Inaugural-Dissertation

zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich–Mathematischen Gesamtfakultät
der
Ruprecht–Karls–Universität
Heidelberg

vorgelegt von
Diplom–Informatiker Daniel Kondermann
aus Kassel

Tag der mündlichen Prüfung: 23.09.2009

Modular Optical Flow Estimation With Applications To Fluid Dynamics

1. Gutachter: **PD Dr. habil. Christoph Garbe**
Digitale Bildverarbeitung
Interdisziplinäres Zentrum
für wissenschaftliches Rechnen
Universität Heidelberg

2. Gutachter: **Prof. Christoph Schnörr**
Image and Pattern Analysis
Interdisziplinäres Zentrum
für wissenschaftliches Rechnen
Universität Heidelberg

Zusammenfassung

Der Begriff "Optischer Fluss" bezeichnet die scheinbare Bewegungen von Intensitäten einer Bildfolge. Seine Schätzung wird bereits seit fast dreißig Jahren untersucht. Die Ergebnisse können für eine Vielzahl von Anwendungen in Bereichen wie der experimentellen Strömungsschätzung und der medizinischen Bildverarbeitung bis hin zu mobilen Computerspielen verwendet werden. Die Entwicklung eines einzigen Schätzers für alle Probleme des optischen Flusses scheint ein erstrebenswertes Ziel zu sein. In dieser Arbeit argumentieren wir jedoch, dass dieses Ziel wahrscheinlich nie erreicht werden wird. Diese Hypothese motivieren wir gründlich mit theoretischen Überlegungen und praktischen Ergebnissen. Anhand dieser Ergebnisse identifizieren wir zwei wichtige Probleme, die die weitere Forschung und Entwicklung behindern. Erstens gibt es nur wenige öffentlich verfügbare Implementierungen von Flussschätzungsverfahren; zweitens werden nicht alle relevanten Eigenschaften dieser Verfahren publiziert.

Im ersten Teil dieser Arbeit tragen wir zur Lösung beider Probleme bei. Dazu diskutieren wir erst einige Eigenschaften, auf die ein solcher Schätzer geprüft werden sollte. Weiterhin zerlegen wir existierende Methoden in ihre algorithmischen Bausteine (genannt Module). Wir schlagen vor, diese Module unabhängig vom gesamten Schätzer bezüglich ihrer inhärenten Eigenschaften zu studieren. Eine große Zahl von Flusschätzern besteht aus einer relativ geringen Zahl von verschiedenen Modulen. Wir haben diese Module in einer Softwarebilbliothek namens Charon implementiert. Dadurch tragen wir zur Erreichbarkeit von Referenzimplementierungen und zur Möglichkeit des Experimentierens mit vorhanden Algorithmen bei.

Im zweiten Teil dieser Arbeit stellen wir zwei neue Module zur Strömungsmessung vor, die speziell entwickelt wurden für Bilddaten, die zur "Particle Tracking Velocimetry" (PTV) erzeugt wurden. Das erste Modul nennen wir "Schätzbarkeitsmaß" (Estimatibility Measure). Es erkennt alle Pixelpositionen, an denen ein verlässlicher Fluss geschätzt werden kann. Es basiert auf der Idee, dass die Bilddaten unter Anwendung mehrerer Schwellwerte Zusammenhangskomponenten mit fast identischen Schwerpunkten pro Schwellwert enthalten. Dieses Modul benötig lediglich wenige und intuitive Parameter. Experimente weisen darauf hin, dass dieses Verfahren sehr robust bezüglich gaussförmigem Rauschen mit räumlich variierenden Mittelwerten und Varianzen ist. Um diese Eigenschaften zu bestimmen, schlagen wir außerdem ein Programm vor, dass die Erzeugung von Partikelbildern simuliert.

Das zweite Modul ist ein Bewegungsmodell, das auf unüberwachtem Lernen mittels Hauptkomponentenanalyse basiert. Trainingsdaten werden durch Computational Fluid Dynamic (CFD) Simulationen bereit gestellt. Das Modell beschreibt lokale Ensembles von Trajektorien deren Parameter über die Bilddaten mittels eines Ähnlichkeitsmaßes optimiert werden können. Zusammen mit einem üblichen Ähnlichkeitsmaß und einem einfachen Optimierungsverfahren setzen wir ein neues PTV-Verfahren zusammen. Verglichen mit existierenden Techniken erreich wir genauere Ergebnisse auf realen und synthetischen Sequenzen mit bekannten Flüssen.

Den gesamten, während dieser Arbeit entwickelten Quellcode bieten wir im Rahmen der GNU Lesser General Public License (LGPL) als Open Source an.

Abstract

Optical flow is the apparent motion of intensities in an image sequence. Its estimation has been studied for almost three decades. The results can be used in a wealth of possible applications ranging from scientific applications like experimental fluid dynamics over medical imaging to mobile computer games. The development of a single solution for all optical flow problems seems to be a worthwhile goal. However, in this thesis, we argue that this goal is unlikely to be achieved. We thoroughly motivate this hypothesis with theoretical and practical considerations. Based on the results, we identify two major problems that significantly complicate the research and development of new optical flow algorithms: First, very few reference implementations are publicly available. Second, not all relevant properties of the proposed algorithms are described in literature.

In the first part of this thesis, our contribution is to alleviate both problems. First, we discuss a number of algorithm properties which should be known by the user. Second, by decomposing existing optical flow methods into their individual algorithm building blocks, shortly called modules, we propose to individually analyze the properties of each module independently. A large number of existing techniques is composed of relatively few existing modules. By implementing these modules in a software library called Charon and adding tools for the evaluation of the results, we contribute to the accessibility of reference implementations and to the possibility of analyzing algorithms by experiments.

In the second part of this thesis, we contribute two modules which are vital for the estimation of fluid flows. They are specifically tuned to the imagery obtained for particle tracking velocimetry (PTV). We call the first module estimatibility measure. It detects those particle locations where fluid motion can be estimated. It is based on the constant position of the center of gravity of the connected components generated by a large number of thresholded versions of the original image. The module only needs a few intuitive parameters. Experiments indicate its robustness with respect to noise with varying mean and variance. To analyze the properties of this module we also provide a framework for simulating the particle image generation.

The second module is a motion model based on unsupervised learning via principal component analysis. Training data is provided through Computational Fluid Dynamic (CFD) simulations. The model describes local ensembles of trajectories which can be fitted to the image sequence by means of a similarity measure. Together with a standard similarity measure and a simple optimization scheme we derive a new PTV method. Compared to existing techniques, we obtained superior results with respect to accuracy on real and synthetic sequences with known ground truth.

All source code developed during the thesis is available as Open Source following the GNU Lesser General Public License (LGPL).

Acknowledgments

Many thanks go to my supervisor Christoph Garbe who supported me with inspiring critical questions, countless hours of paper-reviews and helpful discussions. I also thank Bernd Jähne for hiring and supporting me. Furthermore, huge thanks go to André Berthe for all the great discussions on the phone, the nice days in Berlin, Stuttgart, Münster, Heidelberg and Karlsruhe and for supporting my work with all the image data you sent me for evaluation.
Thanks also go to all of my colleagues who supported and inspired my work. As inspiration seems to be one of the most important ingredients for a successful dissertation, I also thank the following people (who probably did not even notice that they contributed to my work): Daniel Cremers (well, you might know), David Tschumperlé, the PETSc developers at the Argonne National Laboratory, Michael Kelm, Michael Felsberg, Vernor Vinge, Gregory Benford and probably many more people I just did not notice while influencing me as well.

During the last three years so many things happened in my private life that submitting this thesis without mentioning those people involved would be submitting only half of the work. In order of appearance: I thank Claudia Kondermann for the years we shared. I will keep it in fond memory. I thank Hans Schlief for great times and a deep insight into myself. I thank the whole family Wiemer, simply for your existence. I also thank all the friends I won and lost during my darkest days. Special thanks go to Jonas Andrulis: sooner or later, we will rule the world! I also thank Christian Bischoff for countless mind-opening, inspiring and truly helpful hours of discussions. Last but not least I thank Jessica Rhodes. I thank you for being wild and weak, for being scientific and esoteric, for being self-critical and self-opinionated, for being open-minded and yet still focused on your great goal, for being irrepressibly emotional and still restrained when needed, for being misanthropic and nature-loving. I thank you for being there, when I need you, and for showing me who I am.
There are many more important people I met in the last three years, but this is a PhD thesis and not a biography ;)

Finally, I thank my parents and siblings for making me who I am today - especially for making me the improvident rebel I believe to be every now and then.

Contents

Part I.

Problem Statement

> The scientist is not a person who gives the right answers, he's one who asks the right questions.
>
> *(Claude Lévi-Strauss, Le Cru et le cuit, 1964)*

1. Introduction

> Every honest researcher I know admits he's just a professional amateur. He's doing whatever he's doing for the first time. That makes him an amateur. He has sense enough to know that he's going to have a lot of trouble, so that makes him a professional.
>
> *(Charles Franklin Kettering (1876-1958) U. S. Engineer and Inventor.)*

1.1. Motivation and Overview

Informally, motion estimation is the task of finding out how pixels or objects move in a given digital image sequence. At a first glance this might sound simple but at second sight one runs into more and more problems. Indeed, this task has been studied for more than 30 years now. One might wonder what exactly the challenges are. This work is about a closer look at some of them. Although many of the topics addressed here are of interest in all motion estimation techniques, in this work we focus on optical flow estimation, whose aim is to compute a motion vector at each pixel location in the image. Most research done in this field is dedicated to the accuracy of motion estimation methods and to sophisticated ideas on how to generally solve the problem once and for all with a single algorithm.

1.1.1. Modular Optical Flow Estimation

In their paper titled "Towards Ultimate Motion Estimation: Combining Highest Accuracy with Real-Time Performance" by Bruhn and Weickert [1], the authors focus on the improvement of the method of Brox et al. [2] and its near-real-time implementation. Although the authors' contribution of this work is highly valuable for optical flow research and also addresses the very important problem of reducing computation times for global techniques, their title is misleading in various ways. First, the term *highest accuracy* implies that the implemented algorithm yields the best achievable accuracy possible independent of the image data. Second, the term *ultimate* suggests that a single algorithm can exist that solves all problems for all types of image sequences. Third, the term *motion estimation* implies that this not only holds true for optical flow algorithms but also for motion estimation techniques in general. In this work we argue against these concepts.
Even though accuracy and speed are two major goals in motion estimation, we believe

that there are more requirements to motion estimation algorithms. In this work we would like to emphasize the importance of reference implementations, experimental design and the study of other requirements of optical flow algorithms such as applicability, accessibility and modularity.
Along these lines we will argue that it is highly likely that the motion estimation problem will never be solved with a single algorithm. We support this hypothesis with theoretical discussions as well as experimental results in the first part of this thesis. However, we will show that many of todays methods can be unified in one single framework, allowing for a modular implementation with interchangeable parts that constitute most of todays elements of motion estimation. By breaking down the problem of motion estimation into all of its subproblems, each component can be studied separately. By substituting specific modules of interest with others while all of the other modules remain exactly the same (in theory as well as in the implementation), we obtain much more experimental and analytical freedom. We provide an open source software library named Charon which can be used to study those research topics implied by the previous theoretical discussions.

1.1.2. Application to Fluid Flow Estimation

The second main part of this thesis applies the theoretical and practical results of the first part to the field of fluid flow estimation.
The instantaneous measurements of near wall flows are of great interest in biomedical research in order to obtain a deeper understanding of physiological and pathological processes. In the field of biofluid mechanics many medical issues, such as thrombotic events and atherosclerosis, depend on shear stresses and shear rates near non-planar, deformable walls such as, for example, blood flows in aneurysms and displacement blood pumps. The measurement of flow fields and shear rates near non-planar walls is challenging. Therefore, very specific motion estimation techniques are required which exploit every available prior knowledge of the measurement system. One of the main properties of such image data is that motion information is mainly distributed in the temporal domain (instead of the spatial domain).
We will discuss a new particle tracking velocimetry (PTV) algorithm which mainly utilizes temporal image information. It is implemented in Charon. We consider PTV as a very special case of optical flow estimation, where the various models are highly adapted to the image data at hand. Therefore, PTV is an ideal candidate to substantiate our hypothesis that general motion estimation is infeasible. The PTV technique consists of two algorithms: one which detects particle locations and one which estimates the motion. For both algorithms we offer source code, test data and an analysis with respect to the algorithm requirements described in the first part.
Together with the Biofluidmechanics Laboratory at the Charité in Berlin, we developed this technique to get one step closer to the aim of measuring fully three-dimensional motion trajectories. Our algorithm currently estimates two-component trajectories and accomodates for the brightness variations induced by the inherent three-dimensionality of the motion. As discussed in Chapter 8, this algorithm can easily be extended in a

final step to a fully three-dimension, three-component trajectory estimation scheme as soon as the experimental setup developed at the Charité becomes available.

1.2. Contribution

We thoroughly discuss the hypothesis that a general solution for motion estimation problems is not feasible. The large diversity of applications comes along with a host of differing demands. Therefore, we suggest a list of requirements for motion estimation algorithms. We argue that any new method should be tested against these properties in order to obtain a specification that can be used by those who need to select the correct algorithm for their individual application.
Furthermore, we contribute to two aspects of such desirable algorithm properties, namely modularity and accessibility for optical flow algorithms. Based on the proposed set of desirable properties, we give an overview of a number of publications in the field of optical flow estimation. Thereby, we identify a common modular framework in which all of these algorithms fit into. We offer an implementation of this framework along with a number of these methods in order to be able to exemplarily analyze some of their properties.
Furthermore, based on this framework, we contribute a new fluid flow estimation algorithm based on particle tracking velocimetry. By choosing the best instances of all available modules defined by the framework, we are able to define a new algorithm that specifically exploits the information in the image data at hand. Finally, we analyze this new algorithm with respect to its desired properties. Therefore, specially designed real-world ground truth data is generated with a new method well-suited for a broad range of particle tracking velocimetry applications.

1.3. Related Work

On the one hand there is an unmanageable amount of related work ranging from the actual motion estimation algorithms to confidence measures, performance evaluation, real-time-implementations on CPUs [1] or graphic card processors [3, 4] and several fields of related research. In most papers focusing on optical flow estimation some of the algorithm properties described in Chapter 3 are addressed. Sometimes, as for example in the cases of [5] and [6, 7], reference implementations are supplied in the form of source code. In many computer vision libraries such as OpenCV, CImg and FlowJ offer some basic optical flow algorithm implementations. Performance evaluations have been carried out by various authors as for example [8, 9, 10]. An important theoretical discussion on performance evaluation in the field of computer vision has been published in [11]. Many papers address certain modules of motion estimation. For example, Bruhn et al. [12] studied the effect of the Gaussian filtering as preprocessing step for differential optical flow methods. Various types of models of brightness variations have for example been discussed in [13, 14]. The influence and optimization of scale-spaces (or, more specifically, pyramids) is e.g. discussed in [15, 16] but has seldomly been addressed

again, although it seems clear that multi-scale strategies can significantly improve the accuracy of motion estimates.
On the other hand, the first part of this work focuses on an implementation of a broad range of optical flow methods within a single framework in order to increase the degree of comparability, availability and modularity. To the best of our knowledge this has not been addressed so far.
However, a number of related fields of research exist in which motion estimation algorithms are developed and in which our approach could be of advantage as well. In stereo vision the correspondences between points on epipolar lines are sought [17, 18]. Registration is the task of aligning two images onto each other. Often, these images are of different modality (e.g. ultrasound and x-ray images) [19]. The associated problems are very similar to those of optical flow estimation and the developed strategies are often the same. (For example, the variational matching of images based on cross-correlation [20] combines global regularization with cross-correlation based block-matching techniques [21]).
Although tracking can be understood as a much more general problem (namely the estimation of the evolution of a state in time [22]) it is often understood as locating objects in long image sequences. Such algorithms usually need to work in real-time and are therefore of limited accuracy. Another major difference is that the motion estimates sought are not dense and usually more than two images are used.
Two other fields of research, which until recently have often been ignored by the computer vision community, are particle image velocimetry (PIV, [23]) and particle tracking velocimetry (PTV, [24, 25]). Both fields have a highly correlated but seemingly disjunct history of research (cf. 7.2.3). The most significant difference, with respect to optical flow estimation, is that both techniques not only describe software algorithms for the estimation of fluid flows: they also describe the physical experimental setup that is used to acquire the images needed by the algorithms. This closely coupled research process brings both advantages and disadvantages. On the one hand the image acquisition process is carefully set up, analyzed and modeled appropriately in the software algorithms. On the other hand, research is focused both on experiments and on algorithm design which sometimes blurs the boundaries of both to some extend. We discuss relations between both fields in detail in Chapter 7. All work related to PTV is reviewed in these chapters as well.
All six fields of research (stereo, registration, tracking, PIV, PTV, optical flow) deal with very similar tasks and solutions. Therefore, the availability of open source implementations of such algorithms in a common framework could even facilitate interdisciplinary research in these fields and might result in a much broader perspective on the tasks at hand.

