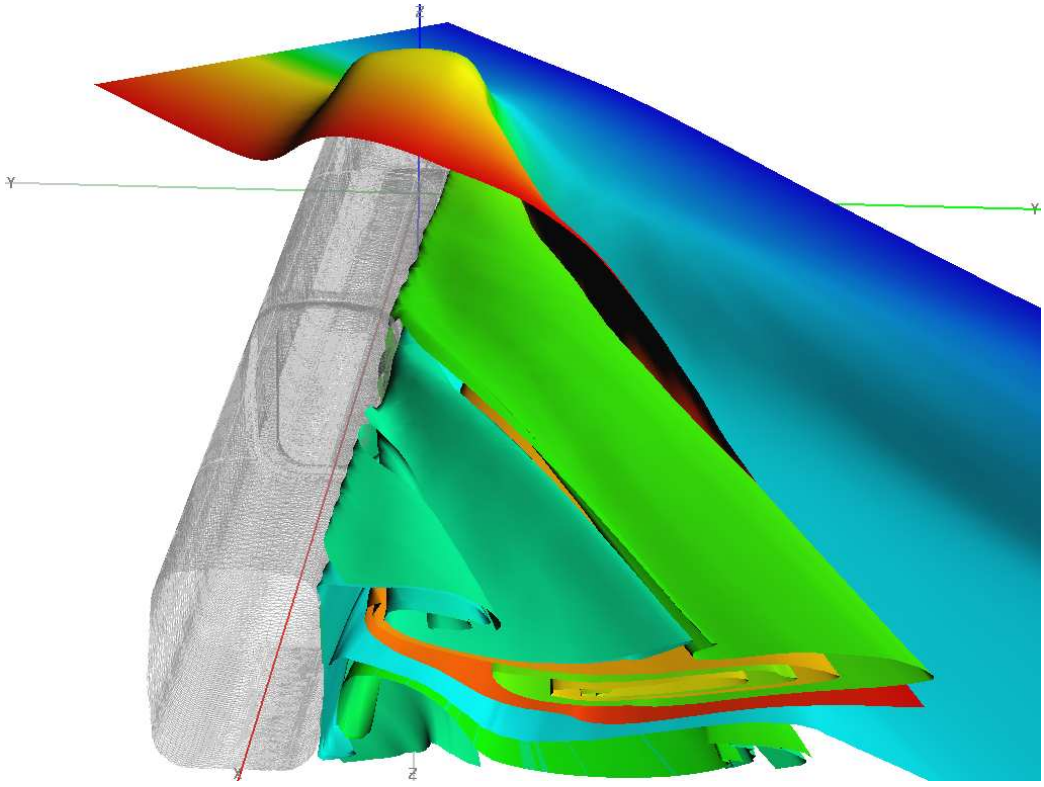

Figure 6.12: Vortices generated by an ICE train. A streamsurface started in front of the train clearly depicts the roll-up of vortices besides the train. At the border of the train, abrupt separation of the flow can be seen. Figure courtesy of Gerik Scheuermann.

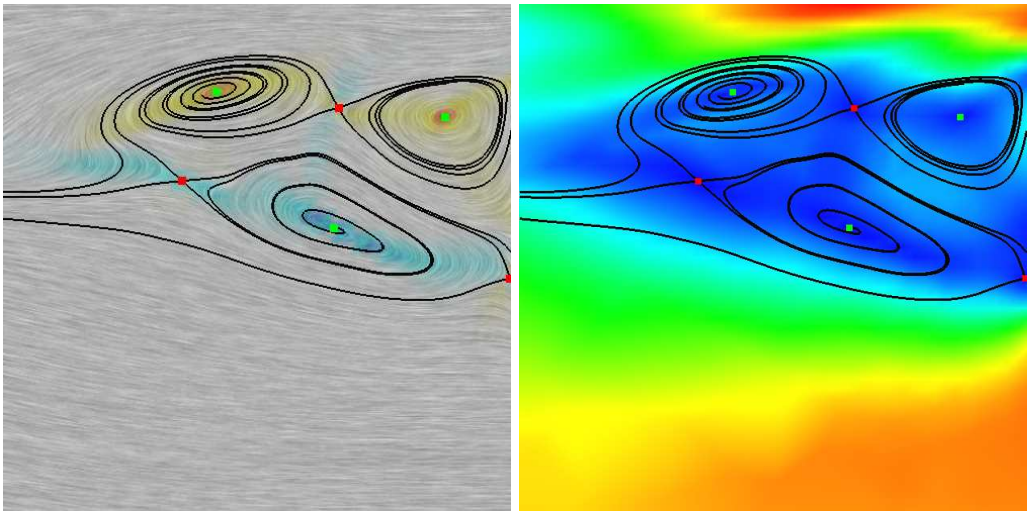

Figure 6.13: A section plane through the vortices alongside the train. **Left**: LIC, topology, and color coding of vorticity of the normalized data set. **Right**: Color-coding of the magnitude of the velocity from zero (blue) to high values (red), and topology. Note that the vortices are within an area of low velocity magnitudes.

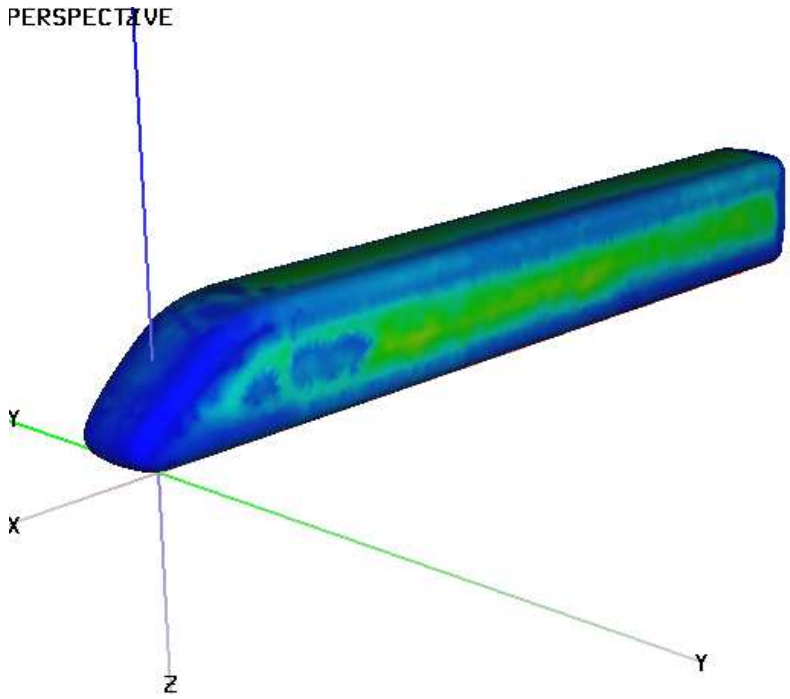

Figure 6.14: The front wagon of the ICE train. Color-coding of the magnitude of the velocity at the surface from zero (dark blue) to maximum (dark red).

6.6 HART II

In 2001, a major international cooperative research program was conducted to investigate the physics of blade pressure, noise radiation, and vibrations caused by the wake of helicopter rotors [10, 100]. Concurrently, a comprehensive experimental database for code development and validation has been generated. There are three major sources for blade pressure fluctuations, noise and vibrations – the superposition of flight speed and blade rotation, the aerodynamic interference between the rotor and the main body of the helicopter, and the wake vortices of the rotor hitting other blades. This research program concentrates on the latter phenomena. It is called HART II for Higher-harmonic-control (HHC) Aeroacoustics Rotor Test II and it is a follow up on the HART program of 1994. (HHC describes the process of influencing the blade pitch angle, 3,4 or 5 times per revolution, to reduce noise and vibration.) The German DLR, the French ONERA, the Netherlands DNW, the US Army Aeroflightdynamics (AFDD) and NASA Langley all take part in the cooperation.

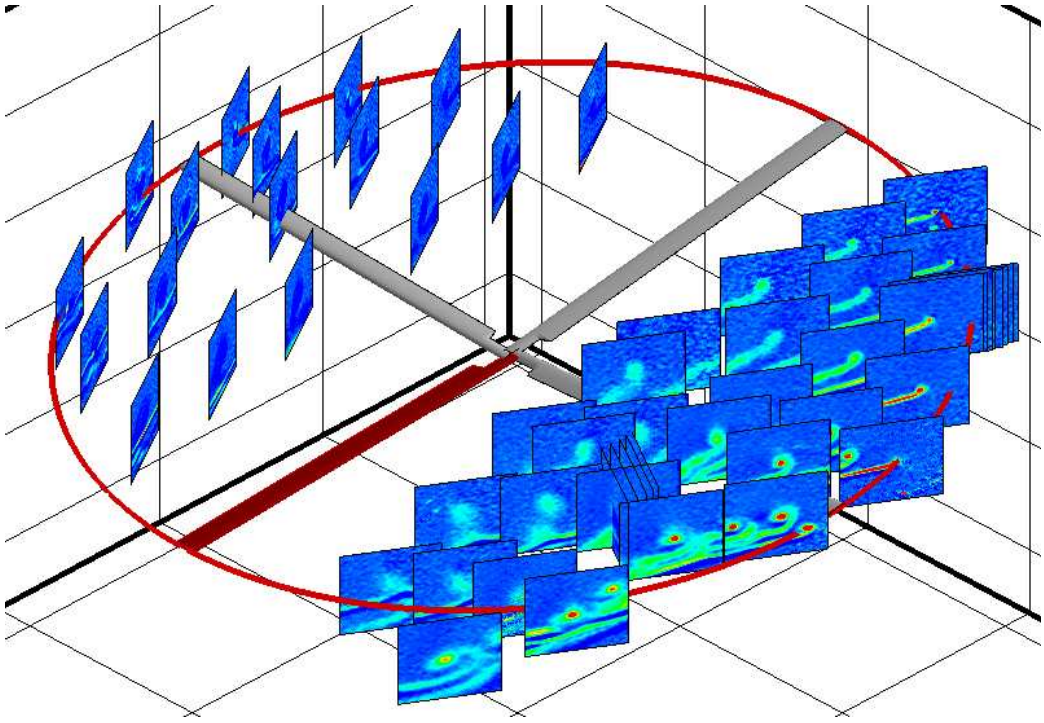

Figure 6.15: PIV measurement positions in the wake of a helicopter rotor. The red blade is at the rear position, and the wind comes from right. Figure courtesy of DLR Braunschweig.

6.6.1 The Test

The HART II test was conducted in the open-jet, anechoic test section of the Large Low-speed Facility (LLF) of the DNW. The set-up for the PIV measurements is shown in Figure 6.16. The rotor is a 40-percent, dynamically and Mach-scaled model of the Bo105 main rotor and operated counter clockwise when looking from above. The model is 4m in diameter and has four hingeless blades with a precone of 2.5° at the hub. For the HART II test, the rotor was operated at a nominal rpm of 1041, thrust coefficient C_T=0.0044, hover tip speed of 218m/s, and an advance ratio of 0.15, for a range of rotor angles and conditions with and without HHC. More detailed information can be found in the literature [10, 100].

6.6.2 3-C PIV Measurements

The rotor wake was measured on both the advancing and retreating sides of the rotor using 3-component particle image velocimetry (3C-PIV) [43, 75]. The measurement locations (cut planes) are shown in Figure 6.15. Two rotor azimuthal orientations were used in order to keep the blade from interfering with the measurements. There were approximately 50 locations on each side of the advancing and retreating sides for the baseline and two HHC conditions. For some locations on the advancing side, PIV measurements were made for six different shaft angles to get more precise information about various flight conditions from steep climb to steep descent. At every PIV measurement location, 100 instantaneous vector fields were obtained, not to get time-dependent behavior, but to average the results to get a statistically based mean behavior at the positions. The time-dependent behavior can be studied by tracing the vortices through the different measurement positions as these are placed along the path of the vortices.

The PIV setup for the HART II test consisted of five digital cameras and three double pulse Nd:YAG lasers. The lasers and cameras were mounted on a common traversing system in order to keep the distance between the cameras and the light sheet generated by the lasers constant, even when moving to different measurement locations (Figure 6.16). Thus, measurements could be continued without recalibration. The five cameras were located on the tower and the lasers underneath the rotor. To obtain measurements on the retreating side, the entire support structure and tower was repositioned.

The three laser systems generated a light sheet of 1.5m, 7mm thick, with an orientation of 30.6° with respect to the wind tunnel axis. The cameras and lasers were synchronized with a one-per-rev signal given by the rotor, allowing for recording at desired phase-angles of the rotor blade. One camera was used for visual checking of the seeding of the particles in the flow prior to the PIV data acquisition to guide the other cameras to the vortex center. The other four cam-

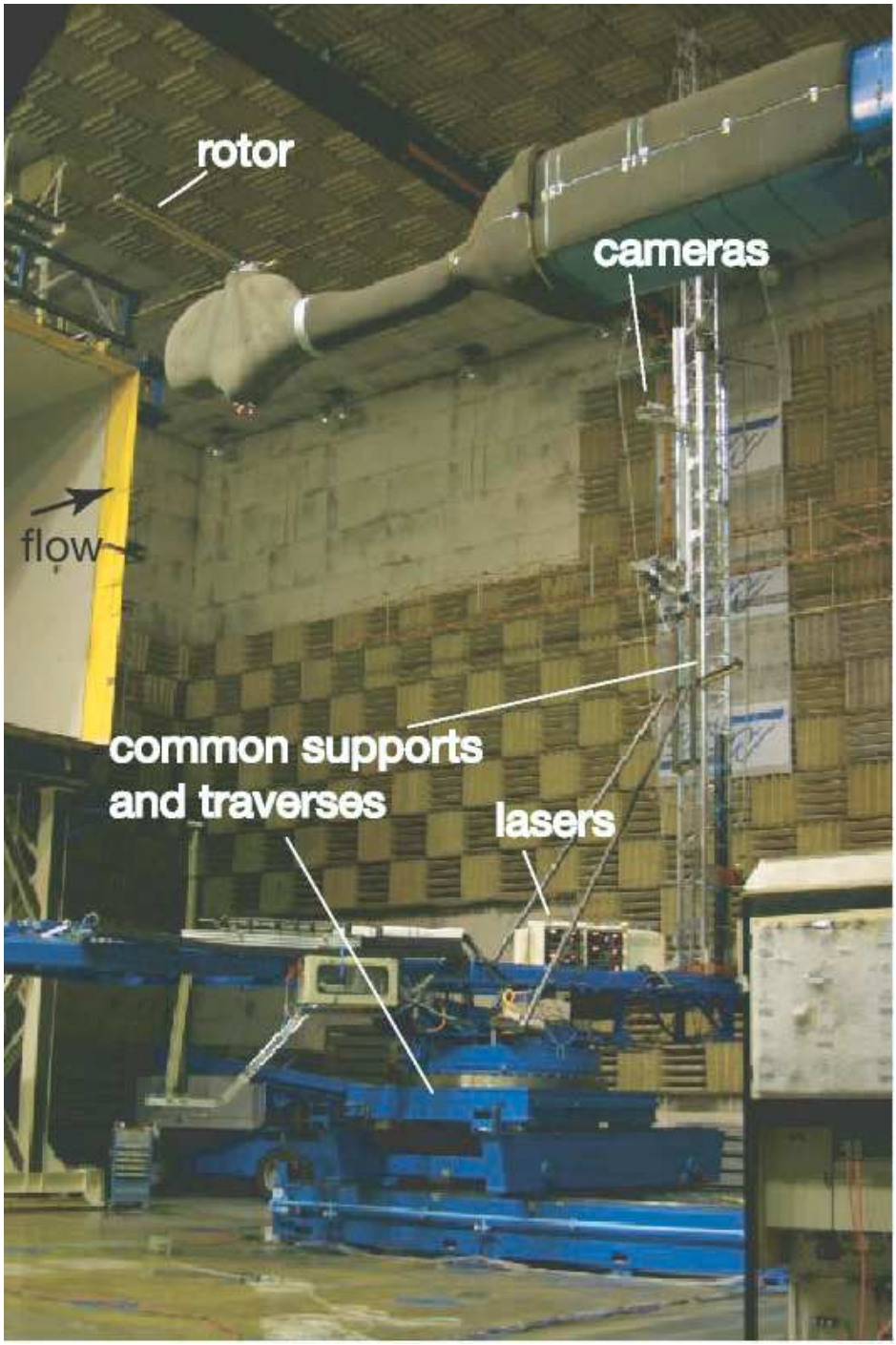

Figure 6.16: HART II measurement configuration. Figure courtesy of DLR Braunschweig.

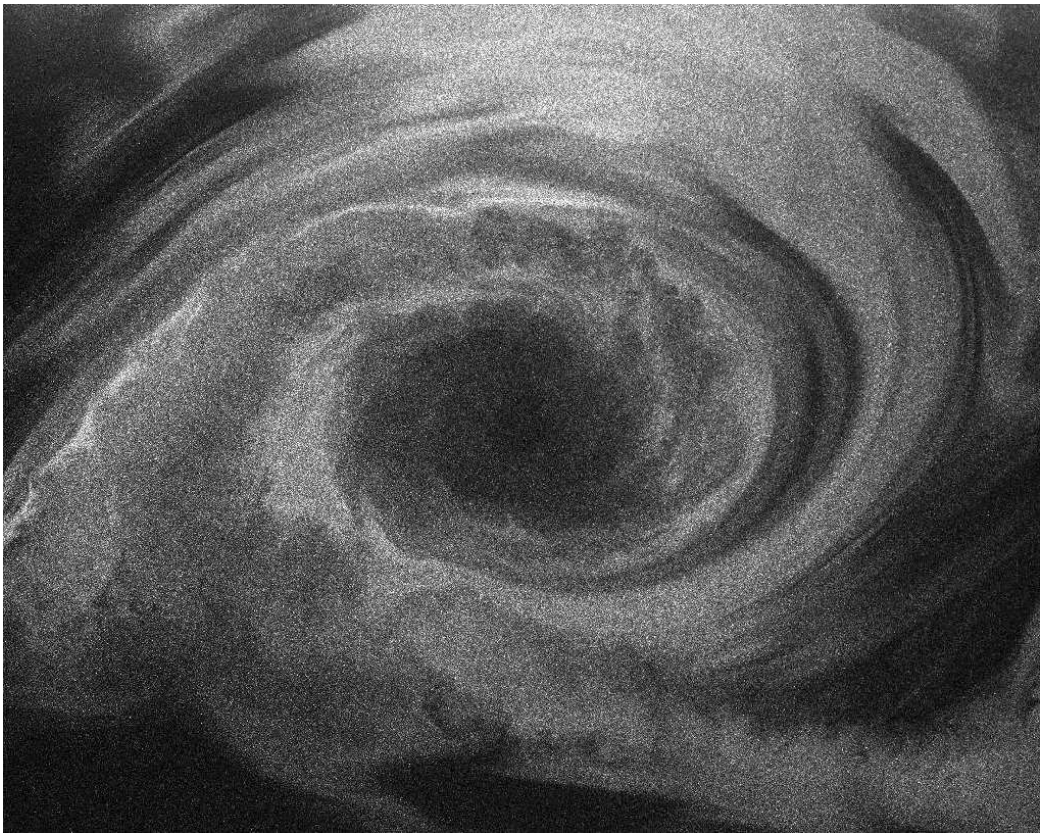

Figure 6.17: A raw PIV image. Out of a stereo pair of these images, 3-C PIV vector fields are computed. Figure courtesy of DLR Braunschweig.

eras were used for two simultaneous PIV measurements of the overall flow and a small, higher resolution image focused on the vortex core region. Each camera had a resolution of 1024 by 1280 pixel, digitized to 12 bit. One camera from each system was located above and below the rotor plane, respectively. The difference in spatial resolution was obtained by using different lenses.

Flow seeding was accomplished by a specially designed seeding rake located in the settling chamber. The rake was 3m by 4 m and was connected to Laskin nozzle particle generators. Di-2-Ethylhexyl-Sebacat (DEHS) was used as seed material. The mean diameter of the particles generated was below 1 μm. More detailed information can be found in the literature [10, 11, 75, 100].

6.6.3 Previous processing of the HART II PIV data

3C-PIV measurements have been applied before, and not only to the HART II data. Some measurement of the wake of a hovering helicopter as well as first

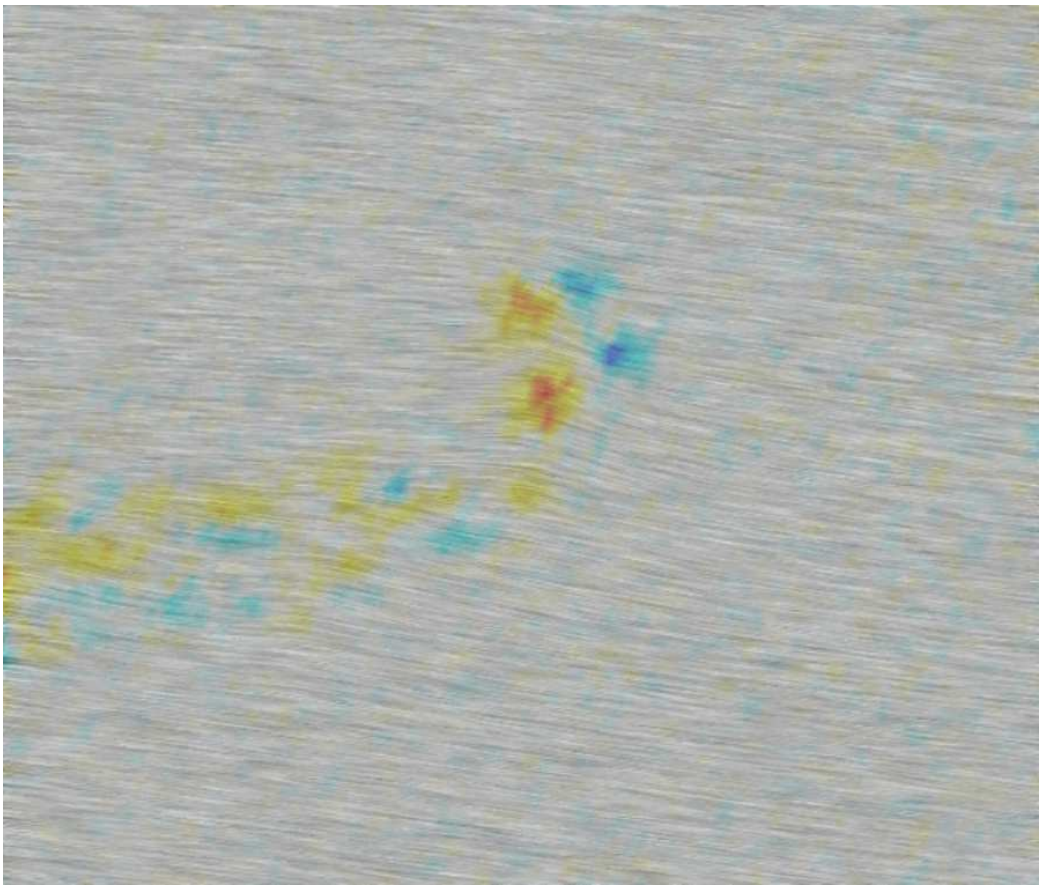

Figure 6.18: One vector field out of the HART II measurements. Displayed are LIC and color coding of vorticity from high negative values (clockwise rotation, blue) to high positive values (counter-clockwise rotation, red).

attempts of processing the vector fields has been done, for example, by Heineck et al. [43].

Note that 3C-PIV measurements result in three-component vector fields of the flow measured in image planes (Figure 6.15). There are immediately two challenges concerning these data sets. First of all, the velocity of the particles in the major flow direction is often, though not always, much bigger than the vortex components and thus hides the vortices. Furthermore, the data has been measured and therefore contains measurement error besides the natural turbulence of this flow. An analysis of the size of the errors of the PIV measurement has been done by Raffel et al. [75]. Most of the time, the vortices are not orthogonal to the measurement plane. Methods for determining, for example, the size of the vortices will therefore give biased results. It is necessary to determine the direction of the

vortex and to correct the data by projecting the (three-component) vectors onto a plane orthogonal to the vortex. There is often more than one vortex in the data and the vortices and the wake sheet can influence each other, resulting again in a non-optimal vortex shape in the image plane. Every vortex has to be found and corrected separately by removing the influence of other vortices, which is a recursive problem.

The rotor wake contains vortices in all creation, aging and destruction phases, the destruction being due to bursting or interaction with blades. Regarding direct visualization, the vortices are often hidden by mean flow components (Figure 8.11). Removing the mean flow is not as simple as averaging the vectors and subtracting the result as the vortices influence the average. To be independent of the mean velocity components, the out-of-plane component of vorticity ω_z, that is, the vorticity of the two in-plane components, has been used mainly for vortex detection and classification.

Figure 8.11 shows the vorticity of one of the instantaneous data sets and the effects of measurement errors. In [10], the vortex core is defined as the center of vorticity (CoV):

Definition 6.6.1 *Let $\boldsymbol{p}(i) \in \Omega$ denote the positions within an area Ω, and let $\omega_z(i)$ the vorticity at the position $\boldsymbol{p}_i$. Then the* **center of vorticity** *of Ω is defined as*

$$CoV = \frac{\sum_i \boldsymbol{p}(i) * \omega_z(i)}{\sum_i \boldsymbol{p}(i)}.$$

The sum is either taken over the whole frame or only within a region around the suspected vortex center, which can be found by manual inspection or locally extremal vorticity. The latter approach has the advantage that the CoV is only influenced by one vortex, but the general area of the vortex and the size have to be determined beforehand.

Two methods to determine the size of the vortices have been used so far [10,43, 100]. First, velocity cuts through the vortex can be analyzed, but these are quite ragged due to the noise. Better results are obtained by integrating the vorticity within a disc that is successively enlarged, and plotting the results as a function of the radius. The maximal integration result, divided by the radius, gives the size of the vortex, and the profile, if convex, gives the velocity at the core radius and the parameter N of the Vatistas vortex model [102] (Section 6.1.1) as described in [10].

A problem is the fact that the vortices are not orthogonal to the light sheet where the data is acquired. The angle between vortices and measurement plane can be in the order of 45° or even more [11]. In the images, this presents itself in an elliptical instead of circular shape of the vortices. For a proper evaluation of these vortices, the direction of the vortices has to be determined and the images

projected onto a plane normal to the vortex direction. Otherwise, for example, the computed size of the vortices will be larger than the actual size.

Two approaches for determining the orientation are described by Burley et al. [11]. The first approach is to remove the mean vectors from the data. Then, within the vortex, averaging of the vectors results in the direction on the vortex as the rotational parts of the vectors erase each other. This approach has to be combined with projecting the data onto the computed orthogonal image plane and iterating the whole process to give good results.

The second approach is to plot the up-wash angles given by the out-of-plane component of the vectors and fit the results with a sine-wave. The maximal value and the zero-crossings of the sine-wave give two angles which determine the direction of the vortex.

Both methods suffer from the facts that the measurement error is largest in the vortex core due to less seeding within the core, and the out-of-plane component itself can not be determined as well as the in-plane components. The averaging of the first method and the fit of the sine-wave in the second method counterbalance some of the errors. Nevertheless, some uncertainty is to be expected. Raffel et al. [75] show that for well-formed vortices, both methods result in generally the same direction, but in the early stages of formation, the second method is more stable as the relative magnitude of the axial velocity is smaller.

The images at each measurement position were averaged after aligning the vortex core positions as the positions differ due to fluctuations in the rotor tip positions (Figure 6.20). In some extraneous images, the vortex was not defined well enough, so images where the maximum vorticity ω_z does not exceed a certain threshold are discarded.

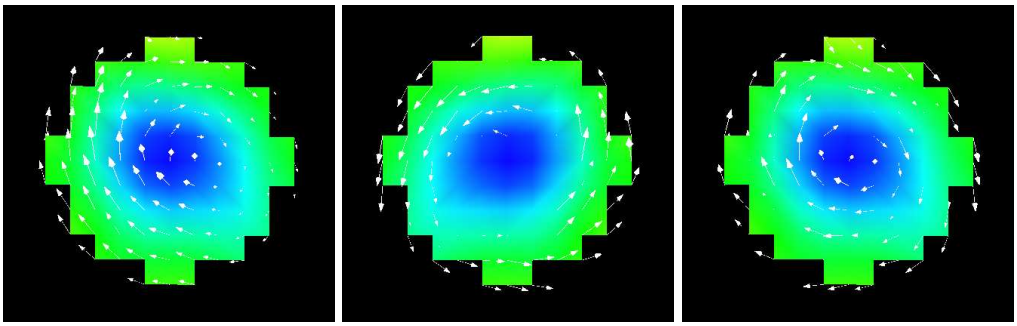

Figure 6.19: Part of a 3-C PIV image around a vortex. The global average has already been removed. **Left**: Velocity in the direction of the vortex. The image is shown from front. **Middle and right**: The local average has been removed, too. Now the vectors are only orthogonal to the vortex direction. As the vortex direction is not orthogonal to the image plane, the vectors come out of the plane. The image is shown from front (**middle**) and back (**right**).

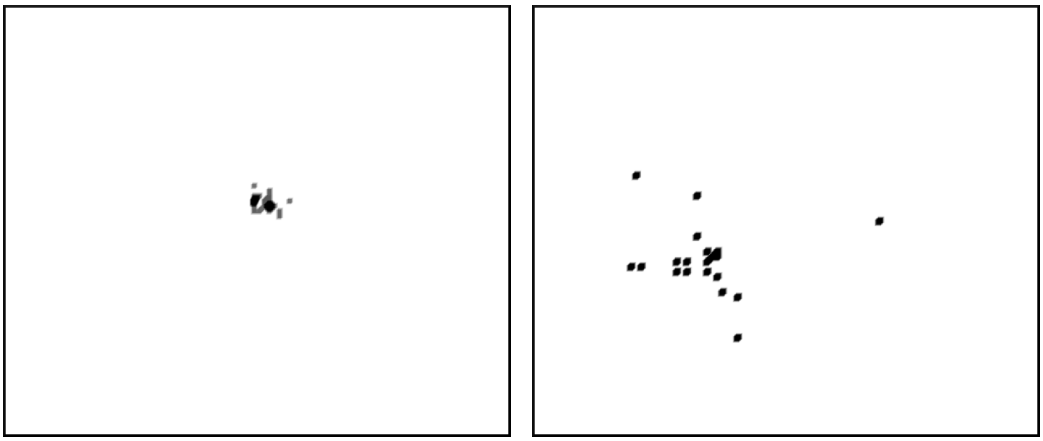

Figure 6.20: Vortex core positions at two measurement positions, 20 images each. **Left**: The measurement position is shortly after vortex creation. **Right**: Measurement position is at a place where the vortex is quite old.