1.4. Organization

The remainder of this work is organized as follows. We first review challenges of optical flow techniques in Chapter 2. We conclude this chapter with the observation that general-

purpose optical flow algorithms are unlikely to be developed. As choosing between existing application-specific algorithms for a new flow estimation task is difficult, we identify a set of requirements which can be used for the classification and evaluation of existing algorithms in Chapter 3.
We review existing methods in Chapter 4 for finding the largest common denominator in the form of a modular framework described in Section 5. We develop this framework into a set of classes and interfaces that can directly be used as an implementation. The second main part of this thesis deals with the application of the developed concepts to a special field of application: Chapters 6 and 7 deal with the description and analysis of a new particle tracking velocimetry algorithm with applications in biofluidmechanics.
We conclude this work with Chapter 8 by discussing the results and future research topics.

Part II.

Modularity in Optical Flow Estimation

> The mind likes a strange idea as little as the body likes a strange protein and resists it with similar energy. It would not perhaps be too fanciful to say that a new idea is the most quickly acting antigen known to science. If we watch ourselves honestly we shall often find that we have begun to argue against a new idea even before it has been completely stated.
>
> *(Wilfred Batten Lewis Trotter (1872-1939) English surgeon.)*

2. Challenges in Optical Flow Estimation

> A fact is a simple statement that everyone believes. It is innocent, unless found guilty. A hypothesis is a novel suggestion that no one wants to believe. It is guilty, until found effective.
>
> *(Edward Teller)*

In this chapter we show that an abundance of situations exist, in which optical flow estimation is either very difficult or even impossible. We give a detailed overview of these challenges in order to strongly emphasize that even from a theoretical point of view the design of an ultimate optical flow algorithm is neither likely to be found nor practically useful. Based on this argumentation we will motivate a set of optical flow algorithm requirements in the next Chapter 3.
From the implications of these two chapters we will draw the conclusion that a modular and accessible software framework is an important step for the study of the resulting research problems.

To define motion estimation more properly a few definitions suffice. Let $O \in R^3$ be a spatio-temporal region (e.g. the point of view of a video camera and the period of time it is recording this view). Let $I : O \rightarrow R$ be an image sequence mapping from spatio-temporal locations to image intensities. Then, let $F : O \rightarrow R^2$ be the mapping that defines the motion (or flow) of the associated intensity in I with respect to time at each image sequence location. (Please note that this definition is still continuous. When talking about image sequences in this work we always refer to discrete videos acquired e.g. by digital cameras. However, the continuous definition is useful for global methods in which the discretization is introduced only for solving the problem on the discrete sequences.)
Models of motion and brightness variations are needed to obtain meaningful motion estimates. All these models encode prior knowledge in one way or another. The better these models are, the more likely they can be used to obtain highly accurate results. As we will discuss below, the problems in optical flow estimation are so diverse that any model general enough to deal with all of these problems is very likely to be too unconstrained to yield highly accurate results. Therefore, we believe that general solutions for the flow estimation problem are not suitable to obtain the best results. Instead, we suggest to find common modules of optical flow techniques which can be assembled for each task at hand.
So what are the challenges in optical flow estimation? Without any further theory one problem immediately becomes obvious: For each single pixel in the image sequence a

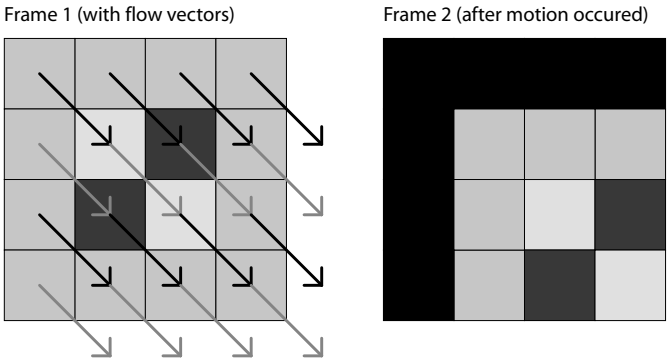

Figure 2.1.: Occlusions occur at the boundaries even if the flow is well-defined.

flow vector with (at least) two values is sought. Where does this information come from? The probably most intuitive idea is to relate pixel intensities or locations, e.g. by comparing or grouping them. This constitutes the very core of motion estimation, be it optical flow estimation, registration, tracking or any other more specific field of research such as particle image velocimetry for fluid flow estimation. Yet, even if this very basic problem is addressed via adequate modeling, a host of additional problems occur due to the image acquisition process. We divide these into three groups of major problems, namely occlusions, ambiguities and ground truth generation. Of course, many of these problems are either obvious or well-known. We will provide a thorough and more general discussion to motivate our hypothesis in the forthcoming chapters.

2.1. Occlusions

In the commonly used definition of occlusion, this term relates to a simple problem: Consider a (discrete) image sequence with two time frames where each pixel exactly moves one pixel to the left and one to the bottom. Obviously, there is a one to one correspondence between the two frames for (almost) each pixel. Thus, at least theoretically, the flow field between the two frames could be computed. But pixels that moved outside the image region cannot be matched. The same holds true for an object suddenly appearing in the second frame, occluding the pixels of the first frame that should actually be matched to compute the correct flow. As a result, even theoretically the motion between the two frames can no longer be estimated for all the occluded pixels. At this point the problem is obvious. Yet, by observing the problem from another perspective, a set of other problems can be understood as occlusion as well.

The first example for a more general notion of occlusion is the following: One might argue to add a binary variable to each location that is true whenever the current pixel is occluded in the second frame. But what if the object appearing in front of the background is half-transparent (e.g. a glass-cup of tea, 2.2)? Clearly, occlusion is nothing digital, it should, perhaps, rather be called opacity.

But this again would not describe the problem in a more delicate thought experiment:

Figure 2.2.: Occlusions cannot be easily modeled by binary variables: This image impressively illustrates various problems ranging from semitransparent objects, optical distortion and light effects. Copyright: HarQ Photography (`naturea.blog94.fc2.com`)

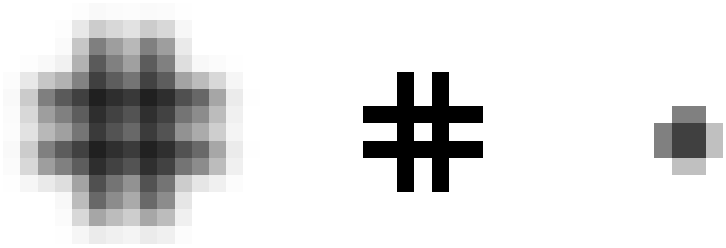

Figure 2.3.: Zooms can be understood as occlusions if the images are discrete. Here we have an extreme case caused by aliasing: The original image in the center looks very different from its enlarged (left) and shrinked (right) version.

consider a camera looking at a purely two-dimensional scene without any motion (e.g. a hash symbol, 2.3) and then zooming in. What happens inside the image? By zooming in, each pixel in the first frame corresponds to multiple pixels in the second frame. Now, what is the "correct" flow? Vice versa, when the camera zooms out, several pixels of the first frame "condense" to a single pixel in the second frame, which can be considered as some kind of occlusion even though the actual, physical image being recorded did not change at all. Of course, we would not have this kind of occlusion-problem with continuous images as zooming in would change nothing in the actual image data.
By adding zooms to the definition of occlusions, in fact any kind of divergence or convergence of motion vectors in a discrete image sequence (that means any motion field that is not constant everywhere as in the first example) contains occlusions. They either stem from the projective nature of the image acquisition system or from the quantization of space and light intensities.
Finally, sensor noise can (arguably) also be understood as occlusion, another major issue of motion estimation. Please note, that interpreting physical occlusions, flow convergences and divergences and noise as several kinds of occlusion is a philosophical subsumption of three problems with different causes. Nonetheless, the effect is similar in that the original intensities are corrupted by other signal sources (be it another physical object, sensor noise or the accumulation of multiple locations at the same image sensor.) These factors can neither be (completely) overcome by technical improvements nor by algorithmic developments. In some examples the image acquisition scheme can be carefully adapted to greatly reduce these problems: Physical occlusions can sometimes be removed physically, noise can (under certain circumstances) be greatly reduced by averaging a large number of frames before recording the next image. Convergences and divergences pose no problem whenever an accurate continuous image representation can be computed e.g. because the analytical form of recorded signal is known a priori. Anyhow, all this idealizing assumptions can only be made if the image acquisition is carefully tailored to the specific type of results one might want to obtain. Therefore, motion estimation algorithms should address those problems specific to the problem at hand: It makes no sense to waste computation time on the reconstruction of a continuous image representation when either subpixel-accuracy is not required or the motion in the sequence contains no occlusions. The same holds true for the next set of problems which we summarize with the term ambiguities.

2.2. Ambiguities

Again, we start with two very simple problems to illustrate in a next step, that these are only instances of a much more general problem. Consider two white images. Where did any pixel of the first image move to in the second image? We cannot know because all pixels are white. Now consider a black line crossing the whole first image and the same black line in the second image, shifted by one pixel to the right (2.4). Assuming that the black line is a foreground object (and not a slit in the actually white foreground) we can again ask where it moved. Did it just move to the right? Or did it also move to the

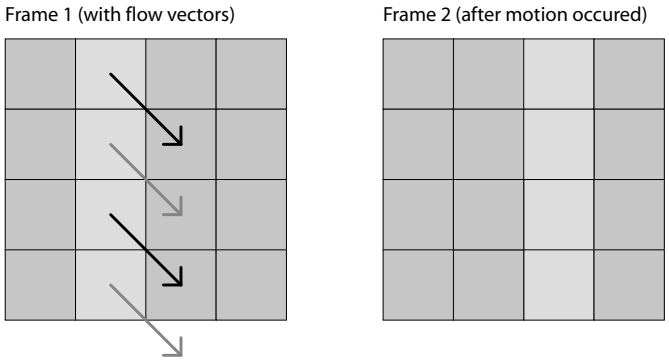

Figure 2.4.: A simple case of ambiguities known as aperture problem.

top? We cannot know as each pixel of the black line in the first image can be matched to any black pixel in the second frame. This problem is generally referred to as aperture problem. After all, this term is rather restricted to the physical interpretation of the image sequence seen through the aperture.

A more general view (which sometimes is called generalized aperture problem) can be illustrated by the following two examples. First, consider a random gray value image consisting of only two distinct values: white and black. The second frame is just another random image of the same type. Any white pixel could now be matched to any white pixel in the second frame: The interpretation very much depends on prior knowledge on the motion in the scene. A second example is a bright, two-dimensional Gaussian on a black background in both frames. In the second frame, the Gaussian becomes smaller or darker or both. It is impossible to say which is true when looking at a few pixels only. However, we could find a model that assumes that the decrease of brightness is associated with a three-dimensional motion, where a greater distance to the camera results in darker pixels. Hence, by introducing a third motion parameter for the depth, this problem can be solved unambiguously.

This leads to the generalized aperture problem: Whenever the used image information and/or model does not contain enough information to retrieve all unknown parameters (be it motion in 2d, 3d or even additional parameters), the number of degrees of freedom that remain constitute the generalized aperture problem. In the example of the black line, only the motion perpendicular to the line can be estimated correctly; in a homogeneous image region, both motion parameters can have arbitrary values. This is just a rough overview of these problems. A more detailed discussion with a proper formal definition, using the notion of intrinsic dimensions, can be found in [26]. Yet, ambiguities are another example why an ultimate motion estimation algorithm is unlikely to be found: Whenever ambiguities exist in the image data, one has to decide for one of the possible solutions. Therefore, prior knowledge is necessary, which is usually application-specific. The model encoding this knowledge should neither be too general nor too specific (to either reduce the chance of interpreting the ambiguities wrongly or to reduce the chance of model overfitting).

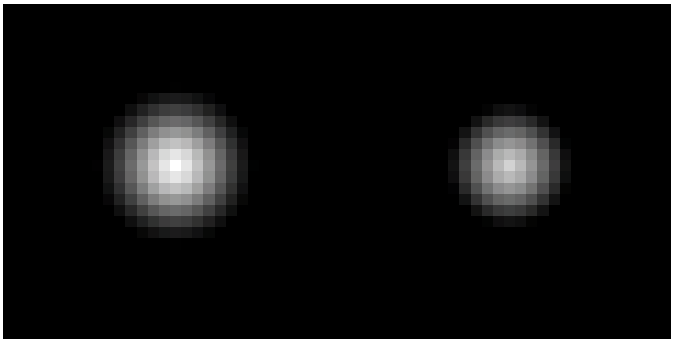

Figure 2.5.: An instance of the more general interpretation of the aperture problem arises when objects change their intensities relative to their distance to the camera: Is the right image smaller or just further away from the camera? Both interpretations are correct without additional knowledge.

2.3. Ground Truth Generation

We have shown that there is an abundance of situations where optical flow estimation is either very difficult or even impossible per se (due to occlusions) or where not all motion parameters can be estimated reliably (due to ambiguities). In both cases application-specific knowledge can be used to infer accurate motion fields or at least to infer at which locations motion can be estimated at all.
Another question is how the results of an algorithm can be validated. The typical approach is to design so-called ground truth image sequences where the motion is known. Such sequences can either be synthetic and as simple as those described in the examples above or they can be more complex as for example those generated with e.g. raytracing programs. The most famous examples are the Yosemite sequence [27], the street and office sequences [28] and the diverging tree sequence [9]. These have become the standard synthetic test sequences. Of course, they do not cover all types of applications and can therefore only be used as a hint on how the algorithm might perform on other sequences. The Yosemite sequence contains very small and relatively large motion vectors with a magnitude of up to roughly four pixels. The office sequence contains a zoom and a few motion boundaries. The street sequence contains a car with very strong motion boundaries. One problem with such sequences is that it is largely unknown whether they represent important or typical cases of motion together with the rendered images.
Furthermore, there are sequences which are acquired with a real camera. The first well-known example is the marbled block sequence [29] which contains a few block-shaped, textured objects standing on a textured underground.

Recently, a number of new synthetic and real sequences have been generated by [30]. We believe that their effort is an important step towards more ground truth data. Furthermore, they encourage the publication of results based on a website were everyone can submit new motion fields. This is another relevant contribution to facilitate further research.

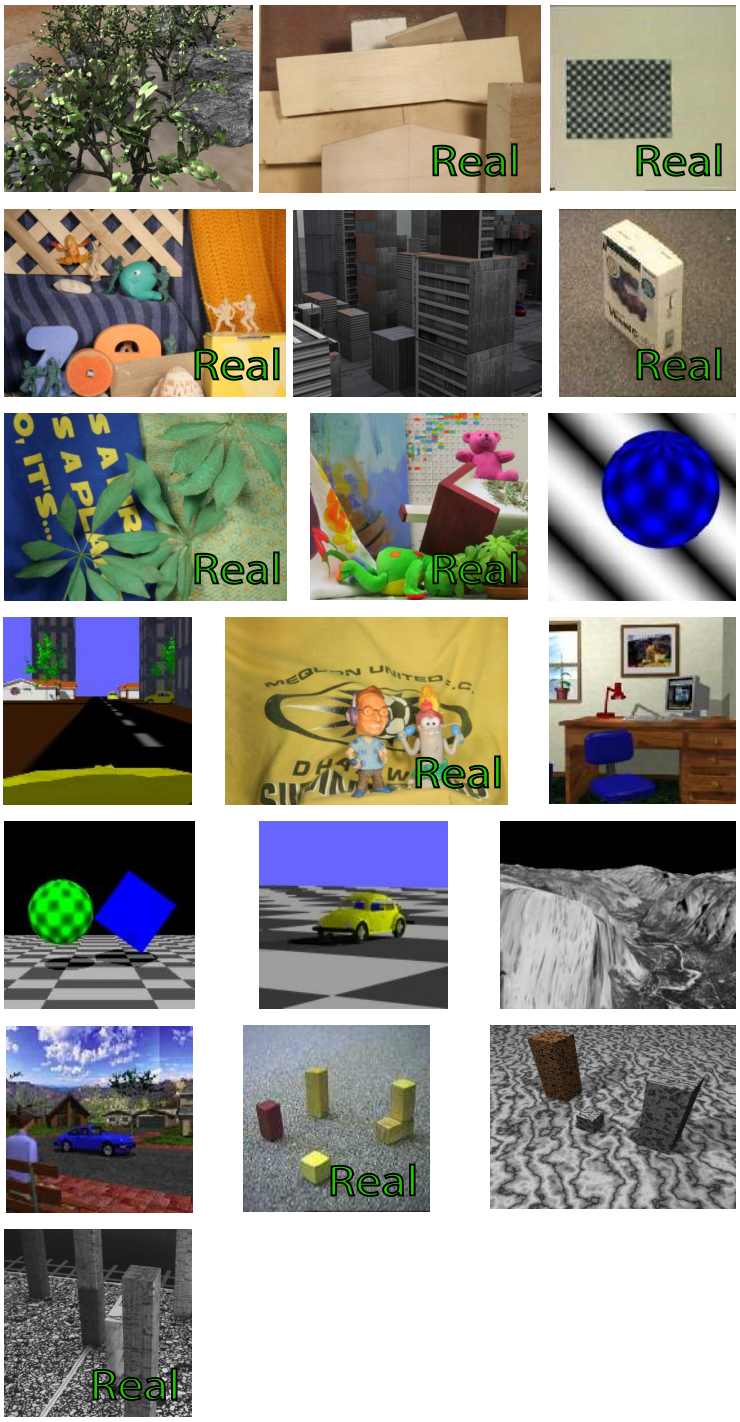

Figure 2.6.: The set of commonly used optical flow sequences with known ground truth. As can be seen immediately, the data does not cover a wide range of image sequence types.

The generation of ground truth data is a challenging optical measurement task itself. Its accuracy should ideally be magnitudes above the accuracy that can be achieved by motion estimation algorithms. The typical problem of real sequences is the estimation of this accuracy. In both publications mentioned above the information supplied from an optical measurement perspective seems to be insufficient to clearly state accuracy limits. Hence, even though in real sequences all physical imaging effects from lens distortion and noise to light reflections and refractions are modeled properly, it remains unclear whether their ground truth is good enough.
New motion estimation algorithms are usually tested against a subset of these sequences. Until recently, they where often solely tested against the Yosemite sequence. As an error measure usually the so called average angular error (defined by [31] and used by [9] and most successive papers) and its standard deviation over a single frame of this sequence is reported. Not only is this error measure unmotivated, it also is inappropriate for the comparison of some typical problems of motion estimation as is e.g. laid out in [32].
To put it in a nutshell, to the best of our knowledge, it is unknown whether current test sequences actually can be used to describe the accuracy of an algorithm and the performance measures are questionable. A lot of future research could be carried out in this field. One especially interesting topic would be to find out whether synthetic test sequences can sufficiently approximate real test sequences. This would help create large amounts of sequences for all kinds of possible applications.
Even if ground truth data could be easily generated, it would still be questionable, whether a generalization of the image data created across all fields of applications can be found. Thus, the quality assessment of the quality of a general-purpose optical flow algorithm might still be impossible. (In fact, the term "general-purpose" looses its meaning with the diversity of possible applications in mind.) Therefore, even if a general-purpose algorithm would be found, we would probably never be able to identify it.

2.4. Conclusion

We have discussed that optical flow estimation still is a difficult task as well as the evaluation of its performance. The challenges described in this chapter are numerous and diverse. As well, the possible ranges of applications for optical flow methods are broad. Thus, we believe that it would be insensible to aim at designing an ultimate optical flow algorithm. On the other hand, new techniques should still try to be as general as possible in order to cover as many applications as possible. We believe that a modular design of optical flow algorithms as described in Chapter 5 can ease this conflict to some degree. Furthermore, such a modular design has many more advantages such as the ability to more effectively compare the effects of exchangeable modules as will be discussed later.
The problem of how to evaluate such newly designed algorithms not only with respect to their accuracy is another open problem. Therefore, in the following chapter, we suggest a broader set of optical flow algorithm requirements which can be used to evaluate and to

describe optical flow algorithms. Based on these requirements and the insight that many of todays optical flow algorithms are already implicitly designed in a modular manner (cf. Chapter 4) we will then discuss the implementation of a modular framework in Chapter 5. In the same chapter we will further substantiate our hypothesis postulated here by showing some experimental results obtained by this framework.

3. Requirements for Optical Flow Algorithms

> Every science begins as philosophy and ends as art.
>
> *(Will Durant, The Story of Philosophy, 1926)*

Occlusions and ambiguities often render the motion estimation problem unsolvable per se and systematic ground truth generation is very difficult. To deal with application-specific subsets of the problems described above, specialized models can be helpful in obtaining superior results. Philosophically, motion estimation rather is the art to adequately exploit known constraints of the application-specific image acquisition process and the actual information recorded to achieve results as good as possible. Or, in other words: We believe that (once a set of useful models has been developed by scientists) motion estimation mostly is an application specific engineering problem. How can computer science help engineers to develop a motion estimation algorithm that perfectly meets the requirements? As discussed above, it would be helpful to break down the whole problem into subproblems which can be studied more thoroughly. These algorithmic elements could then be plugged together to new motion estimation algorithms. Additionally, a set of properties (or a specification) needs to be found that fully describes the input and the output of each algorithm under every circumstance so that engineers can deal with them as black boxes. The accuracy of the output is one of the most important properties. Nevertheless, we believe that it is not the only desirable property of a motion estimation algorithm. In the following we will suggest a general optical flow algorithm specification scheme. A subset of these points has already been addressed in literature, some points have not. We also shortly discuss the challenges of defining these general specifications in the field of optical flow estimation. The algorithms described in the third part of this work will exemplarily be tested against these algorithm requirements.

3.1. Accuracy Limits

As stated above, the study of the accuracy of the outcome of an motion estimation algorithm is of major importance. There are several ways to test and compare accuracies of such algorithms. A major problem is how to measure the error because there is no order (or ranking) defined between two vectors. Hence, each pair of vectors (i.e. ground truth and measured flow vector) first has to be transformed into scalar values in order to be comparable. There are many ways to turn vectors into scalar values. One common way

is to compute the magnitude of both vectors. This is problematic when ground truth vector and measured flow vector are on the one hand equally long but on the other hand point into opposite directions. The magnitude error would still be zero. Another way would be to compute the angle between two vectors which raises the analog problem: The vectors can be of different magnitude. Another problem here is the singularity for vectors of very small magnitude.
To weight these two components of magnitude and angle the so-called angular error defined by [31] has been suggested.
But this error weights both parts of the errors in a nonlinear and unintuitive manner which is actually not motivated in the paper [33].
Recently, another effort has been made by defining the so-called endpoint error [30] which simply is the magnitude of the difference vector between ground truth and measured flow.
Depending on the application one error measure or another might be favorable, a fact that should be taken into account when stating the accuracy limits of the algorithm.
Once an error measure has been defined, the error distribution needs to be sufficiently motivated. The problem here is, that this distribution actually depends on image data, ground truth and measured flow. For example, testing of the accuracy with a highly textured region that moves a constant velocity everywhere yields very low errors with most algorithms. If the images were of constant color the results could be completely wrong; if the motion of the ground truth was absolutely arbitrary, too. Hence, testing on a sequence like Yosemite does not adequately represent the quality of the algorithm. It just gives a hint that for this type of scene (highly textured, smooth and mostly small motion) the algorithm might actually work well. Furthermore, representing the error distribution only by its mean and variance is not sufficient, because only the Gaussian distribution can be fully described by these first two moments. As motion estimation errors are far from Gaussian it might be more helpful to actually visualize the whole distribution (or parts of it) which in turn raises the problem of density estimation.
Finally, it would be helpful if it was known under which circumstances the most accurate results can be achieved by an algorithm. At first sight this sounds easy to answer: Constant motion through time and much texture certainly is a simple case. Yet, a Gaussian intensity distribution in a 32 bit quantized image might even yield very accurate results for non-constant motions such as a rotation. Furthermore, it is interesting to which degree the results deteriorate with respect to more challenging image data.
Although the accuracy is a very important part of a motion estimation algorithm specification, and much more research efforts can be put into this topic it is not part of this work. However, we will keep the considerations made here in mind for a new algorithm proposed in Chapter 7.

3.2. Estimatibility and Confidence

Even though an exact solution of the optical flow problem is impossible, a host of algorithms have been developed which show surprisingly high accuracies on some test image

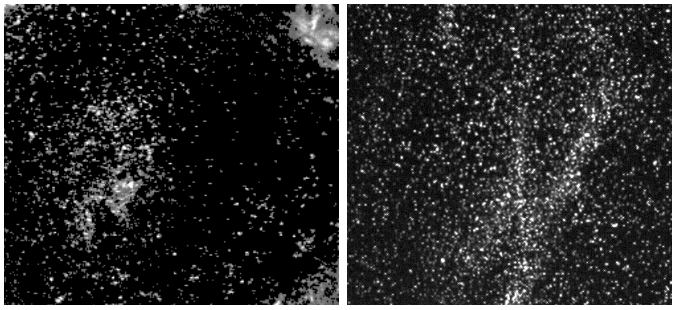

Figure 3.1.: Two examples for data resulting from particle tracking velocimetry experiments. General optical flow estimation techniques often fail on such sequences. Please note that this data significantly differs from the commonly used data sets in Figure 2.6.

sequences. Countless industrial applications for optical flow estimation exist. Hence the question is: How do occlusion and ambiguities affect the results?

For example, a typical image sequence for particle tracking velocimetry (PTV) (cf. Figure 3.1) consists of a mainly black background and some hundred (or thousand) bright moving spots. Occlusion occurs whenever two particles are crossing due to the projective nature of the image acquisition. In the black (homogeneous) regions of the background no motion can be estimated: A black spot at any location can be matched to almost any other location in the next frame. We do not care about the occlusion of the background and ambiguities in the background and assume that there is no motion at all. However, when two or more particle images are crossing each other, we are interested in which particle is which, once the crossing is over in order to properly reconstruct each of their trajectories.

Even though there is a lot of occlusion in this image sequence, it does not matter much because we can still quite accurately infer all particle trajectories from the images. Hence, what matters is how much information we need to obtain from the given data, depending on the intended later use of the resulting motion. In PTV we are satisfied with a few but accurate trajectories of particles. In other applications, such as image stitching, only the global affine motion, together with some lens distortion correction is sought. In tracking, just the location or the 3d-transformation of a single object relative to the camera is the interesting motion. In registration only the boundary motion of a heart recorded via radiography might be of importance.

All the above examples show that occlusions can often be ignored or treated separately by some motion postprocessing or image preprocessing. On the other hand it is obvious that an occlusion of the object to be found in a tracking approach needs to be dealt with or that a discontinued heart boundary should be correctly interpolated in an ultrasound image. From a broader perspective dealing with occlusions and ambiguities can be understood as dealing with estimatibility: Instead of assuming that at each pixel of an image sequence a full flow can be estimated, we pose the question whether motion can be estimated at all and, if so, how many parameters of it. The intuition (based on

algorithm complexity notions) here is that the decision whether motion can be estimated at all is easier to make than to actually carry out the estimation. Details on this topic can be found in [26].
Yet, once the flow has been estimated at all those locations which are estimatible, the question remains on how accurate the estimate is. Here we use the notion of confidence measures. Intuitively, the complexity of estimating the confidence of a flow vector lies between the complexity of estimatibility and optical flow computation. Again, this is a difficult problem and some research has focused on it up to now. Two recent publications are [7, 8].
Hence, the notions of estimatibility and confidence somewhat relax the problem of motion estimation by introducing continuous variables that supply the user with additional information on the success of the algorithm. Hence, a requirement of a motion estimation algorithm would be to supply the user with such information.

3.3. Range of Applications

Most development of motion estimation algorithms is driven by the need to solve a specific problem like weather forecasting, tumor growth analysis, industrial quality inspection, etc. Therefore, each algorithm focuses on a specific aspect of motion that needs to be estimated accurately. For example, block matching (or correlation based) algorithms as e.g. [36] are able to find very large displacements easily; yet they often are only pixel-accurate. Another example are local least squares methods [17, 37] where large motions cannot be estimated well but small motions are computed with very high accuracy.
Not only the length of the flow, but also its spatial and/or temporal smoothness are of interest. Some methods can very well deal with affine image transformations as they directly model such motion patterns. Dealing with motion discontinuities (occlusions in the images) can also be such a feature.
Summing up, it would be interesting to know which algorithm performs especially well on which kind of data and also whether it can be applied to which range of other images and flows. To broaden this range, learning might play a special role: If it where possible to learn models of motion and image data, an algorithm could be adapted to a specific application by choosing an appropriate training set. It is often unclear, which algorithm is the state of the art in which type of application. Future research could therefore e.g. focus on the design of a decision process for the choice of an appropriate algorithm given a set of typical image sequences in a specific application. Automatizing this process would be of great help for users with little expertise. Therefore, the range of applications is another important algorithm property that needs to be carefully described in the algorithm specification.

3.4. Execution Speed

The time an algorithm needs to actually estimate the motion of an image sequence usually is a major issue in industrial applications. Basically, four aspects can be addressed to improve the speed. These aspects range from practical over completely theoretical to technically highly intricate considerations; to each of these a complete field of research is dedicated. Therefore, it is very difficult to judge the execution speed of an algorithm even though it is one if its important properties.

3.4.1. Data Reduction

Sometimes, it suffices to only compute motion at a few locations. Hence, computation time can be saved by finding algorithms that reduce the number of locations. This is a typical approach in tracking [22] where usually only very few pixels of an image sequence are investigated.

3.4.2. Mathematics

For example in global motion estimation techniques (which often include the solution of a system of partial differential equations) large linear systems of equations are generated from the image sequence. The solution of these systems of equations can be carried out by a host of methods, ranging from Gaussian Elimination Schemes over Krylov Subspace Methods to Algebraic Multigrid Schemes. Exploiting mathematical properties of the problem to be solved can dramatically reduce computation times. This was for example shown by [38, 1].

3.4.3. Parallelization

At least with the dawn of multicore desktop computers, parallelization has become a major topic for all types of applications. Especially in image processing where huge amounts of data are often processed with the same operations (consider e.g. the convolution of an image with a mask), parallelization is surprisingly easy to implement. But also solving large linear systems of equations can be done in parallel with a little more effort.

3.4.4. Code Optimization

It might sound trivial to optimize source code but with the advent of a diversity of large image processing libraries for major programming languages used for image processing (as e.g. C++ and Matlab) code optimization is far from simple. Nonetheless, this part can also affect theoretical considerations: If it were for example easier to optimize code for matrices than for other data structures, the choice of the optimization method interfered with the actual code design. The main problem today is that the programmer needs to have a deeper understanding on how image processing libraries implement their functionality in order to optimally exploit its internal structures. Another problem

is that the ways in which compilers optimize code is rather unintuitive: One cannot implement all functions in the same way to yield the same code optimizations by the compiler. The typical approach here is trial and error - a method which is not simplified by the fact that each compiler optimizes its code differently so that the same code can be orders of magnitude faster when compiled with a different compiler.
Hence, investigations into the various execution speeds of an optical flow algorithm are another property to be specified.