Chapter 7

Clifford Fourier Transform

The Fourier transform (Section 3.3) is a basis transform from image space to frequency space. This is useful since images can be analyzed in frequency space where it is easier to describe the phase and frequency of the image data. Filter responses are often better analyzed in the frequency domain because of the convolution theorem. Thus, applying the Fourier transform to vector fields in the context of Clifford convolution opens up a whole new approach for analyzing vector fields. Furthermore, in signal and image processing, fast Fourier transforms are used to accelerate the computation of convolutions. This would be beneficial for a Fourier transform within Clifford algebra, too. In this section, a Fourier transform within the Clifford algebras G_2 and G_3 is defined.

The basic idea concerning the definition of the Fourier transform of arbitrary multivectors is to use the pseudoscalar $\boldsymbol{i}$ to replace the complex i in the Fourier kernel. This works well in 2D and 3D as $\boldsymbol{i}$ shares the most important property of i, that is $\boldsymbol{i}^2 = i^2 = -1$. However, this approach does not work for arbitrary dimensions as $\boldsymbol{i}_n = (-1)^{\frac{1}{2}n(n-1)}$. As the vector fields from flow visualization are mostly 2D and 3D, sometimes plus an additional time-dimension, the restriction to 2D and 3D is not hindering though it is mathematically unsatisfying.

In Section 7.1, previous and subsequent definitions of other Fourier transforms within Clifford algebra are introduced and discussed. The definition of the Clifford Fourier transform as well as the most important theorems of this transform and their proves can be found in Section 7.2 and 7.3. In Section 7.4, further issues like discretization (Section 7.4.1), fast algorithms for the computation of the Clifford Fourier transform (Section 3.4), the connection to the vector convolution (Section 7.4.2) and the definition of Clifford Gabor filters (Section 7.4.4) can be found. First steps towards an analysis of basic vector valued flow patterns in frequency domain are taken in Section 7.5. Finally, results of applying Clifford Fourier transform and Clifford Gabor filters to template matching can be found in Section 7.6.

7.1 Related Work

Extensions of the Fourier transform of multidimensional signals arose from different areas of research, namely disparity estimation and texture segmentation. There, the analytic signal, which consists of a signal and its Hilbert transform, is used to analyze local phase and amplitude. As the analytic signal is only defined for intrinsically one-dimensional structures, extensions of this signal became necessary. While extending this signal to multidimensional structures, Bülow [9] and Felsberg [34] defined Fourier transforms within Clifford algebra.

The first extension of the Fourier transform to Clifford algebra stems from the analysis of the local structure and phase of a signal, which can be analyzed using Gabor filters (Section 3.5). To obtain symmetries for more than one direction, Bülow [9] defined the analyzing filters in 2^n-dimensional algebras. As the Gabor filter is closely related to the Fourier transform – it is in fact a windowed Fourier transform – the path of extending the Gabor filter to these algebras led via the definition of corresponding Fourier transforms.

Bülow [9] used a Clifford algebra where $\mathbf{e}_j^2 = -1$ and defined the n-dimensional Fourier transform by using the bases $\{\mathbf{e}_1, ..., \mathbf{e}_n\}$ in the Fourier kernel:

Definition 7.1.1 *Let $\boldsymbol{F} : \mathbb{R}^n \to G_n$ be a multivector-valued signal. Let $x, u \in \mathbb{R}^n$. Let the product $\prod_{k=1}^n$ be performed in the fixed order of the indices. Then the* **Bülow Clifford Fourier transform** *of $\boldsymbol{F}$ is defined as*

$$\mathcal{F}_B\{\boldsymbol{F}\}(\boldsymbol{u}) = \int_{\mathbb{R}^n} \boldsymbol{F}(\boldsymbol{x}) \prod_{k=1}^{n} e^{(-\boldsymbol{e}_k 2\pi x_k u_k)} |d\boldsymbol{x}|.$$

If $n = 2$, this transform is also called **Quaternionic Fourier transform**.

The corresponding convolution theorems are rather complicated, and only given for $n = 2$. The complex form of the kernel and the non-commutativity of the multiplication present a problem, especially when trying to establish a fast version of this Fourier transform. Therefore, Bülow [9] makes also use of the commutative hypercomplex algebra HCA_n (Definition 4.4.2) for another definition of a Fourier transform on multidimensional signals:

Definition 7.1.2 *Let $\boldsymbol{F} : \mathbb{R}^n \to HCA_n$ be a multidimensional signal. Let $x, u \in \mathbb{R}^n$. Let*

$$I_n = \begin{pmatrix} \boldsymbol{e}_1 & 0 & \cdots & 0 \\ 0 & \boldsymbol{e}_2 & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \boldsymbol{e}_n \end{pmatrix}$$

be the matrix given by the basis vector $\boldsymbol{e}_j$ of HCA_n. Then the **Commutative Hypercomplex Fourier transform** *of $\boldsymbol{F}$ is defined as*

$$\mathscr{F}_H\{\boldsymbol{F}\}(\boldsymbol{u}) = \int_{\mathbb{R}^n} \boldsymbol{F}(\boldsymbol{x}) e^{(-2\pi \boldsymbol{u}^T I_n \boldsymbol{x})} d\boldsymbol{x}$$

The Commutative Hypercomplex Fourier transform results in the well-known convolution theorem:

Theorem 7.1.3 *Let $\boldsymbol{F}, \boldsymbol{H} : \mathbb{R}^n \to HCA_n$ be two signals. Then*

$$\mathscr{F}_H\{\boldsymbol{F} * \boldsymbol{H}\}(\boldsymbol{u}) = \mathscr{F}_H\{\boldsymbol{F}\}(\boldsymbol{u}) \mathscr{F}_H\{\boldsymbol{H}\}(\boldsymbol{u})$$

However, the step away from the Clifford algebra results in loosing the geometric information inherent in this algebra. Furthermore, Bülow [9] uses these tools for image processing, that is only for scalar-valued fields.

Felsberg [34] extended the analytic signal in 1D to the monogenic signal in 2D by using embedded functions to obtain additional phases. Therefore, he composed vectors by combining spatial coordinates and the corresponding signal value, $v(x_1, x_2) = (x_1, x_2, f(x_1, x_2))$. He defined convolution only for vector-valued fields and spinor-valued signals in G_2 and G_3. Furthermore, the filters are only applied from the left.

Definition 7.1.4 *Let $\boldsymbol{f} : \mathbb{R}^n :\to \mathbb{R}^n \subset G_n$ be a vector-valued field and $\boldsymbol{h} : \mathbb{R}^n \to S_n \subset G_n$ a spinor-valued filter for n=2,3. Then* **convolution** *of $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{x})$ based on the geometric product is defined as*

$$(\boldsymbol{h} * \boldsymbol{f})(\boldsymbol{x}) = \int_{\mathbb{R}^n} \boldsymbol{h}(\boldsymbol{x'}) \boldsymbol{f}(\boldsymbol{x} - \boldsymbol{x'}) |d\boldsymbol{x'}|.$$

Definition 7.1.5 *Let $\boldsymbol{F} : \mathbb{R}^{n-1} \to G_n$, $n = 2, 3$, be an embedded function with $\boldsymbol{x} = x\boldsymbol{e}_1$ in 1D and $\boldsymbol{x} = x_1\boldsymbol{e}_1 + x_2\boldsymbol{e}_2$ in 2D. Then the* **Felsberg Fourier transform** *is defined as*

$$\mathscr{F}\{\boldsymbol{F}\}(\boldsymbol{u}) = \int_{\mathbb{R}} e^{(-2\pi \boldsymbol{i}_n \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} \boldsymbol{F}(\boldsymbol{x}) dx_1$$

for n=2 and

$$\mathscr{F}\{\boldsymbol{F}\}(\boldsymbol{u}) = \int_{\mathbb{R}} \int_{\mathbb{R}} e^{(-2\pi \boldsymbol{i}_3 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} \boldsymbol{F}(\boldsymbol{x}) dx_1 dx_2$$

for n=3.

Note that these definitions are inherently tuned to embedded functions. As the convolution was restricted to vector-valued signals and spinor-valued filters, so

was the convolution theorem. Concerning other theorems like derivative and Parseval's theorem, only special cases of the multivector-valued signals were considered. We extend this approach to genuine vector fields by using Clifford convolution and proving theorems for arbitrary multivector-valued signals.

Another definition of a Fourier transform within Clifford algebra comes from mathematical theory. Brackx et al. [8] pursue the idea of refining the Fourier transform by using operator notation and defining a pair of transformations whose harmonic average is the classical Fourier transform. Note that they use the Clifford algebra where the square of the unit-vectors is $\mathbf{e}_j^2 = -1$.

The classical Fourier transform

$$\mathscr{F}\{f\}(\mathbf{u}) = \int_{\mathbb{R}_n} f(t) e^{(-2\pi i \langle \mathbf{x}, \mathbf{u} \rangle)} d\mathbf{x}$$

of a continuous signal $f : \mathbb{R}^n \to C$ can also be written as

$$\mathscr{F}\{f\} = e^{\left(\frac{-i\pi H}{2}\right)}$$

with the scalar-valued differential operator

$$H = \frac{1}{2}(-\Delta_n + \|\mathbf{x}\|^2 - n).$$

where Δ_n is the Laplace operator computing the second derivative. In a Clifford algebra with $\mathbf{e}_j^2 = -1$, this is equivalent to

$$H = \frac{1}{2}(-\partial_{\mathbf{x}}^2 - \mathbf{x}^2 - n).$$

A split of this operator leads to the refinement of the Fourier transform:

Definition 7.1.6 *Let*

$$H^+ = \frac{1}{2}(\partial_{\boldsymbol{x}} - \boldsymbol{x})(\partial_{\boldsymbol{x}} + \boldsymbol{x}) - \frac{n}{2},$$

$$H^- = \frac{1}{2}(\partial_{\boldsymbol{x}} + \boldsymbol{x})(\partial_{\boldsymbol{x}} - \boldsymbol{x}) - \frac{n}{2}.$$

Then, for n-dimensional signals, the pair of transformations

$$\mathscr{F}_{H^+} = e^{\left(\frac{-i\pi H^+}{2}\right)}$$

$$\mathscr{F}_{H^-} = e^{\left(\frac{-i\pi H^-}{2}\right)}$$

defines the **Brackx Clifford Fourier transform**.

For 2D, the Fourier kernel is, up to constants, given as $e^{(\mathbf{x} \wedge \mathbf{u})}$. A huge disadvantage for the application of this transform is that no closed form is given for other dimensions yet.

7.2 Clifford Fourier Transform in 3D

Definition 7.2.1 *Let $\boldsymbol{F}:\mathbb{R}^3\to G_3$ be a multivector-valued signal. Let $x,u\in\mathbb{R}^3$. The* **Clifford Fourier transform** *(CFT) of $\boldsymbol{F}$ is defined as*

$$\mathscr{F}\{\boldsymbol{F}\}(\boldsymbol{u})=\int_{\mathbb{R}^3}\boldsymbol{F}(\boldsymbol{x})e^{(-2\pi\boldsymbol{i}_3\langle\boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}|.$$

The inverse transformation is then given by

$$\mathscr{F}^{-1}\{\boldsymbol{F}\}(\boldsymbol{x})=\int_{\mathbb{R}^3}\boldsymbol{F}(\boldsymbol{u})e^{(2\pi\boldsymbol{i}_3\langle\boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{u}|.$$

Note that the Clifford Fourier kernel $e^{(-2\pi\boldsymbol{i}_3\langle\mathbf{x},\mathbf{u}\rangle)}$ is multivector valued. To be more exact, it consists of a scalar and a pseudoscalar.

Theorem 7.2.2 *The Clifford Fourier transform is a linear combination of four classical Fourier transforms.*
Proof*: For a multivector field $\boldsymbol{F}:\mathbb{R}^3\to G_3$, we have*

$$\begin{aligned}\boldsymbol{F}(\boldsymbol{x}) &= \boldsymbol{F}_0(\boldsymbol{x})+\boldsymbol{F}_1(\boldsymbol{x})\boldsymbol{e}_1+\boldsymbol{F}_2(\boldsymbol{x})\boldsymbol{e}_2+\boldsymbol{F}_3(\boldsymbol{x})\boldsymbol{e}_3\\ &+ \boldsymbol{F}_{23}(\boldsymbol{x})\boldsymbol{e}_{23}+\boldsymbol{F}_{31}(\boldsymbol{x})\boldsymbol{e}_{31}+\boldsymbol{F}_{12}(\boldsymbol{x})\boldsymbol{e}_{12}+\boldsymbol{F}_{123}(\boldsymbol{x})\boldsymbol{e}_{123}\\ &= \boldsymbol{F}_0(\boldsymbol{x})+\boldsymbol{F}_1(\boldsymbol{x})\boldsymbol{e}_1+\boldsymbol{F}_2(\boldsymbol{x})\boldsymbol{e}_2+\boldsymbol{F}_3(\boldsymbol{x})\boldsymbol{e}_3\\ &+ \boldsymbol{F}_{23}(\boldsymbol{x})\boldsymbol{i}_3\boldsymbol{e}_1+\boldsymbol{F}_{31}(\boldsymbol{x})\boldsymbol{i}_3\boldsymbol{e}_2+\boldsymbol{F}_{12}(\boldsymbol{x})\boldsymbol{i}_3\boldsymbol{e}_3+\boldsymbol{F}_{123}(\boldsymbol{x})\boldsymbol{i}_3\end{aligned}$$

which can be regarded as four complex signals:

$$\begin{aligned}\boldsymbol{F}(\boldsymbol{x}) &= [\boldsymbol{F}_0(\boldsymbol{x})+\boldsymbol{F}_{123}(\boldsymbol{x})\boldsymbol{i}_3]1\\ &+ [\boldsymbol{F}_1(\boldsymbol{x})+\boldsymbol{F}_{23}(\boldsymbol{x})\boldsymbol{i}_3]\boldsymbol{e}_1\\ &+ [\boldsymbol{F}_2(\boldsymbol{x})+\boldsymbol{F}_{31}(\boldsymbol{x})\boldsymbol{i}_3]\boldsymbol{e}_2\\ &+ [\boldsymbol{F}_3(\boldsymbol{x})+\boldsymbol{F}_{12}(\boldsymbol{x})\boldsymbol{i}_3]\boldsymbol{e}_3\end{aligned}$$

This can be interpreted as an element of $\mathbb{C}^4$. Considering the linearity of the Clifford Fourier transform, we get

$$\begin{aligned}\mathscr{F}\{\boldsymbol{F}\}(\boldsymbol{u}) &= [\mathscr{F}\{\boldsymbol{F}_0(\boldsymbol{x})+\boldsymbol{F}_{123}(\boldsymbol{x})\boldsymbol{i}_3\}(\boldsymbol{u})]1\\ &+ [\mathscr{F}\{\boldsymbol{F}_1(\boldsymbol{x})+\boldsymbol{F}_{23}(\boldsymbol{x})\boldsymbol{i}_3\}(\boldsymbol{u})]\boldsymbol{e}_1\\ &+ [\mathscr{F}\{\boldsymbol{F}_2(\boldsymbol{x})+\boldsymbol{F}_{31}(\boldsymbol{x})\boldsymbol{i}_3\}(\boldsymbol{u})]\boldsymbol{e}_2\\ &+ [\mathscr{F}\{\boldsymbol{F}_3(\boldsymbol{x})+\boldsymbol{F}_{12}(\boldsymbol{x})\boldsymbol{i}_3\}(\boldsymbol{u})]\boldsymbol{e}_3.\end{aligned}$$

The Fourier kernel consists of a scalar and a pseudoscalar or trivector. Considering the multiplication rules for a multiplication with them concludes this proof.

Note that dual pairs form Fourier pairs. The multivector space is divided into four orthogonal spaces which are then transformed separately. Because of Equation (4.2.4), the 3D Clifford Fourier kernel commutes with every multivector (although Clifford multiplication is not generally commutative). All of the well-known theorems hold. Because of the non-commutativity of the Clifford multiplication, we present theorems for the application of a filter from the left and right.

Theorem 7.2.3 (Shift theorem) *Let $\boldsymbol{F} : \mathbb{R}^3 \to G_3$ be multivector valued and let $\mathcal{F}\{\boldsymbol{F}\}$ exist. Then*

$$\mathcal{F}\{\boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})\}(\boldsymbol{u}) = \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x'},\boldsymbol{u}\rangle)} .$$

Proof*:*

$$\begin{aligned} & \mathcal{F}\{\boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})\}(\boldsymbol{u}) \\ = & \int_{\mathbb{R}^3} \boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}| \\ = & \int_{\mathbb{R}^3} \boldsymbol{F}(\boldsymbol{k})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{k},\boldsymbol{u}\rangle)}e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x'},\boldsymbol{u}\rangle)}|d\boldsymbol{k}| \\ = & \int_{\mathbb{R}^3} \boldsymbol{F}(\boldsymbol{k})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{k},\boldsymbol{u}\rangle)}|d\boldsymbol{k}|e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x'},\boldsymbol{u}\rangle)} \\ = & \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x'},\boldsymbol{u}\rangle)} . \end{aligned}$$

Theorem 7.2.4 (Convolution theorem) *Let $\boldsymbol{F},\boldsymbol{H} : \mathbb{R}^3 \to G_3$ be multivector valued and let $\mathcal{F}\{\boldsymbol{F}\}$ and $\mathcal{F}\{\boldsymbol{H}\}$ exist. Then*

$$\begin{aligned} \mathcal{F}\{\boldsymbol{H} *_l \boldsymbol{F}\}(\boldsymbol{u}) &= \mathcal{F}\{\boldsymbol{H}\}(\boldsymbol{u})\mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) \\ \textit{and}\quad \mathcal{F}\{\boldsymbol{F} *_r \boldsymbol{H}\}(\boldsymbol{u}) &= \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u})\mathcal{F}\{\boldsymbol{H}\}(\boldsymbol{u}). \end{aligned}$$

Proof*:*

$$\begin{aligned} & \mathcal{F}\{\boldsymbol{H} *_l \boldsymbol{F}\}(\boldsymbol{u}) \\ = & \int_{\mathbb{R}^3} \left(\int_{\mathbb{R}^3} \boldsymbol{H}(\boldsymbol{x'})\boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})|d\boldsymbol{x'}|\right) e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}| \\ = & \int_{\mathbb{R}^3} \left(\int_{\mathbb{R}^3} \boldsymbol{H}(\boldsymbol{x'})\boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x'}|\right) |d\boldsymbol{x}| \\ = & \int_{\mathbb{R}^3} \left(\int_{\mathbb{R}^3} \boldsymbol{H}(\boldsymbol{x'})\boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}|\right) |d\boldsymbol{x'}| \\ = & \int_{\mathbb{R}^3} \boldsymbol{H}(\boldsymbol{x'}) \left(\int_{\mathbb{R}^3} \boldsymbol{F}(\boldsymbol{x}-\boldsymbol{x'})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}|\right) |d\boldsymbol{x'}| \\ = & \int_{\mathbb{R}^3} \boldsymbol{H}(\boldsymbol{x'})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x'},\boldsymbol{u}\rangle)}\mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u})|d\boldsymbol{x'}| \\ = & \int_{\mathbb{R}^3} \boldsymbol{H}(\boldsymbol{x'})e^{(-2\pi \boldsymbol{i}_3\langle \boldsymbol{x'},\boldsymbol{u}\rangle)}|d\boldsymbol{x'}|\mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) \\ = & \mathcal{F}\{\boldsymbol{H}\}(\boldsymbol{u})\mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) . \end{aligned}$$

Because of the commutativity of the Clifford Fourier kernel, see Equation (4.2.4), the proof of the other case is analog to the one above.

Theorem 7.2.5 (Derivative theorem) *Let the preconditions be the same as in 7.2.4. Then*

$$\begin{aligned} \mathcal{F}\{\nabla \boldsymbol{F}\}(\boldsymbol{u}) &= 2\pi \boldsymbol{i}_3 \boldsymbol{u} \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) , \\ \mathcal{F}\{\boldsymbol{F}\nabla\}(\boldsymbol{u}) &= \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) 2\pi \boldsymbol{i}_3 \boldsymbol{u} , \\ \mathcal{F}\{\Delta \boldsymbol{F}\}(\boldsymbol{u}) &= -4\pi^2 \boldsymbol{u}^2 \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) , \\ \textit{and} \quad \mathcal{F}\{\boldsymbol{F}\Delta\}(\boldsymbol{u}) &= -4\pi^2 \boldsymbol{u}^2 \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) . \end{aligned}$$

Proof*: Since*

$$\begin{aligned} (\nabla \boldsymbol{F})(\boldsymbol{x}) &= \nabla \mathcal{F}^{-1}\{\mathcal{F}\{\boldsymbol{F}\}\}(\boldsymbol{x}) \\ &= \nabla \textstyle\int_{\mathbb{R}^3} \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) e^{(2\pi \boldsymbol{i}_3 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} |d\boldsymbol{x}| \\ &= \textstyle\int_{\mathbb{R}^3} \nabla \left(\mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) e^{(2\pi \boldsymbol{i}_3 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} \right) |d\boldsymbol{x}| \\ &= \textstyle\int_{\mathbb{R}^3} \nabla \left(e^{(2\pi \boldsymbol{i}_3 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} \right) \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) |d\boldsymbol{x}| \\ &= \textstyle\int_{\mathbb{R}^3} 2\pi \boldsymbol{i}_3 \boldsymbol{u} e^{(2\pi \boldsymbol{i}_3 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) |d\boldsymbol{x}| \\ &= \mathcal{F}^{-1}(2\pi \boldsymbol{i}_3 \boldsymbol{u} \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u})) \end{aligned}$$

we get

$$\mathcal{F}\{\nabla \boldsymbol{F}\}(\boldsymbol{u}) = 2\pi \boldsymbol{i}_3 \boldsymbol{u} \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u})$$

and

$$\mathcal{F}\{\Delta \boldsymbol{F}\}(\boldsymbol{u}) = 2\pi \boldsymbol{i}_3 \boldsymbol{u} \mathcal{F}\{\nabla \boldsymbol{F}\}(\boldsymbol{u}) = -4\pi^2 \boldsymbol{u}^2 \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) .$$

The application of the derivative from the right can be proved analogously.

Theorem 7.2.6 (Parseval's theorem) *Let the preconditions be the same as in 7.2.3. Then*

$$\|\boldsymbol{F}\|_2 = \|\mathcal{F}\{\boldsymbol{F}\}\|_2 .$$

This is also true for the different grades of the multivector-valued signal $\boldsymbol{F}$ *such that*

$$\|\langle \boldsymbol{F} \rangle_j\|_2 = \|\mathcal{F}\{\langle \boldsymbol{F} \rangle_j\}\|_2, \ j = 0, .., 3 .$$

Proof*: The theorem for the classical Fourier transform is* $\|f\|_2 = \|\mathcal{F}\{f\}\|_2$*. The proof for the Clifford Fourier transform follows directly since the CFT is a linear combination of several classical Fourier transforms.*

7.3 Clifford Fourier Transform in 2D

Definition 7.3.1 *Let* $\boldsymbol{F} : \mathbb{R}^2 \to G_2$ *be a multivector-valued signal. Let* $x, u \in \mathbb{R}^2$*. The* **Clifford Fourier transform** *(CFT) of* $\boldsymbol{F}$ *is defined as*

$$\mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) = \int_{\mathbb{R}^2} \boldsymbol{F}(\boldsymbol{x}) e^{(-2\pi \boldsymbol{i}_2 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} |d\boldsymbol{x}|.$$

The inverse transformation is then given by

$$\mathscr{F}^{-1}\{\boldsymbol{F}\}(\boldsymbol{x}) = \int_{\mathbb{R}^2} \boldsymbol{F}(\boldsymbol{u}) e^{(2\pi \boldsymbol{i}_2 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} |d\boldsymbol{u}|.$$

Note that this Clifford Fourier kernel $e^{(-2\pi \boldsymbol{i}_2 \langle \mathbf{x}, \mathbf{u} \rangle)}$ again consists of a scalar and a pseudoscalar. In G_2 this implies that the kernel is spinor-valued.

Theorem 7.3.2 *The Clifford Fourier transform is a linear combination of two classical Fourier transforms.*
Proof*: For a multivector field $\boldsymbol{F} : \mathbb{R}^2 \to G_2$, we have*

$$\boldsymbol{F} \;=\; \boldsymbol{F}_0 + \boldsymbol{F}_1 \boldsymbol{e}_1 + \boldsymbol{F}_2 \boldsymbol{e}_2 + \boldsymbol{F}_{12} \boldsymbol{e}_{12},$$

which can be regarded as two complex signals

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{x}) &= 1[\boldsymbol{F}_0(\boldsymbol{x}) + \boldsymbol{F}_{12}(\boldsymbol{x}) \boldsymbol{i}_2] \\ &+ \boldsymbol{e}_1[\boldsymbol{F}_1(\boldsymbol{x}) + \boldsymbol{F}_2(\boldsymbol{x}) \boldsymbol{i}_2,] \end{aligned}$$

which can be interpreted as an element of $\mathbb{C}^2$. Considering the linearity of the Clifford Fourier transform, we get

$$\begin{aligned} \mathscr{F}\{\boldsymbol{F}\}(\boldsymbol{u}) &= 1[\mathscr{F}\{\boldsymbol{F}_0(\boldsymbol{x}) + \boldsymbol{F}_{12}(\boldsymbol{x}) \boldsymbol{i}_2\}(\boldsymbol{u})] \\ &+ \boldsymbol{e}_1[\mathscr{F}\{\boldsymbol{F}_1(\boldsymbol{x}) + \boldsymbol{F}_2(\boldsymbol{x}) \boldsymbol{i}_2\}(\boldsymbol{u})] \end{aligned}$$

which means that the 2D Clifford Fourier transform is the linear combination of two classical Fourier transforms.

Again, dual pairs form Fourier pairs. This time, the Fourier kernel does not commute with every multivector, rather it commutes with the spinor part and anti-commutes with the vector part. Therefore, we present convolution theorems for vector and spinor-valued fields separately. Note also that the multiplication of the Fourier kernel from the right is not quite the same as that from the left as in 3D.

Theorem 7.3.3 (Shift theorem) *Let $\boldsymbol{F}$ be multivector valued and let $\mathscr{F}\{\boldsymbol{F}\}$ exist. Then, we have*

$$\mathscr{F}\{\boldsymbol{F}(\boldsymbol{x} - \boldsymbol{x'})\}(\boldsymbol{u}) = \mathscr{F}\{\boldsymbol{F}\}(\boldsymbol{u}) e^{(-2\pi \boldsymbol{i}_2 \langle \boldsymbol{x'}, \boldsymbol{u} \rangle)}\,.$$

Proof*: Analogous to the 3D equivalent (Theorem 7.2.3).*

Theorem 7.3.4 (Convolution theorem) *Let $\boldsymbol{F},\boldsymbol{H}$ be multivector valued, $\boldsymbol{f},\boldsymbol{h}$ be vector valued and $\mathbf{f},\mathbf{h}$ be spinor valued. Note that $\boldsymbol{F},\boldsymbol{f}$ and $\mathbf{f}$ are fields and $\boldsymbol{H},\boldsymbol{h}$ and $\mathbf{h}$ are filters. Let $\mathscr{F}\{\boldsymbol{F}\}$, $\mathscr{F}\{\boldsymbol{H}\}$, $\mathscr{F}\{\boldsymbol{f}\}$, $\mathscr{F}\{\boldsymbol{h}\}$, $\mathscr{F}\{\mathbf{f}\}$ and $\mathscr{F}\{\mathbf{h}\}$ exist. Then, we have*

$$\begin{aligned}
\mathscr{F}\{\boldsymbol{H}*_l\mathbf{f}\}(\boldsymbol{u}) &= \mathscr{F}\{\boldsymbol{H}\}(\boldsymbol{u})\mathscr{F}\{\mathbf{f}\}(\boldsymbol{u})\,,\\
\mathscr{F}\{\boldsymbol{H}*_l\boldsymbol{f}\}(\boldsymbol{u}) &= \mathscr{F}\{\dot{\boldsymbol{H}}\}(\boldsymbol{u})\mathscr{F}\{\boldsymbol{f}\}(\boldsymbol{u})\,,\\
\mathscr{F}\{\boldsymbol{F}*_r\boldsymbol{h}\}(\boldsymbol{u}) &= \mathscr{F}\{\boldsymbol{F}\}(\boldsymbol{u})\mathscr{F}\{\boldsymbol{h}\}(\boldsymbol{u})\,,\\
\textit{and}\quad \mathscr{F}\{\boldsymbol{F}*_r\boldsymbol{h}\}(\boldsymbol{u}) &= \mathscr{F}\{\dot{\boldsymbol{F}}\}(\boldsymbol{u})\mathscr{F}\{\dot{\boldsymbol{h}}\}(\boldsymbol{u})\,.
\end{aligned}$$

Proof*: Let $\mathbf{f}$ be spinor valued. Then we have*

$$\begin{aligned}
&\mathscr{F}\{\boldsymbol{H}*_l\mathbf{f}\}(\boldsymbol{u})\\
=\;& \textstyle\int_{\mathbb{R}^2}\left(\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')\mathbf{f}(\boldsymbol{x}-\boldsymbol{x}')|d\boldsymbol{x}'|\right)e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}|\\
=\;& \textstyle\int_{\mathbb{R}^2}\left(\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')\mathbf{f}(\boldsymbol{x}-\boldsymbol{x}')e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}'|\right)|d\boldsymbol{x}|\\
=\;& \textstyle\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')\left(\int_{\mathbb{R}^2}\mathbf{f}(\boldsymbol{x}-\boldsymbol{x}')e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}|\right)|d\boldsymbol{x}'|\\
=\;& \textstyle\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')\mathscr{F}\{\mathbf{f}\}(\boldsymbol{u})e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x}',\boldsymbol{u}\rangle)}|d\boldsymbol{x}'|\\
=\;& \textstyle\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x}',\boldsymbol{u}\rangle)}|d\boldsymbol{x}'|\mathscr{F}\{\mathbf{f}\}(\boldsymbol{u})\\
=\;& \mathscr{F}\{\boldsymbol{H}\}(\boldsymbol{u})\mathscr{F}\{\mathbf{f}\}(\boldsymbol{u})\,.
\end{aligned}$$

Let $\boldsymbol{f}$ be vector valued, we have

$$\begin{aligned}
&\mathscr{F}\{\boldsymbol{H}\star_l\boldsymbol{f}\}(\boldsymbol{u})\\
=\;& \textstyle\int_{\mathbb{R}^2}\left(\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{x}')|d\boldsymbol{x}'|\right)e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}|\\
=\;& \textstyle\int_{\mathbb{R}^2}\left(\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{x}')e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}'|\right)|d\boldsymbol{x}|\\
=\;& \textstyle\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')\left(\int_{\mathbb{R}^2}\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{x}')e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x},\boldsymbol{u}\rangle)}|d\boldsymbol{x}|\right)|d\boldsymbol{x}'|\\
=\;& \textstyle\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')\mathscr{F}\{\boldsymbol{f}\}(\boldsymbol{u})e^{(2\pi\boldsymbol{i}_2\langle\boldsymbol{x}',\boldsymbol{u}\rangle)}|d\boldsymbol{x}'|\\
=\;& \textstyle\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x}',\boldsymbol{u}\rangle)}\mathscr{F}\{\boldsymbol{f}\}(\boldsymbol{u})|d\boldsymbol{x}'|\\
=\;& \textstyle\int_{\mathbb{R}^2}\boldsymbol{H}(\boldsymbol{x}')e^{(-2\pi\boldsymbol{i}_2\langle\boldsymbol{x}',\boldsymbol{u}\rangle)}|d\boldsymbol{x}'|\mathscr{F}\{\boldsymbol{f}\}(\boldsymbol{u})\\
=\;& \mathscr{F}\{\boldsymbol{H}\}(\boldsymbol{u})\mathscr{F}\{\boldsymbol{f}\}(\boldsymbol{u})\,.
\end{aligned}$$

and therefore

$$\mathscr{F}\{\boldsymbol{H}*_l\boldsymbol{f}\}(\boldsymbol{u}) = \mathscr{F}\{\dot{\boldsymbol{H}}\}(\boldsymbol{u})\mathscr{F}\{\boldsymbol{f}\}(\boldsymbol{u})\,.$$

The other cases of the convolution theorem can be proved analogously.