3.5. Modularity

A common practice in the publication of motion estimation algorithms in all fields of research is to describe the whole algorithm and to test its output against test sequences. Regularly a few crucial parts of the algorithm are either left out or parameterized differently in order to estimate its effect on the overall results. Without diving into details, for example the most sophisticated optical flow algorithms are built up from many algorithmic elements, such as multiple similarity measures, image derivative kernels, interpolation techniques, pyramid computation schemes, regularization terms and so on. Each of these elements has parameters and can even be replaced by completely different methods. For example, subpixel image intensities can be interpolated by a large number of interpolation schemes ranging from linear to sinc interpolation; an image pyramid can be computed by scaling the original image down by a factor of two or smaller or it can even scale the image up to some degree [39]; the derivative of an image can be computed by a huge set of kernels or even other filtering techniques ranging from simple central differences to sophisticated filters specially designed to estimate motion with a specific similarity measure [40]. Any subtle change in these settings can influence the overall accuracy of the results and is therefore worth further investigation.
At the core of this problem lies the fact that any motion estimation algorithm is actually plugged together from a large set of modules available. Some of these modules as for example image derivative computation are fields of research on their own. It would be helpful if there were a set of known slots (constituting the elements of the most general motion estimation algorithm and clearly defining input and output data) and a variety of possible modules that could be plugged into each appropriate slot. Then, each slot or module could be scientifically investigated separately and also in its combination with other modules. We call this property modularity. A motion estimation algorithm is modular if it fits into a general motion estimation framework and each of its algorithmic elements can be exchanged. Furthermore, in an implementation of such an algorithm we demand that all of its parameters can be adapted from the outside of the actual program so that other scientists can experiment with the algorithm. One major contribution of this work is the proposal of such a general framework in which most of todays known algorithms fit into and a software environment in which many of these algorithms are implemented in a modular manner. The modules of an optical flow method are another interesting algorithm property.

3.6. Accessibility of Implementations

Hundreds of motion estimation algorithms have been suggested in the past 30 years. A few of them have become famous (e.g. [17, 41, 36]) as predecessor of most of todays known techniques. A typical problem of most techniques is that papers being published describe the idea and some results, but usually do not supply the reader with the actual implementation of the algorithm. As a result, a comparison of a large number of algorithms is merely impossible due to two reasons: First, each implementation of the same paper will slightly differ due to imprecise formulations in the publication, forgotten details and so on. This is for example the case with the method of Black and Anandan [5]: There are three implementations listed in the Middlebury data set [30] which differ in their results. Second, the implementation of a paper can be very time-consuming. This limits the number of algorithms that can be implemented in a reasonable amount of time.
From an engineering perspective, this problem becomes even worse. Modern motion estimation algorithms utilize for example new optimization techniques (as e.g. graphical models) which had not been developed a few years ago, so that their implementation can be very time consuming. Furthermore, as many of the above described algorithm properties of motion estimation algorithms are unknown for many published papers, one can easily spend large amounts of time implementing techniques which are actually inappropriate for the task at hand. Thus, the implementation and comparison of such algorithms for any scientific or industrial applications often is extremely costly and, therefore, infeasible.
One might argue that the purpose of public fundings of science by a country is to create an open and free institution that allows for unconstrained research in order to discover knowledge which turns out to be useful for its citizens in the long run. Assuming that this philosophical understanding of science is valid at least to some degree, the accessibility of implementations of motion estimation algorithms as a direct research result should be a requirement for all involved parties, including scientists and engineers.
Accessibility has several aspects. First, the source code of an algorithm should be open to the public to facilitate further research and industrial feasibility studies. Second, the implementation should be as simple as possible as the theory allows and be well documented. Third, the parameters influencing the output of the algorithm should be small in their number, intuitive to understand and insensitive with respect to input data. We contribute to this point by defining and implementing a framework in which motion estimation algorithms can defined in a platform-independent, open source environment.

3.7. Conclusion

In this chapter we have discussed the importance of algorithm requirements to describe the properties of optical flow techniques. Requirements can be used by scientists as well as engineers to design improved algorithms or choose between several options for a given application. We have identified six algorithm properties on which requirements can be

defined: The accuracy of an algorithm should e.g. be studied to validate the model choice. Ideally, each algorithm should provide information about the estimatibility and flow confidence of each image location. It should be clear in which fields of application the algorithm can be applied. Complexity, computation speeds and possibilities for the improvement of both properties should be investigated. The algorithm should ideally be implemented in a modular way so that individual modules can be replaced by improved ones upon availability. Finally, we believe that the implemented algorithms are an integral part of the research results and should therefore be made available as open source code.

We have carefully motivated these requirements. Furthermore, we have discussed the problems of testing for this suggested set of algorithm properties.

To overcome these problems, we will specifically address the modularity of existing techniques in the following Chapter. The resulting implementation described in Chapter 5 can then be used to carefully analyze whether the algorithm requirements are fulfilled.

4. On the Modularity of Existing Optical Flow Techniques

> The important thing in science is not so much to obtain new facts as to discover new ways of thinking about them.
>
> *(William Lawrence Bragg)*

Motion estimation can be divided into three main steps: First, a model is defined that describes how the actual motion results in an image sequence. If this model is formulated with continuous variables, these are discretized in the second step. Finally, the model is fitted to the image data which is the optimization step.

4.1. Algorithmic Building Blocks

Hence, for a modular software an investigation of these steps is used as a guide to design the algorithmic building blocks. We will first investigate various models in local and global optical flow estimation. Then, we will discuss how these models are discretized and which options exist for the optimization step.

4.1.1. Modeling

In 1981, Horn and Schunck [41] suggested to define an energy functional consisting of two terms. The first term is called the Brightness Constancy Constraint Equation (BCCE) and is derived as follows. Let $I(\vec{x}, t)$ be a two-dimensional image sequence and let $\vec{u}(\vec{x} - \vec{x}')$ be a two-dimensional flow vector at location $\vec{x}$. Assuming that the brightness of a single pixel remains constant along its trajectory of motion, the difference of the intensity between the first image at location $\vec{x}$ and the second image at location $\vec{x} + \vec{u}(\vec{x} - \vec{x}')$ should be small. Horn and Schunck decided to penalize the deviation from this brightness constancy by defining an energy $E_{data}(\vec{x})$:

$$E_{data}(\vec{x}) := (I(\vec{x}, t) - I(\vec{x} + \vec{u}(\vec{x} - \vec{x}'), t + \delta t)) \tag{4.1}$$

In a second step they chose to linearize this energy by a first-order Taylor-Expansion which (after some rearrangements) results in the BCCE:

$$E_{bcce}(\vec{x}) := (\nabla_{x,y} I(\vec{x}, t)\vec{u}(\vec{x} - \vec{x}') - \frac{\delta}{\delta t} I(\vec{x}, t)) \tag{4.2}$$

Here, $\nabla_{x,y}$ defines the image derivative in x and y-direction. In a second term, they assumed constant motion in the whole image. Any deviations from constant motion are penalized by evaluating the derivative of the flow field:

$$E_{reg}(\vec{x}) := \nabla_{x,y}\vec{u}(\vec{x} - \vec{x}') \tag{4.3}$$

This term relates neighboring flow vectors to each other. Both energies are weighted quadratically and integrated over the whole image:

$$E := \int_{\Omega} E_{reg}(\vec{x})^2 + E_{reg}(\vec{x})^2 d\vec{x} \tag{4.4}$$

Due to the regularization term, the energies cannot be optimized locally. Therefore, methods based on this technique are called *global*. Until today, the method of Horn and Schunck is the basis for most global optical flow techniques.

In the same year, Lucas and Kanade proposed another method for optical flow estimation. It is based on the same linearized brightness constancy equation 4.2. They also assume that motion is locally constant. The only difference between the methods is the way how spatially connected flow vectors are related to each other. Instead of a global regularization term that connects all pixels to each other, they proposed to collect BCCE equations in a neighborhood around the pixel location to be estimated in order to create an overconstrained linear system of equations which they solve by means of a least squares fit. The system is linear because the BCCE has been linearized before any optimization is carried out. A least squares fit of a nonlinear data term would be possible as well.

Due to the finite neighborhood which is investigated per pixel location, the method of Lucas and Kanade is called *local*. It also is the basis for many other optical flow algorithms.

Both methods have been refined by a number of extensions which will now be reviewed. This review is by no means exhaustive. Yet, it gives many examples for a well-motivated, modular concept of motion estimation software.

Regularization Techniques A host of regularization terms $E_{reg}(\vec{x})$ has been suggested for global methods. Nagel and Enkelmann [42] modified the spatially isotropic regularizer based on the image gradient. Furthermore, it is possible to define anisotropic regularization based in diffusion tensors. The diffusion direction can either be determined by the flow or by the image data. An overview of such convex regularizers can be found in [43]. More complex regularizers have for example been developed by Cremers et al. [44] by simultaneously segmenting the image into regions of constant motion.

Data Terms The method of Horn and Schunck as well as the method of Lucas and Kanade rely on gray scale images. However, the BCCE can easily be extended to color images: Since the equation should also hold for other color channels than the first, one can simply include those constraints with additional BCCE equations. Other options have been investigated by [45, 46, 47].

Another data term is not to linearize the BCCE equation 4.2 and use the nonlinear version 4.1 instead. This results in a nonlinear problem which has for example been studied for global methods by Papenberg et al. [14].
To model physically meaningful brightness changes, Haussecker et al. [13] derived BCCEs with additional parameters that allow to simultaneously estimate brightness variations with respect to physical models. These equations have originally been designed for local methods, but they can easily be inserted into global methods as well. Further brightness variation models have been developed by Gennert et al.[48] and Garbe et al.[49].
To render the data term more illumination invariant, higher order image derivatives have been proposed as data terms [50, 51, 14].
Finally, Bruhn et al. [12] realized that a combination of local and global methods is possible by integrating the spatial neighborhood of local methods as data term into a global approach.
All approaches have in common that a data term per pixel location based on its local surroundings is used as brightness variation model. This model is based on a set of two or more (in the case of [13]) parameters that are inserted into this model.

Robust Measures Robust data terms are mainly used to deal with motion discontinuities usually caused by occlusion. Seen from a more general perspective, the spatial relation between flow vectors can no longer be described by a single set of parameters in such cases. Therefore, the model assumptions become invalid for a subset of pixels in the local neighborhood. For local methods this yields wrong flow vectors in those regions. For global methods motion discontinuities have an impact on the whole flow field.

Lai and Vemuri [52] analyzed the second image derivative to estimate whether the image curvature at a given location is sufficient to estimate a flow at these locations. In regions were this is not the case, they dropped the data term and used the regularizer only. This was an early approach for robust data terms and has been followed by many researchers.

Black et al. [53] suggested robust statistical measures such as the truncated quadratic or the Lorentzian function to nonquadratically weight the brightness model residuals. Other authors suggested to use various non-Euclidean norms as e.g. the L1-norm [14]. Bruhn et al. [12] chose to use the Charbonnier-function for weighting. Such functions are not only used for the data term in global methods but also for the regularization term. Usually, the choice of weighting function is found out experimentally and motivated by improved results.

The same methods for weighting the residuals of brightness models can be used for local methods. For example, a robust method has been developed by Bab-Hadiashar and Suter [54] which iteratively samples from the set of BCCE equations and filters the results with respect to such residuals. Other ways to weight the BCCE equations have been analyzed by Farnebäck [6]. From a more general perspective, robust least squares fitting is a whole field of dedicated research [55]; any of such methods can be applied

to local methods. Here, the boundary between the optimization scheme and the actual data term can become blurred, as the iterative refinement of the neighborhood can be driven by the optimization process.

Motion Models Constant motion is a model so simple that it often is an invalid assumptions in real world applications. Therefore, whenever neighborhoods of more than one pixel are considered, a more accurate model of non-constant motion in this neighborhood is interesting for improved results.

Cremers et al. [44] showed that with a piece-wise affine motion model in a simultaneous segmentation framework, very good results can be achieved on suitable sequences (e.g. the flower garden sequence `www.cs.brown.edu/~black/images.html`). Recently, Nir et al. [56] incorporated motion models into a standard global approach by estimating all model parameters at each pixel location. Instead of regularizing the flow, they regularize the parameter set that describes the flow. With this technique, on the Yosemite sequence they showed results superior to previous methods.

Local optical flow techniques have often been augmented by motion models as well. Contributors are for example Bergen et al. [57] and Farnebäck [58]. An overview over motion models is given in the review-paper of [59].

Furthermore, the estimation of multiple motions per pixel has been studied both in global [60] and local methods [61, 5, 62, 63]. For the latter, this is for example achieved by fitting two or more separate models to the same data while reweighing those BCCE equations which belong to the other models respectively.

4.1.2. Discretization and Optimization

Without diving into details here, the next two steps would be discretization and optimization. For discretization both image and flow derivatives need to be computed which can be done in various ways. The optimization of global methods is usually carried out by computing the associated Euler-Lagrange Equations, a system of partial differential equations which can be derived analytically by means of calculus of variations (e.g. [64]). On the other hand, this energy term can be directly optimized, e.g. by gradient descent. For local methods, least-squares techniques like ordinary least squares, total least square, least median of squares and others are applied. An overview of such methods can for example be found in [65]. In the past years, statistical methods have become increasingly popular (e.g. [66, 67]). In recent work, the models are nonlinear, so choosing the best suitable optimization technique is of high importance in order to avoid the convergence to local minima. However, regardless of the chosen optimization technique, the motion and brightness models remain the same, only the point at which the problem is discretized varies due to the choice of optimization method.

Multiresolution Techniques A big problem of the BCCE is that it cannot capture large motion when the image texture is too fine. The reason is the limited accuracy of the first-order approximation of the image derivatives. To facilitate this problem, an image pyramid is computed, following the idea that for a smaller image, the motion also is

smaller and can hence be estimated more easily. The result on a coarse scale is then upsampled to the next finer scale on which a new flow estimate is computed based on the results of the coarse scale. As simple as this method sounds, it brings a number of problems and parameters with it:

- Image and flow need to be resized. As both are defined on a discrete grid, an interpolation scheme is needed. Especially for upsampling flows in regions with occlusion boundaries results in large artifacts on the next finer scale. Many schemes exist [68], but usually only linear and bicubic interpolation are used.
- A downsizing factor and a number of pyramid levels have to be decided. These numbers can have significant effects on the overall-outcome of the algorithm. This has for example been studied by [16]
- Following this approach, any non-linear function could be used for downsizing. It would even be possible to try to locally find the optimal sampling parameters for flow estimation. To the best of our knowledge this approach has not been tried yet.
- The images could even be upsized for more accurate flows. This idea has for example been studied by [39].
- Additional problems occur when the downsizing of the image modifies its content. Consider for example a checkerboard-image with alternating black and white pixels. Downsampling by a factor of two with local averaging would yield a gray image. This is not a purely academic problem as for example in particle image velocimetry [23] very small image structures (which visualize a physical flow) are of high importance for the motion estimation. Downsampling can simply remove this information.

It is important to note that pyramids are used to overcome problems which are caused by the linearization of an essentially nonlinear problem. Using pyramids therefore accounts to employing more sophisticated optimization strategies while maintaining the same model. Bruhn et al. [12] showed that such multiresolution methods are closely related to multigrid methods for the solution of large linear systems of equations. Major differences are that multigrid methods still solve a linear problem, iterate on the residuals of these problems and that they cycle through the levels instead of doing a single sweep from coarse to fine.

Computation of Derivatives The computation of image derivatives is another field of research that plays an important role in image processing. Beginning with simple forward difference schemes and local derivative averaging approaches (e.g. Sobel and Prewitt), many approaches to image derivative estimation have been studied. In [33] the Sobel filter was optimized with respect to rotational invariance. Scharr [40] as well as Krajsek and Mester [69] constructed derivative filters specially suited for BCCE-based optical flow estimation and showed significant accuracy increases for various applications. Farid

and Simoncelli [70] studied the problem in a more general manner and therefore retrieved another set of linear derivative filters.
Closely related is the field of interpolation. For example the approach of Unser [71, 72] creates a polynomial approximation of the image where accuracies increase with the order of the polynomial. Knowing the polynomials that constitute the image, its derivatives can analytically be calculated at very little computational cost. Please note that knowing the exact image derivatives at each image location is equivalent to knowing the flow, as the BCCE can directly be solved. (Here, we still have to assume that exactly two neighboring pixels have the same flow but linearly independent gradient vectors.)
Hence, the choice of the derivative estimation algorithm can greatly alleviate the problem of motion estimation.