Theorem 7.3.5 (Derivative theorem) *Let the preconditions be as in 7.3.4. Then,*

we have

$$\begin{aligned}
\mathcal{F}\{\nabla \boldsymbol{f}\}(\boldsymbol{u}) &= -2\pi \boldsymbol{i}_2 \boldsymbol{u} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) , \\
\mathcal{F}\{\boldsymbol{f}\nabla\}(\boldsymbol{u}) &= 2\pi \boldsymbol{i}_2 \boldsymbol{u} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) , \\
\mathcal{F}\{\nabla \boldsymbol{f}\}(\boldsymbol{u}) &= 2\pi \boldsymbol{i}_2 \boldsymbol{u} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) , \\
\mathcal{F}\{\boldsymbol{f}\nabla\}(\boldsymbol{u}) &= 2\pi \boldsymbol{i}_2 \boldsymbol{u} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) , \\
\mathcal{F}\{\boldsymbol{F}\nabla\}(\boldsymbol{u}) &= \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u})(-2\pi \boldsymbol{i}_2 u) , \\
\mathcal{F}\{\Delta \boldsymbol{F}\}(\boldsymbol{u}) &= 4\pi^2 \boldsymbol{u}^2 \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) , \\
\textit{and} \quad \mathcal{F}\{\boldsymbol{F}\Delta\}(\boldsymbol{u}) &= 4\pi^2 \boldsymbol{u}^2 \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}) .
\end{aligned}$$

Proof*: For spinor* $\boldsymbol{f}$*, the proof of the derivative theorem is analogous to that of (7.2.5). For vector valued* $\boldsymbol{f}$*, we have*

$$\begin{aligned}
(\nabla \boldsymbol{f})(\boldsymbol{x}) &= \nabla \mathcal{F}^{-1}\{\mathcal{F}\{\boldsymbol{f}\}\}(\boldsymbol{x}) \\
&= \nabla \int_{\mathbb{R}^2} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) e^{(2\pi \boldsymbol{i}_2 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} |d\boldsymbol{x}| \\
&= \int_{\mathbb{R}^2} \nabla \left(e^{(-2\pi \boldsymbol{i}_2 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) \right) |d\boldsymbol{x}| \\
&= \int_{\mathbb{R}^2} \nabla \left(e^{(-2\pi \boldsymbol{i}_2 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} \right) \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) |d\boldsymbol{x}| \\
&= \int_{\mathbb{R}^2} -2\pi \boldsymbol{i}_2 \boldsymbol{u} e^{(-2\pi \boldsymbol{i}_2 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) |d\boldsymbol{x}| \\
&= \int_{\mathbb{R}^2} -2\pi \boldsymbol{i}_2 \boldsymbol{u} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}) e^{(2\pi \boldsymbol{i}_2 \langle \boldsymbol{x}, \boldsymbol{u} \rangle)} |d\boldsymbol{x}| \\
&= \mathcal{F}^{-1}(-2\pi \boldsymbol{i}_2 \boldsymbol{u} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}))
\end{aligned}$$

and

$$\mathcal{F}\{\nabla \boldsymbol{f}\}(\boldsymbol{u}) = -2\pi \boldsymbol{i}_2 \boldsymbol{u} \mathcal{F}\{\boldsymbol{f}\}(\boldsymbol{u}).$$

Since ∇ *is a vector and anti-commutes with the 2D Clifford Fourier kernel, the derivative theorem for the application of the derivative from the right can be proved analogously. Thus, we have*

$$\mathcal{F}\{\Delta \boldsymbol{F}\}(\boldsymbol{u}) = \mathcal{F}\{\boldsymbol{F}\nabla\}(\boldsymbol{u}) = 4\pi^2 \boldsymbol{u}^2 \mathcal{F}\{\boldsymbol{F}\}(\boldsymbol{u}).$$

Note that the frequency u is vector valued and anti-commutes with $\boldsymbol{i}_2$*.*

Theorem 7.3.6 (Parseval's theorem) *Let the preconditions be as in 7.3.3. Then, we have*

$$\|\boldsymbol{F}\|_2 = \|\mathcal{F}\{\boldsymbol{F}\}\|_2.$$

Proof*: Analogous to the 3D equivalent (Theorem 7.2.6).*

7.4 Further Definitions and Properties

7.4.1 Discrete Clifford Fourier Transformation

Properties of the classical Fourier transform, like the sampling theorem and issues regarding discretization and periodicity, can be extended easily since the Clifford Fourier transform is a linear combination of several classical Fourier transforms.

Theorem 7.4.1 (Sampling theorem) *Let $\boldsymbol{F}$ be a multivector-valued signal sampled on a uniform grid having spacings Δx_w. If $\boldsymbol{F}$ is bandlimited, that is*

$$\|\boldsymbol{F}\|_2 = 0 \; \forall |k_w| \geq k_{nyq},$$

then it can be reconstructed without error provided that $\frac{1}{\Delta x_w} > 2k_{nyq}$, where k_{nyq} is the Nyquist rate.

Proof*: The sampling theorem holds for f when f is complex-valued. $\boldsymbol{F}$ can be understood as a linear combination of two complex signals in 2D or four complex signals in 3D, all sampled on the same grid. The Clifford Fourier transform is a linear combination of two or four classical Fourier transforms. Thus, the sampling theorem for multivector-valued signals follows directly from the sampling theorem of complex-valued signals.*

7.4.2 Clifford Fourier Transform and Vector Convolution

Since the vector convolution on vector fields, as given by Heiberg et al. [42], is part of the Clifford convolution, we can also analyze it within this context. However, the theorems for vector convolution and correlation are not as simple as those for Clifford convolution and correlation. Consider convolution in 3D where $\mathbf{f}, \mathbf{h} : \mathbb{R}^3 \to \mathbb{R}^3 \subset G_3$ are two vector fields.

$$\begin{aligned} \text{Since} \qquad (\mathbf{h} *_v \mathbf{f})(\mathbf{x}) &= \langle (\mathbf{h} *_l \mathbf{f}) \rangle_0 \\ \text{we get} \quad \mathscr{F}\{(\mathbf{h} *_v \mathbf{f})\}(\mathbf{u}) &= \langle \mathscr{F}\{\mathbf{h}\}, \mathscr{F}\{\mathbf{f}\} \rangle + \langle \mathscr{F}\{\mathbf{h}\} \mathscr{F}\{\mathbf{f}\} \rangle_3 \end{aligned}$$

Since the Clifford Fourier transforms of 3D vector fields contain a vector and a bivector part, $\langle \mathscr{F}\{\mathbf{h}\} \mathscr{F}\{\mathbf{f}\} \rangle_3$ is generally nonzero. In Figure 7.3, one can see that there are substantial vector and bivector parts, even for typical vector patterns.

7.4.3 Fast Clifford Fourier Transform

One of the reasons for the success of the Fourier transform in image processing is the existence of fast Fourier transform (Section 3.4). Algorithms for the fast computation of the Fourier transform take a divide and conquer approach based on recursively dividing even and odd elements. The basic approach assumes that the dimensions of the images are of the form 2^k. Since the Clifford Fourier Transform can be computed as a linear combination of several regular Fourier transforms, FFT-like algorithms can be applied directly for acceleration of the CFT. The Clifford Fourier transforms used for Figures 7.1 and 7.5 have been computed using a fast Clifford Fourier transform. Thus, the computational time of the CFT for this data set was reduced to less than a second.

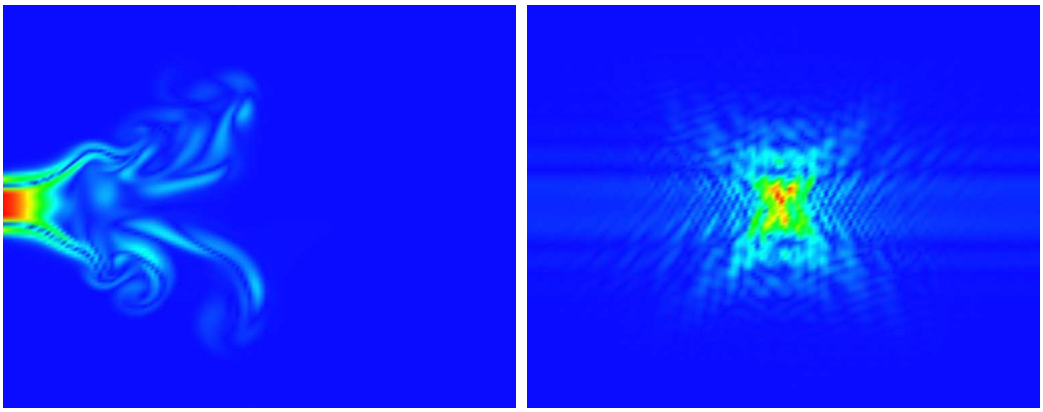

Figure 7.1: OM06(2^k) (Section 6.2). **Left**: Color coding of the absolute magnitude of the vectors. The colors are scaled from zero (blue) to the maximal magnitude (red). **Right**: (Fast) Discrete Clifford Fourier transform of the dataset. Zero frequency is located in the middle of the image. Vectors transform to multivectors when using Clifford algebra in frequency domain, thus, color coding is based on the magnitude of the multivectors. Scaling of the colors is the same as the left image.

7.4.4 Clifford Gabor Filter

The definition of Gabor filters in 2D and 3D Clifford algebra is the convolution of the Clifford Fourier kernel with a Gaussian filter analog to Definition 3.5.3. Thus, the complex i in their original definition is replaced by $\boldsymbol{i}_2$ or $\boldsymbol{i}_3$ respectively. As the CFT can be reduced to several complex transforms in the coordinates, it directly inherits most of the properties of the Fourier transform of scalar fields. The Gabor Filter can be understood as short-time Fourier transforms. Thus, the Gabor filters on multivector fields inherits the properties of the Gabor filters on scalar fields, too.

Definition 7.4.2 *Let $g : \mathbb{R}^n \to \mathbb{R}$ be a Gaussian filter with variance σ. The impulse response $h : \mathbb{R}^n \to \mathbb{C}$, $n = 2, 3$, of the* **Clifford Gabor filter** *is defined as*

$$\boldsymbol{h}(\boldsymbol{x}) = g(\boldsymbol{x}) * e^{2\pi \boldsymbol{i}_n \langle \boldsymbol{x}, \boldsymbol{U} \rangle}$$

The frequency response is

$$\boldsymbol{H}(\boldsymbol{u}) = e^{-2\pi^2 \sigma^2 \|\boldsymbol{u}\|)}$$

7.5 Analysis of Basic Vector-Valued Patterns

Now that the Clifford Fourier transform is defined, the connection between waveforms like sine and cosine and basic patterns found in flow fields can be studied. First of all, a sine wave in one direction was taken as one of the two components of a vector field resulting in a pattern similar to shear flow. This sine wave was then shifted in the spatial domain to observe the change of phase in frequency domain. The results are analogous to scalar fields, and only the phase of the 2D vector, which is interpreted as a complex number, changes. By adding a second waveform in the other direction, flow patterns which are well known are generated (Figure 7.2).

This leads to the analysis of vector valued flow patterns in frequency domain. Therefore, the Clifford Fourier transforms of some 3D patterns (Figure 7.3) were studied. 3D patterns are easier to understand due to the interpretation of the 3D vector as three complex numbers in contrast to the 2D vector which is interpreted as only one complex number.

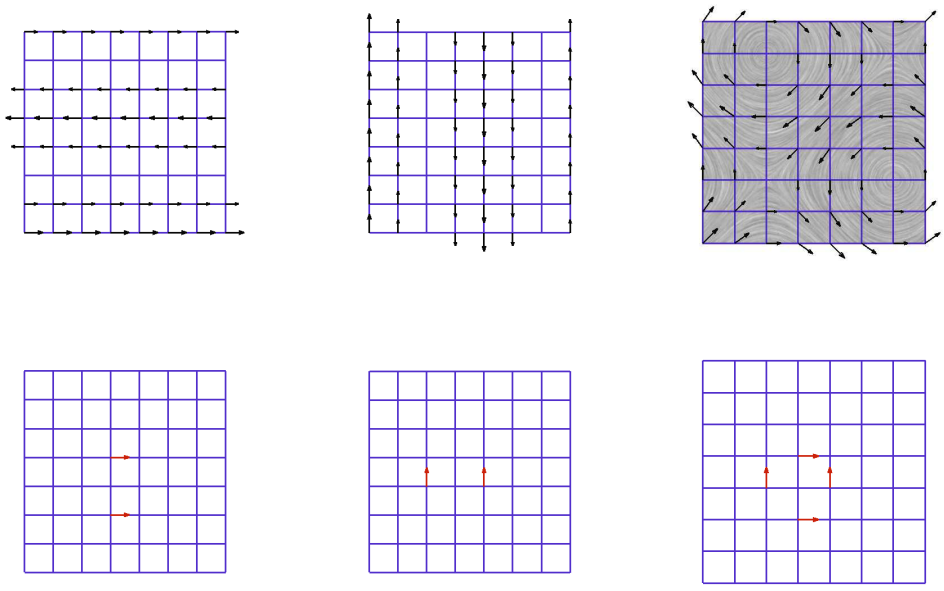

Figure 7.2: Two single waveforms result in complex flow pattern when overlaid. **Top**: Grid, hedgehog and LIC of the patterns in spatial domain. **Bottom**: Grid and hedgehog of the DCFT of the patterns. **Left**: A waveform in y-direction in the x-coordinate of the vector. **Middle** A waveform in x-direction in the y-coordinate. **Right**: The flow pattern resulting from a superposition of the single waves.

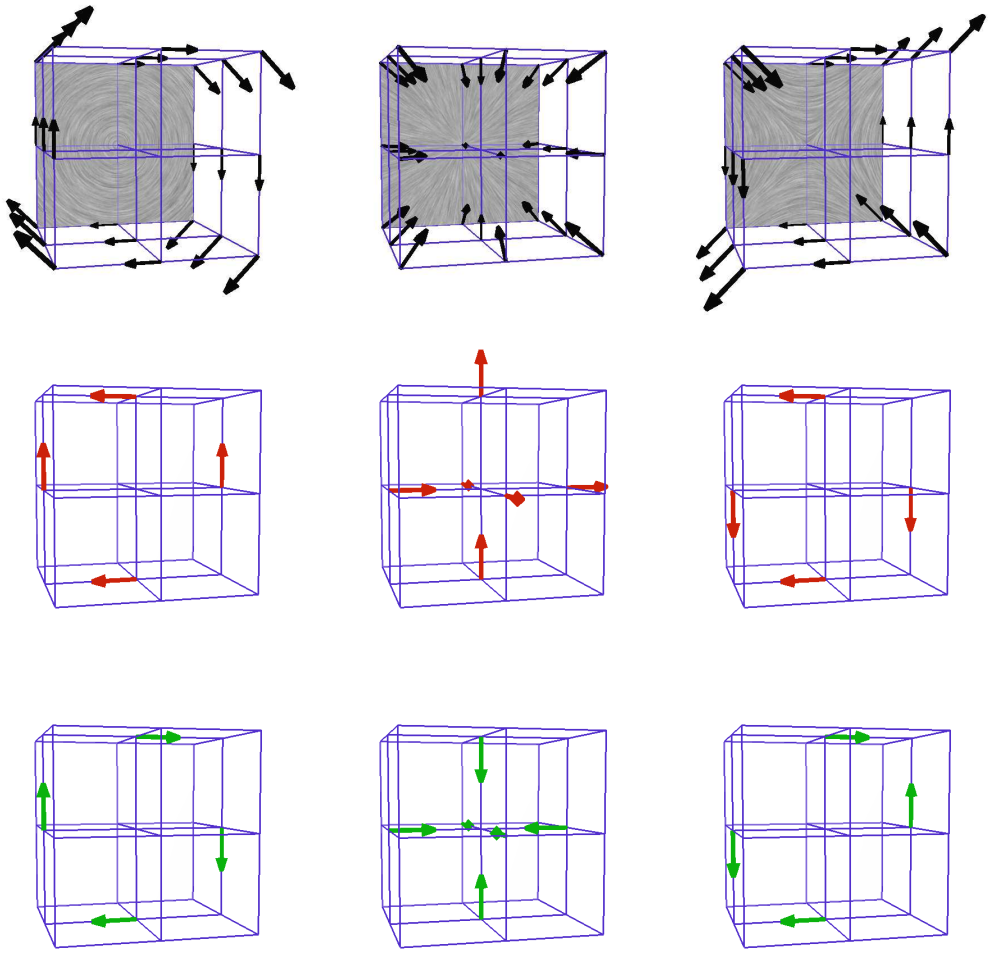

Figure 7.3: **Top**: Various 3D patterns. **Middle**: The vector part of their DCFT. **Bottom**: The bivector part of their DCFT, displayed as normal vector of the plane. **Left**: $3 \times 3 \times 3$ rotation in one coordinate plane. **Middle**: $3 \times 3 \times 3$ convergence. **Right**: $3 \times 3 \times 3$ saddle line. The mean value of the DCFT is situated in the center of the field. In 3D, the waves forming the patterns can be easily seen in the frequency domain. The magnitude of the bivectors of the DCFT is only half the magnitude of the corresponding vectors, though both are displayed with same length.

This split of the components of a 3D vector can be found in the DCFTs of the three patterns showing rotation and a saddle along one axis, and a convergence to a point (Figure 7.3). Rotation and saddle line, which are a repetition of the corresponding 2D pattern (Figure 7.4) along the third axis, have no non-zero Clif-

ford Fourier coefficients corresponding to this third direction as the values do not change in this direction. The DCFT of the convergence, on the other hand, has non-zero coefficients for the third direction as convergence takes also place in this direction.

Note that all 3D patterns are real valued. Thus, the real or vector valued parts of DCFTs are symmetric (Theorem 3.3.11), that is

$$\langle \mathscr{F}_r\{f\}(u)\rangle_1 = \langle \mathscr{F}_r\{f\}(-u)\rangle_1.$$

The imaginary or bivector parts are antisymmetric,

$$\langle \mathscr{F}_i\{f\}(u)\rangle_2 = -\langle \mathscr{F}_i\{f\}(-u)\rangle_2.$$

This is in contrast to the visual perception of these patterns, which would rate the vector part as antisymmetric and the bivector part as symmetric due to the orien-

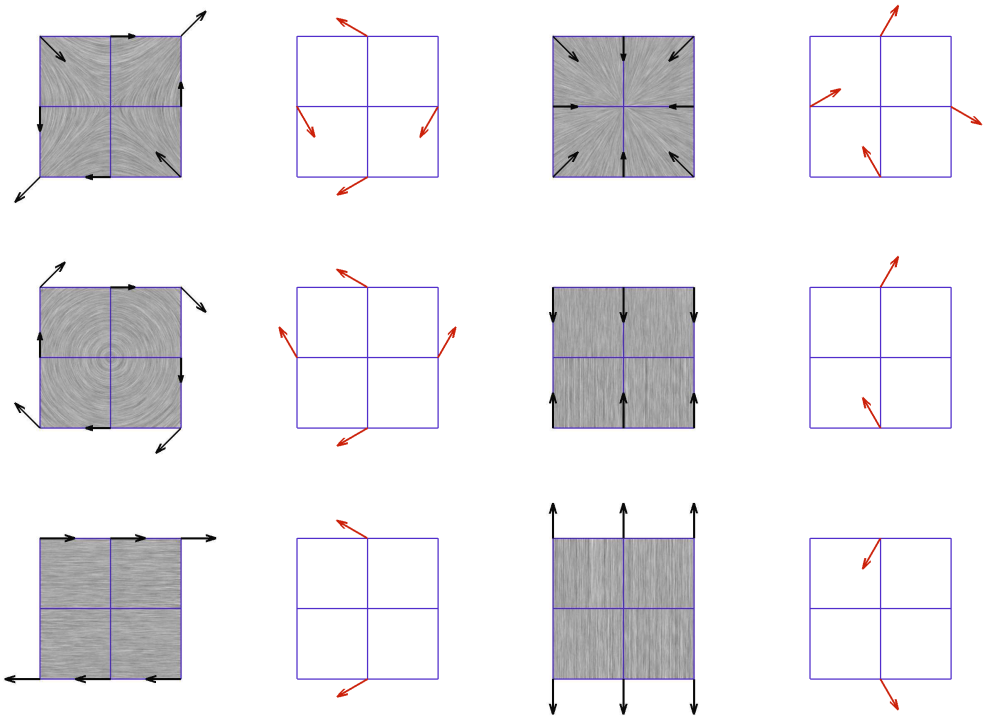

Figure 7.4: Various 2D patterns (black hedgehogs) and their DCFT (red hedgehogs). **Top left**: 3×3 saddle. **Top right**: 3×3 convergence. **Middle left**: 3×3 rotation. **Middle right**: 3×3 convergent line. **Bottom left**: 3×3 shear flow. **Bottom right**: 3×3 divergent line. The mean value of the DCFT is situated in the center of the fields. The results of the 2D DCFT looks a bit confusing at first since vectors are mapped to vectors.

tation of the vectors. However, this visual perception of the symmetry of vector fields is in contrast to the mathematical definition of symmetry and antisymmetry.

The interpretation of the DCFTs of these patterns is odd at first. A careful distinction between the direction of the waveforms, and the direction of the resulting vectors given by the component of the vector in which the waveform is present, has to be made. An example is the DCFT of the rotation. The rightmost vector and bivector correspond to the waveform with direction x. As the vector is $a\mathbf{e}_2 = ay$ and the bivector $-b\boldsymbol{i}\mathbf{e}_2 = -biy$, the wave is in the second component of the vector. Out of the complex number $(a-ib)$ corresponding to the second component of the multivectors, the amplitude and phase of the signal in this component of the vector can be determined. Note that for the determination of the properties of this waveform, the leftmost vector and bivector can be ignored due to symmetry properties. Another property of these patterns is that the rotation and the saddle line differ only in the phase of the waveform in direction x. This can also be seen in 2D (Figure 7.4). Remember that 2D vectors are interpreted as one complex signal. Thus, the symmetry properties of real signals are not valid. However, as the x-component corresponds to the real signal, it is conjugate symmetric, and the y-component or imaginary part is conjugate-antisymmetric (Theorem 3.3.11).

Regard the shear, divergence line and convergence line patterns. The correspondence between convergence and divergence line, which can be transformed into each other by negating every vector, can also be seen in frequency domain. Furthermore, the shear flow is only a phase-shifted copy of these flows. The same connection is true for the rotation and convergence patterns.

To summarize, rotation, convergence, divergence and saddle points in 2D and rotation in a single coordinate plane, convergence and divergence in 3D, were constructed by using (half) waveforms in the coordinates of the vectors. Some of these patterns and their discrete CFT's are shown in Figures 7.3 and 7.4. Some of their formulas can be found in the following Lemma 7.5.1, the rest can be defined analogous.

The patterns in Lemma 7.5.1 are all defined on $[-1,1]^2$ and $[-1,1]^3$ for 2D and 3D, respectively. The same patterns can be defined on $[-k,k]^2$ and $[-k,k]^3$ by replacing x, y and z with x/k, y/k and z/k, respectively, where the real number $2k+1$ is the overall size of the patterns in x,y, and z-directions. Subsequently, the patterns can been multiplied with a Gaussian window function (Definition 3.5.2) to reduce the influence of values by the distance to the center of the patterns.

Lemma 7.5.1

$$\textit{Rotation (3D)} \quad : f\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ -sin(2\pi z) \\ sin(2\pi y) \end{pmatrix},$$

$$\textit{Convergence (3D)} \quad : f\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -sin(2\pi x) \\ -sin(2\pi y) \\ -sin(2\pi z) \end{pmatrix},$$

$$\textit{Rotation (2D)} \quad : f\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} sin(2\pi y) \\ -sin(2\pi x) \end{pmatrix},$$

$$\textit{Convergence (2D)} \quad : f\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -sin(2\pi x) \\ -sin(2\pi y) \end{pmatrix}, \textit{ and}$$

$$\textit{Saddle point (2D)} \quad : f\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} sin(2\pi y) \\ sin(2\pi x) \end{pmatrix}.$$

7.6 Results

7.6.1 Clifford Fourier Transform

We chose a turbulent swirling jet entering a fluid at rest (OM06(2^k), Section 6.2) for first tests. Figure 7.1 shows the application of a fast discrete Clifford Fourier transform to the resampled field where the absolute values of the original and transformed fields are shown and the zeroth Clifford Fourier coefficient is drawn in the center of the image. A 2D vector field transforms into a 2D vector field since it forms one complex signal. A 3D vector field transforms into a multivector field where only the vector and bivector parts do not equal zero since vectors and bivectors form three complex signals.

The use of fast discrete CFT can also speed up the convolution computation of large data sets. Figure 7.5 shows the result of matching a 5×5 rotational pattern to the swirling jet data set. The vectors of the data set have been normalized to enhance smaller rotations. The pattern matching was computed using fast CFT and multiplication in the frequency domain. The results of the matching were color coded so that red denotes high positive similarities and corresponds to right handed rotation and blue denotes high negative similarity values and left handed rotations. The difference of computing the convolution in spatial or frequency

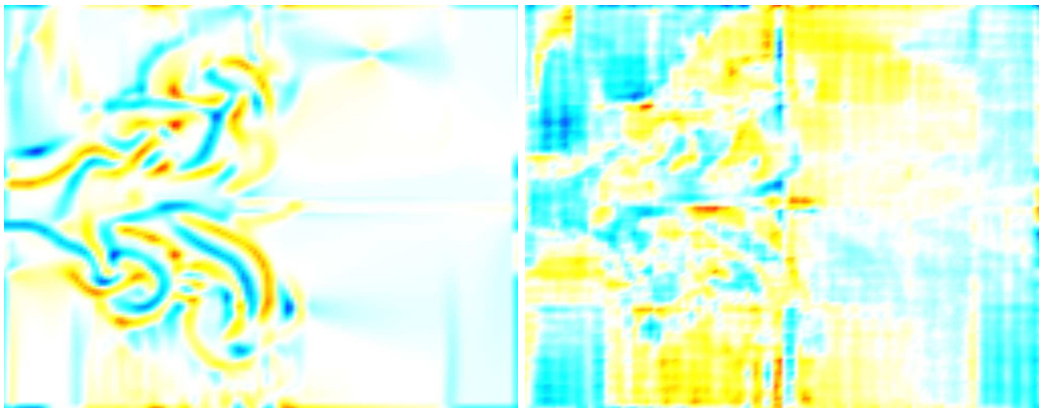

Figure 7.5: **Left**: Correlation of a swirling jet data set (OM06(2^k)) with a counter-clockwise 5×5 rotational pattern. The data set has been normalized to emphasize small structures. The image shows correlation computed in frequency domain with the result transformed back to spatial domain. Color coding of the scalar part is blue for high negative similarities (-0.94) and a left-handed rotation and red for high positive similarities (0.94) and a right-handed rotation. **Right**: Difference in the computation of the correlation operation in spatial domain using a periodic boundary condition. The colors are scaled from -1×10^{-13} (blue) to 1×10^{-13} (red).

domain (Figure 7.5), which is due to numerical errors, is at most 1×10^{-13} and is insignificant. Some representative timings of the convolution operation on a 1.6 GHz Intel Centrino are listed in Table 1. The timings for the 5×5 mask correspond to the convolution computations used for Figure 7.5.

computation	size of pattern	time (sec)
spatial domain	5×5	7
spatial domain	15×15	27
using fast CFT	5×5	5
using fast CFT	15×15	5

Table 7.1: Timings for the convolution computation on a 128×128 2D vector field, comparing direct computation and using fast discrete CFT and multiplication in frequency domain.

Symmetric Phase-Only Matching

A normalization of the data set is usually done to get a similarity which is independent of the magnitude of the velocities. This magnitude corresponds to the energy of the signal. Another technique to obtain a similarity measure independent of the energy of a signal, and with a much sharper peak at local extrema, is symmetric phase-only matching (Section 3.6.2). With the definition of the Clifford Fourier transform, this can be applied to vector fields within this framework (Figure 7.6). For simplicity, this approach was studied for a rotational symmetric pattern, here a rotation.