4.2. An Optical Flow Model Suited For Modular Implementations

As described above, motion estimation can more generally be understood as the parameter estimation of the motion model. Therefore, let $\mathcal{P} \in \mathbb{R}^n$ be the set of parameters that describe the motion and $\mathcal{U} \in \mathbb{R}^m$ be the actual motion. Please note that U can have more than two dimensions when the motion is three-dimensional and also estimates brightness variations. We call the mapping

$$M(\vec{p}(\vec{x})) : \mathcal{P} \to \mathcal{U}, \tag{4.5}$$

which converts a parameter set into an actual motion vector, the *motion model*.
Furthermore, we define the *brightness variation model* to be the mapping

$$B(\vec{u}(\vec{x})) : \mathcal{U} \to \mathbb{R}, \tag{4.6}$$

which inserts the motion vector into the image sequence at location $\vec{x}$ and returns a scalar value which can for example be called matching quality, energy, potential or log-likelihood e. In the simplest case M is the identity function and B is equivalent to Equation 4.2.
In both global and local methods, the result of B is often nonlinearly weighted with a function

$$\Psi(e) : \mathbb{R} \to \mathbb{R}. \tag{4.7}$$

This function is regularly used to gain robustness with respect to outliers generated by B. For B, often a local neighborhood $\mathcal{N}_{\vec{x}}$ is considered around $\vec{x}$. Furthermore, sometimes more than one brightness variation model is applied with linear weights λ_i:

$$E_{data} = \sum_{i=1}^{k} \lambda_i \sum_{\vec{x}' \in \mathcal{N}_{\vec{x}}} \Psi_i(B_i(M_i(\vec{p}(\vec{x} - \vec{x}')))) \tag{4.8}$$

In global methods, one or more additional regularization terms are added that relate motion parameters or vectors locally:

$$E_{reg} = \sum_{j=1}^{l} \lambda_j \mathcal{R}_j(\vec{p}(\vec{x}), M_j(\vec{p}(\vec{x}))) \tag{4.9}$$

For local methods, $\mathcal{R}$ is the identity function so that local optimization techniques as for example least squares methods can be applied. These two terms describe all of the above described models (4.1.1) for motion estimation:

$$E(M, B, \Psi, \mathcal{N}_{\vec{x}}, \vec{p}(\vec{x}), \vec{x}) = \sum_{i=1}^{k} \lambda_i \sum_{\vec{x}' \in \mathcal{N}_{\vec{x}}} \Psi_i(B_i(M_i(\vec{p}(\vec{x} - \vec{x}')))) + \sum_{j=1}^{l} \lambda_j \mathcal{R}_j(\vec{p}(\vec{x}), M_j(\vec{p}(\vec{x}))) \tag{4.10}$$

This energy can be formulated using formal mathematical tools from statistics, analysis, or other terms. Here, we concentrate on a way how to implement such methods, regardless of their mathematical language used to describe the model.

4.3. Conclusion

By reviewing some exemplary publications on optical flow estimation, we have shown that these algorithms can be described by a single equation consisting of interchangeable functions for the various types of models. These functions define a clear interface of input and output parameters. They formalize a large number of existing optical flow algorithms in a common framework. Therefore, they can be implemented via abstract base classes in an object oriented programming language as for example Java and C++. Some methods as for example those based on Fourier Transformations [31] or correlation-based methods [36] have not explicitly been addressed. Optical flow techniques based on learning, such as [67] have not been mentioned as well. Yet, many of such methods can still be implemented in this framework: For example the learning-based method [67] consists of learned motion and brightness variation models which can directly be applied here. Furthermore, correlation-based methods define a neighborhood and use motion models as well - only the optimization strategy varies. In methods based on Fourier Transformations a local neighborhood is used to define an energy which results in an optimization problem as well. (In this case the choice of the neighborhood is restricted so that modules for neighborhood selection need to communicate these restrictions.)
It is unlikely that any optical flow algorithm can be described by this general model formulation. Yet, we believe that the discussed advantages of such a unifying framework are so valuable for further research that a continued effort to unite the concepts of motion estimation is worth to be undertaken.
In the following chapter we will discuss the structure of our software library that follows these abstractions in order to allow a modular implementation of optical flow algorithms.

5. Charon - If you want to cross the river, you need to know the flow

Facts are not science - as the dictionary is not literature.

(Martin H. Fischer)

To contribute to the above described aims of comparability, availability and modularity we implemented the model in equation 4.10 and a number of optimization techniques. We call the resulting software Charon (kaaron according to ancient Greek pronunciation). It consists of a library and an executable, platform-independent front-end that can be used to experiment with motion estimation techniques. A simple tool for the evaluation of the results is provided as well. In this chapter we will describe underlying concepts of Charon and additional tools currently being developed. Please note, that here we describe a work in progress. With the advent of more complex optical flow techniques we will extend the architecture of Charon and its components as becomes necessary.

5.1. Design Choices

To create a sustainable software library, the choice of tools is a first important step. As a programming language we chose between Java, Matlab and C++. We chose the latter due to the following reasons:

- Matlab does not provide sophisticated tools for object oriented programming.
- Some operations in Matlab require more computation time and memory as is needed in our specialized application.
- Most implementations of algorithms available in Matlab are not accessible as source code.
- C++ is accessible on a broader range of platforms and to a broader range of potential users.
- Most image processing libraries are written in C++ as well. Although we did not investigate the appropriateness of Java exhaustively, this is a strong indicator that C++ suits our needs.

- Finally, Java is conceptually similar to C++. We believe that the choice between both is of little importance with respect to the final outcome.

Furthermore, data structures for image processing and tools for image loading, saving and display are needed. Instead of reinventing the wheel, we chose an existing image processing library. There is a large number of libraries available. We chose the CImg library of David Tschumperlé (located at `cimg.sourceforge.net`) due to the following reasons:

- It is platform-independent.
- The open source license is not restrictive (CeCILL-C which is close to the GNU LGPL).
- It is easy to use due to its simple interface and single header file.
- It is templated and supports a large number of image formats.
- In the back-end, it optionally connects to specialized, performance-tuned libraries such as ImageMagick `imagemagick.org` and fftw `fftw.org`.
- Its use is widely spread in the computer vision and image processing community.
- It supports OpenMP `openmp.org` for parallel computing on multicore-CPUs.
- Finally, the support is excellent as the author usually responds to questions within a few hours.

Finally, as most optimization methods boil down to the solution of either small or large linear systems of equations (or specifically the solution of discretized systems of partial differential equations in the case of variational approaches) an appropriate library is needed. A vast number of mathematical libraries exists. We chose the portable, extensible toolkit for scientific computation (PETSc) for the following reasons:

- It is platform-independent.
- There is no restrictive licensing model (cf. `www.mcs.anl.gov/petsc/petsc-as/`).
- It is well documented.
- In the back-end, it optionally connects to a large number of mathematical libraries (more than 30).
- Its use is widely spread and many libraries are based on PETSc itself (such as SLEPc and TAU).
- It supports MPI `open-mpi.org` for parallel computing on supercomputers and clusters.

- PETSc allows to choose the algorithms it uses via command-line arguments which greatly helps in experimenting with various solvers.

- Finally, PETSc is developed by the Argonne National Laboratories and the support is excellent as the authors usually respond to questions within a few hours as well.

5.1.1. Model Implementations and Optimization Routines

In the current version Charon is implemented with a focus on optimization. Therefore, two sets of base classes are used to model global an local motion estimation techniques respectively. For global techniques, two base classes describe the optimization framework: One is used for data terms and regularizers (shortly: terms) and one is an interface to the solver. The term base class has methods for the computation of its energy, the derivative of the energy and the application of robust measures (outlier functions) and motion models. The last two are implemented in two additional classes, so that they can be applied to local methods as well. Its properties contain information about a global weight of the whole term and individual weights per pixel. These can for example be used for inpainting methods.

The solver receives a list of terms. As these are additive, each term has its own derivative that does not interfere with others. Since outlier functions can be derived by applying the chain rule, there is no need to derive each combination of outlier function and term. In Charon, both are derived separately and then inserted into each other appropriately. The solver can perform any optimization task that takes the discrete results of the terms and their derivatives. (Hence, if one or more terms are nonlinear, the solver has to deal with this.) For the optimization of a set of terms a pyramid class is used that offers an interface to pyramid generation from an image sequence. A Gaussian pyramid implementation is based on this class and can be used to create pyramids with arbitrary bases. The solver has an option to utilize this pyramid with two options: If the terms consist of a convex optimization problem, any initial guess leads to the same solution. In this case, each two images in the motion estimation problem can be warped onto each other. This corresponds to a new linearization of the problem after each pyramid level. Upsampling of intermediate motion estimates is carried out by arbitrary interpolation schemes which are implemented in a separate interpolator class. The second option can be used for nonlinear methods: Here, the pyramid images are not warped. Instead, only a new initial guess for the next finer level is generated. The rediscretization of the optimization problem is then performed by the solver itself. For a discussion of these to options, please refer to [14]. Note that algorithms using more than two image frames at the same time cannot be used when the algorithm is based on warping. Specifically image derivative filters should not have a temporal support of more than two frames. This also is the case for three-dimensional regularizers.

Currently implemented terms are the BCCE, the non-linearized image difference as e.g. used in [14], the data term of the CLG method [12] and the regularizers of Horn and Schunck [41] and Nagel and Enkelmann [42]. All terms can be combined with one of the implemented outlier functions: Charbonnier, Huber, Lorentzian and P-Norm. The

latter renders the optimization problem nonlinear, so that appropriate algorithms need to be applied here. Additional data terms can easily be added to the library by deriving the base class and overwriting the methods for energy and derivative computation.

For many data terms image or flow derivative filters are needed. We implemented this based on a class named Discretizer, which essentially contains a filter kernel and its associated properties (such as the center of the kernel and its dimensions). Appropriate boundary handling is included. As well as the described interpolators, these discretizers can be plugged into any of the terms described above.

Currently, we implemented the standard forward and central difference first and second order derivative filters, the Sobel-filter and its optimized version [33] and the BCCE-optimal filters of Scharr [40] in three different sizes. Again, further filters can be included by simply deriving from the Discretizer class and defining the filter mask. Due to a flexible interface, filter masks which are not based on linear filter kernels can be implemented as well.

For local linear optical flow methods, we adopted a very similar approach. The data term of the CLG method utilizes these functions to minimize duplicated code to guarantee comparability between local and CLG-based methods. As described in Chapter 4, four aspects have to be modeled for local methods: The neighborhood operator, the motion model, the data term and the optimizer. The neighborhood operator assembles the data of those pixel locations where constant motion model parameters are assumed. For the method of Lucas and Kanade [17] this is simply a boundary box around the current pixel location. For the method of Bigün et al. [37] a Gaussian weighting is added to this neighborhood. More complex methods can be based on a segmentation of the image (e.g. [58]) or other heuristics. To collect the needed data at those pixel locations, the neighborhood operator uses the motion model and discretizers for image derivatives, respectively. This data can be individually weighted or masked with associated objects. The resulting list of linear equations is then passed to the optimization routine.

For weighting we implemented constant weighting as in [17], Gaussian weighting as in [37] and masking (i.e. pixel locations are only added to the neighborhood if the mask is non-zero). As data terms we implemented the BCCE and those variations of Haussecker et al. [13] that allow for the simultaneous estimation of intensity variations. Motion models are currently not implemented (only the "identity-model").

Finally, block-matching methods are very similar to local linear methods. As they have not been very popular in recent optical flow research, we implemented a simple interface that defines a pixel neighborhood (region mask) and a similarity measure. For optimization, only exhaustive search is implemented for now.

5.1.2. Helper Classes

To facilitate the further development of Charon, a set of platform-independent helper classes has been implemented. Without going into details, these classes include:

- A sophisticated set of classes for the serialization of any objects into parameter files. As each Charon object is derived from this set of classes, the library is

able to be configured by parameter files which in turn can be created by visual tools. Furthermore, the library can be built in a self-documentation mode so that a detailed description of the algorithms and their parameters can be generated automatically for novice users.

- To experiment with various combinations of parameter settings, we implemented a tool for parameter iterations. The user supplies a set of parameter ranges and increments for a given parameter file. The tool then automatically iterates through all combinations of the settings and saves the results to subdirectories. Furthermore, this tool is designed to allow parallel computations of several parameter settings at the same time on different machines.

- Many tools for enhancing the CImg library such as Region of Interest-support, motion estimate error computation, convolution helpers and so on are implemented

- A set of regularly needed statistical functions such as for drawing from a Gaussian distribution.

- A class for the computation of the principal component analysis.

- A set of classes that conveniently wraps the dialog capabilities of the CImg library in an object oriented, event-driven manner.

- A tool to create simple plots either in a vector format (EPS) or rasterized format (normal images).

- Simple convenience-tools for file handling string handling and profiling.

- A Windows-specific class for creating AVIs of image sequences from within the library.

5.1.3. Visualization and Evaluation

Charon comes with a simple tool called WarpViewer which was designed to quickly visualize the results of an optical flow algorithm (Figure 5.1). It offers three views: The sequence view, the error view and the flow view. The sequence view visualizes the post-processed image sequence. The flow can be shown as a configurable vector overlay (color, sampling, scale). If a mask is present, it can be used to either mask the flow, the sequence or both. The error view contains all error measures described in the Middlebury data set [30]. Furthermore, it is able to generate an error histogram and to visualize only a given range of errors. The flow view shows a HSV visualization of either the ground truth, the estimated flow or the difference of both. Screenshots can easily be saved and error statistics can be generated as text-file. An AVI-File can be generated using the current view settings.

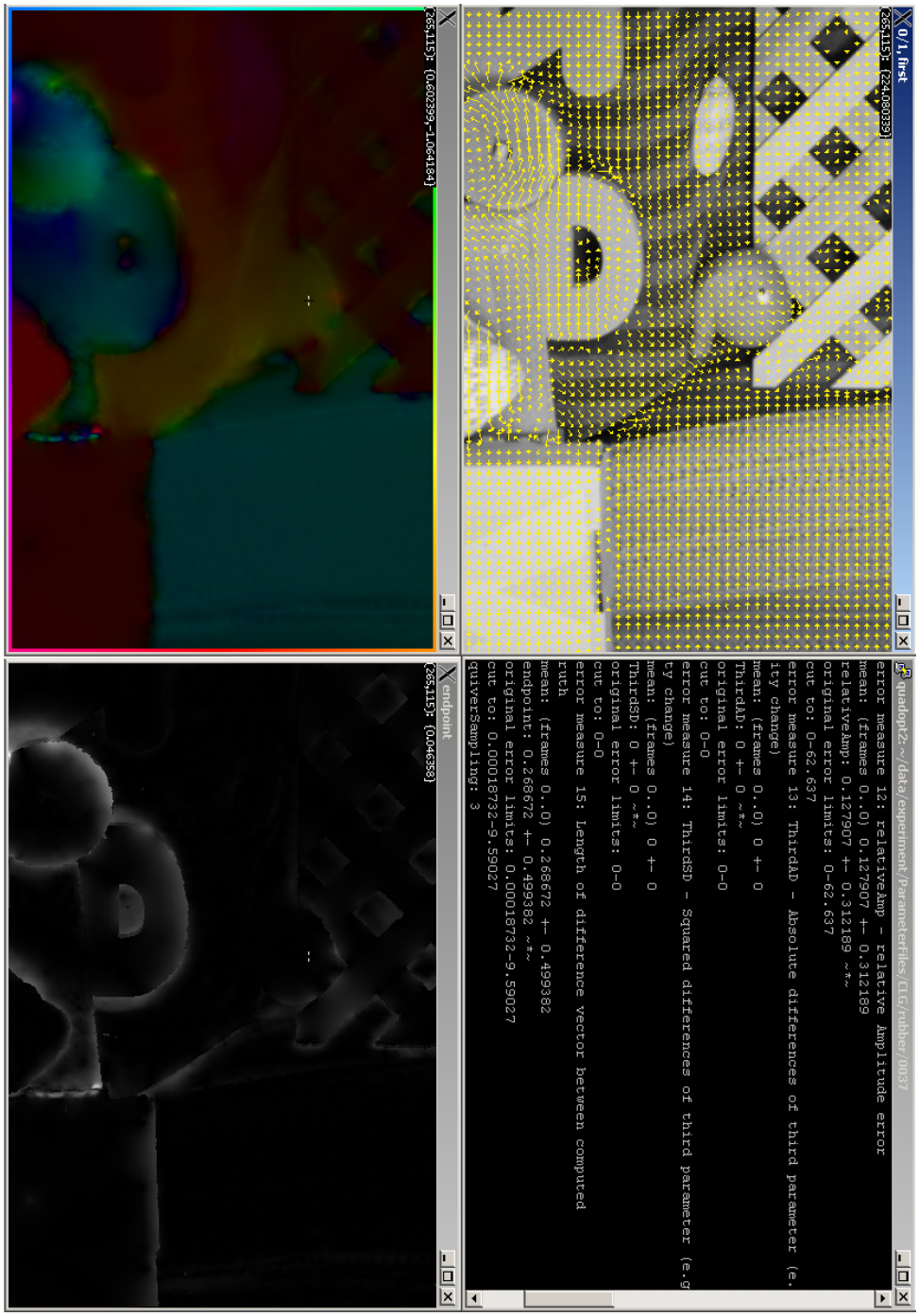

Figure 5.1.: The WarpViewer tool can be used to evaluate motion estimation results. Next to the console, where information about the current actions are displayed, the error and the flow window are similar to the visualizations at the Middlebury website [30].