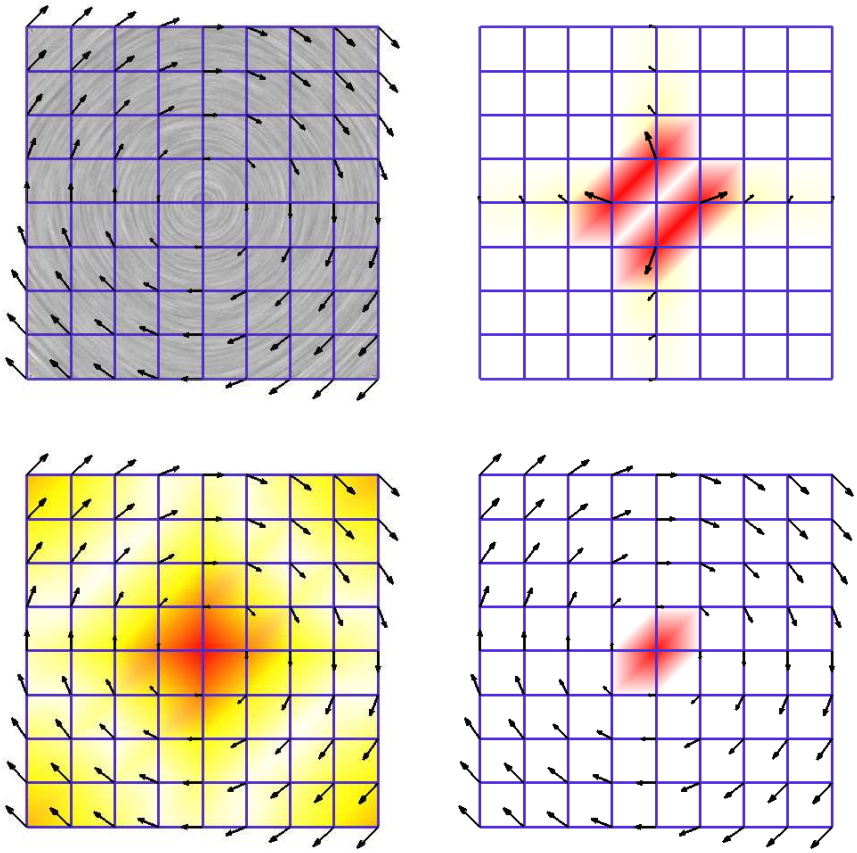

Figure 7.6: Symmetric phase-only matched filtering (SPOMF) results in a sharper peak. **Top left**: 9×9 clockwise rotation. Hedgehog, LIC and grid. **Top right**: DCFT of the same data. Hedgehog, color coding of the magnitude, and grid. **Bottom**: Color-coding of the similarities, hedgehog of the original data set, and grid. **Bottom left**: Result of auto-correlation. **Bottom right**: Result of SPOMF of the auto-correlation results in a sharper peak.

In 2D, the vectors form a complex signal. Thus, normalizing the 2D vectors of template and data set in frequency domain, multiplying the resulting signals, and computing the inverse Clifford Fourier transform, corresponds to symmetric phase-only matched filtering as the energy of the signal in frequency domain is normalized. The analogon in 3D is interpreting vector plus bivector as three complex signals as described in Section 5.4. These complex signals can then be normalized separately for symmetric phase-only matching in 3D.

However, the results of symmetric phase-only matching of vector fields are not convincing (Figure 7.7 and 7.8). The results resemble the original template matching with sharper peaks, but also with more noise. A connection of the results of SPOMF on the original data and template matching on a normalized data set can not be detected. This is due to the fact that magnitude changes are not only an amplitude property of the features, but also part of the waveforms in each of the components which is destroyed during the normalization.

Note also that this symmetric phase-only matching resulted in a misclassification, for example the classification of the saddle point as a rotation in Figure 7.7. Therefore, this approach was not used at all. However, some of these problems may be due to the fact that the global phases were compared. Thus, symmetric phase-only matched filtering was also investigated in combination with Gabor filters (Section 7.6.2), as these can be understood as a localized Fourier transform (Section 3.5) and thus allow an analysis of the local phase.

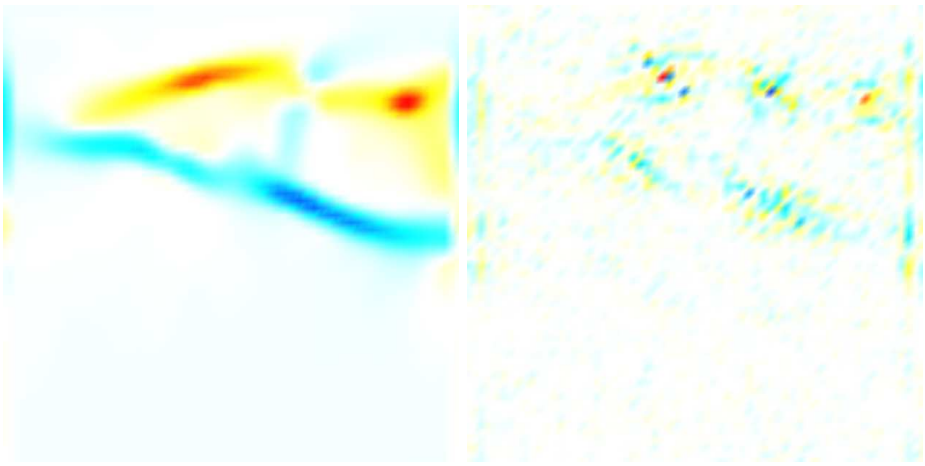

Figure 7.7: Normal and symmetric phase-only matched filtering (SPOMF) of the ICE section plane resampled to a 64×64 grid. Color coding of the similarities to a counter-clockwise 5×5 rotational mask, periodic boundary condition **Left**:Similarities of direct template matching. **Right**: Result of SPOMF.

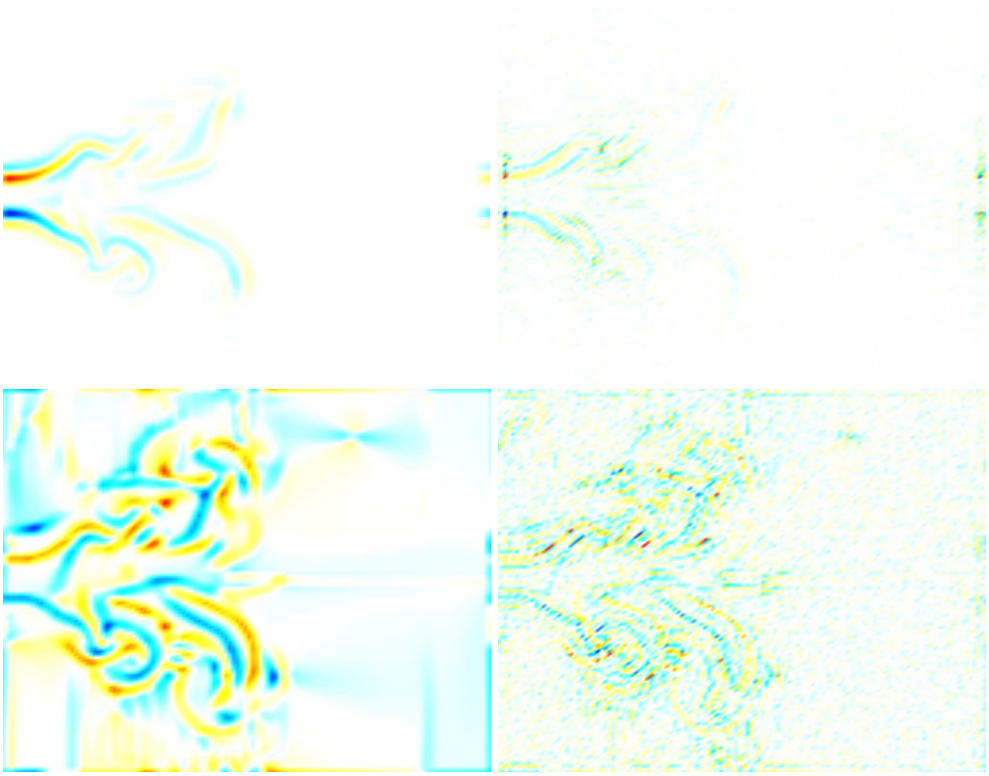

Figure 7.8: Normal and symmetric phase-only matched filtering (SPOMF) of the OM06(2^k) data. Color coding of the similarities to a counter-clockwise 5×5 rotational mask, periodic boundary condition. **Top left**: Similarities of direct template matching. **Top right**: Result of SPOMF. **Bottom left**:Similarities of direct template matching of the normalized data set. **Bottom right**: Result of SPOMF of the normalized data set.

7.6.2 Clifford Gabor Filter

Gabor filters can be applied to compute a Gabor expansion, they can be used for a wavelet decomposition, or directly as filters (Section 3.5). Clifford Gabor filters were used to study symmetric phase-only matched filtering of local phase (Figure 7.9 and 7.12). Again the rotation, which is a rotational symmetric pattern, was used to study this approach.

The results of SPOMF of local phase and direct template matching in spatial domain and matching via Clifford Gabor filters were compared (Figure 7.9). Note that the results of direct matching of the normalized data set in spatial domain and SPOMF of local phase results in similar structures. However, template matching of the normalized data set results in sharper features. Furthermore, the rotation in the upper left of the data set is not detected using SPOMF of local phase. Template

matching of the normalized data set via Gabor filters is similar to direct matching. However, it introduces some new features, for example the lines in the right of the image. Adaptive color coding of the magnitude of the velocities reveals a difference in the velocities there, which might be the reason for this feature. However, it can not be verified using other techniques like LIC. Thus, it is probably an artifact of the matching technique.

Note that template matching via Gabor filters, SPOMF or not, is computationally more expensive than direct template matching in spatial domain. An idea to reduce the effort comes from the investigation of basic patterns in frequency domain (Section 7.5). As these patterns consist of half-waves in the different components of the vector, the low frequencies should be enough to describe these patterns. Thus, only five values in 2D and seven values in 3D have to be compared. The Gaussian of the Gabor filter introduces new frequencies as a multiplication with a Gaussian in spatial domain equals a convolution with the Gaussian in frequency domain. Thus, this approach has to be studied carefully for use with Gabor filters as information of the Gabor coefficients is lost.

For a complete overview, the different template matching techniques were computed, without normalization, with normalization in spatial domain, and with SPOMF (Figure 7.10). The difference of matching the normalized data set using all coefficients or only the five low-frequency values were at most 10%, though the structures are the same. Matching using lesser coefficients resulted in higher similarity values. The highest difference is found when using SPOMF. Here, using only the low frequencies results in even fuzzier images. Note that SPOMF of the normalized data set makes no sense as both SPOMF and template matching of the normalized data set are similar. No new information is found but the results are blurred beyond recognition (Figure 7.11).

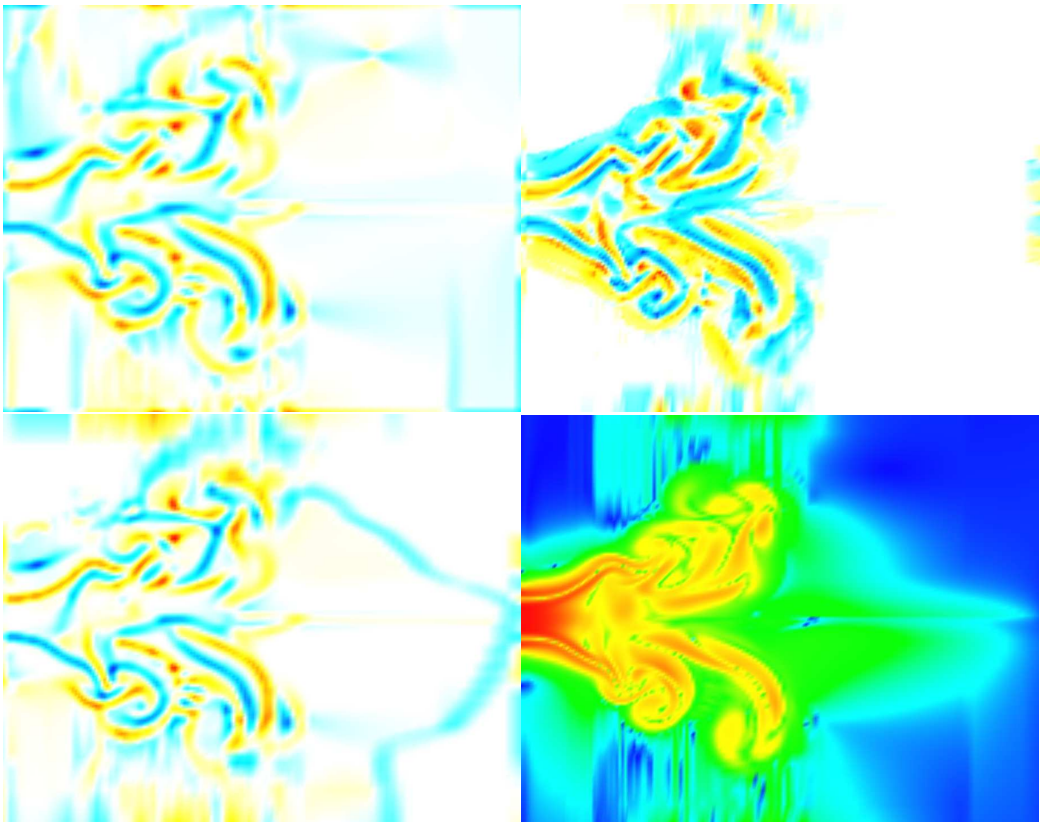

Figure 7.9: OM06 data set. Template matching using Gabor filters. **Top left**: Similarities of direct template matching in spatial domain using the normalized data set. Red denotes high positive similarities and a counter-clockwise rotation and blue high negative similarity values and a clockwise rotation. **Top right**: Result of SPOMF using Gabor filters. **Bottom left**: Template matching via Gabor domain using the normalized data set. Note the similarities to the image top left as well as the blue line in the right of the data set. **Bottom right**: Adaptive color coding from zero (blue) to high values (red) of the magnitude of the velocities reveals a possible reason for the blue line in the right of the bottom left image.

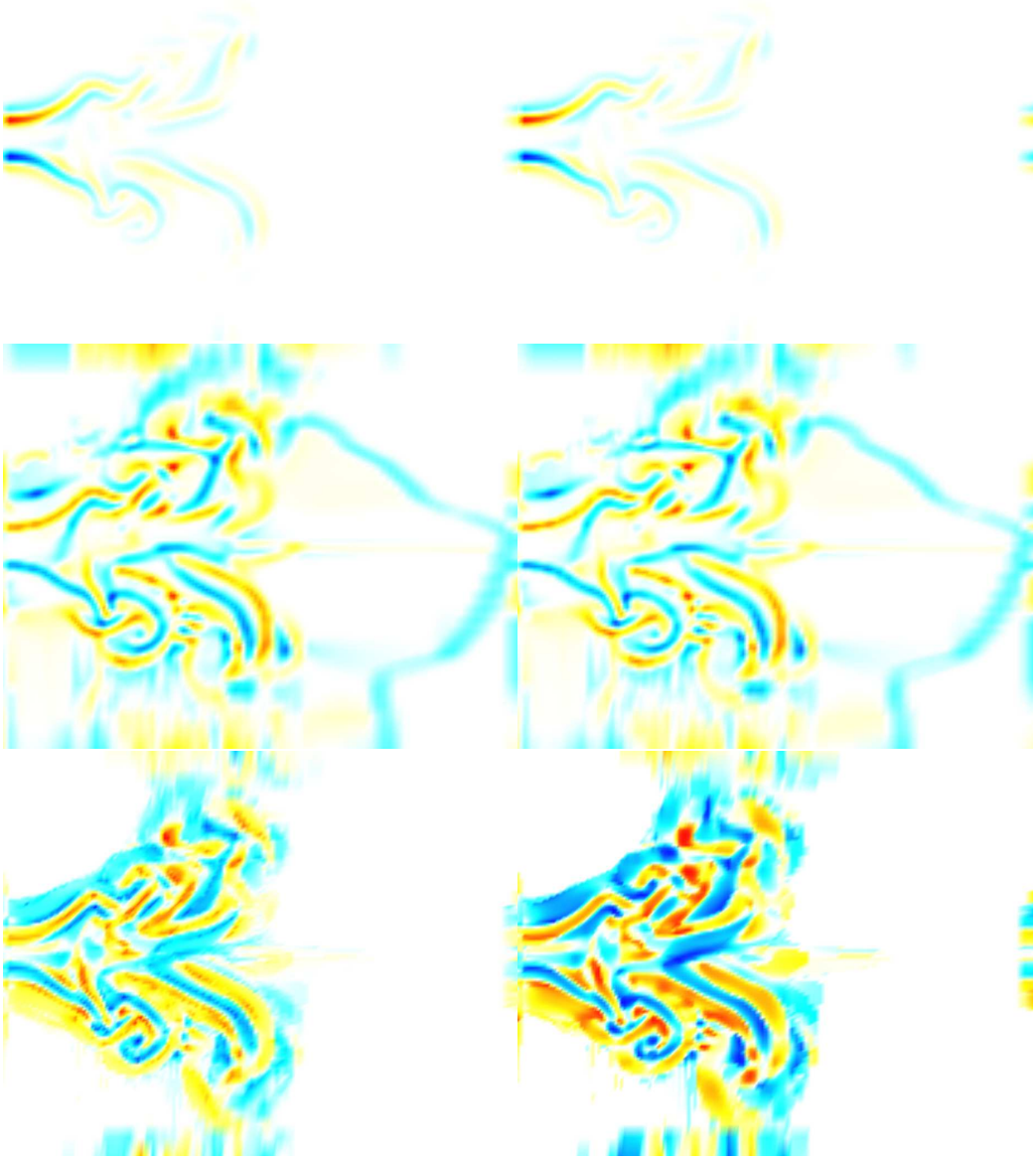

Figure 7.10: OM06 data set. Pattern matching with a 5×5 rotational template in Gabor domain using all coefficients (**left**) versus matching of low frequency values only (**right**). Color coding of the similarity values. Red denotes high positive similarities and a counter-clockwise rotation and blue high negative similarity values and a clockwise rotation. **Top**: No normalization. **Middle**: Normalization of the vectors in spatial domain. **Bottom**: Symmetric phase-only matching.

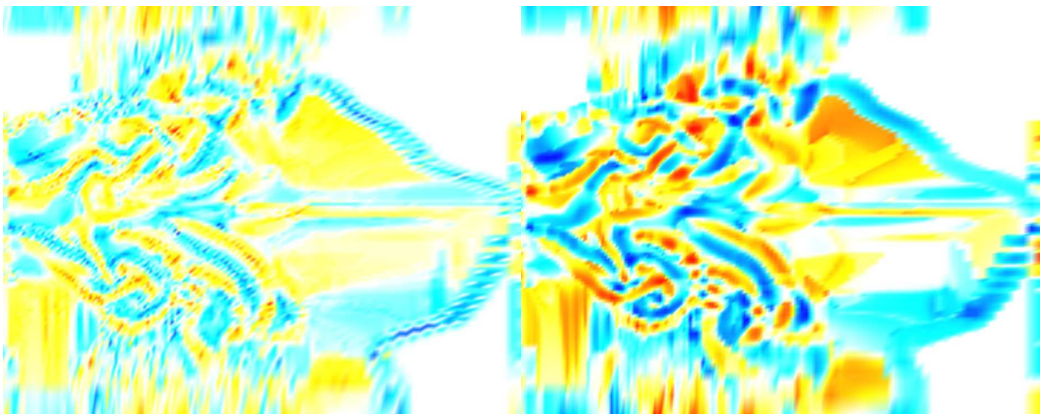

Figure 7.11: OM06 data set. Normalization in both spatial and frequency domain is not advisable. Color coding of the similarity values. Red denotes high positive similarities and a counter-clockwise rotation and blue high negative similarity values and a clockwise rotation. Pattern matching with a 5×5 rotational template in Gabor Domain. **Left**: Pattern matching. **Right**: Matching of the five low frequency values only.

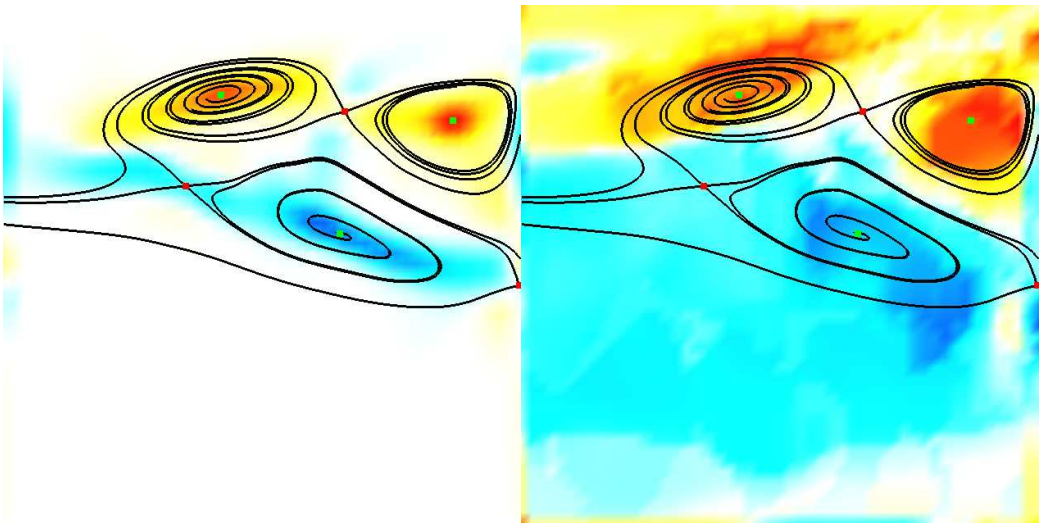

Figure 7.12: Section plane of ICE data set. Color coding of the similarity values and LIC. Red denotes high positive similarities and a counter-clockwise rotation and blue high negative similarity values and a clockwise rotation. **Left**: Template matching of the normalized data set in spatial domain. **Right**: SPOMF with Gabor filters.

Chapter 8

Analyzing Vector Fields using Template Matching

In this section, several issues of applying template matching to flow fields are examined and discussed. First of all, many flow fields are defined on irregular grids, or even on surfaces. In Section 8.1, techniques to transfer template matching to work directly on these data sets are discussed. The next question which will be discussed in this chapter is the issue of superposition, which plays a central role in signal processing. However, in flow analysis, it is often ignored so far. Superposition effects in flow fields, and the appropriateness of basic flow visualization and analysis tools, are examined in Section 8.2. Determining features parameters, and not only position and size, with the help of template matching in flow fields is another focus. This part of the work is motivated by the HART II data (Section 6.6). The vector fields measured there are defined on a uniform grid and noisy, thus convolution should be an ideal tool for the analysis of the described flow. Details on the analysis, which serve as an example of principal flow feature analysis based on template matching, are given in Section 8.3. Last but not least, the insight and techniques gained by this work are brought together, and an algorithm for feature based segmentation of flow fields is developed in Section 8.4.

8.1 Irregular Grids and Surfaces

Most often data sets from flow visualization are defined on irregular grids. The cell sizes differ greatly in size as they are very small in regions of interest and pretty large in regions where the flow is mostly homogeneous. This is illustrated in Figure 8.1. The Clifford convolution described so far only works on vector fields defined on uniform grids. Simple regridding of irregular grids results in a high number of grid points and oversampling in most parts of the vector field.

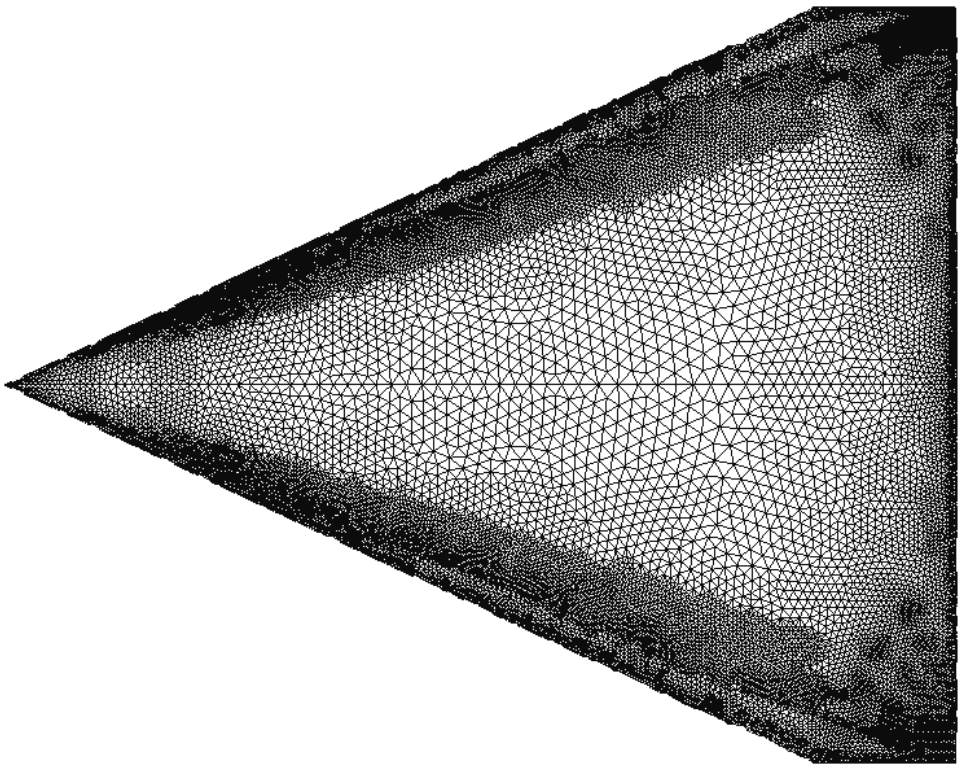

Figure 8.1: Grid of the flattened surface of the delta wing

However, this is most often the best approach due to computational efforts of subsequent algorithms which are applied to the data. In this section we discuss several approaches of template matching of irregular grid and surfaces in order to extend Clifford convolution and correlation to these data sets. The principal ideas of local resampling of field and template are investigated in Section 8.1.1. In Section 8.1.2, they are used to extend template matching to arbitrary surfaces. The results are given and discussed in Section 8.1.3.

8.1.1 Local Resampling

Scaling the Mask

The size of the cells of a flow field can differ greatly, and a unifying approach for scaling the template accordingly has to be determined. There are two principal approaches: Using one size at all matching positions, or scaling the template according to the neighboring cells. Note that in this subsection, the size of a template denotes its spatial extent and not the number of nodes in each dimension. As the templates are defined on uniform grids, the length of each of the edges is the

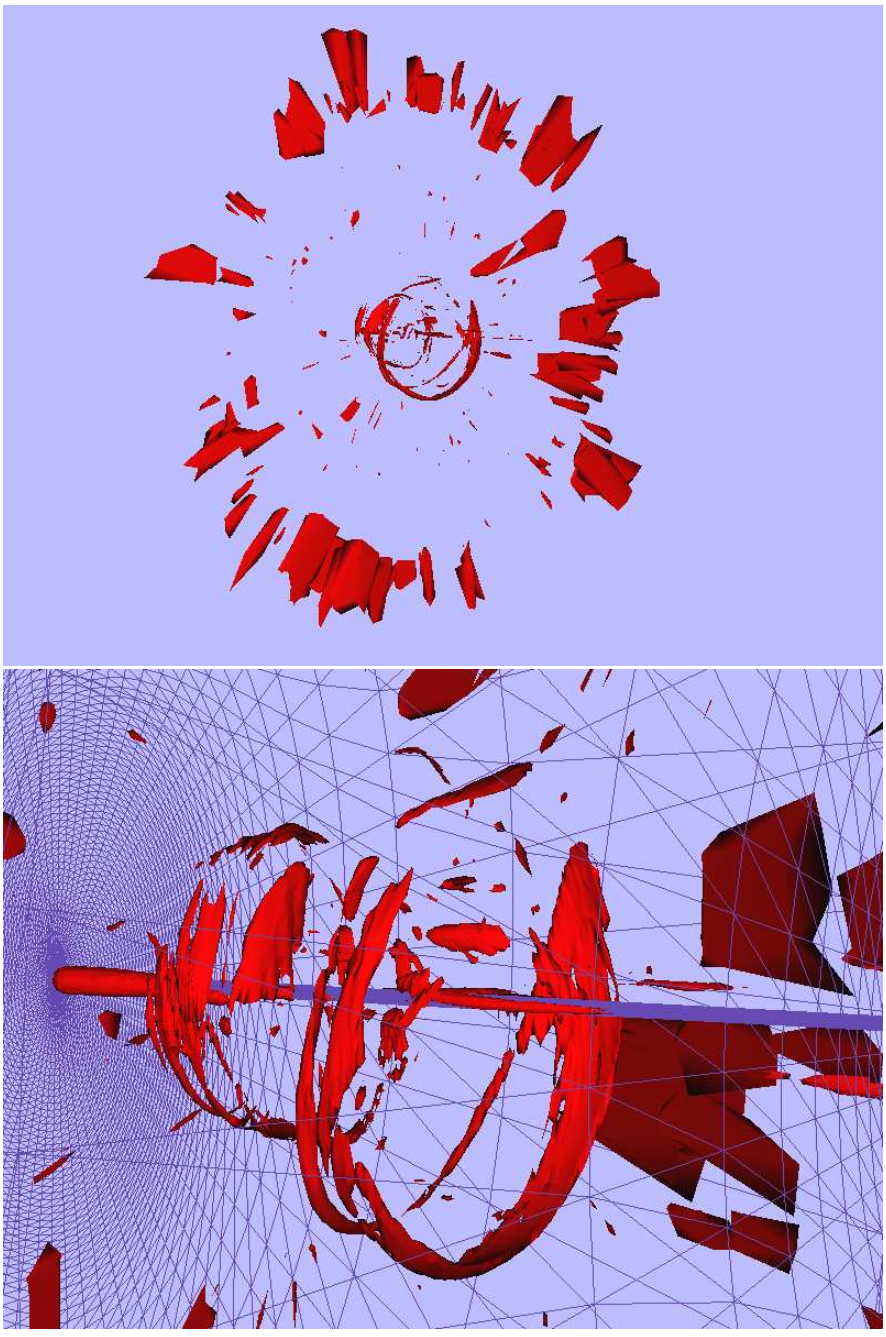

Figure 8.2: Swirling jet entering a fluid at rest, 3D simulation. Pattern matching with a $5 \times 5 \times 5$ rotational template. As the cells differ greatly in size, features of different scales are detected

property which is regarded here.