5.2. Exemplary Results

As laid out in Chapter 2 we believe that general-purpose optical flow algorithms cannot be used to obtain highly accurate motion estimates. Hence in this section we want to show a few results that indicate that this hypothesis at least holds true for some regularly applied algorithms. We will continue to implement more recent methods for Charon to find further evidence.
Please note that we do not intend to provide new results on the latest optical flow techniques. Instead, this section shows the versatility, modularity and ease of use of Charon. In all experiments we compared endpoint errors as used in [30] instead of the angular error used in many publications before. The motivation is that this error is much more intuitive to interpret.

5.2.1. Image Derivatives

One interesting question for differential optical flow algorithms is the choice of image derivative filters. To evaluate under which circumstances the best results can be achieved, we compared all implemented filters. (i.e. central differences, Sobel, optimized Sobel and the filters of Scharr [40] in the sizes 3x3x3, 5x5x5 and 7x7x7). As larger filters are likely to be inaccurate on image sequences with motion boundaries, we first applied them to the Yosemite sequence without clouds. Thus, we can see whether those improved filters yield better results at all. We applied the methods of Bigün [37] and Horn and Schunck [41] because they are so simple that other parts of the algorithm (as e.g. robustness terms) are unlikely to interfere with the derivative filters. We did not apply image pyramids during the optimization because these assume that image derivative filters only operate on the two images being evaluated. This is not the case for the filters of Scharr. We also varied the regularization strength (in the global method) and the integration scale (in the local method). We optimized the results for the global optimal average endpoint error for these parameter settings. Please note that, although it is unrealistic to choose optimal parameters in real scenarios, this approach is very helpful in finding out the theoretic accuracy limit that can be obtained with these methods.
The results show that a larger regularization strength λ in the global method compensates the improvements of image derivative filters: For the Yosemite sequence, the optimal parameters are the central differences filters (with forward differences in time) and a regularization strength of $\lambda = 2000$. In this case, the average endpoint error is 0.74 ± 0.87. As soon as we use image pyramids (as discussed in the next subsection), we are able to obtain much smaller errors. From this point of view the choice of derivative filter seems unimportant. However, in this case the regularization term smooths over large areas and the average endpoint error is computed on a very large number of locations. The errors mainly occur along occlusions and in the bottom left region where flows are large. Therefore, for data similar to the Yosemite sequence (i.e. smooth flows, with small and large magnitudes and some occlusion boundaries) image derivatives do not significantly improve the results.

To see whether this statement is valid for a wider range of sequences, we applied the same techniques with the same parameters to another synthetic sequence. Here, we reduced the complexity of the scene so much that we can test under which circumstances derivative filters can still be useful. We chose to draw a Gaussian function ($\sigma = 10px$, maximum intensity 1.0) on a float-valued image. The center of the Gaussian moves in a circle around the center of the image. The speed of the motion is 0.75 pixels per frame. Although the motion magnitude is constant from one frame to the next, the orientation changes due to the circular motion. These conditions are almost ideal due to the small magnitude of motion, the smooth gradients of the Gaussian and the lack of ambiguities, noise or other image corruptions.
With the same parameter iteration we retrieved an almost constant endpoint error of $0.48 \pm 2.89 \cdot 10^{-5}$ with the $7x7x7$-filter of Scharr and $\lambda = 10$. Hence, the global method is severely biased with this filter. (Closer investigations of this effect indicate that this is due to the anisotropy of the discretization of the regularizer in the Euler-Lagrange Equations. Depending on the angle of the flow vector, differing biases occurred. However, further experiments have to be carried out to validate this hypothesis.)
The best results we obtained with the global method had en error of $0.0029 \pm 7.6 \cdot 10^{-6}$. The small standard deviation is due to the constant motion in the image and the regularization strength of $\lambda = 50$. The error was reduced by choosing central difference in space and forward differences in time. Furthermore, we rediscretized the problem by applying the image pyramid with a downsampling factor of 1.0 and two levels. This is similar to the nonlinear approach of Papenberg et al. [14] where the nonlinear problems are solved by a sequence of linearized versions of the problem. This shows on the one hand, that the error of the linearization error of the BCCE is not necessarily negligible. On the other hand, very accurate results can be obtained with simple derivative filters with this global method. Yet, this method exhibits a considerable computational complexity.
Next, we applied the TLS method with the same filters to the same image sequence. The optimal results were achieved with a large integration scale of $\sigma = 7$ and the $7x7x7$-Scharr-filters: The achieved average endpoint error is 0.012 ± 0.0029. Reducing the integration scale to $\sigma = 1$ increased the error by roughly 23% to 0.015 ± 0.0074. Hence, the best errors are still several times larger than with the global method. But, using the central differences filters (as done in the global method), we obtain even worse errors of 0.059 ± 0.041 with $\sigma = 7$ and 0.22 ± 0.12 with $\sigma = 1$. These results are summarized in Table 5.1.

Can we therefore conclude that the global method is always more accurate and that optimized image derivative filters have an actually negative impact on the quality of the results? To answer this question, we carried out a third experiment. The filter kernels of Scharr are three-dimensional and therefore obtain information from several frames of the image sequence. However, during filter design, it was assumed that the gradient is constant in the whole region of support. In our experiment this assumption is violated (especially with respect to the temporal gradient) as the direction of the motion changes in each frame. To rule out the influence of this violation of the model used during filter design, we created a new sequence that contains the same images. But instead of creating a circular motion, the new Gaussian moves on a straight line from top to bottom.

	Horn & Schunck (Warping)	Total Least Squares ($\sigma = 7$)
Central Differences	$0.0029 \pm 7.6 \cdot 10^{-6}$	0.059 ± 0.041
Scharr 7x7x7	$0.48 \pm 2.89 \cdot 10^{-5}$	0.012 ± 0.0029

Table 5.1.: Average endpoint errors and standard deviations on a Gaussian image rotating in a circle with a flow magnitude of 0.75. Two methods and two sets of derivative filters have been applied. (Warping has only been applied to the central differences scheme with the HS method.)

	Horn & Schunck (Warping)	Total Least Squares ($\sigma = 7$)
Central Differences	$5.6 \cdot 10^{-3} \pm 2.9 \cdot 10^{-5}$	0.044 ± 0.026
Scharr 7x7x7	$0.48 \pm 1.4 \cdot 10^{-5}$	$1.3 \cdot 10^{-5} \pm 6.1 \cdot 10^{-6}$

Table 5.2.: Average endpoint errors and standard deviations on a Gaussian image straightly moving down with a flow magnitude of 0.75. Two methods and two sets of derivative filters have been applied. (Warping has only been applied to the central differences scheme with the HS method.)

The main error in the image derivative computation now stems from the fact that the derivative of the Gaussian image is not exactly constant in space and time, whereas the solution to the problem is now constant in space as well as time.
Thus, instead of optimizing the algorithm design to fit the data at hand, we turned around the problem by tuning the sequence to perfectly meet the model assumptions. Therefore, we can see which of the algorithms is more accurate under almost ideal conditions. The results are summarized in Table 5.2. Here, the local method outperforms the global method by two orders of magnitude: The error of the global method is $5.6 \cdot 10^{-3} \pm 2.9 \cdot 10^{-5}$. Even with a small integration area of $\sigma = 1$, the error of the local method is $3.4 \cdot 10^{-5} \pm 2.7 \cdot 10^{-5}$.
In some applications, as e.g. particle image velocimetry, such almost ideal conditions with small and constant motions over several frames can be achieved with high speed cameras and carefully designed markers. Though, in these applications noise corrupts the image data. Therefore, we added spatially independent Gaussian noise with the variances of 1%, 5% and 10% of the maximum image intensity to the sequences. As can be seen in Figure 5.2, the local method performs better as soon as some noise corrupts the image sequence.

What can we learn from these examples? First, we believe that choosing the correct derivative filters for a given application can be very difficult for users with little expertise in the field of motion estimation. Furthermore, image derivative filters can be used to dramatically improve the results - but only in the special case of reasonably well met model assumptions for both the derivative filter design and the optical flow technique. The applied local method is easier to implement, faster to compute and (in the case of a good model) even more accurate for small motions. Therefore, the choice of the algorithm strongly depends on the kind of input data at hand. A general-purpose algorithm

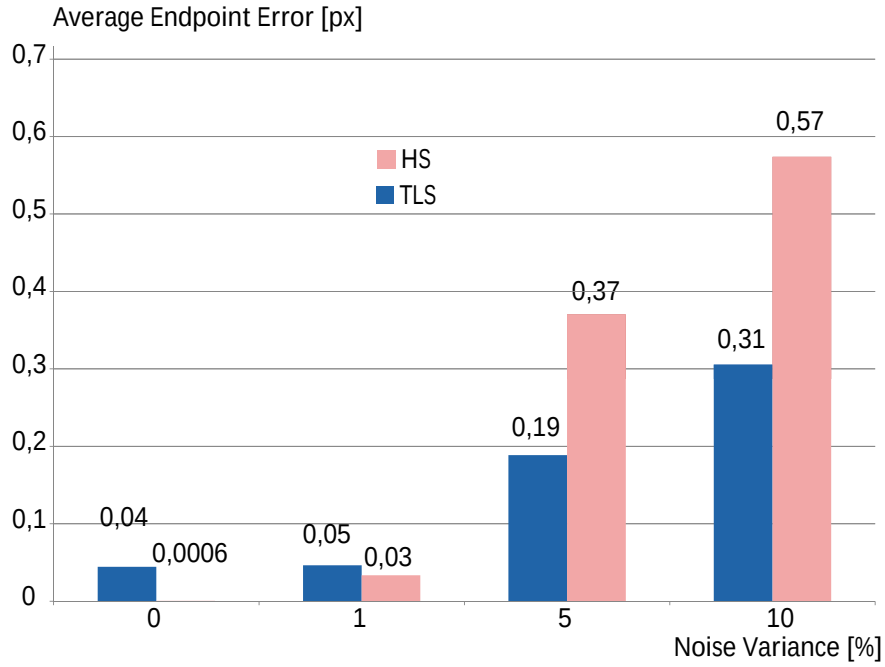

Figure 5.2.: The linear TLS method with optimized gradient filters yields highly accurate results for a flow magnitude of 0.75px and temporally constant motion in the presence of noise. It even outperforms the nonlinear version of the HS method which needs considerable more computation time. Please note, that in our case the standard deviations of the errors of the HS method can be arbitrarily scaled with the regularization strength and are therefore omitted.

	Horn & Schunck	Combined Local/Global
Yosemite	0.12 ± 0.26	0.09 ± 0.1
Rubber	0.26 ± 0.49	0.27 ± 0.5
Dimetrodon	0.16 ± 0.19	0.16 ± 0.2

Table 5.3.: Average endpoint errors and standard deviations with optimal parameters for regularization strength and pyramids (downsampling factor and number of levels) on three sequences with known ground truth data.

does not seem to be feasible in these scenarios. Since we have only experimented with flows of magnitude 0.75, it remains to be analyzed how the accuracy depends on other magnitudes. Nonetheless, knowing the model assumptions and the data can help to design a new algorithm that is simply put together with existing modules of known optical flow techniques. Further research could for example concentrate on the automatic choice of modules for a given image sequence. Charon can be used to experiment with such techniques.

5.2.2. Pyramid Levels

In almost all of todays global techniques (such as [12, 14, 56]), multiresolution methods are applied to capture large motions. Yet, the choices which have to be made for image pyramids as described in Chapter 4 are often not discussed. For example, even though the results are tuned for accuracy and not for small computation times, in [56] a standard downsampling factor of 2.0 is applied. Sometimes, the interpolation method for downsampling or the one for upsampling the flow is not mentioned at all. Although Alvarez et al. [73] proposed to downsample with other factors and to use Gaussian scale spaces instead of image pyramids, these approaches have seldom been addressed in later publications. To get an idea how much the pyramid implementation affects the accuracy we ran a parameter iteration by simultaneously varying the number of levels of the pyramid, the downsampling factor and the regularization strength. We experimented with three sequences (Yosemite, Rubber Whale and Dimetrodon of the Middlebury dataset) and applied the method of Horn and Schunck (HS) as well as the 2d, linear combined local global method (CLG) of Bruhn et al. [12]. For the CLG method we iterated over the integration scale of the structure tensor, the regularization strength, the number of pyramid levels and the downsampling factor. We used bicubic interpolation.
Furthermore, before downsampling the images they are convolved with a Gaussian. The variance of the filter kernel is the effective downsampling factor at the current level divided by 1.4. Downsampling is always carried out based on the original image instead of iteratively resizing the already downsampled image. Thus, downsizing with a factor of 1 slightly blurs the image which effectively results in a scale space.
The optimal parameters and the results with both the HS and the CLG method are shown in Table 5.3. In all examples, the optimal downsampling factor is smaller than 1.1. The number of pyramid levels varies between 4 and 15. We can draw two conclu-

sions from these results: First, in our test scenarios, downsampling by a factor of two only speeds up the computation time but does not improve the accuracy of the results. Larger motions can hence as well be captured by the scale space and the iterative linearization of the problem. Second, the number of pyramid levels (or rediscretizations) strongly varies with the input data. In most examples the results became less accurate for more than the optimal number of levels. On the other hand, as soon as a sufficient number of rediscretizations has been carried out, the errors roughly converge to a constant value. It remains unclear how the minimum number of levels for a given error limit can be determined based on the original image data.
Please note that again we use simple sequences and simple methods. Yet, even for such well-known methods the de facto standard pyramid settings are not optimal with respect to accuracy. By starting to experiment with these simple methods, we would like to encourage further experimentation with more sophisticated methods. But with more and more methods, a combinatorial explosion of modules renders this task difficult. Nevertheless, we believe that by more carefully studying existing methods and thereby describing their properties in greater detail can facilitate further research and motivate new approaches.

5.2.3. Parameter Choice

We have shown that the optimal choices of pyramid levels and downsampling factor for a few test sequences are not the same as regularly employed in literature. Additionally, the examples described above also vary in the optimal choice of the regularization strength. Although Yosemite and Dimetrodon sequence are normalized to the same intensity range of [0..255] (which strongly affects the regularization strength), we obtain optimal strengths differing by an order of magnitude. If we apply the optimal parameters of the Dimetrodon sequence to the Yosemite sequence we obtain an average endpoint error of 0.6 ± 2.66 which is five times worse than with optimal parameters. Using the parameters of Yosemite for Dimetrodon, we obtain an error of 0.21 ± 0.22 which is 30% worse than with optimal parameters.
The CLG method is more stable with respect to the regularization strength in the three test sequences. Yet, the additional parameters for the integration scale have to be investigated as well. For example, the Yosemite sequence needs strong regularization of $\lambda = 500$, a large integration scale of $\sigma = 4$ and the optimized Sobel-filters as image derivative kernels. In contrast, the rubber whale sequence needs a smaller regularization strength of $\lambda = 100$, a much smaller integration scale of $\sigma = 1$ and the standard filter kernels. Exchanging the parameters of Yosemite and rubber whale sequence yields the results shown in Tables 5.5 and 5.4. Hence, applying the same parameters of these methods to a wide range of sequences can result in quite different results.
Yet, with optimal parameters both methods perform almost equally well, except for the Yosemite sequence where the CLG method is more accurate. Therefore, research into the optimal choice of parameters with a given image sequence can sometimes (e.g. under the given circumstances described here) be a fruitful approach to improve existing motion estimation techniques. We believe that comparability, availability and modularity

	Horn & Schunck	Accuracy Loss
Yosemite (parameters for Dimetrodon)	0.6 ± 2.7	80.0%
Rubber (parameters for Yosemite)	0.38 ± 0.6	31.6%
Dimetrodon (parameters for Yosemite)	0.21 ± 0.22	23.8%

Table 5.4.: Average endpoint errors and standard deviations with suboptimal parameters for the HS method applied on three sequences with known ground truth data.

	Combined local/global	Accuracy Loss
Yosemite (parameters for Dimetrodon)	0.19 ± 0.58	47.4%
Rubber (parameters for Yosemite)	0.35 ± 0.6	22.9%
Dimetrodon (parameters for Yosemite)	0.20 ± 0.22	20.0%

Table 5.5.: Average endpoint errors and standard deviations with suboptimal parameters for the CLG method applied on three sequences with known ground truth data.

of optical flow methods can facilitate and focus this further research.

5.3. Further Developments And Additional Tools

We consider this release of Charon as a very first step towards an open framework for motion estimation researchers. With that in mind, we currently refactor the library to more strictly follow the model equation 4.10 described in Chapter 4. The interoperability of similarity measures (or data terms) is currently restricted to the optimization method used. In future releases we aim at defining a single data term base class for all optimization techniques. These data terms can easily be interpreted as unnormalized log-likelihoods in graphical models whose interaction between nodes is defined by the discrete versions of regularization terms. With a few minor modifications, our model equation can directly be interpreted as graphical model. As a result, statistical optimization algorithms derived from graphical models are another topic for future implementations. This would enable researchers to define models independent of the optimization method or mathematical philosophy behind the actual idea how the optical flow problem can be solved.
Furthermore, the support for motion models is being extended by those models described in [59].
Finally, two additional software projects [1] are going to be fully integrated into Charon for facilitating good documentations, intuitiveness, reproducibility of results and their interpretation. The first program is called Argos. It implements a much more sophisticated viewer for motion estimation results. The second program is Tuchulcha (whose name also is derived from Greek mythology). It is a visual editor that helps generating motion estimation algorithms from Charon modules.

[1] developed during student projects by Stephan Meister and Jens-Malte Gottfried

5.3.1. Argos

On the one hand Argos (Figure 5.3) is a general, platform-independent, Qt-based image viewer that can be used to view images and sequences in the CImg library format as well as any format supported by ImageMagick. On the other hand, Argos is a tool to specifically generate workflows for the interpretation of motion estimates. The functionality of the WarpViewer is implemented in a much more general way so that individual views of flow results can be generated. With flows in mind, the analysis of sequences with two or more dimensions per pixel and their visual connection to conventional image sequences becomes possible with just a few mouse clicks. Motion error analysis, sophisticated masking methods, information about mean and standard deviation with respect to given masks and many more tools are included. Furthermore, the results can be saved for figures in scientific publications or websites. As well as Charon, Argos is written in a completely modular way so that specialized extensions can be implemented with little knowledge about the program itself.

5.3.2. Tuchulcha

Tuchulcha (Figure 5.4) is a visual tool to generate workflow graphs based on predefined rules. It is also Qt-based and can be used for many other applications as well, but it tightly integrates with Charon. The generation of rules for the graph design is carried out by simply compiling Charon. Along with the rules comes an automatically generated documentation that describes the individual modules of Charon and their parameters. Hence, with Tuchulcha, the learning of Charon is a lot easier. New modules of Charon are automatically integrated into Tuchulcha without any additional steps. Finally, the designed algorithms can be saved to a single parameter file. These can in turn be saved to a database with experiment results, used for unit tests, or published on a website. Researchers then only need to compile the current release of Charon and then download parameter files of other researchers to verify their results.

5.4. Conclusion

In this chapter we have introduced Charon, a software library for the estimation of motion in image sequences. Charon comprises a huge set of features that have been developed during the course of the authors' optical flow research in the past three years. Thanks to a modular design, a large number of optical flow algorithms can be generated, including classical [17, 37, 36, 41] and more recent algorithms [73, 12, 14]. This modular framework derived from a general perspective on motion estimation techniques minimizes the implementation time of new local and global optical flow techniques. Simultaneously, it allows for a comparable experimentation with known and new algorithms where only modules of interest are changed, while the rest of the algorithm remains exactly the same.
Although the library is not designed for performance, it allows for the computation of results on multicore CPUs, network clusters and supercomputers.

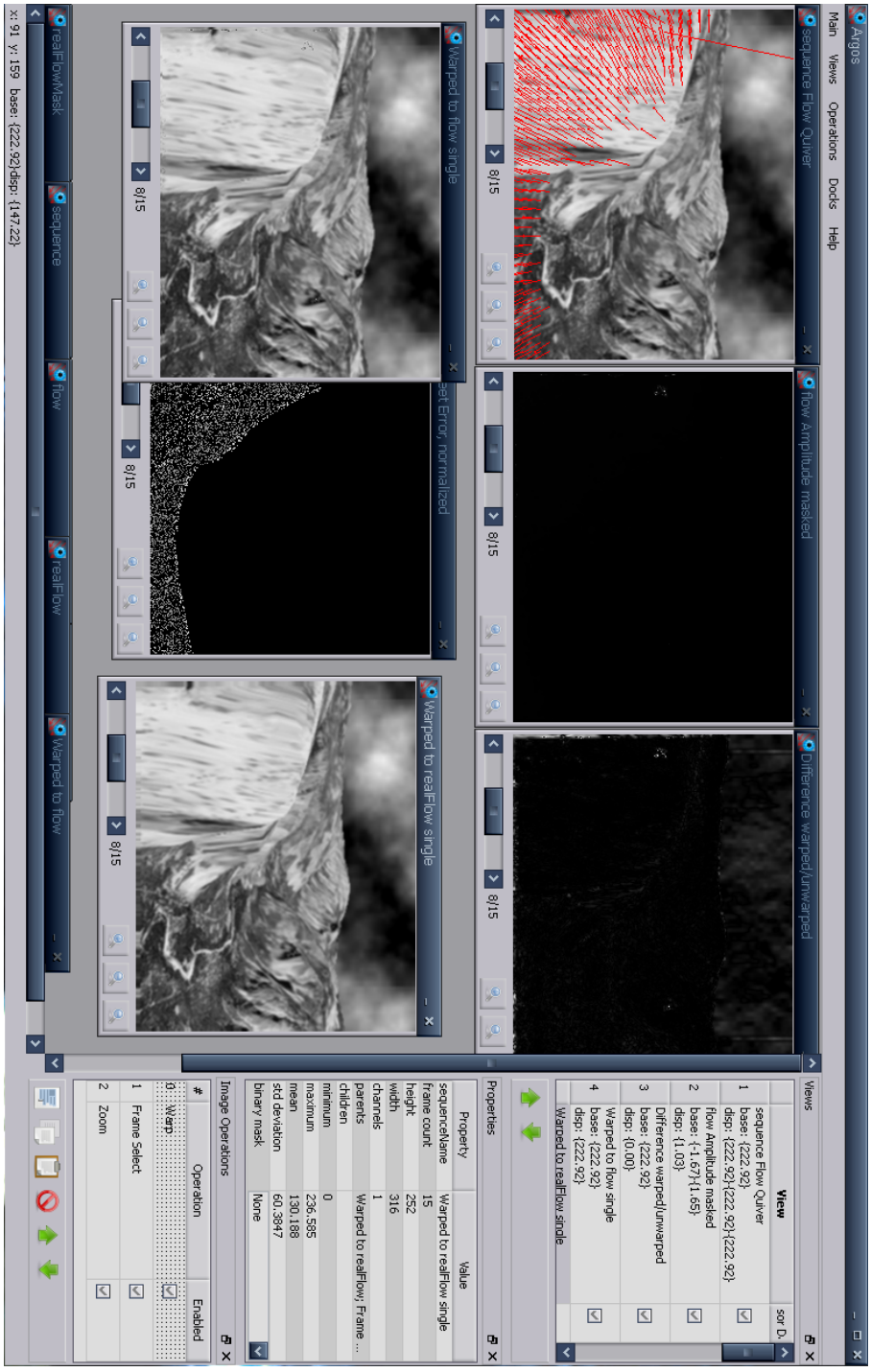

Figure 5.3.: Argos is a general image viewing tool which is specifically tailored for scientific image analysis. The currently implemented modules are designed for flow vector evaluation with respect to ground truth.

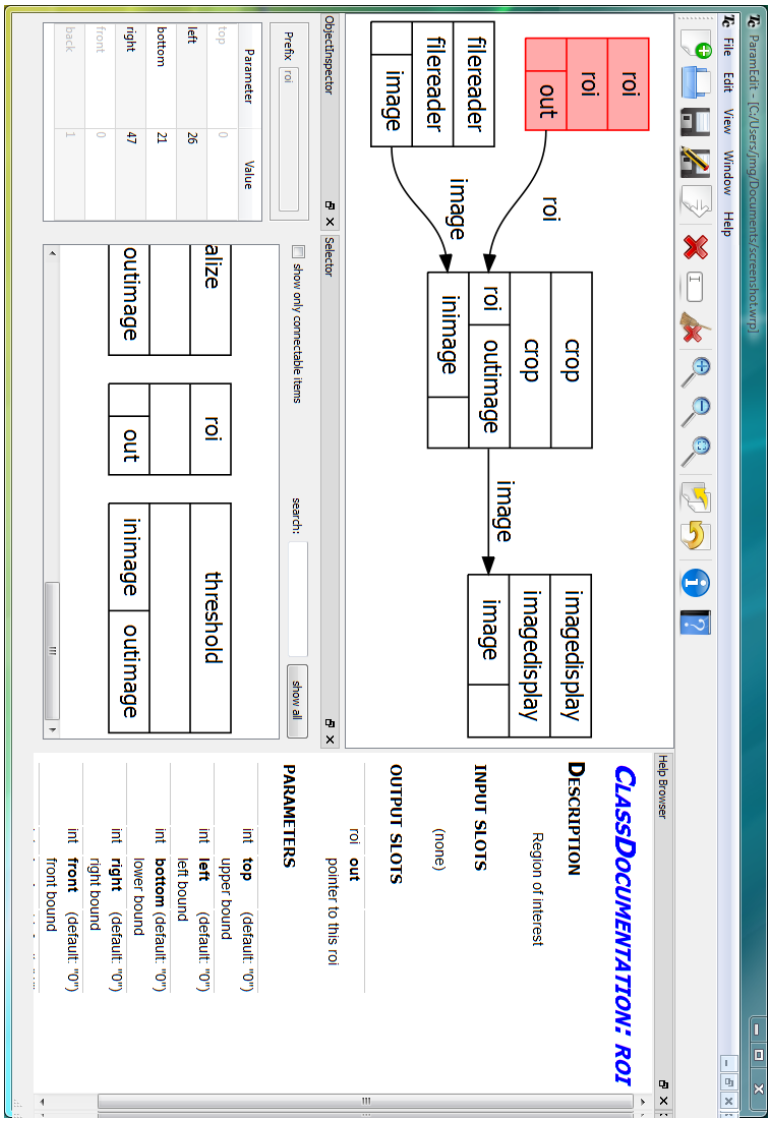

Figure 5.4.: Tuchulcha is a tool which can be used to graphically design optical flow algorithms from known modules with integrated documentation.

To encourage other researchers to use Charon for both experimentation and new algorithm implementation we are willing to offer high quality future support (inspired by the developers of CImg and PETSc) and will continue the development as laid out in the previous Section.

Based on some preliminary experiments, we have shown that the choice of modules has an impact on optical flow techniques that cannot be easily predicted without a decent amount of expert knowledge or extensive experimenting. These experiments for example show a strong dependence between the optimal regularization strength and the given input data, the optimal choice of the best pyramid parameters and unintuitive behaviors of derivative filters. This indicates that, for example, more research has to be put into the automatic choice of parameters or into methods with less parameters. Furthermore, we have shown that the accuracy of the discussed optical flow algorithms is best optimized by choosing those modules which are best suited for the image data at hand. In our case, the local structure tensor method with the derivative filters of Scharr compared to any combination of modules for the HS method yielded superior results already for small noise levels.

We believe that, given the support of the motion estimation community, Charon can greatly facilitate further research in motion estimation by providing a set of reference implementations and efficient evaluation tools as Open Source code. We sincerely hope to inspire other researchers to use Charon as their software framework for new optical flow algorithms in order to create an algorithm database similar to the reference data sets provided by [30].

Part III.

Applications to Fluid Flow Estimation

> The scientist is a practical man and his are practical (i.e., practically attainable) aims. He does not seek the ultimate but the proximate. He does not speak of the last analysis but rather of the next approximation. His are not those beautiful structures so delicately designed that a single flaw may cause the collapse of the whole. The scientist builds slowly and with a gross but solid kind of masonry. If dissatisfied with any of his work, even if it be near the very foundations, he can replace that part without damage to the remainder. On the whole he is satisfied with his work, for while science may never be wholly right it certainly is never wholly wrong; and it seems to be improving from decade to decade.
>
> *(G. N. Lewis. Quoted in Stochiometry by Leonard K. Nash. Addison-Wesley 1966. p. vii.).)*

6. Particle Detection By Momentum Stability

Science is simply common sense at its best.

(Thomas Huxley)

In the previous part of this work we have motivated and developed a framework for the implementation of optical flow techniques. The resulting software library is capable of abstracting the modules of optical flow algorithms so that the implementation of new techniques is fast and easy to carry out. In the following two chapters we will now demonstrate the versatility and the advantages of this framework by introducing a new algorithm in the field of fluid flow estimation. More precisely, we will develop an algorithm for image data with relatively few visible particles in the image sequence. This is a rather exotic application in terms of optical flow estimation, as motion can only be estimated at pixel locations where particle images are visible. By detecting these particle locations we tackle the optical flow algorithm requirement of an estimatibility measure. As this measure is an important step limiting the accuracy of PTV approaches in general, we will discuss it in the following.

The performance of this initial part of the PTV algorithm also has significant consequences for subsequent tracking algorithms, particularly in the case of dense seedings of particles. Different PTV algorithms have been proposed in literature. But usually only the accuracy of the whole approach is presented, including experimental set-up, segmentation and tracking. Here, we will present a novel algorithm and a thorough evaluation framework for the first step of particle detection which can be considered as an estimatibility computation module of Charon. The algorithm is based on momentum stability over a set of thresholds and exhibits superior results. With the latter the performance of our proposed algorithm is analyzed with respect to particle image size, intensity, noise and overlap. Our technique is thus well suited as an initial building block for PTV algorithms. To facilitate incorporating our technique into PTV algorithms (such as the one presented in Chapter 7) and testing this module for accuracy, we provide reference images and C++ source code in the form of Charon modules.

6.1. Introduction

For visualizing fluid flow, neutrally buoyant particles are used to record an image sequence as they follow the motion of the fluid. The resulting data is studied by image

processing algorithms which try to reconstruct the fluid flow from this data. Particle tracking velocimetry (PTV) is a good choice whenever one wants to extract Lagrangian particle trajectories or when the flow to be studied does not allow for high particle densities. This is for example the case if the visualized flow is highly three-dimensional and illumination is carried out with a volume light source which would results in an intractable number of occlusions (regions were particle images of different depths overlap each other). Such a PTV algorithm usually consists of the following parts:

- Preprocessing: E.g. image sensor noise is reduced and the background is subtracted.
- Particle Detection: For each pixel a decision is made, whether it belongs to a particle or not.
- Particle Localization: The center of each particle is located with sub-pixel-accuracy.
- Particle Tracking: The correspondences of the particles between image frames are determined.
- Postprocessing: E.g. interpolation or confidence estimation of the sparse motion estimates is performed.