The first approach of scaling has the advantage that all detected features are of similar size. However, not all features will thus be detected, and the matching will be computed at positions where a corresponding feature can not be in the field due to sampling constraints according to the Nyquist theorem (Theorem 3.4.6). A multiscale approach is necessary for a thorough inspection of the data (Section 3.6.3).

Often, the cells are already scaled to the size of the feature that is expected. Then, it can be advantageous to use an adaptive template size. However, the size of the template has to be determined at every node of the grid of the flow field, adding computational effort. The template is usually scaled according to some measure of the cell sizes of the cells surrounding grid point P. Coping with all kinds of different cell types like tetrahedron, prism, cube and hexahedron, taking the longest edge connected to P can be used for the scaling. Let s be the length of this edge. Then the template defined on a uniform grid is scaled with s. That means that every edge of the template has length s now. Then the template can be used for the convolution at P. As the template is scaled differently at grid points with different maximal edge length s, features of different scales are found with the same template (Figure 8.2). Note that for uniform grids, this scaling corresponds to the usual convolution computation based only on the values at the grid nodes.

The techniques for Clifford convolution on irregular grids based on local resampling of field or template all use this scaling. Therefore we will not mention it every time and just assume that the template is scaled properly. Some vector field might require another scaling measure like an averaged length. All these measures will have some degenerated fields where they will not work well.

Local Resampling of the Field

The first idea is to place the template onto the field, the center of the template aligned with the grid point P to be convolved. Then the field is sampled on those points where the grid points of the template ”lie” on the field (Figure 8.3, left). These values are then multiplied with the corresponding values of the template.

Local Resampling of the Mask I

Most of the computational cost of the previous approach comes from point location for the sampling. This leads to the next idea. The field is placed onto the template. The center of the template is again aligned with the grid point P to be convolved. The vector field is cut off at the border of the template. Then the template is sampled on those points where the grid points of the vector field ”lie” on

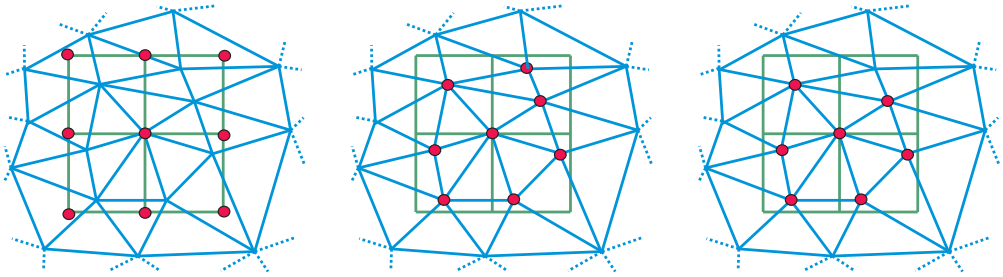

Figure 8.3: **Left**: Local resampling of the field. **Middle**: Local resampling of the template. **Right**: Local resampling of the template, only 1-neighborhood

the template (Figure 8.3, middle). The sampled values are then multiplied with the values of the vector field that "lie" on the same spot. This time the computation is even more expensive, as all grid points of the vector field in a bounding box have to be found.

Local Resampling of the Mask II

To avoid this overhead of the last approach, only the points in the n-neighborhood, where $(2n+1)^2$ or $(2n+1)^3$ is the size of the template, were used. The n-neighborhood of P is computed, that is the set of all grid points which are connected to P by n edges at most. All points of the n-neighborhood are projected on the template, the template is sampled there and the sampled values are multiplied with the values of the field. (Figure 8.3, right).

8.1.2 Template Matching on Surfaces

The algorithms described so far do not work for arbitrary surfaces. The 2D algorithm assumes a planar grid and the 3D algorithm only works for grids describing volumes. When the surface can be projected onto a planar grid like the delta wing in Figures 6.9 and 6.11, the 2D algorithm can be applied. It is also possible to project the surface only locally onto a planar grid, which is a lesser constraint.

For a 2D irregular grid, one idea was to use a local resampling of the field. For a surface, however, determining the positions for the convolutions is not straight forward as the template is planar and the surface can have a complicated form. Thus, the template has to be deformed to fit onto the surface. One approach to implement this is to use geodesics:

Definition 8.1.1 *A* **geodesic** *is the locally shortest path between two points.*

In an Euclidean space, a geodesic is always a straight line. On a surface, however, the form of the geodesic can be quite complex. Furthermore, it is not always

unambiguous, e.g. in the case of two antipodal points on a sphere there are many shortest connections.

To determine the sampling positions, the geodesics are started at the position where the convolution is to be computed. The direction and length of the geodesics is given by the vectors from the center of the template to the corresponding nodes. The end points of these geodesics determine the positions on the surface which are needed for the convolution computation (Figure 8.4).

Note that it is not advisable to transfer the other approaches of local resampling where the template is resampled at the grid nodes of the flow field. For these approaches, geodesics from the neighboring nodes of the surface to the position where the convolution is computed would need to be determined. This is not solvable other then by brute force as the path of the geodesics is not predictable enough.

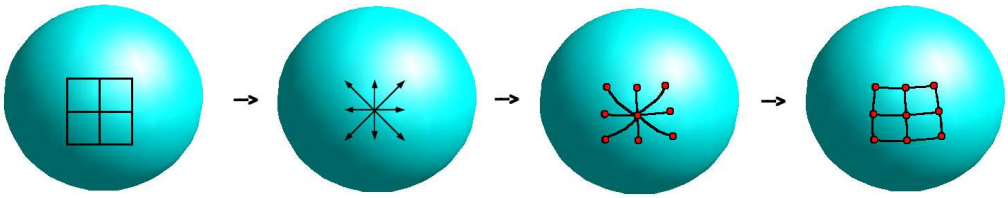

Figure 8.4: For matching on surfaces, the planar 2D templates have to be fitted to the surface, here a sphere. This can be done using geodesics: At the position where the convolution is to be computed, geodesics are started with direction towards the template nodes and with corresponding length. Thus, the positions for the computation of the convolution can be determined. **Left**: Sphere and planar template. **Middle left**: Direction and length of the geodesics are determined. **Middle right**: The actual geodesics determine the convolution positions on the sphere. **Right**: The template was fitted on the surface.

8.1.3 Results

Irregular Grids

For template matching on irregular grids, we have chosen two test data sets from real applications. The first data set is the OM06 (Section 6.2), and the second is the flattened surface of the delta wing (Section 6.4).

For the OM06 data set, which is defined on a regular grid, the results differ only slightly (Figure 8.5). The highest difference can be seen at the border, due to the different approaches of dealing with the missing values outside the field

(Section 3.6.1). For the computation of Figure 8.5, the missing values were assumed to be zero for simplicity of the computation. Using local resampling of the template automatically assumes ideal behavior of the flow outside the data set, as only the existing values influence the number of compared positions. Higher similarities at the border of the data set are the result. Note that using only the 1-neighborhood corresponds to matching with a 3×3 template, which is the reason for the difference to the results when using a 5×5 template.

Looking at Figure 8.6, the images of the results of resampling the template seem to be splotchy. This has a couple of reasons. First, the data set has some cells of size zero which distorts the results of the convolution. This is an extreme

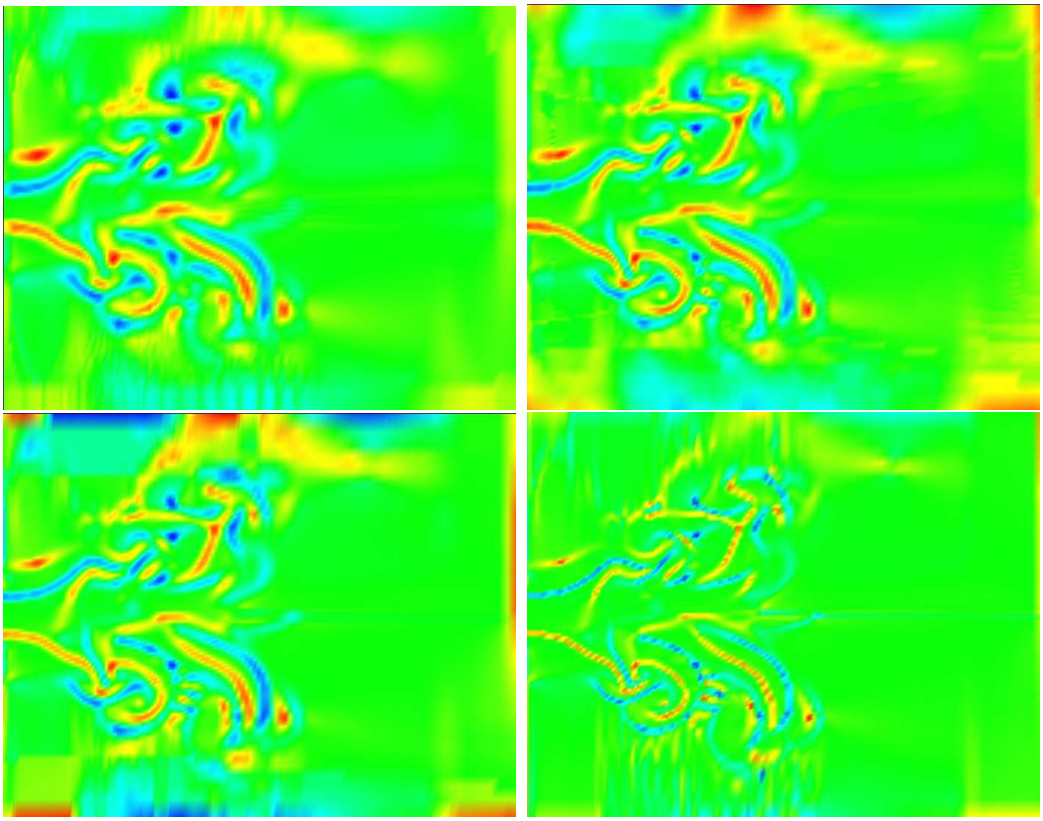

Figure 8.5: Pattern matching of a 2D vector field with a 5×5 rotation template. The grid is regular. The similarity values are normalized. Light areas corresponds to the highest similarity and to a righthanded rotation in the field and dark areas to a lefthanded rotation, as a righthanded rotation template is changed into a lefthanded by multiplication with -1. **Top left**: Local resampling of the field. **Top right**: Local resampling of the template. **Bottom left**: *n*-neighborhood. **Bottom right**: 1-neighborhood.

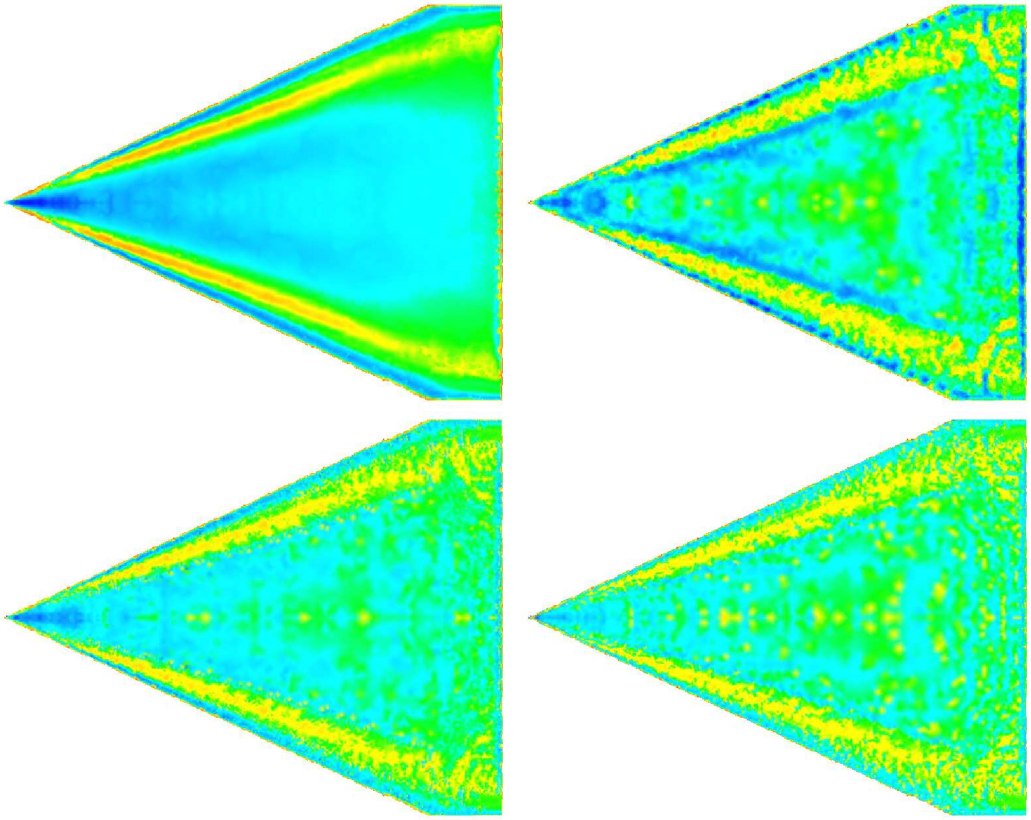

Figure 8.6: Regions of convergence (red) and divergence (blue) on the wing. Adaptive color coding of the results of pattern matching with a 5×5 convergence template. The grid is irregular. **Top left**: Local resampling of the field. **Top right**: Local resampling of the template. **Bottom left**: *n*-neighborhood. (**Bottom left**: 1-neighborhood.

example distortion which different cell sizes introduce into the convolution using *n*-neighborhood or 1-neighborhood. Cells which have large aspect ratios cause the same problem. Another reason for the differences is the resampling process which is based on interpolation of the grid points. Thus the first two approaches show different results although in the continuous case they would have exactly the same results.

In Table 8.1 and 8.2, timings of Clifford convolution and pattern matching on two different data sets are given. Local resampling of the template is the slowest approach for both data sets. The OM06 is defined on a regular grid, thus local resampling of the field is faster than local resampling of the template based on *n*-neighborhood as the point location is not as expensive as on irregular grids.

Resampling of the template is the slowest approach, and using the *n*-neighbor-

hood is more sensitive in terms of noise. Therefore, local resampling of the field is the best approach so far.

Table 8.1: Timings for computing one Clifford convolution on the OM06 and the flatwing data sets on a 1,3 Ghz computer

dataset	size of template	resampling of field	resampling of template	n-neighborhood
swirl. jet	3×3	4 s	75 s	5 s
(12524 p)	5×5	6 s	314 s	24 s
delta wing	3×3	33 s	88 s	10 s
(25800 p)	5×5	90 s	390 s	40 s

Table 8.2: Timings for complete template matching on the OM06 and the flatwing data sets on a 1,3 Ghz computer

dataset	size of template	resampling of field	resampling of template	n-neighborhood
swirl. jet	3×3	8 s	88 s	10 s
(12524 p)	5×5	16 s	352 s	40 s
delta wing	3×3	65 s	103 s	21 s
(25800 p)	5×5	183 s	441 s	66 s

Surfaces

Due to the computation of the geodesics, the computation of template matching on surfaces is slow. It took several hours to compute the template matching for the ICE front wagon (Section 6.5). The results are robust and as expected when comparing it to the LIC image (Figure 8.7). The region of divergence at the tip of the ICE train, where the wind hits the train, is clearly visible. Thus, template matching on arbitrary surfaces is possible though the computational effort is high.

Conclusion

Local resampling of either field or template is slow in comparison to the timings on a regular grid. This is due to the computation of the interpolation position, and the interpolation itself. The results are not better and can be quite noisy. Thus, global resampling of the field is preferred though it can greatly enlarge the number of positions and cells. Even then, template matching on the resampled grid will be faster, especially if the grid size in each dimension is 2^k for some,

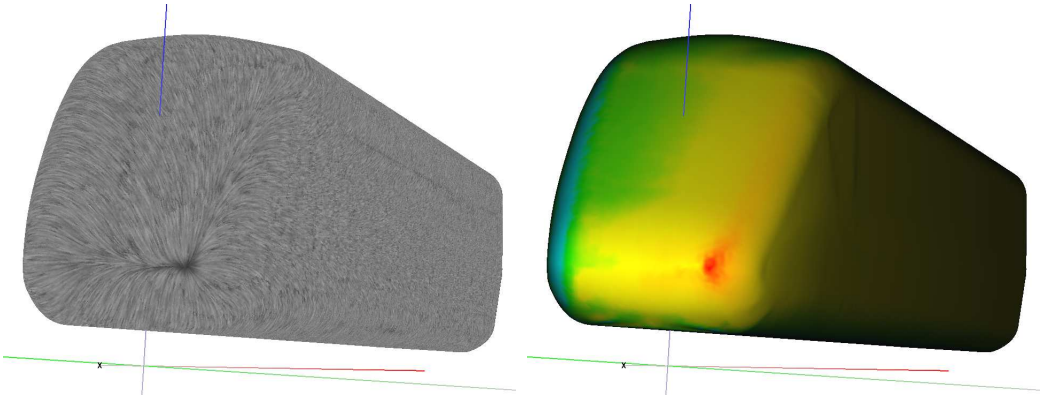

Figure 8.7: The surface of the front wagon of the ICE data set. **Left**: LIC. **Right**: Similarities of the normalized flow to a 3×3 divergent template. Color coding from dark blue (convergence) to dark red (divergence).

possibly different, $k \in \mathbb{N}$. Then, the fast Clifford Fourier transform can be used for acceleration of the convolution computations (Section 7.4.3).

The only case when global resampling of the field will not work at all is for flow fields defined on a surface. There, local resampling of the field using geodesics to determine the sampling positions has to be used though it is slow.

8.2 Flow Fields, LSI Filter and Superposition

For signal processing, linear, shift invariant systems (Section 3.2) play an important role as most systems can be described, or at least approximated, by them. The linearity property of these systems is also known as the superposition principle, stating that the result of two superposed signals equals the superposed results of the single signals. This also implies that a complex signal can be understood as a linear combination of several simpler signals (Figure 8.8).

Please note the difference between the linearity of LSI systems, which are mostly used to compute the similarity of a template to a flow field in this thesis, and the (non-) linearity of the description of the signals themselves, which in this case are flow fields. These are two quite different issues which do not influence each other.

The perception of systems as a superposition of less complex systems is fundamental in signal processing. However, many flow analysis and visualization techniques do not take this principle into consideration. Or, in other words, not all analysis and visualization techniques are LSI filters. Thus, several problems can occur, for example in measurements of wind tunnel experiments. There, dominant passing flow induced by the blower often hides vortices and other features so

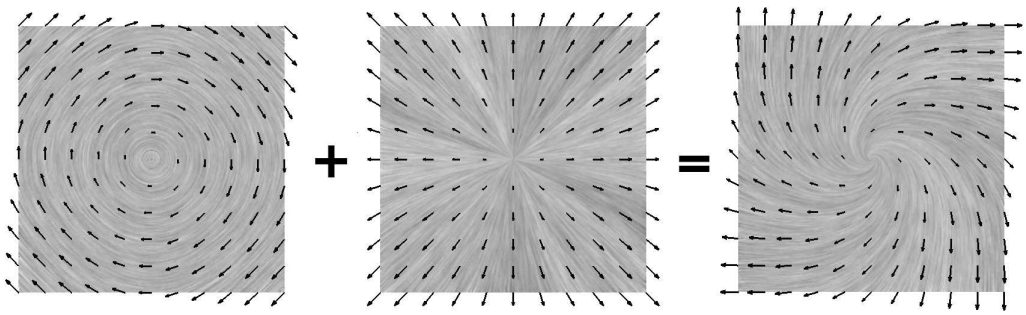

Figure 8.8: Linearity of features - the superposition principle: The flow on the right is the direct sum of the two different flows on the left.

a direct visualization may not reveal all features or even none at all [99] (Figure 8.9). Furthermore, the parameters of the detected features can be altered due to superposition effects.

In the vector fields measured in the HART II project (Section 6.6), for example, superposition phenomena are ubiquitous (Figure 8.9). Not only because of the dominant passing flow, but also because each crossing of a blade creates, among other things, a new vortex which is added to the flow created by previous blade crossings, the movement and the shape of the helicopter. Thus, for this application, a superposition perspective of the flow comes naturally. To understand the wake of the rotor blades, and to be able to create a model of it, all vortices and other features have to be detected and their parameters have to be determined. For accurate determination of the parameters, the superposition effects and their consequences for the accuracy of the analysis methods have to be studied.

The data sets from HART II (Section 6.6) were analyzed in the context of superposition phenomena. The data sets consist of 3-C PIV images, that is 3D vectors in a 2D image plane which have been measured in the simulation [30]. For simplicity, the third or out-of-plane component was neglected for the analysis here. First of all, global methods for obtaining Galilean invariant features (Section 2.4.2) were tried. Note that these methods try to minimize the effects of global superpositions for the detection of features.

The difference in removing the average and computing the localized flow (Section 2.4.5) can be seen in Figure 8.9. The wake area, clearly visible in the topology of the data set with average removed, crosses the border of the data set and is thus partly removed by the computation of the localized flow. However, with both methods, only one of the 4 vortices revealed by vorticity or pattern matching with a vortex mask is detected and visualized. Thus, not only global but also local superposition phenomena have to be considered in the analysis and visualization of this data set.

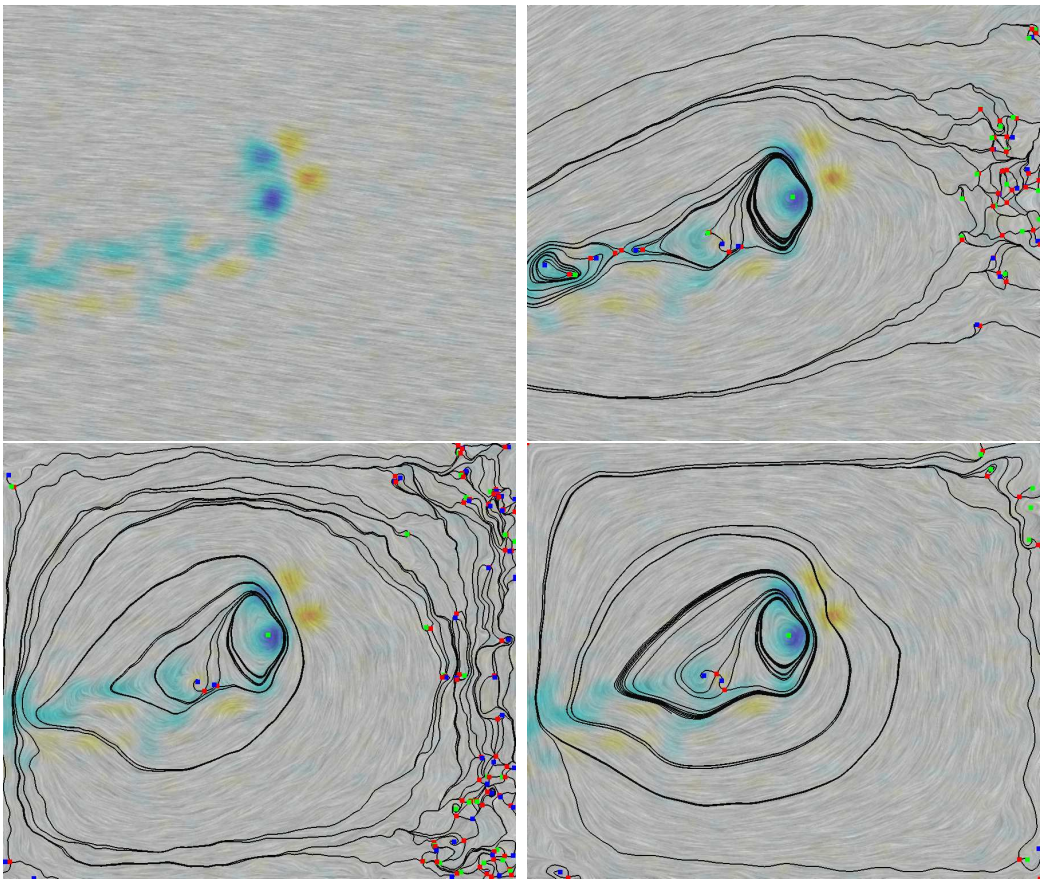

Figure 8.9: One vector field of the HART II test measurements. **All images**: Color coding of similarity to a 5×5 counter-clockwise rotational mask, dark blue: high negative similarity, red: high positive similarity. **Top left**: LIC of the original data set **Top right**: LIC and topology of the data set after removing the average. **Bottom left**: LIC and topology of the localized flow. **Bottom right**: LIC and topology of the flow after removing the harmonic component as computed by the hodge-decomposition. In all images, the streamlines do not reveal more than one of the four vortices indicated by vorticity.

In this section, the effects of local superposition in vector fields are examined. Furthermore, basic approaches for detecting and visualizing features in flow fields are investigated regarding their robustness towards these superposition effects. The properties are demonstrated using the Vatistas vortex [102] (Section 6.1.1), a popular vortex model in fluid dynamics, as an example.

The first two data sets that are used for the examination of the behavior of flow visualization and analysis techniques in the presence of superposition phenomena were computed by superposing two Vatistas vortices. For the first data set, the

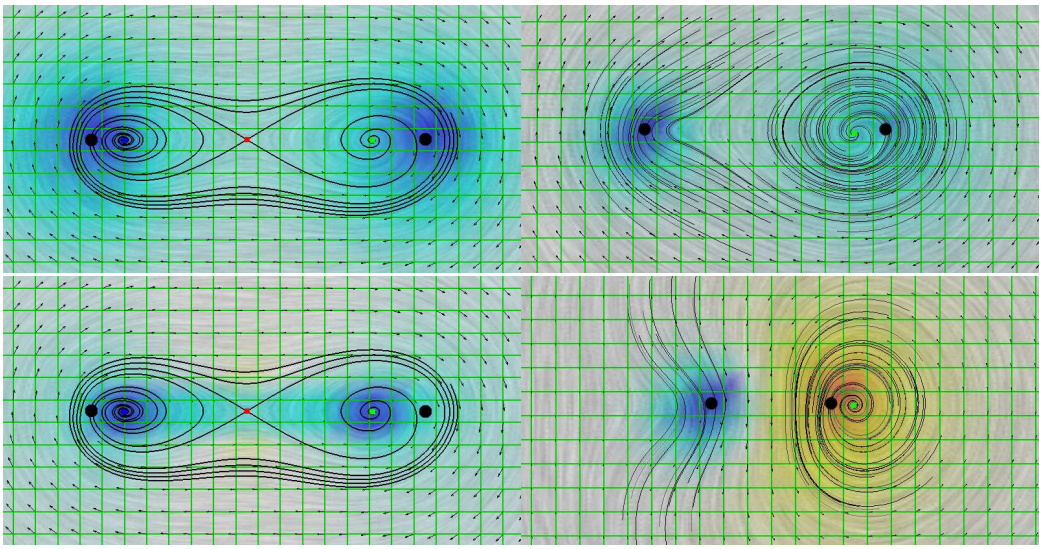

Figure 8.10: Superposition and interaction of two Vatistas vortices [102]. **All images**: The original vortex centers are displayed as black dots. Grid (green), hedgehogs (black arrows), color coding of an additional property from high negative values (blue) to high positive values (red). **Left**: Two Vatistas vortices [102] with both radius 5, radial velocity 1 and N=1. **Top left**: Color-coding of similarity to 3×3 rotational mask detects the true vortex centers. **Bottom left**: Normalizing the field and matching afterwards yields results more similar to the topological features. **Right**: The left vortex has radius 2 and radial velocity 1, the right one has radius 5 and radial velocity 3, both have N=1. **Top right**: Both vortices have the same rotation direction. **Bottom right**: The left vortex rotates clockwise and the right vortex counterclockwise. **Both right images**: Color-coding of vorticity. Topology only detects one center each (green dot), therefore some streamlines are added.

parameters of both vortices were equal, and the distance of the two vortex core centers was twice their vortex core radius (Figure 8.10, left). In the second example, the strength and radius of one vortex is smaller than the other. In direct visualizations like hedgehogs or streamlines, it is thus hidden in the flow of the other vortex and only bends the shape of the flow (Figure 8.10, right).

Topology, vorticity and pattern matching with a simple rotational mask were applied in order to detect the vortex centers. Topological and local streamline based features are good at detecting and describing features of the actual flow, but not for features hidden or moved by superposition. The resulting errors in the analysis of the feature parameters are high in the case of superposition, even resulting in not detecting a vortex at all (Figure 8.10, right). This is also due to the fact that streamlines are independent of the actual velocity of the flow at one

position (Section 2.1.1).

As vorticity and template matching are LSI systems, they clearly depict the vortices in the resulting scalar fields. However, even these results can be influenced by superposition. Assimilation effects, like two vortices with same parameters but different rotation direction which annihilate each other, effect the results of feature detection using any feature definition. Note that this is no counterexample for the linearity of the systems described by the templates as the results of matching the single vortices have same strength but different signs and thus annihilate each other, too.

Superposition effects should be considered for the visualization of vector fields. This perception of vector fields is necessary for some applications, for example the analysis of the HART II data (Section 6.6). Template matching is inherently suited for detection and analysis of features in the presence of superposition effects as computing a similarity based on convolution with a template is a linear, shift invariant system. For other applications, the actual streamlines may be of higher importance. There, streamline based approaches of visualizing the flow, like topological methods, are better at describing the flow than velocity and direction based feature models like the Vatistas vortex.

8.3 Feature Analysis

In this section, it is shown how to use template matching for feature detection and analysis in flow fields. It is quite obvious that the maximal similarity value will hint at the position of the corresponding feature. Detailed analysis, however, clearly goes beyond this straightforward approach.

This section is motivated by a cooperation with the DLR Braunschweig concerning the evaluation of the HART II data (Section 6.6). As the measurement noise is quite a challenge in these vector fields, and the data is aligned on a regular grid, convolution based feature analysis of this data is a promising approach. The vortices within the data are assumed to be perfectly circular. In practice, this is not always the case. Especially in the early stages of vortex creation, elliptical or band-shaped vortices have been observed. Nevertheless, for most of the measurement positions of the HART II PIV images, this assumption is warrantable.

First steps of an analysis of the flow fields consist in detecting all vortices and determining their parameters applying the Vatistas model (Section 6.1.1). The desired precision, however, is quite high. The positions of the vortices, for example, have to be determined with subpixel accuracy, that is, the exact position within a cell is needed. This presents a challenge to convolution based approaches, as these are usually evaluated at grid points. In this section, it is shown how subpixel accuracy can be obtained nonetheless.

It is not feasible to give a complete overview of the determination of all possible features and their parameters. The chosen examples in this section, however, demonstrate the principal application of template matching as well as some fine-tuning for a specific application. Due to the described task, the Vatistas vortex (Section 6.1.1) is again chosen as a feature example.

8.3.1 Similarity Value and Normalization Issues

As used by Heiberg et al. [42] (Section 5.2), the similarity of two vector-valued patterns is defined by the sum of the scalar products of their vectors:

$$s(\mathbf{x}) = \langle (\mathbf{h} * \mathbf{f})(\mathbf{x}) \rangle_0 \tag{8.1}$$

The other properties of the Clifford convolution are used to obtain rotation invariant matching, and to have a unified notation for the convolution of scalar and vector-valued data for further processing of the data.

The similarity value itself depends on the magnitude of both the patterns in the field and the template itself. Therefore, the obtained similarity values are usually scaled by the magnitude of the template pattern:

$$s(\mathbf{x}) = \frac{(\mathbf{h} * \mathbf{f})(\mathbf{x})}{\sum_{\mathbf{x'} \in \mathbf{h}} |\mathbf{h}(\mathbf{x'})|} \tag{8.2}$$

Often, this similarity is more influenced by the velocity of the vectors than their orientations. Then, it can be sensible to normalize the similarity by the velocities of the pattern in the data set as well. It can be achieved quite easily by normalizing the vector field beforehand. Matching on a normalized data set corresponds to a matching of the streamlines rather than the vectors themselves (Figure 8.10). It is often used to enhance weak features, or to obtain matching results more similar to the results gained by streamline-based analysis.

Normalization of the data set will work very well in some cases, but not when the features are hidden by other components of the flow (Section 8.2). Furthermore, normalization will shift the position of features (Figure 8.10), and can drastically change their size. When the velocity of vectors in the feature is important, for example the velocity profile of a Vatistas vortex (Section 6.1.1), normalization should not be used at all. Thus, for accurate determination of the position of features, the perception of the flow has to be determined beforehand, that is whether a superposition perception (Section 8.2) is used or not.

8.3.2 Determining Existence and Position

Vortices are quite challenging in flow visualization and analysis as no unified definition of a vortex is existent at the moment. To detect vortices with the use of

a convolution, a vector-valued template describing the vortex can be used. Templates that have been successfully used are pure rotation and a Rankine vortex (Section 6.1, e.g. Figure 6.3). An example of a similarity image thus computed can be found in Figure 8.11. Note that it is quite similar to smoothing of a vorticity field. The templates shown in Figure 6.3 are rotational symmetric. Therefore, it is enough to compute one convolution only in order to get a rotational invariant similarity. Furthermore, the templates have zero mean and thus they are not affected by any mean flow (Figure 8.11).

This far, the similarity values are only given at the nodes of the grid. To get subpixel accuracy, two different methods are proposed here. First of all, the center of similarity can be computed:

Definition 8.3.1 *Let $\boldsymbol{p}(i)$ denote the positions in Ω, and let $s(i)$ be the similarity value at $\boldsymbol{p}_i$. Then the center of similarity (CoS) of Ω is defined as*

$$CoS = \frac{\sum_i \boldsymbol{p}(i) * s(i)}{\sum i \boldsymbol{p}(i)}.$$

CoS result in exactly the same positions. To determine the neighborhood before the CoS computation automatically, the size of the vortex (without subpixel accuracy) should be approximated first. The region thus defined, or a multiple thereof, can be used before the computation of CoS.

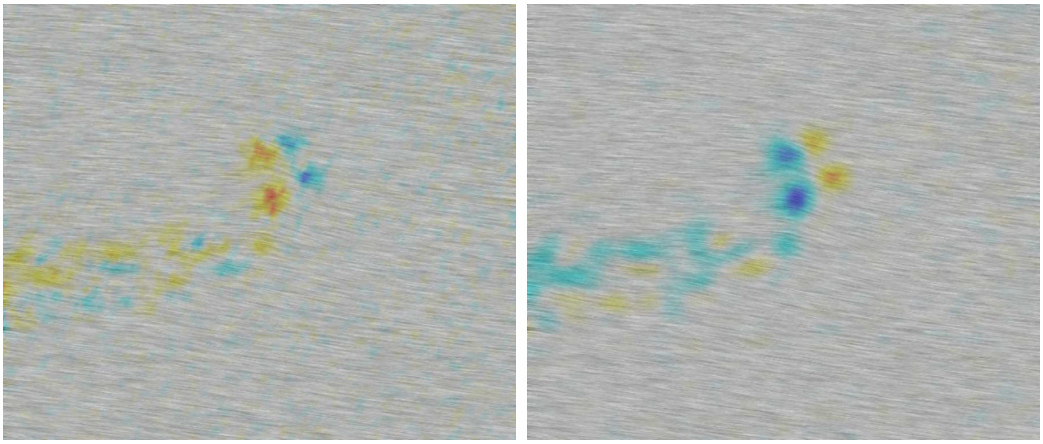

Figure 8.11: **Left**: LIC and Vorticity. **Right**: LIC and a similarity image. The similarity to a 5^2 rotational template is shown where blue denotes high similarity to a counter-clockwise rotation and red a high similarity to a clockwise rotation. Therefore the color is inverted in comparison to the vorticity image. Note that the scaling differs, between $\pm$ 4500 in the case of vorticity and ± 4 for the similarity image.

Another approach is to use a kind of bisection method. Note that linear interpolation, which is most often used in grids, can be computed via a convolution with a triangle filters and that (scalar) convolutions are commutative. That means that computing the similarity at an arbitrary point in a cell of the grid results in the same value as computing the similarity at the nodes and then interpolating the results. Thus, the maximal similarity can only be at a grid node. However, templates can be generated with the rotation center at every arbitrary point in space. So, the template is not moved for subpixel accuracy, but the vortex core or rotation center within the template. It has the disadvantage that the template has to be computed for every new position, but the similarity values of different subpixel positions are nearly equal within a few iterations. This bisection method needs less data than CoS. For simple test cases, if the neighborhood for the CoS computation is large enough and well behaved, the bisection method is less precise. As soon as the data becomes complex, the bisection method produces results which are more reasonable than the CoS approach.

Another issue which should not be neglected is the connection between scale and position, computing the position of a feature at different scales can yield quite different results. Thus, position and size can only be determined accurately when their detection is coupled. This will be discussed further in the next section.

8.3.3 Determining Size

When the grid node with (locally) maximal similarity has been determined, the size of the vortex can be found by using successively larger templates (Figure 8.12) at the position of the vortex center until the similarity values begin to drop off (Figure 8.13). Another method is to compute a scale space, for example a Gaussian pyramid (Section 3.6.3), and compute the similarities in each of the scales. Note that the vortices are assumed to be circular for the moment. For the analysis of the size of the vortex, the feature center in the templates can be either at a grid point or at the sub-pixel position. There is not much difference in the results of those templates due to the averaging effects of convolution.

When the vortices in the image plane have an non-circular shape, the detection of the vortex with a smaller template can give wrong results (see isolines in Figure 8.16). Then, the computation of the size should be done within a small region of the assumed vortex center. The region can be detected automatically by using all positions in the neighborhood which have similarity measures above a certain percentage of the maximal similarity value. There is a relationship between computational effort and a stable threshold, but the regions are not that large, and, for example, 33% gives stable results for all datasets tested. The position which gave the largest similarity in the computation of the size of the vortex is the grid node next to the true vortex center and thus both determine the position, though not with

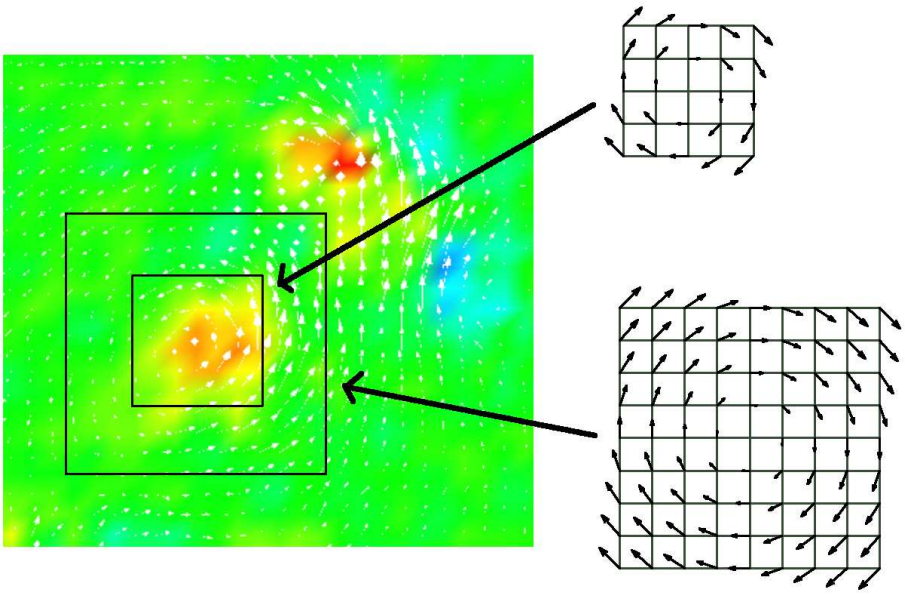

Figure 8.12: The size of a vortex can be determined by convolution with successively larger templates.

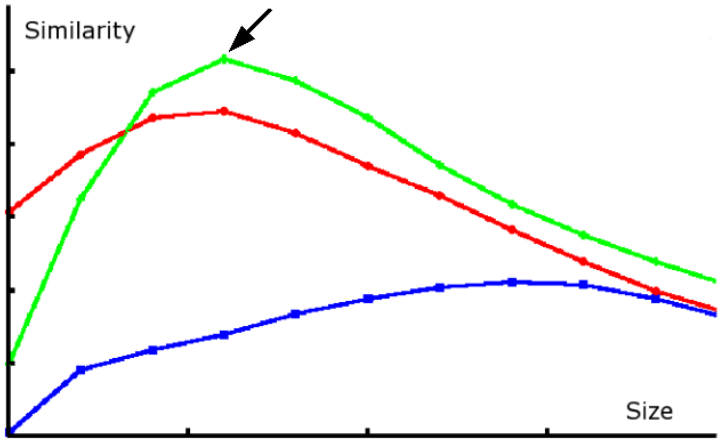

Figure 8.13: Similarity with a pure rotational template (y-axis) vs. template size (x-axis) for three positions. Shown are the position with maximal similarity to the 5^2 rotational template (red), the template with maximal similarity within the size computation (green), and the position with maximal template size within the region defined by a percentage threshold (blue). The maximal similarity of the green line (black arrow) determines position and size of the vortex.

subpixel accuracy, and the size of the vortex. It is not automatically the position where the maximal size was computed as can be seen in Figure 8.13. Note that it is not sensible to determine subpixel accuracy of position and size before the true vortex center has thus been identified. An example for the positions and sizes of the vortices in one data set of the HART II test can be found in Figure 8.14.

Again, the size is only determined with an accuracy of edge length so far. For subpixel accuracy, the trick from the last subsection has to be used again. Here, this means that the size of the template stays the same but the assumed core radius is changed. This way, an accuracy of one tenth of the edge length can be achieved. A pure rotational template is no longer optimal. The Rankine template

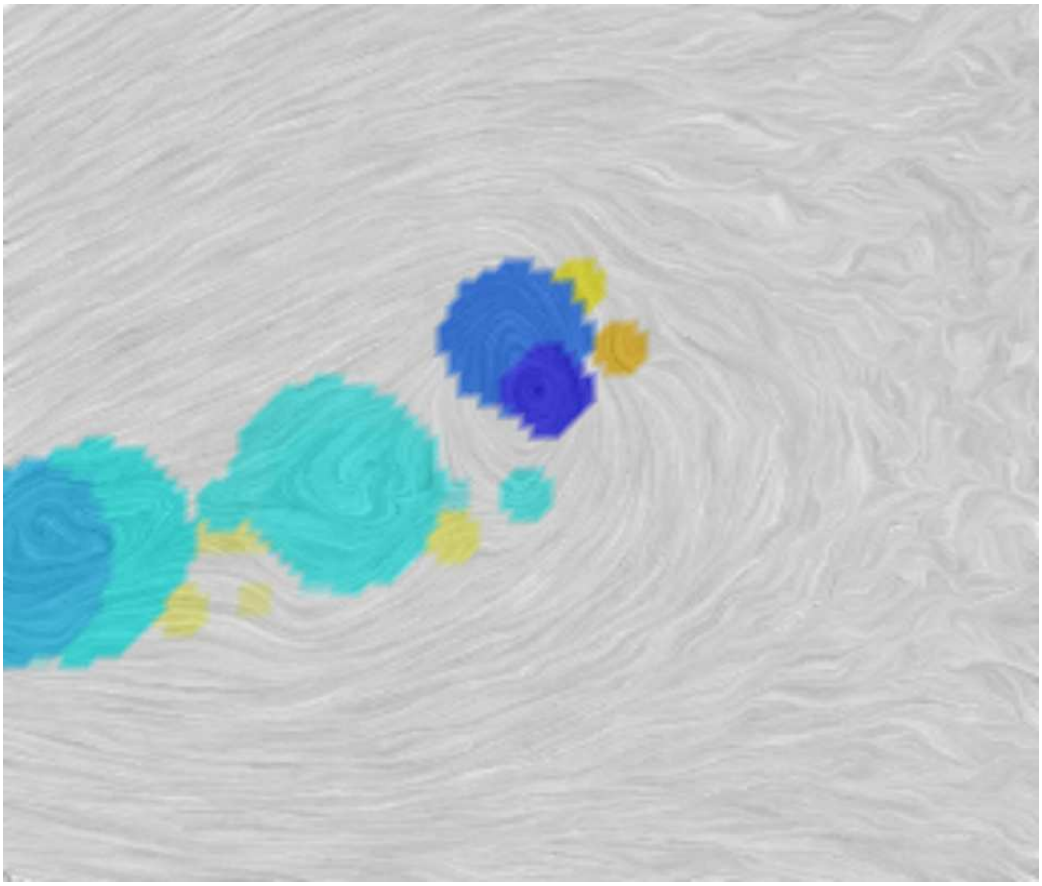

Figure 8.14: Visualizing position and size of all detected vortices with similarity ≥ 1. Blue denotes high similarity to a counter-clockwise rotation and red a high similarity to a clockwise rotation. In the wake area, the similarities are smaller. Note the differences in position in comparison to the LIC of the data set with average removed.

itself behaves not bad, but for subpixel accuracy, the velocity at the core radius should be enlarged to get a significant drop-off in the similarity values once the radius is bigger than the vortex core radius. Another possibility is to use only the vectors within half an edge or less of the core radius and zero the rest. With this template, the velocity at the core radius v_r can be read of directly from the similarity values. Because of the averaging, the result is a little bit smaller than the actual values, but not significantly.

8.3.4 Elliptical Shape and Orientation of Vortices

So far, the vortices have been assumed to be circular. In the case of the HART II data (Section 6.6) this is a valid assumption, but the circular vortices are not orthogonal to the image plane and so the profile of the vortex within the data set is elliptical (Figure 8.15). This introduces problems for the determination of the parameters of the vortex model, and thus an orientation correction has to be computed. In this section, an approach for determining the orientation of a vortex based on the elliptical shape in the image plane is given. In the process, the major axes of elliptical vortices are determined as well as their position in space. Out of these parameters and additional information coded in the out-of-plane component of the vectors, the orientation of the vectors can be determined.

Looking at Figure 8.15 and 8.16, the idea for the determination of the vortex direction can be explained quite well. First of all, it is assumed that position and size of the vortex are known, though not necessarily with subpixel accuracy. Then, a local coordinate system with origin in the vortex center is used. The direction of the major axis of the ellipse has to be aligned with the local x-axis. The rotation angle around the local out-of-plane axis (z-axis) can be computed quite easily

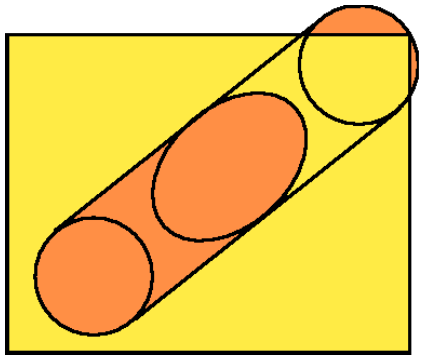

Figure 8.15: When the direction of the vortex is not orthogonal to the image plane, the vortex shape in the image plane will be elliptical.

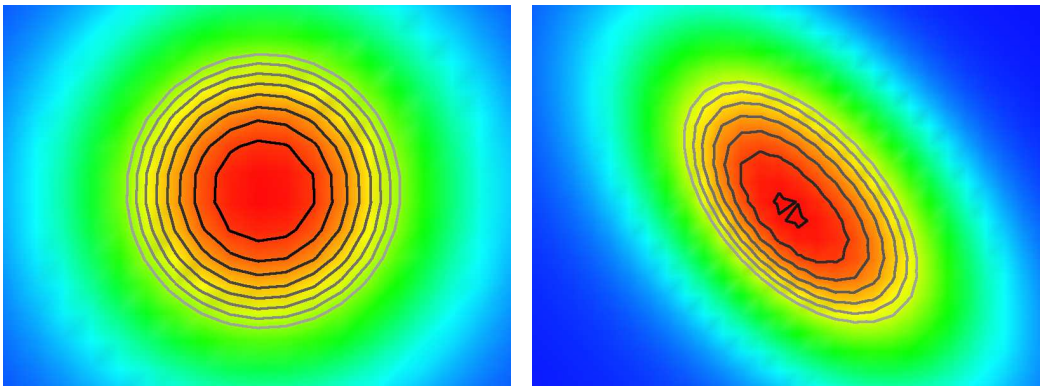

Figure 8.16: Looking at similarity images gained by convolving a data set with a rotational template or a Rankine vortex, the shape of the resulting image tells whether the vortex direction is orthogonal to the cutting or image plane (**left**) or not (**right**). Out of the shape of the ellipse in the right image, the direction of the vortex can be computed except for the sign of the second rotation angle. The color map of the similarity image, where red denotes high similarity and blue low similarity, is overlaid with isolines.

from the direction of the major axis. Then, the plane itself has to be tilted. The angle for this rotation is given by the arc cosine of the ratio of the size of the major and the second axis of the ellipse. The last step is to determine the direction of the tilt - positive or negative.

A tensor is optimal for describing ellipsoids. Thus, the orientation tensor is used here in combination with the convolution. Then, the major axis of the ellipse is given by the first eigenvector and the ratio of the two axes by the two eigenvalues. To determine the sign of the tilt, the vectors itself have to be used again. Here, the total average is subtracted and then the vectors are averaged within the vortex region. The sign of the y-component of the result gives the sign of the tilt.

One idea was to determine the shape of the ellipse out of isolines as shown in Figure 8.16. But because of the noise, the actual isolines in the HART II PIV images are quite ragged and thus transport the noise into the results. Therefore, elliptical templates are used which can also be interpreted as an integration of several isolines.

Several templates were tried for this computation. First of all, an elliptical Rankine vortex with ratio 1:2 was used in combination with the orientation tensor. The size of the larger axis was chosen as the diameter given by the approximated size of the vortex. The next idea was to use a circular Rankine vortex or a simple rotational template with half the approximated size. These resulted in a similarity image similar to Figure 8.16. On this scalar-valued similarity image, convolutions

with a scalar-valued, elliptical template, which has the same shape as the vector-valued elliptical template of the previous method, was performed and combined with the orientation tensor.

The second approach gave slightly better results as the form of the ellipse was not as restricted as for the vector-valued elliptic template. Furthermore, the scalar similarity image is pretty smooth which might be the second reason for the better behavior of the second approach. The accuracy of the angles was better than $\pm 5^\circ$. A better result for the second angle could be obtained by rotating the template into the direction of both axes of the ellipse and convolving again. Using the resulting similarity measure instead of the eigenvalues resulted in an improved accuracy of $\pm 2^\circ$, because the similarity encoded in the eigenvalues of the orientation tensor is often a little bit too small. The results degrade for tilt angles larger than $\pm 60^\circ$ due to the shape of the template. For the Hart II data all reported tilt angles are smaller than this, so this presented no disadvantage.

8.3.5 Results

Accuracy

Several ”gold standard” data sets have been generated for testing, for example the data shown in Figure 8.16. There, the vortex characteristics are known and can be compared to the results of our algorithms. In some of these test data, the vortex direction is orthogonal to the image plane, in others it is not. Furthermore, the effects of noise to the results of the algorithms can be studied quite well. The methods presented so far are all pretty robust in terms of noise due to the averaging effect of the convolution. Furthermore, as the templates all have zero mean, the methods are not affected by any mean flow.

Detecting the vortex core with different templates like a pure rotation or a Rankine vortex always gives essentially the same results. Concerning the size of a vortex, a pure rotational template or Rankine vortex results in diameters which are too large. Here, a modified Rankine vortex gives better results for first approximations as well as subpixel accuracy. The Rankine vortex was modified by enlarging the velocity at the core radius, or within a small neighborhood of the core radius, and setting the velocity outside of the core radius to zero. Another rotational template, where only the velocities at a ring centered at the core radius are larger then zero, behaves similar. To determine the vortex direction, a rotational template or Rankine vortex in combination with a scalar-valued elliptical template on the resulting similarity image gives the best results as described in Section 8.3.4. The methods give really good results even if values like position or size, which have to be determined beforehand for the computation of size or orientation, do not have subpixel accuracy.

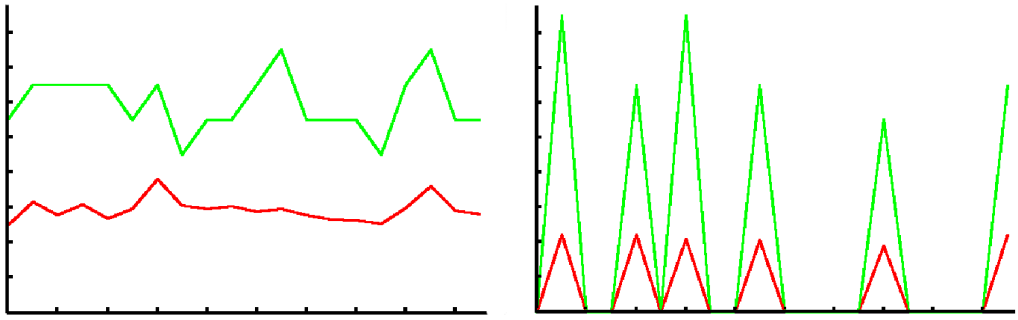

Figure 8.17: Similarity (red) and size (green) of the vortices of 20 images each of two measurement positions. **Left**: A young and well defined vortex. **Right**: The vortex is quite old and often the vortex can not be found at all. The x-axis gives the number of the image and the y-axis gives the similarity values in red and the diameter of the vortices in green. The results are computed without orientation correction or subpixel accuracy.

After determining the accuracy of the methods, these methods were applied to several of the 3-C PIV images from the HART II test. Some results can be seen in Figures 6.20, 8.11, 8.13, 8.14, and 8.17. As the methods are all robust in terms of noise and independent of mean flow, they behave well on these data sets. The vortex characteristics thus computed can now be used for visualization.

The techniques are applicable to other settings as well, especially the computation of feature position and size. One example is the computation of position and size of vortex systems above a 3D delta wing (Section 6.4). In Figure 8.18, isosurfaces of the similarities of this data set to a rotational pattern and the corresponding sizes can be seen.

Naturally, the orientation correction is only applicable to perfectly round vortices. For deformed vortical structures, the rest of the methods will only perform as well as for round vortices if the shape is determined beforehand and coded into the masks. In the future, the interaction between nearby vortices could be modeled directly with masks. Furthermore, the whole wake has to be identified and classified. Therefore, the vector fields have to be segmented into several vortices, the mean flow component, and the wake areas with a lot of vorticity where no vortices have formed yet.

Comparison to Streamline-based Techniques

For the Vatistas vortex, the vectors describing the flow are more important than the actual streamlines. The velocity of the flow is not only indispensable for the engineers whereas it is neglected in topological methods, but the projection of the

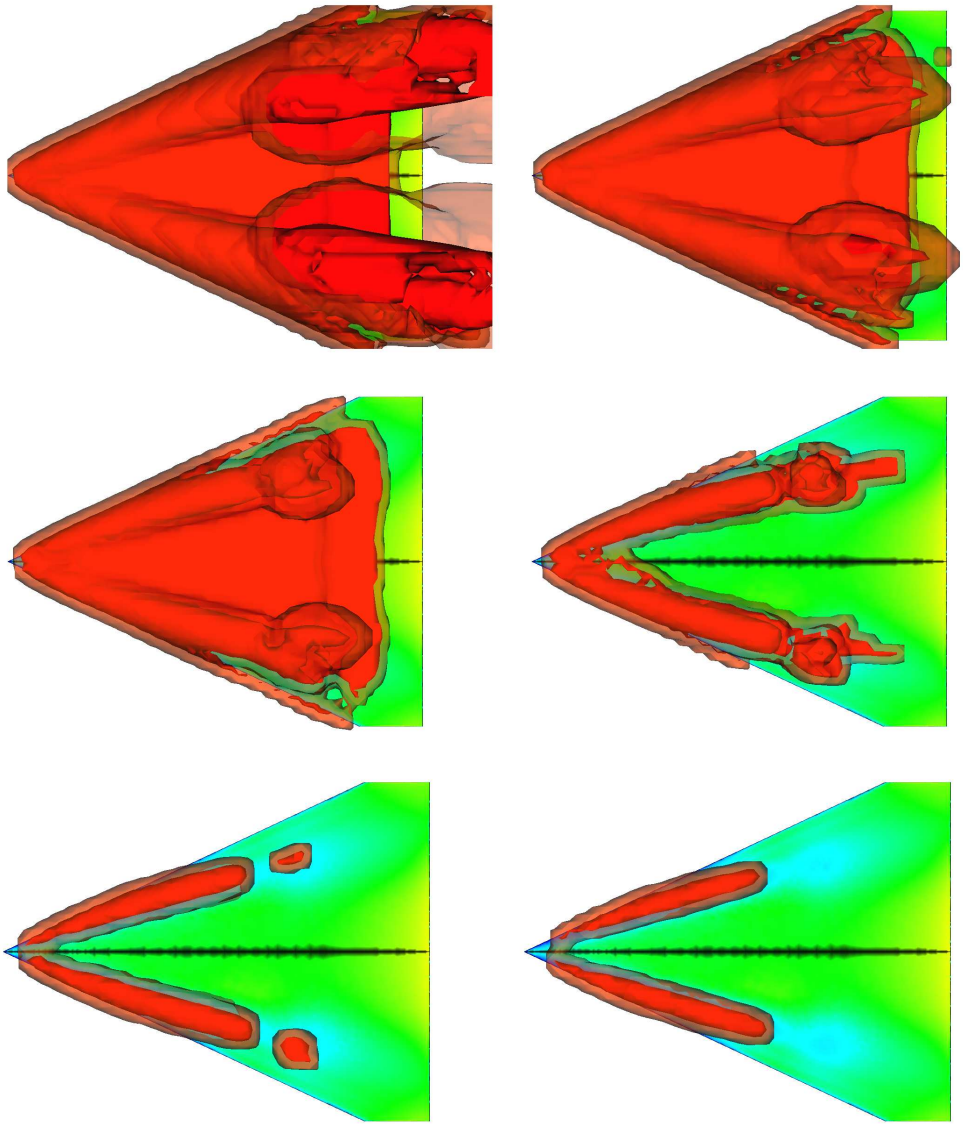

Figure 8.18: The 3D delta wing data set, resampled to a uniform grid. Displayed are the pressure on the wing and several isosurfaces concerning similarity to and corresponding size of a rotational template. From top left image to bottom right, the value of the displayed isosurface grows from 10 to 35 in steps of 5. The vortices at the front of the wing are small and strong. In the area of the vortex burst bubbles, large sizes and lesser similarity were computed.

vectors describing the actual flow onto the vectors describing the Vatistas vortex can be more important than the flow itself. Thus, the use of templates for the

analysis of these vortices is much better suited than streamline-based approaches like e.g. topological methods.

The Vatistas vortex is an abstraction of vortices. Though tunable by some parameters like vortex core radius and velocity distribution, this model assumes a perfectly circular vortex and thus only approximates real flow. The vortex is assumed to spread out infinitely, though the influence of the vortex will converge to zero with increasing distance to the center. This means that the region of significant influence will be larger than the actual vortex core and spread out over regions separated by topology.

Flow field topology itself is based on critical points, that is points where the velocity in the field is zero. As discussed in Section 8.2, it is quite sensitive to local and global superposition effects. As the center of the Vatistas vortex is a critical point in 2D, the position of the vortex can be easily determined using vector field topology. In contrast to other feature definitions based on vorticity or pattern matching, the center position is automatically determined with subpixel accuracy. This is a distinct advantage of topology as subpixel accuracy is often hard to obtain [30]. However, topology is sensitive to noise, therefore subpixel results are meaningless for noisy data as e.g. obtained by measurements and the only solution is smoothing the data.

The determination of the size of a vortex – or its vortex core region – is much more challenging. First of all, for the 2D Vatistas Model itself, no size can be determined using streamline-based approaches and topological methods as there are no separatrices and closed orbits are at all distances to the center. When two vortices interact, as in Figure 8.10, often a saddle point confines the regions of the two vortices. The separatrices as defined by the saddle point seem to enclose the vortex regions. However, in the case of a spiraling separatrix, no size can be determined numerically. For a closed orbit, the radius of the orbit or the area of the region can be used as a parameter describing the size of the vortex. Note that for the visualization itself, vector field topology will mostly characterize the size of a vortex as given by separatrices surrounding the vortex core.

Comparing these quantities with the Vatistas parameters again reveals the two different approaches taken by streamline-base and template based descriptions. Vector field topology segments the vector field into regions of same flow behavior, that is every particle within the vortex region as defined by topological methods will pass into the critical point or has emerged there. The vortex core radius as defined by the Vatistas vortex can be either larger or smaller than that region (Figure 8.10), as the topology depends entirely on the result of the superposition of features and not on the original properties of the Vatistas vortex (Section 8.2).

From this discussion it can be seen quite clearly that streamline-based and template-based models take quite different perspectives on the definition of interesting features. The first globally describes regions of same flow behavior in

relation to inflow and outflow regions while the other is more interested in local properties like velocity and vorticity, which can not be determined using topology based methods at all (Section 2.1.1). However, it is application dependent which of these two perspectives is more appropriate or beneficial.

8.4 Classification and Segmentation

Due to the amount of data nowadays, automatic detection, classification and visualization of features is necessary for a thorough inspection of flow data sets. In the last section, it was discussed how to apply template matching with vector valued templates successfully for the detection of features in flow fields. In this section, the approach is extended to automatically compute feature based segmentations of flow data sets. Different problems of the segmentation like the influence of thresholds, overlapping features, and classification errors are discussed. Subsequent visualizations of the segmentation display important structures of the flow and highlight the interesting features.

In Section 8.4.1, the advantages and disadvantages of previous approaches for segmentations of images and vector fields are discussed, and it becomes clear that a successful feature based segmentation of flow fields has not been developed so far. In Section 8.4.2, issues like overlapping features and classification errors are treated. The resulting segmentation algorithm is given in Section 8.4.4, and the results presented in Section 8.4.5.

8.4.1 Related Work

When the amplitude sufficiently characterizes the features, amplitude thresholding is useful (Section 3.6.1). The results can afterwards be used for component labeling, where the connectivity of pixels with their neighbors is examined in order to assign the pixel to objects. In vector fields, the amplitude or velocity of the vector field normally does not provide enough information for segmentation. On the other hand, amplitude thresholding of derived values like vorticity or similarity values from pattern matching can be quite useful for first analysis steps.

Another classical approach for segmentation in image processing is edge based (Section 3.6.1). The edges of objects are combined to form boundaries, which then determine the objects. In vector fields, this might work for shock waves, shear flow, and separation and attachment lines, as these can be interpreted as edges. However, segmentation should also classify these features. Furthermore, feature models of vortices, sinks, sources and saddles often have no real boundary, e.g. the Vatistas vortex [102], a vortex model used by engineers. There, the vortex is assumed to spread out infinitely though the influence of the vortex to the flow

is nearly zero outside a certain region around the vortex center. The vortex core center is given by the maximum of the velocity profile, but the transition from inside the vortex core to the outside is usually smooth. This behavior is also typical for other flow features. Therefore, edge based segmentation in flow fields will not yield satisfactory results.

Looking at the vectors within a vortex or a swirling motion (e.g. Figure 6.6), it becomes quite clear that region based approaches and clustering [38] will not work for a feature centered segmentation of vector fields, either. Computing the topology of a vector field yields a segmentation of the flow into regions of same flow behavior. However, the features can not be classified or quantified well. Furthermore, the resulting visualization is not centered on the features themselves (Figure 8.19 and 8.24).

Segmentation based on anisotropic diffusion of LIC images [21] results in a feature based segmentation. However, there is no criteria to stop the diffusion process, making the results not easily qualifiable. Furthermore, the problem of classification and quantification of the segmented features remains.

Pattern matching has been used for segmentation of images as well. There, similarity information of several different templates is computed at all pixels in the image, the features are classified according to the results and the image is then segmented into the regions of the features and background information. This approach is transfered to vector fields in this paper.

Before starting a segmentation, it has to be determined which features are of interest, and should form the template set. This includes specifying the type of feature like vortex, shear, sink, saddle or source (Figure 6.1) as well as the strength and size of the features and the scale at which they appear.

8.4.2 Challenges

The basic idea of segmenting a data set via pattern matching is quite easy: Determine all features present in the data set, compute their size and shape, and label all positions within the feature as belonging to it. All positions not labeled at all will be background, or not interesting at the moment. However, there are several challenges when trying to use this approach on vector fields.

Position, Size and Scale

Scale space considerations should not be neglected within the segmentation (Section 3.6.3). The size of the features, and thus the scales at which they appear and dissappear, can play an important role for the segmentation (Figure 8.19). The classification of the features, and thus the segmentation, can be done for each scale separately and be combined with scale space visualizations like Gaussian

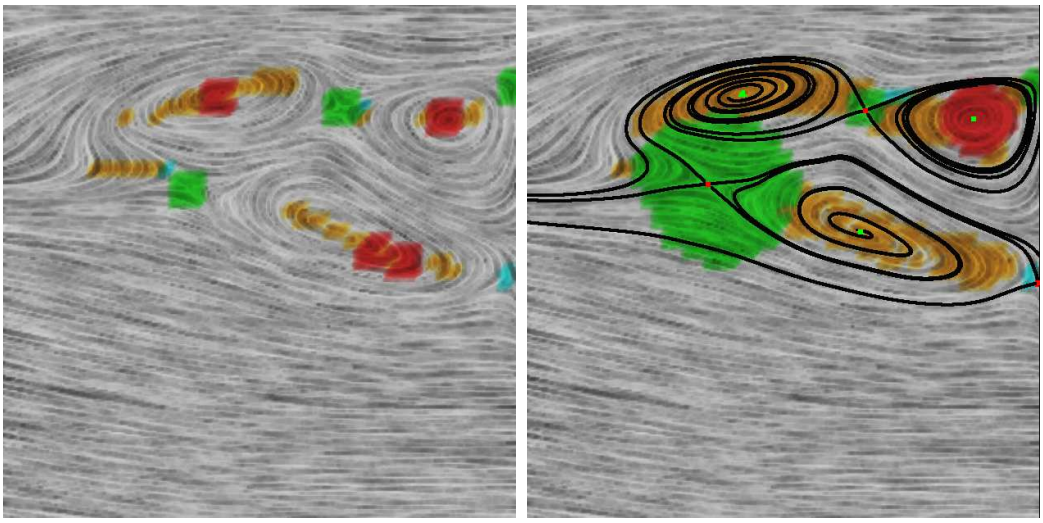

Figure 8.19: Vortices generated by an ICE train. Segmentation of a section plane through the flow (threshold=0.5), overlaid with LIC. The data set was normalized. Red: rotation, orange: shear flow, light blue: separation/attachment line, green: saddle point. **Left**: Only 3×3 templates were used to determine the line features shear flow and separation/attachment lines. **Right**: Templates growing from size 3×3 till no significant similarities were gained were used to detect the features. Topology added to the segmentation results. Note that the elliptical vortices are classified as shear flow when using larger templates.

pyramids (Section 3.6.3). Another possibility is to match each template using different template sizes. The resulting similarity images can then be combined into one scale invariant similarity image by using the maxima of the values at each position.

A scale-invariant similarity will also ensure a scale-invariant detection of the position of a feature as these are usually detected by local maxima of the magnitude of the similarity values. Note that the position of a feature would otherwise depend on the scale at which the feature is evaluated, e.g the center of a vortex with an elliptical shape will have different positions for different scales. Furthermore, the scale and template size resulting in the maximal similarity also gives size information of the feature (Section 8.3.2 and 8.3.3). Though subpixel-accuracy is possible, it was better to start with a template of size 3Δ for each direction where Δ is the (uniform) edge length. To continue with all uneven template sizes will result in a growth of the radius of Δ.

The computation of the convolutions with different template sizes can be accelerated by computing the convolutions in Fourier domain (Section 7.4.3). Another possibility is to start with a small template size and only compute similarities

with larger templates where the similarities with the smaller templates were above a certain threshold. The results depend on the choice of the threshold, and large features may be missed because they are hidden at the small scale. While this approach seems to be stable for e.g. rotational patterns, the similarities obtained for shear like patterns were often to small. However, this approach can give fast and useful results, especially for a first overall view of the data set.

The size of a feature in a flow field is usually hard to define. Point based features, like saddle points, usually have no size, and are visualized using only a very small area. However, for segmentation and visualization issues, larger areas are preferred as they are not easily overlooked. The region around a point or line based feature classifies this feature, and therefore can be regarded as belonging to it. This is also the size that is computed by the approach above.

The size of a line based feature, e.g. a 2D shear flow, can be the length of the line, or the scale at which they appear. Due to the smoothing effect of template matching, the region with similarity values above a threshold will be larger for larger features. Thus, segmenting and visualizing thresholded similarities is often good enough in this case.

For region based features, the size of a feature model can be infinite, as e.g. in the Vatistas vortex model [102]. However, the size of the vortex core region is an important information there. Again, this is exactly the size information given by scale invariant template matching.

Non-Orthogonal Feature Definitions

Orthogonality is also defined for vector fields:

Definition 8.4.1 *Two vector fields* $\boldsymbol{u}(\boldsymbol{x})$ *and* $\boldsymbol{v}(\boldsymbol{x})$ *are* **orthogonal** *if*

$$< \boldsymbol{u}, \boldsymbol{v} >= \int < \boldsymbol{u}(\boldsymbol{x}), \boldsymbol{v}(\boldsymbol{x}) > d\boldsymbol{x} = 0.$$

Note that $< \mathbf{u}, \mathbf{v} >=< \mathbf{u} \star \mathbf{v} >_0$. Using vector valued templates for the feature definition, orthogonality of features can also be determined by correlation of the different templates: A pair of templates is orthogonal if, and only if, their (rotation invariant) similarity is zero. When features are orthogonal to each other, their description and subsequent matching will not respond to the other features at all. One example of an orthogonal feature pair is pure rotation and pure divergence in 2D (Figure 8.20, top). Here, the classification of the features based on template matching and subsequent segmentation is obvious. But feature definitions can also overlap, for example a rotation and a shear flow both describe part of the phenomena of the other feature (Figure 8.20, bottom). Other pairs of features which describe similar phenomena are sinks and convergence lines, and

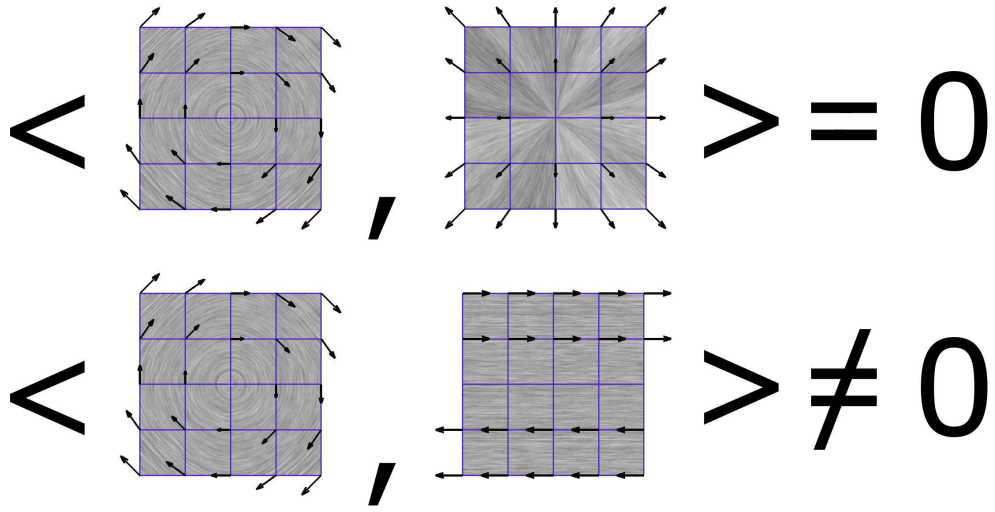

Figure 8.20: An example of orthogonal and non-orthogonal templates. **Top**: 5×5 rotation and divergence are orthogonal. **Bottom**: 5×5 rotation and shear are non-orthogonal.

sources and divergence lines (see e.g. Figure 7.4). In these cases, more than the similarity of one template to the flow can be non-zero at one position. Therefore the different similarity values have to be compared, and the feature classified according to the largest similarity value which has been computed. This also means that misclassification can take place. An elliptical rotation, for example, can be more similar to the shear flow description than to a circular rotation (Figure 8.19), though it surely is a vortex.

Overlapping Features

There are also other reasons why more than one template will respond to the flow at a position. Convolution with a template is a linear operation. The linearity property is also known as the superposition principle (Section 8.2). It means that complex flow can be analyzed by matching with several quite simple templates where the similarity values will indicate how much one template resembles the flow. It also indicates how much of the flow at this position is due to this particular feature model, and how much has to be described by other templates. An example is a swirling vortex which is a superposition of a perfectly circular rotation and a divergent flow (Figure 8.8). Another example is a wind tunnel, where the overall velocity of the air flow will usually hide smaller vortices. But this means that there may be more than one feature at a position, depending on whether this point of view is taken. In this case, segmentation is more challenging.

One solution is to classify the flow according to the most dominant feature at this position. But this is short-sighted as e.g. in the wind tunnel experiment only the overall flow will remain. Regarding the swirling vortex, one could classify it into a new class of swirling features. These swirling flows are then detected by using the already computed similarities to rotation and divergence templates. The percentage of these two similarities also gives a measure of the skewness of the swirl.

The issue of superposition and overlapping features leads to the proposition of computing a classification of the flow at every position into all of the features found there and the percentage in which they contribute to the flow. This is also a kind of segmentation, but one of the flow at one position into each of the interesting features. Note that when the data set has been normalized beforehand, the

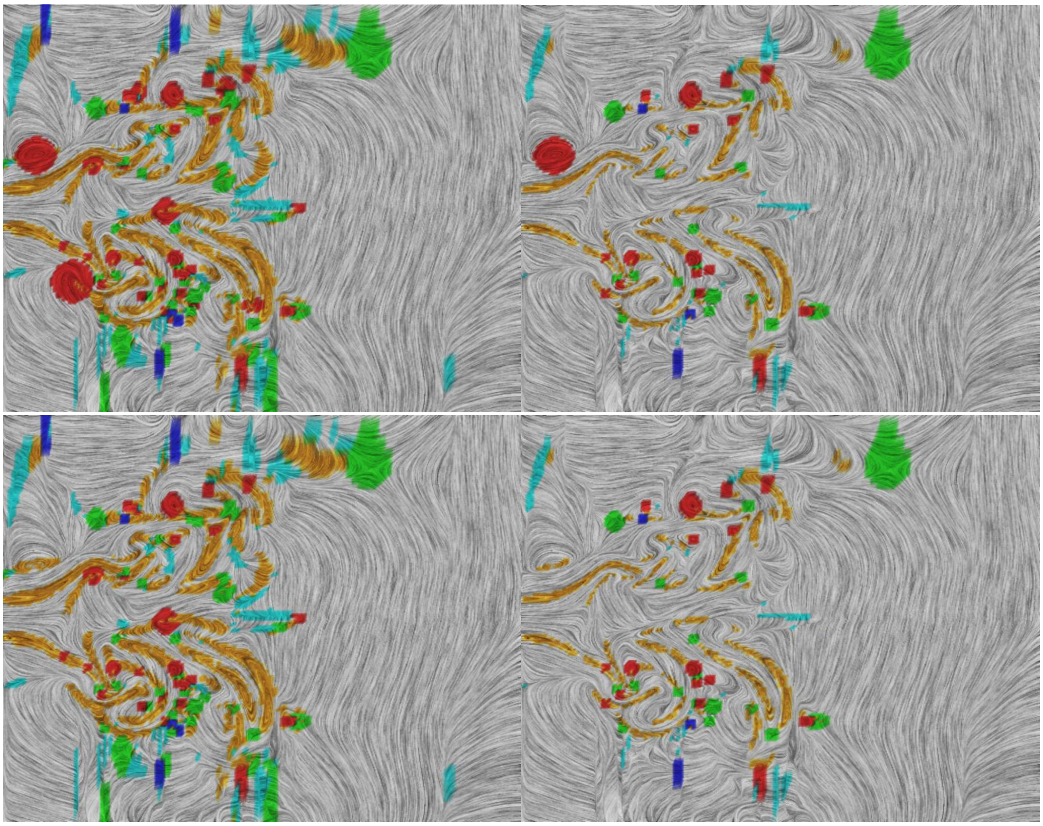

Figure 8.21: A swirling jet data set. Segmentation and LIC. Red: rotation, orange: shear flow, light blue: separation/attachment line, dark blue: sink/source, green:saddle. **Left**: Threshold of 0.5. **Right**: Threshold of 0.7. **Top**: Segmentation using the similarity values. **Bottom**: when the difference between shear and rotation was below 0.05, the flow was classified as shear rather than rotation.

similarity values equal this percentage. Otherwise, the similarity values have to be scaled by the energy of the flow to obtain this information.

Choice of Thresholds

The choice of suitable thresholds naturally depends on the properties of the data set to be analyzed. When the data set has been normalized, and no constant flow hides the features, a threshold of 0.5 was a robust start value for the examined data sets. When the data has not been normalized, another approach can be to determine all similarities larger than one. Generally, half the maximal computed similarity value is a valid choice. Note that the use of different thresholds will result in different segmentations (Figure 8.21).

8.4.3 2D, 3D and Time-Dependent Data Sets

The vector-valued convolution is defined for arbitrary nD data sets. However, the rotation invariant matching is only defined for 2D and 3D so far. Here, the use of the orientation tensor as defined by Heiberg [42] can be advantageous as it is defined for nD, too. However, it is assumed that the values change only in one direction, which is usually not the case, even for simple 2D data sets. As 2D and 3D data sets are most common, the rotation invariant matching approach using Clifford convolution poses no disadvantages.

For time-dependent data sets, each time slice can be segmented separately and the resulting regions can be traced over time (Figure 8.22). Tracing algorithms are well known from image processing [51, 52], and can be applied directly as the similarity data is usually scalar valued. As time dependent data is often visualized

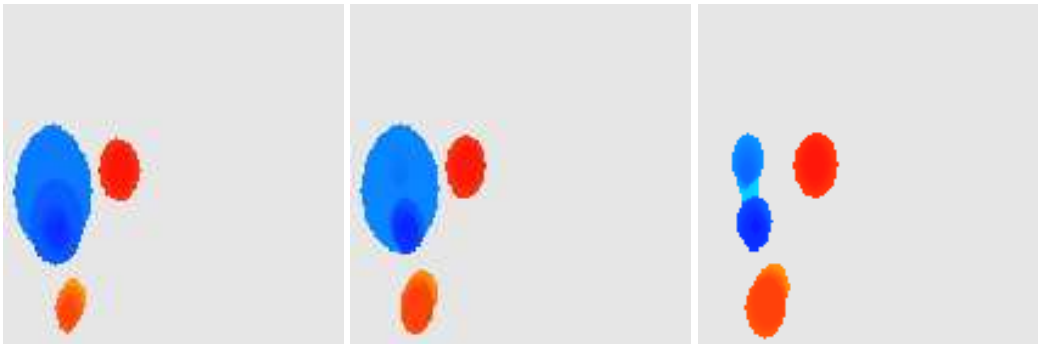

Figure 8.22: Segmentations of three successive time steps of a swirling jet data set data set (**from left to right**, threshold=0.5). The data has been normalized before matching. Color coding: clockwise rotations in red and counter-clockwise rotations in blue. From light to dark colors: increase in similarity values.

using movies, the tracing itself can often be left to the eye of the user (Figure 8.22). As, due to the averaging effect of the convolution, the similarity values are robust in terms of noise, visual discontinuities over different time steps are not expected.

8.4.4 The Algorithm

To summerize, the algorithm for the segmentation of vector fields based on Clifford correlation is computed as follows:

1. Determination of the features of interest and the parameters to be computed
2. Grouping of the features for segmentation and visualization (e.g. all rotations, all shear flows, etc.)
3. Determination of (non-)orthogonality of the feature definitions
4. Generation of (vector-valued) templates describing the features or feature groups
5. For each feature:
   (a) Template matching (rotational invariant) using different features sizes
   (b) Thresholding of the resulting similarity values
   (c) Computation of the maxima of the similarity values at each position, storing the corresponding feature sizes.
   (d) Local maxima of the results determine position and size of the features
   (e) Computation of possible other parameters of the features
6. For non-orthogonal features at a certain position, the features resulting in the smaller similarities are discarded
7. For orthogonal features at a certain position, the dominant feature is determined but all features are stored

8.4.5 Results

The first data set the segmentation was applied to is the section plane through vortices generated by an ICE train (Section 6.5, Figure 8.19). This data set was also included to remind of scale issues (Figure 8.19). Furthermore, it clearly shows the problem of classifying rotations and shear flow.

The next vector fields which were segmented are the OM06 and OM08 data sets (Section 6.2). The vector fields have been normalized before processing. The segmentation and subsequent visualization of the data sets highlights the features and thus guides the user through the data set (Figures 8.21, 8.23 and 8.24). Some vortices are found in the data, and the layers of opposite flow, divided by shear flow, are clearly visible. The classification of the most dominant feature was challenging as the similarity values of matching e.g. shear and rotational templates at one position in the data set sometimes differed only by 0.0001, though both similarities were above the threshold. This usually indicates an elliptical vortex or a swirling motion generated by shear flow.

As an example of a time-dependent data set, another swirling jet was used (Section 6.2). The data was normalized and segmented into clockwise and counterclockwise rotations, and background. Three successive time steps are shown in Figure 8.22. In these timesteps, the split of a vortex into two new ones can be observed. The pairing of two vortices each of different rotation orientation, and the path of the moving vortices, can be easily studied using segmentation throughout all timesteps. Only a part of the data sets of these time steps is shown as the rest was classified as background.

An interesting data set is the gas furnace chamber (Section 6.3). For computational issues, the data was resampled onto a uniform grid with dimensions $126 \times 65 \times 57$. The structures in the flow can easily be identified by the segmentation (Figure 8.25). Note that the shear flow at the front bottom (in yellow) is a misclassification, it is actually an elliptical vortex. This is one reason why the vortex core itself (in red) extends into this area. Note also the saddle line behind these vortices (in green), it is clearly visible in the bottom image.

Additional information of the gas furnace chamber can be gained by displaying an isosurface of the velocity of the original data set (Figure 8.25, blue isosurface). Using this isosurface, the gas and air inflow streams are clearly visible. Note the vortices besides them, and how they follow the shape of adjacent air streams.

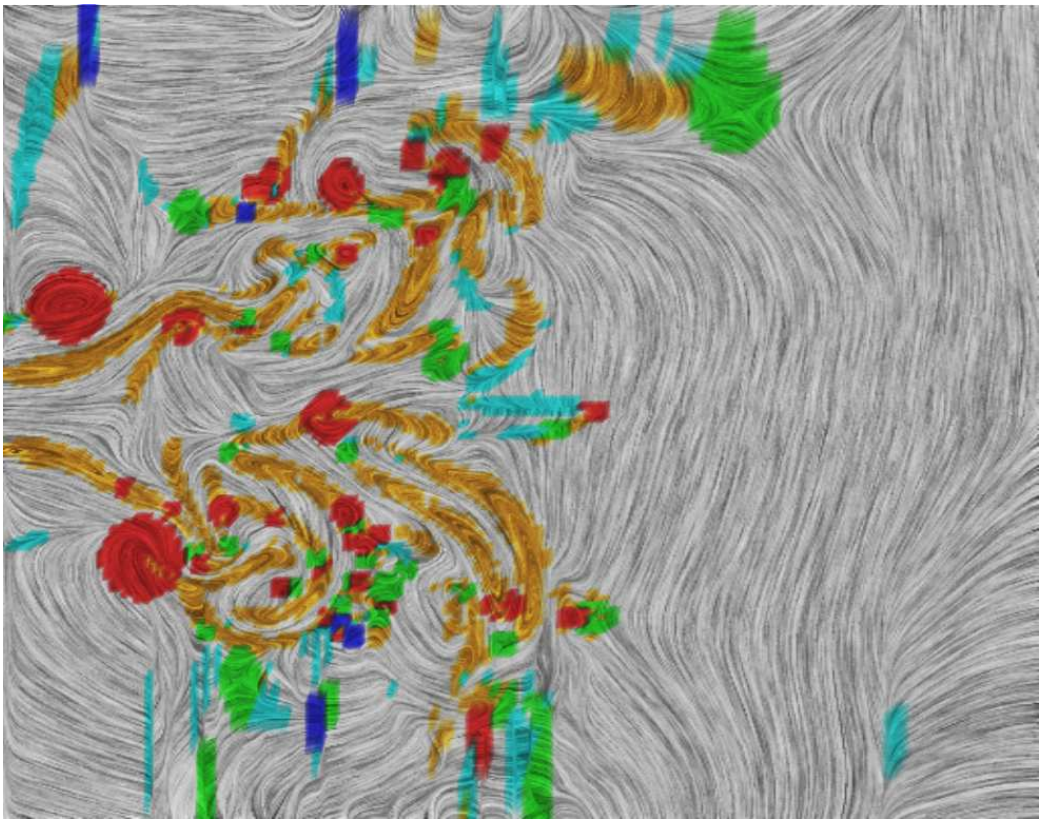

Figure 8.23: A swirling jet data set. Segmentation and LIC. Red: rotation, orange: shear flow, light blue: separation/attachment line, dark blue: sink/source, green:saddle. Threshold of 0.5.

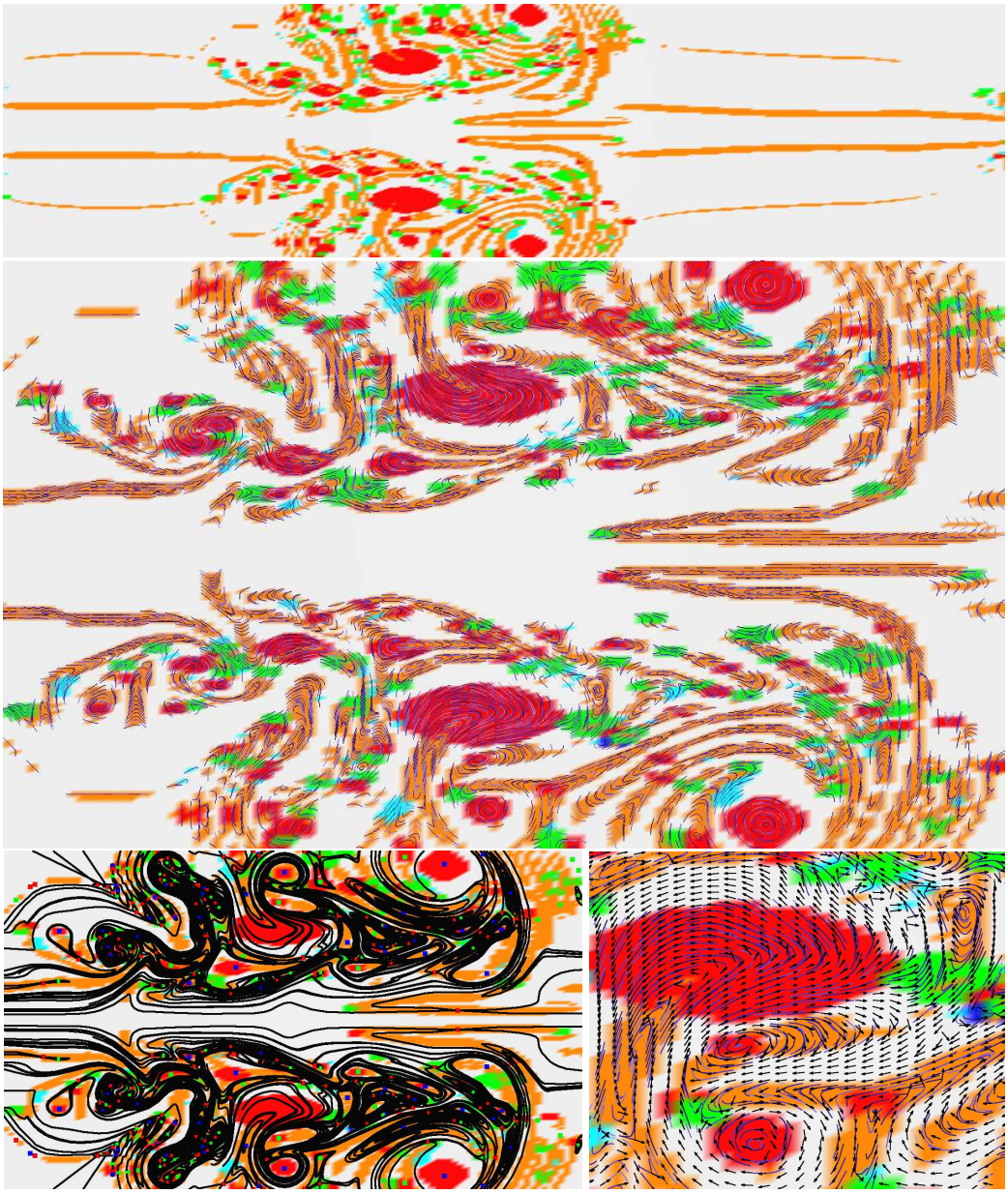

Figure 8.24: Segmentation of a simulation of a swirling jet entering a fluid at rest (threshold=0.5). Red: rotation, orange: shear flow, light blue: separation/attachment line, dark blue: sink/source, green: saddle. **Top**: Whole data set. **Middle** and **bottom left**: Zoomed in. Streamlines respectively topology added. **Bottom right**: Some details. Hedgehogs and streamlines added.

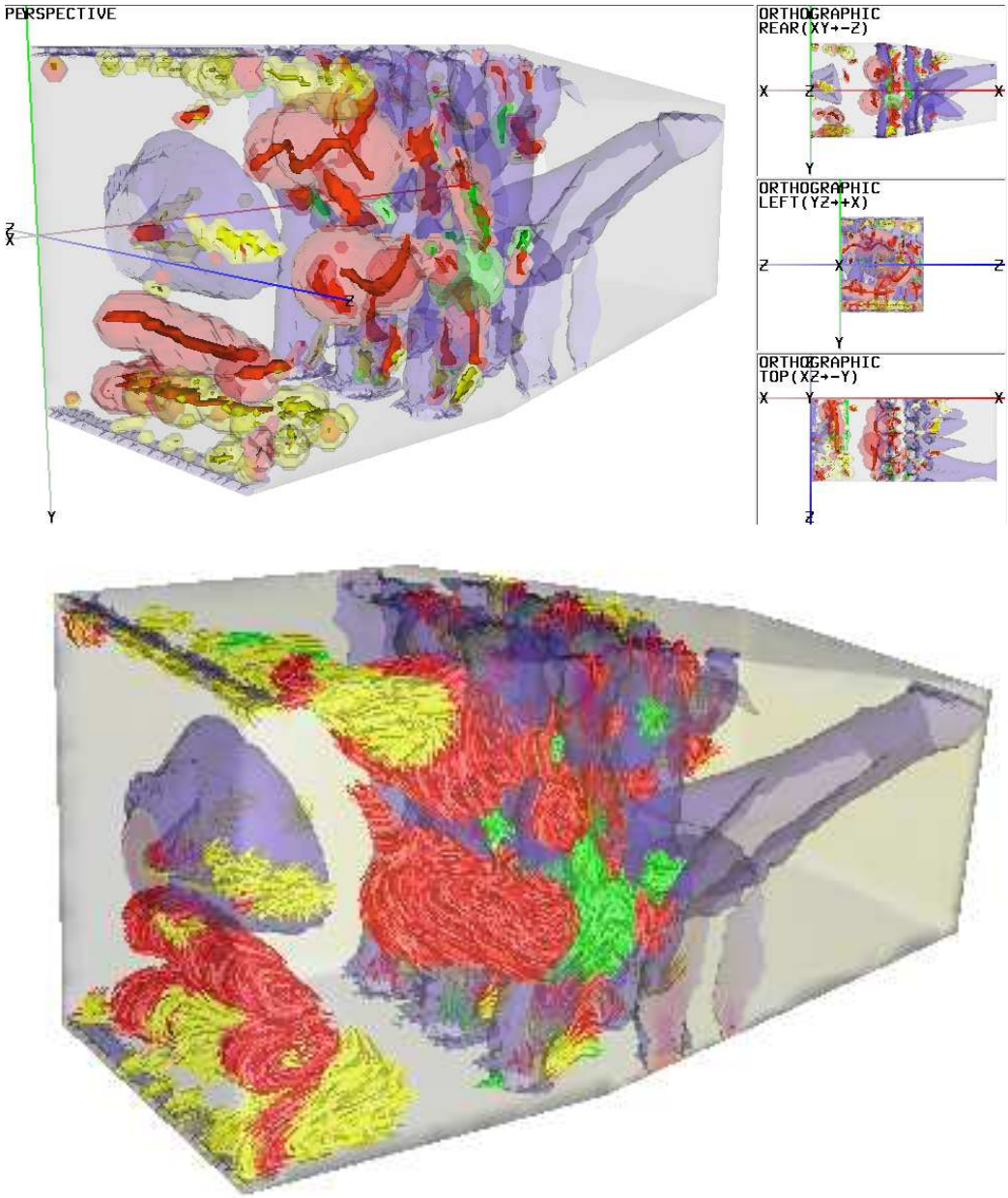

Figure 8.25: Segmentation of the normalized gas furnace chamber (Threshold: 0.5). Isosurfaces of the results (Value 0.5): Red: rotations, yellow: shear flow, green: saddles. The cores of the regions are displayed, too, using the same colors. Templates of divergence/convergence resulted in similarities below the threshold. The velocity of the original data set is displayed at an isovalue of 15 (blue). **Top**: Core lines and region of the features. **Bottom**: The results of the segmentation can also be used for streamline seeding.

Chapter 9

Conclusion

The goal of this thesis has been to transfer analysis methods from image processing to vector fields via the previously defined Clifford convolution, and to use them for automatic feature detection and subsequent visualization of flow fields. The methods developed thereby have all been examined and applied successfully to several complex flow data sets.

The author has presented en extension of the Fourier transform to Clifford algebra in 2D and 3D, and proofed convolution and correlation theorems for the Clifford convolution on multivector fields. Other theorems, like derivative and Parseval's theorem, were derived as well. The theorems extend those applicable to the Fourier transform on scalar fields while still remaining reasonably simple. The existence of fast algorithms for the computation of the Clifford Fourier transform allow an acceleration of Clifford convolution and related template matching algorithms. Analyzing flow pattern in frequency domain allows new insights into the flow. The frequency domain descriptions of simple flow patterns regarded in this thesis, for example, differs mostly in phase.

For flow fields gained by measurement using particle image velocimetry, the underlying grid is uniform and the data is quite noisy. Template matching, being robust due to the inherent averaging, is inherently suited for analysis of this data and behaves well. As flow fields gained by simulations are often defined on irregular grids, several strategies of template matching on these grids have been investigated as well. However, global resampling of the field is preferable as template matching on the resampled grid will be faster, especially if the fast Clifford Fourier transform is used for acceleration of the convolution computations. For flow fields defined on surfaces, however, global resampling will not work. Then, local resampling of the field using geodesics to determine the sampling positions has been used.

Vector-valued templates used for template matching of flow fields are linear, shift-invariant filter. The linearity is also known as the superposition principle.

Thus, the author has investigated the perception of flow fields as a superposition of several, possibly simpler fields. Engineers often think in terms of velocity, vorticity, and resulting feature models like the Vatistas vortex which are tuned to a superposition perspective of the flow field. However, visualizations based on streamline behavior can only study the resulting flow. The effects of superposition on the accuracy of the detection of the original features and their parameters have been discussed for different types of feature definitions, showing that the resulting errors can be quite high.

Based on this observation, several feature analysis methods based on Clifford convolution were investigated. Position, size, direction and velocity at the core radius can be determined quite robust and precise with the presented methods, for an accuracy of edge-length as well as subpixel accuracy.

Bringing together all the developed methods and knowledge, an algorithm for automatic computation of a feature-based segmentation has been presented. The visualizations of the segmentations highlight the detected features and reveal information about their interrelationship as they are displayed in conjunction. All in all, analysis and visualization of flow fields based on Clifford convolution is a profitable approach.

Bibliography

[1] D. Alexander, J. Gee, and R. Bajcsy. Similarity Measures for Matching Diffusion Tensor Images. In *Proceedings of British Machine Vision Conference*, pages 93–102, 1999.

[2] D. Alexander, J. Gee, and R. Bajcsy. Elastic Matching of Diffusion Tensor MRIs. *Computer Vision and Pattern Recognition*, pages 1244–1249, 2000.

[3] D. Bauer and R. Peikert. Vortex Tracing in Scale-Space. In I. Navazo D. Ebert, P. Brunet, editor, *Data Visualisation 2002*, pages 232–240, New York, USA, 2002. Association for Computational Machinery.

[4] D. Le Bihan, J.-F. Mangin, C. Poupon, C. A. Clark, S. Pappata, N. Molko, and H. Chabriat. Diffusion Tensor Imaging: Concepts and Applications. *Journal of Magnetic Resonance Imaging*, 13:534–546, 2001.

[5] A. I. Borisenko and I. E. Tarapov. *Vector and Tensor Analysis with Applications*. Dover Publications, Inc., New York, 1979.

[6] A. C. Bovik, M. Clark, and W. S. Geisler. Multichannel Texture Analysis using Localized Spatial Filters. *IEEE Trans. Pattern Anal. Machine Intell.*, 12(1):55–73, 1990.

[7] R. Bracewell. *The Fourier Transform and its Applications*. McGraw-Hill Book Company, New York, 1965.

[8] F. Brackx, N. De Schepper, and F. Sommen. The Clifford-Fourier Transform. *The Journal of Fourier Analysis and Applications*, 11(6), 2005.

[9] T. Bülow. *Hypercomplex Spectral Signal Representations for the Processing and Analysis of Images*. PhD thesis, University of Kiel, Germany, 1999.

[10] C. L. Burley, T. F. Brooks, B. G. van der Wall, H. Richard, M. Raffel, P. Beaumier, Y. Delrieux, J. W. Lim, Y. H. Yu, C. Tung, K. Pengel, and E. Mercker. Rotor Wake Vortex Definition - Initial Evaluation of Results of

the HART-II Study. In *28th EUROPEAN ROTORCRAFT FORUM*, Bristol, England, 2002.

[11] C. L. Burley, T. F. Brooks, B. G. van der Wall, H. Richard, M. Raffel, P. Beaumier, Y. Delrieux, J. W. Lim, Y. H. Yu, C. Tung, K. Pengel, and E. Mercker. Rotor Wake Vortex Definition Using PIV Measurements - Corrected for Orientation. In *9th AIAA/CEAS Aeroacoustics Conference*, 2003.

[12] T. Salzbrunn C. Garth, X. Tricoche. Surface Techniques for Vortex Visualization. In *Proceedings of Joint Eurographics - IEEE TCVG Symposium on Visualization*, Konstanz, Germany, 2004.

[13] B. Cabral and L. C. Leedom. Imaging Vector Fields Using Line Integral Convolution. In *Proceedings of SIGGRAPH '93*, pages 263–270, New York, 1993.

[14] Q. Chen, M. Defrise, and F. Deconinck. Symmetric Phase-Only Matched Filtering of Fourier-Mellin Transforms for Image Registration and Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(12):1156–1168, 1994.

[15] Wikipedia contributors. Wikipedia, The Free Encyclopedia, 2006.

[16] J.W. Cooley and O.W. Tukey. An Algorithm for the Machine Calculation of Complex Fourier Series. *Math. Comput.*, 19:297–301, 1965.

[17] I. Daubechies. The Wavelet Transform, Time-Frequency Localization and Signal Analysis. *IEEE Transactions on Information Theory*, 36(5):961–1005, 1990.

[18] W.C. de Leeuw and R. van Liere. Visualization of Global Flow Structures Using Multiple Levels of Topology. In E. Gröller, H. Löffelmann, and W. Ribarski, editors, *Data Visualization'99 (Proceedings VisSym '99)*, pages 45–52, 1999.

[19] T. Delmarcelle and L. Hesselink. Visualizing Second-Order Tensor Fields with Hyperstreamlines. *IEEE Computer Graphics and Applications*, 13(4):25–33, 1993.

[20] R. R. Dickenson. A Unified Approach to the Design of Visualization Software for the Analysis of Field Problems. In *SPIE Proceedings*, volume 1083, Bellinham, Washington, 1989. SPIE - The International Society for Optical Engineering.

[21] U. Diewald, T. Preußer, and M. Rumpf. Anisotropic Diffusion in Vector Field Visualization on Euclidean Domains and Surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):139–149, 2000.

[22] H. Doleisch, M. Gasser, and H. Hauser. Interactive Feature Specification for Focus+Context Visualization of Complex Simulation Data. In *Proceedings of the 5th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2003)*, pages 239–248, Grenoble, France, 2003.

[23] J. Ebling. Cliffordfaltung auf Vektorfeldern. Master's thesis, Universität Kaiserslautern, Germany, 2003.

[24] J. Ebling and G. Scheuermann. Clifford Convolution and Pattern Matching on Vector Fields. In *Proceedings of IEEE Visualization 2003*, pages 193–200, Los Alamitos, CA, 2003. IEEE Computer Society.

[25] J. Ebling and G. Scheuermann. Pattern Matching on Vector Fields using Gabor Filter. In *Proceedings of IASTED VIIP 2004*, pages 825–830, 2004.

[26] J. Ebling and G. Scheuermann. Clifford Convolution and Pattern Matching on Irregular Grids. In G.P. Bonneau, T. Ertl, and G.M. Nielson, editors, *Scientific Visualization: The Visual Extraction of Knowledge from Data*, Berlin, Germany, 2005. Springer.

[27] J. Ebling and G. Scheuermann. Fourier Transform on Multivector Fields. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):469–479, 2005.

[28] J. Ebling and G. Scheuermann. Segmentation of Flow Field using Pattern Matching. In *Data Visualization 2006*, 2006.

[29] J. Ebling and G. Scheuermann. Template Matching on Vector Fields using Clifford Algebra. In *IKM 2006*, 2006.

[30] J. Ebling, G. Scheuermann, and B.G. van der Wall. Analysis and Visualization of 3-C PIV Images from HART II using Image Processing Methods. In *Data Visualization 2005*, pages 161–168, 2005.

[31] J. Ebling, A. Wiebel, C. Garth, and G. Scheuermann. Topology Based Flow Analysis and Superposition Effects. In *submitted to Proceedings of TopoInVis 2005*.

[32] J. Fdez-Valdivia, J. A. Garcia, J. Martinez-Baena, and X. R. Fdez-Vidal. The Selection of Natural Scales in 2D Images using Adaptive Gabor Filtering. *IEEE Trans. Pattern Anal. Machine Intell.*, 20(5):458–469, 1998.

[33] H.G. Feichtinger and T. Strohmer. Fast Iterative Reconstruction of Band-Limited Images from Irregular Sampling Values. In D.Chetverikov and W.G. Kropatsch, editors, *Computer Analysis of Images and Patterns*, pages 82–91, Conf.CAIP Budapest 93, 1993.

[34] M. Felsberg. *Low-Level Image Processing with the Structure Multivector*. PhD thesis, University of Kiel, Germany, 2002.

[35] D. Gabor. Theorie of Communication. *J. Inst. Elect. Eng.*, 93:429–457, 1946.

[36] G. H. Granlund and H. Knutsson. *Signal Processing for Computer Vision*. Kluwer Academic Publishers, 1995.

[37] G. H. Granlund and H. Knutsson. *Signal Processing for Computer Vision*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[38] M. Griebel, T. Preußer, M. Rumpf, A. Schweitzer, and A. Telea. Flow Field Clustering via Algebraic Multigrid. In *Proceedings Visualization Conference 2004*. IEEE CS Press, 2004.

[39] M. Hamouz, J. Kittler, J.-K. Kamarainen, and H. Kälviäinen. Hypotheses-Driven Affine Invariant Localization of Faces in Verification Systems. In *Proceedings of the 4th International Conference on the Audio- and Video-Based Biometric Person Authentication*, pages 276–284, Guildford, UK, 2003.

[40] C. D. Hansen and C. R. Johnson. *The Visualization Handbook*. Elsevier Academic Press, 2005.

[41] E. B. Heiberg. *Automated Feature Detection in Multidimensional Images*. PhD thesis, Linköpings Universitet, Sweden, 2004.

[42] E. B. Heiberg, T. Ebbers, L. Wigström, and M. Karlsson. Three Dimensional Flow Characterization using Vector Pattern Matching. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):313–319, 2003.

[43] J.T. Heineck, G.K. Yamauchi, A.J. Woodcock, and L.Laurenco. Application of Three-Component PIV to a Hovering Rotor Wake. In 56^{th} *Annual Forum of the American Helicopter Society*, Virginia Beach, VA, USA, 2000.

[44] J.L. Helman and L. Hesselink. Visualizing Vector Field Topology in Fluid Flows. *IEEE Computer Graphics and Applications*, pages 36–46, 1991.

[45] L. Hesselink, Y. Levy, and Y. Lavin. The Topology of Symmetric, Second-Order 3D Tensor Fields. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):1–11, 1997.

[46] D. Hestenes. *New Foundations for Classical Mechanics*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1986.

[47] D. Hestenes. *Clifford Algebra to Geometric Calculus*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.

[48] A.N. Hirani. *Discrete Exterior Calculus*. PhD thesis, California Institute of Technology, 2003.

[49] M. Hlawitschka, J. Ebling, and G. Scheuermann. Convolution and Fourier Transform of Second Order Tensor Fields. In *Proceedings of IASTED VIIP 2004*, pages 78–83, 2004.

[50] B.K.P. Horn and B.G. Schunk. Determining Optical Flow. *Artificial Intelligence*, 17:185–204, 1981.

[51] B. Jähne. *Digitale Bildverarbeitung*. Springer Verlag, Berlin, Germany, 2002.

[52] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, Englewoods Cliffs, NJ, USA, 1989.

[53] J. Jeong and F. Hussain. On the Identification of a Vortex. *Journal of Fluid Mechanics*, 285:69 – 94, 1995.

[54] B. Jeremic, G. Scheuermann, J. Frey, Z. Yang, B. Hamann, K. I. Joy, and H. Hagen. Tensor Visualization in Computational Geomechanics. *Int. J. Numer. Anal. Meth. Geomech.*, 26:925–944, 2002.

[55] M. Jiang, R.Machiraju, and D. Thompson. Geometric Verification of Swirling Features in Flow Fields. In *Proceedings of IEEE Visualization 2002*, pages 307–314, Los Alamitos, CA, 2002. IEEE Computer Society.

[56] M. Jiang, R.Machiraju, and D. Thompson. A Novel Approach to Vortex Core Region Detection. In *Joint Eurographics - IEEE TVCG Symposium on Visualization*, pages 217–225, Barcelona, Spain, 2002.

[57] D. N. Kenwright, C. Henze, and C. Levit. Feature Extraction of Separation and Attatchment Lines. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):151–158, 1999.

[58] G. Kindlemann, D. M. Weinstein, and D. Hart. Strategies for Direct Volume Rendering of Diffusion Tensor Fields. *IEEE Transactions on Visualization and Computer Graphics*, 6(2):124–137, 2000.

[59] V. Krueger, S. Bruns, and G. Sommer. Efficient Head Pose Estimation with Gabor Wavelet Networks. In *BMVC*, pages 11–14, Bristol, Great Britain, 2000.

[60] V. Krueger and G. Sommer. Gabor Wavelet Networks for Object Representation. *Journal of the Optical Society of America(JOSA)*, 2002. accepted for publication.

[61] V. Kyrki, J.-K. Kamarainen, and H. Kälviäinen. Simple Gabor Feature Space for Invariant Object Recognition. *Pattern Recognition Letters*, 25(3):311–318, 2004.

[62] T. S. Lee. Image Representation using 2D Gabor Wavelets. *IEEE Trans. Pattern Anal. Machine Intell.*, 18(10):959–971, 1996.

[63] T. Lindeberg. *Scale-Space Theory in Computer Vision*. Kluwer Academic Publisher, Boston, 1994.

[64] B. S. Manjunath. Gabor Wavelet Transform and Application to Problems in Computer Vision. In *Proc. 26th Asilomar Conference on Signals, Systems and Computers*, pages 796–800, Pacific Grove, CA, 1992.

[65] S. Marcelja. Mathematical Description of the Responses of Simple Cortical Cells. *J. Opt. Soc. Amer.*, 70:164–177, 1980.

[66] A. V. Oppenheim, R. W. Schafer, and J. R. Buck. *Discrete-Time Signal Processing*. Prentice Hall, 1999.

[67] H.G. Pagendarm and B. Walter. Feature Detection from Vector Quantities in a Numerically Simulated Hypersonic Flow Field in Combination with Experimental Flow Visualization. In *Proceedings of IEEE Visualization '94*, pages 117–123, Los Alamitos, CA, 1994. IEEE Computer Society Press.

[68] S. Pajevic and C. Pierpaoli. Color Schemes to Represent the Orientation of Anisotropic Tissues from Diffusion Tensor Data. Application to White Matter Fiber Tract Mapping in the Human Brain. *Magnetic Resonance in Medicine*, 42(3):526–540, 1999.

[69] K. Polthier and E. Preuß. Variational Approach to Vector Field Decomposition. In R. Liere and F. Post et al., editors, *Eurographis Workshop on Scientific Visualization*, pages 113–134, 2000.

[70] K. Polthier and E. Preuß. Identifying Vector Field Singularities Using a Discrete Hodge Decomposition. *Visualization and Mathematics III*, pages 775–792, 2003.

[71] M. Porat and Y. Y. Zeevi. The Generalized Gabor Scheme of Image Representation in Biological and Machine Vision. *IEEE Trans. Pattern Anal. Machine Intell.*, 10(4):452–468, 1988.

[72] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleitsch. Feature Extraction and Visualization of Flow Fields. In D. Fellner and R. Scopigno, editors, *Eurographics 2002*, pages 69–100, Saarbrücken, Germany, 2002. The Eurographics Association.

[73] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleitsch. The State of the Art in Flow Visualization: Feature Extraction and Tracking. *Computer Graphics Forum*, 22(4):775–792, 2003.

[74] D. Potts, G. Steidl, and M. Tasche. Fast Fourier Transforms for Nonequispaced Data: A Tutorial. In J. J. Benedetto and P. J. S. G. Ferreira, editors, *Modern Sampling Theory: Mathematics and Applications*, pages 247–270, Boston, 2001. Birkhäuser.

[75] M. Raffel, H. Richard, K. Ehrenfried, B. G. van der Wall, C. L. Burley, P. Beaumier, K. McAlister, and K. Pengel. Recording and Evaluation Methods of PIV Investigations on a Helicopter Rotor Model. In *Experiments in Fluids 36*, pages 146–156. Springer Verlag, 2004.

[76] W.J.M Rankine. *Manual of Applied Mechanics*. C. Griffen Co., London, 1858.

[77] M. Roth. *Automatic Extraction of Vortex Core Lines and other Line Type Features for Scientific Visualization*. PhD thesis, ETH, 2000.

[78] J. Ruiz-Alzola, C.-F. Westin, S. K. Warfield, A. Nabavi, and R. Kikinis. Nonrigid Registration of 3D Scalar, Vector and Tensor Medical Data. In *Proceedings of Third International Conference On Medical Robotics, Imaging and Computer Assisted Surgery*, pages 541–550, Pittsburgh, Pennsylvania, USA, 2000.

[79] G. Scheuermann. *Topological Vector Field Visualization with Clifford Algebra*. PhD thesis, University of Kaiserslautern, Germany, 1999.

[80] G. Scheuermann, H. Hagen, H. Krüger, M. Menzel, and A. Rockwood. Visualization of Higher Order Singularities in Vector Fields. In Roni Yagel and Hans Hagen, editors, *Proceedings of IEEE Visualization*, pages 67–74, Los Alamitos, CA, 1997. IEEE Computer Society Press.

[81] G. Scheuermann, H. Krüger, M. Menzel, and A. Rockwood. Visualizing Nonlinear Vector Field Topology. *IEEE Transactions on Visualization and Computer Graphics*, 4(2):109–116, 1998.

[82] M. Schlemmer, I. Hotz, V. Natarajan, B. Hamann, and H. Hagen. Fast Clifford Fourier Transform for Unstructured Vector Field Data. In P. Papadopoulos, P.R. Eiseman, J. Haeuser, B.K. Soni, and J.F. Thompson, editors, *Proceedings of Ninth International Conference on Numerical Grid International Society of Grid Generation*, pages 101–110, College of Engineering, San Jose State University, San Jose, California, 2005.

[83] W. Schroeder, K. Martin, and B. Lorensen. *The Visualization Toolkit 2nd Edition*. Prentice Hall, New Jersey, USA, 1998.

[84] M.P. Scully. Computation of Helicopter Rotor Wake Geometry and It's Influence on Rotor Harmonic Airloads. Technical Report ASRL TR 178-1, Masachussetts Institute of Technology, March 1975.

[85] Ben Shneiderman. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *In Proceedings of the IEEE Symposium on Visual Languages*, pages 336–343, Washington, 1996. IEEE Computer Society Press.

[86] A. Siegfridsson, T. Ebbers, E. Heiberg, and L. Wigström. Tensor Field Visualization using Adaptive Filtering of Noise Fields combined with Glyph Rendering. In *Proceedings of IEEE Visualization 2002*, pages 461–469, Los Alamitos, CA, 2001. IEEE Computer Society Press.

[87] G. Sommer. *Geometric Computing with Clifford Algebras*. Springer Verlag, Berlin, Germany, 2001.

[88] T. Strohmer. Computationally Attractive Reconstruction of Band-Limited Images from Irregular Samples. *IEEE Transactions on Image Processing*, 6(4):540–548, 1997.

[89] B. Stroustrup. *Die C++ Programmiersprache*. Addison-Wesley Verlag, München, Germany, 2000.

[90] D. Sujudi and R. Haimes. Identification of Swirling Flow in 3-D Vector Fields. In *AIAA Paper 95-1715, 12th AIAA CFD Conference*, San Diego, 1995.

[91] H. Theisel, H.C. Hege, and H.C. Seidel. Saddle Connectors - An Approach to Visualizing the Topological Skeleton of Complex 3D Vector Fields. In H. Rushmeier, G. Turk, and J. van Wijk, editors, *Proceedings of IEEE Visualizations 2003*, pages 225–232, 2003.

[92] S.P. Timoshenko and J.N. Goodier. *Theory of Elasticity*. McGraw-Hill Kogakusha, Ltd., 1970.

[93] Y. Tong, S. Lombeyda, A. Hirani, and M. Desbrun. Discrete Multiscale Vector Field Decomposition. *ACM Transactions on Graphics*, 22(3):445–452, 2003.

[94] X. Tricoche. *Vector and Tensor Field Topology Simplification, Tracking, and Visualization*. PhD thesis, University of Kaiserslautern, Germany, 2002.

[95] X. Tricoche, C. Garth, and G. Scheuermann. Fast and Robust Extraction of Separation Line Features. In G.P. Bonneau, T. Ertl, and G.M. Nielson, editors, *Scientific Visualization: The Visual Extraction of Knowledge from Data*, pages 249–264, Berlin, Germany, 2005. Springer.

[96] X. Tricoche, G. Scheuermann, and H. Hagen. A Topology Simplification Method for 2D Vector Fields. In D. Ebert, M. Gross, and B. Hamann, editors, *Proceedings of IEEE Visualization 2000*, pages 359–366, Los Alamitos, CA, 2000. IEEE Computer Society Press.

[97] X. Tricoche, G. Scheuermann, and H. Hagen. Tensor Topology Tracking: A Visualization Method for Time-Dependent 2D Symmetric Tensor Fields. In *Eurographics 2001 Proceedings, Computer Graphics Forum 20(3)*, pages 461–470, Saarbrücken, Germany, 2001. The Eurographics Association.

[98] D. Tschumperle and R. Deriche. Diffusion Tensor Regularization with Constraints Preservation. In *Computer Vision and Pattern Recognition 2001*, Kauai, Hawaii, 2001.

[99] T. Urness, V. Interante, E. Logmire, I. Marusic, and B. Ganapathisubramani. Interactive Poster: Illustrating Different Convection Velocities of

Turbulent Flow. In *Poster Compendium of IEEE Visualization 2004*, Los Alamitos, CA, 2004. IEEE Computer Society Press.

[100] B. G. van der Wall, C. L. Burley, Y. H. Yu, H. Richard, K. Pengel, and P. Beaumier. The HART II Test - Measurement of Helicopter Rotor Wakes. *Aerospace Science and Technology*, 8(4):273–284, 2004.

[101] T. van Walsum, F. H. Post, D. Silver, and F. J. Post. Feature Extraction and Iconic Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):111–119, 1996.

[102] G.H. Vatistas. New Model for Intense Self-Similar Vortices. *Experiments in Fluids*, 14(4):462–469, 1998.

[103] C. Ware. *Information Visualization : Perception for Design*. Morgan Kaufmann Publishers, San Francisco, USA, 2000.

[104] D. M. Weinstein, G. Kindlmann, and E. C. Lundberg. Tensorlines: Advection-Diffusion based Propagation through Diffusion Tensor Fields. In *Proceedings of IEEE Visualization '99*, pages 249–253, Los Alamitos, CA, 1999. IEEE Computer Society.

[105] E.W. Weisstein. MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com.

[106] T. Welsh and K. Mueller. A Frequency-Sensitive Point Hierarchy for Images and Volumes. In *Proceedings of IEEE Visualization 2003*, pages 425–432, Los Alamitos, CA, 2003. IEEE Computer Society.

[107] C.-F. Westin and H. Knutsson. Tensor Field Regularization using Normalized Convolution. In *Proceedings of the Ninth International Conference on Computer Aided Systems Theory (EUROCAST)*, pages 564–572, 2003.

[108] C.-F. Westin, S. E. Maier, B. Khidhir, P. Everett, F. A. Jolesz, and R. Kikinis. Image Processing for Diffusion Tensor Magnetic Resonance Imaging. In *Proceedings of Second Int. Conf. on Medical Image Computing and Computer-assisted interventions*, pages 441–452, 1999.

[109] Alexander Wiebel, Christoph Garth, and Gerik Scheuermann. Localized Flow Analysis of 2D and 3D Vector Fields. In Ken Brodlie, David Duke, and Ken Joy, editors, *Data Visualization 2005*, pages 143–150. Eurographics Association, 2005.

[110] C. M. Wittenbrink, A. T. Pang, and S. K. Lodha. Glyphs for Visualizing Uncertainty in Vector Fields. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):266 – 279, 1996.

[111] X. Zheng and A. Pang. Volume Deformation for Tensor Visualization. In *Proceedings of IEEE Visualization '02*, pages 379–386, Los Alamitos, CA, 2002. IEEE Computer Society.

[112] X. Zheng and A. Pang. HyperLIC. In *Proceedings of IEEE Visualization '03*, pages 249–256, Los Alamitos, CA, 2003. IEEE Computer Society.

[113] L. Zhukov and A. H. Barr. Oriented Tensor Reconstruction: Tracing Neural Pathways from Diffusion Tensor MRI. In *Proceedings of IEEE Visualization '02*, pages 387–394, Los Alamitos, CA, 2002. IEEE Computer Society.

[114] L. Zhukov and A. H. Barr. Heart-Muscle Fiber Reconstruction from Diffusion Tensor MRI. In *Proceedings of IEEE Visualization '03*, pages 597–605, Los Alamitos, CA, 2003. IEEE Computer Society.

Curriculum Vitae

Persönliche Daten

Name:	Ebling
Vorname:	Julia
Geburtsdatum:	27.11.1978
Geburtsort:	Mainz
Eltern:	Richard Jürgen Ebling
	Barbara Schulte-Ebling, geb. Schulte

Lebenslauf

Aug. 1984	-	Juli 1988	Besuch der Grundschule Schornsheim
Aug. 1988	-	Juni 1997	Besuch des Gymnasium Nieder-Olm
Juni 1997			Abitur
Okt. 1997	-	Feb. 2003	Studium der Informatik mit Nebenfach Mathematik an der Universität Kaiserslautern
Feb. 2003			Diplom in Informatik
März 2003	-	Mai 2004	Wissenschaftliche Mitarbeiterin der AG Computergraphik des Fachbereiches Informatik der Universität Kaiserslautern
seit Juni 2004			Wissenschaftliche Mitarbeiterin der AG Bild- und Signalverarbeitung des Fachbereiches Mathematik und Informatik der Universität Leipzig