Each step can significantly influence the quality of the overall results. One crucial step is the particle detection: If particles are missing from one image frame to the next, a correct tracking step becomes much more difficult because the flow for this particle cannot be determined on a frame to frame basis; if spurious particles are detected, the estimated flow at this location becomes arbitrary because any correspondence with particles in the subsequent frame is wrong. Nonetheless, locating particles correctly (or at least knowing the typical problems with the detection step) can significantly facilitate the subsequent tracking step. This is particularly the case when particle densities increase.
Therefore, the optimization and analysis of this single algorithmic building block promises large quality improvements. Yet, this aspect of image processing for PTV is usually included in the whole experimental setup and not analyzed separately. It remains uncertain which part of the whole PTV technique yields which quality improvements. Furthermore, particle detection often is highly adapted to actual imagery obtained by the experimental setup (cf. Section 6.2). A more general solution to the problem seems desirable.
Our contribution in this paper is twofold: In order to be able to compare the quality of such algorithms, we suggest a test environment consisting of reference images and C++ code to evaluate the results and construct new images as was done by Okamoto et al. [74] for PIV image data. This set of reference data is also very useful for a comparable evaluation of complete PTV methods. Unfortunately, these image sequences cannot directly be used to specifically test particle detection routines. The reason is that reference data is not given for all particles showing up in the image data (cf. Figure 6.3 for a visualization of this problem). The estimation of false positive and false negative rates therefore becomes impossible.

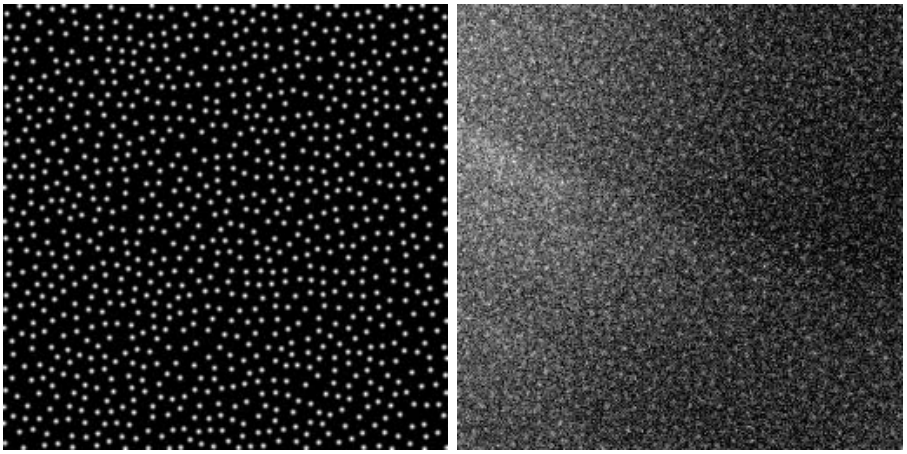

Figure 6.1.: A particle image used for the evaluation of our algorithm. Top: original image. Bottom: image with 800% noise and background image. The number of correctly detected particles is 86.7% (cf. Section 6.6)

Additionally, we propose a new segmentation algorithm with a wider field of applications in mind. We evaluate this algorithm with respect to variable particle image sizes, intensities, noise and particle overlap. The results show that our new algorithm is very robust with respect to noise and produces almost no false positives at all. This suggests that this method can be applied to a wide range of PTV-based applications. But, as there is no common data basis for investigations of previous particle detection algorithms (in contrast to data for black-box tests of whole PTV algorithms), a thorough comparison remains to be undertaken. We understand this paper as a first step towards this direction.

6.2. Related Work

Very few papers actually analyze the particle-detection part separately as will be discussed below. Possible reasons might be a lack of space, the minor importance of this specific part in a paper with another major point of focus or the lack of reference data from other papers. On the other hand, PTV papers analyze the overall outcome of the whole technique in a very detailed manner, so the particle detection quality assessment can be understood as implicitly included.
Almost every PTV algorithm uses its own, specialized particle detection algorithm. Hence, it is very difficult to give a comprehensive overview on this topic. Nevertheless, only a few basic principles are usually applied which will be described here exemplarily.

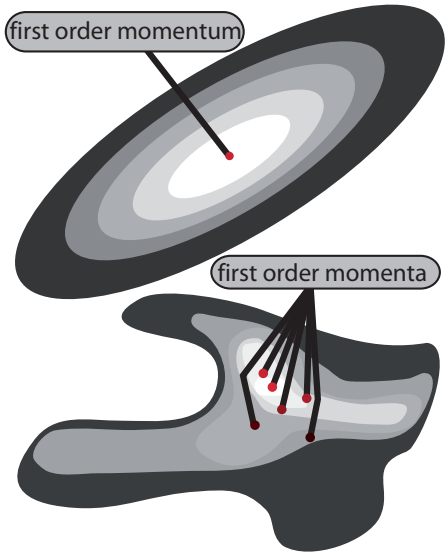

Figure 6.2.: When thresholding a particle image at various thresholds, one of the invariants is the the location of the first statistical momentum of the resulting connected components.

6.2.1. Global Thresholding

One idea to seperate particle images from the background is to determine a global threshold. All pixels above this threshold are assumed to be particles. The main challenge of this approach is to determine a single optimal threshold for all pixels at once. One step in this direction has for example been made by Guezennec et al. [75] by analyzing the histogram of the image. They search for two modes in the histogram: The mode of dark intensities is assumed to be the background and the one with bright intensities is understood as particle. The optimal threshold lies at the intensity farthest away in between both modes.
Problems occur when the background of the image is not black, e.g. to insufficient background subtraction or for pixels with shot noise. Furthermore, it can become difficult if various particle image intensities exist, i.e. the histogram is not bimodal any longer. In [75] this algorithm has been analyzed for 100 particles with respect to several noise levels. A comparison or discussion with other methods was not carried out.

6.2.2. Dynamic Thresholding

When global thresholding fails, a possible solution is to determine an individual threshold for each particle individually. For example in [76], Maas et al. search for local maxima first and then determine a threshold based on its surroundings. To reduce the number of false positives, the region around the maxima is tested for monotonously decreasing intensity values. The authors state that the quality of the detection is good. Yet, they

do not explicitly analyze this part of their algorithm.
A second approach was undertaken by Marxen et al. [77]. For each particle location candidate, they take the average gray value of a region of 32x32 pixels into account. The threshold is a factor $f > 1$ multiplied by this average and is empirically determined based on the given particle density. No individual results for this part of the algorithm are shown and no other possible methods are referenced.
Other dynamic thresholding approaches were introduced by Ohmi et al. [78]. In their paper, problems of existing particle detection algorithms are discussed thoroughly and their properties are mentioned in a qualitative manner. They apply their algorithms to the standard PIV library [74] in order to make their results comparable.
This method was modified by Mikheev et al. [79] so that local maxima intensities are used as input for the threshold estimate and particles too dark are removed from the list of candidates. The paper also includes a comprehensive discussion on the difficulties of known methods but again, the actual quality of this part of the whole PTV algorithm is not investigated.

6.2.3. Correlation-Based Methods

In [80], Etho and Takehara proposed a completely different method to detect particle images. They tried to analyze the visual appearance of the sought particles. From the results a template (or mask) is created which is then correlated with the whole image. Correlation peaks show where the template matches the image well. Thus, a local maxima in the correlation results denotes a particle location.
Ohmi et al.[81] took up this idea of defining a correlation mask which inspired the use of the Moravec operator. The authors state that this operator helps to identify particle images of variables sizes and intensities.

6.2.4. Region-Growing Methods

Similar to dynamic thresholding are region growing methods. Here, pixels are seeded into the image (usually at all local maxima locations) and then grown simultaneously by adding pixels to each region. A first approach in this direction was carried out by Hering et al. [82, 83]. This method was further developed by Jehle et al. by using watershed segmentation in [84]. One of the major problems of these approaches is noise: During the seeding, the algorithm creates regions to be grown which would eventually become particle candidates. If noise severely corrupts these images, the growing sometimes randomly stops or propagates into other particles. As a result their identification becomes very difficult in the final decision step.

6.2.5. Learning-Based Methods

Carosone et al. [85] suggested to use artificial intelligence methods to detect particle locations. In their approach, a Kohonen neural network is trained to detect particle locations. In experiments they found a sensitivity of 97.6% for their algorithm. Furthermore, they state that the algorithm is very good in detecting overlapping particles. But the

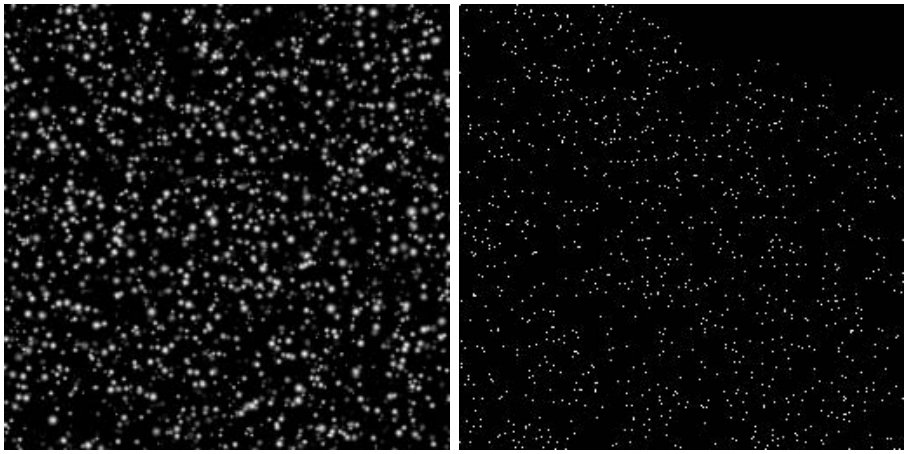

Figure 6.3.: A particle image from the PIV-STD dataset [74]. Left: simulated PIV image. Right: locations of particles in the supplied particle list. Missing particles make the estimation of false positives and false negatives impossible.

exact type of synthetic test-data is not exactly described in the paper. Additionally, by just citing the sensitivity, nothing is known about the false positive rate: the number of spurious particles can be arbitrary. Hence, the overall quality of the algorithm remains to be investigated.
A recent approach to detect particles with neural networks was suggested by Ouellette et al. [25]: The authors trained a multilayer perceptron with a 9x9 input region (the intensities around each pixel location), a single hidden layer of 60 neurons and 2 output neurons with sigmoidial activation functions. The results show very good true positive rates but no false positive or false negative rates are shown. Hence, again the overall quality of this algorithm is unknown. The method was not compared to other methods.

6.3. Particle Detection

The central observation in this paper is that particle images approximately maintain their first momentum (center of gravity) when thresholded at various levels as illustrated in Figure 6.2. Furthermore, we assume that particles are roughly Gaussians and thus are convex in shape. Please note, that in this paper we are only interested in the particle detection and not its localization: Subpixel-accurate particle locations can be estimated as soon as all pixels belonging to a particle image have been determined by our algorithm. It consists of five steps, applied to each image separately in the sequence. The choice of parameters is discussed in the subsequent section.

Step 1: Gaussian Blur As a first step, we convolve the image with a Gaussian kernel. On the one hand, this is merely used to increase the number of intensities from the

discrete set of quantized camera intensities to floating point values (which is useful as shown below). On the other hand (as this kernel is symmetric, and thus equivalent to a correlation) Gaussian-like image structures are enhanced by this process.

Step 2: Thresholding To calculate the momentum stability in step four, we create a stack of binarized images. The set of thresholds is determined as follows: The user supplies the algorithm with a maximum intensity value I_{max} that should be used as maximum threshold and a number of thresholds N to be computed. Thus, the "threshold step" can be expressed as $\frac{I_{max}}{N}$.

Step 3: Connected Component Analysis A connected component consists of all pixels in a binary image that are connected in a four-neighborhood. Each pixel in a binary image belongs to exactly one connected component. Each of the thresholded images of the previous step is analyzed via its connected components. We now reject those connected components which are too small or too large.
Furthermore, we generate the convex hull of each connected component and count the number of pixels of the convex hull that do not lie inside the connected component. This number of pixels is considered as convexity measure: If it is larger than four we reject the component. Finally we compute the first momentum of each remaining component by averaging all of its pixel locations.

Step 4: Momentum Stability Computation After the most obvious non-particle connected components have been removed from the thresholding result, the momentum stability for each connected component is calculated. By now, for each pixel, we have obtained the associated connected component for each threshold. Starting out from the smallest threshold, we now compute the difference vector between the momentum location at a given pixel at the current threshold and the associated momentum location at the same pixel in the next higher threshold result stored in the stack of binarized images. If the length of this vector is smaller than two pixels, we mark this location as stable at the current threshold. If it is larger than two, we mark it as instable at this threshold. In the end, for each location and each threshold we have a boolean stability value. An illustration can be found in Figure 6.4.

Step 5: Maximum Stability Range Detection Once the stability value has been calculated for each pixel and threshold we search for the largest stable range of thresholds at each pixel. This can be easily done (at each pixel) by starting at the smallest threshold's stability. If it is stable, we add one to a counter. Else, the counter is reset to zero. Then, we go to the next higher threshold and continue (cf. Figure 6.4). Thus, we can remember the smallest threshold for the largest stability range and the number of thresholds over which this range was stable. The larger this number, the more likely we identified a connected component that is a particle. If the maximum stability range R_{max} is large enough we accept the connected component with the smallest threshold in this range as particle.

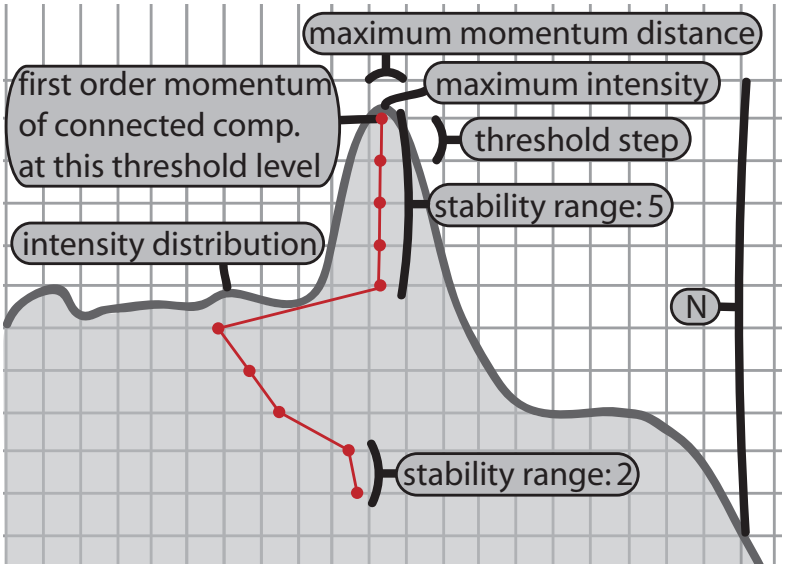

Figure 6.4.: A visualization of the momentum stability computation and maximum stability range detection of steps four and five in the algorithm (cf. Section 6.3).

(Optional) Step 6: Noise Analysis Around each found particle, the average intensity of all non-particle-pixels in a diameter of twice the particle radius is computed. Then, the maximum intensity of the particle is divided by this average to get a rough idea of the signal to noise ratio of this particle image. Based on this value, additional particle image candidates can be neglected. Obviously this method only works for zero-mean noise, hence (if necessary) background subtraction methods have to be applied before this step is carried out.

6.4. Choice of Parameters and Technical Considerations

As already mentioned in Section 6.3, the choice of the parameters is quite intuitive and robust to variations. To give a better overview, we sum up all aspects of parameter choice in this Section.

Gaussian Blur The size of the blur in step one is mainly used to spread out the intensity values in order to reduce the effect of quantization. We modified the parameter in the range between $\sigma = [0.3..2.0]$ and yielded essentially the same results. Only the centers of gravity of the momentums were slightly perturbed without an effect on the detection rate. Since we are only interested in the particle centers, the decreasing intensity and increasing size of the particle images caused by the Gaussian blur can be ignored. Hence, the variance of the Gaussian kernel should be roughly close to the particle size.

Nonetheless, as overlapping increases with larger filter kernels, this value should be as small as possible.

Connected Component Analysis The smallest particle area allowable for the subsequent steps of the algorithms is usually known by the experimenter. Particles smaller than this size are neglected because they could also be noise. The largest particle area is useful to instantly remove particle candidates that are obviously impossibly large. Removing them by this threshold mainly saves computation time. Furthermore, the maximum size can be roughly determined by looking at the image data.
If the connected component is not convex, it is highly unlikely that it is a particle, hence a very small number for non-convexity as e.g. four (for allowing four pixel difference to the convex hull) should be used. This parameter begins to play a role when particles touch each other or background objects were not successfully removed before the algorithm started. Anyhow, it plays only a minor role in the overall results.
The connected component can be carried out highly efficiently by preallocating all memory needed throughout the routine (as memory allocations are very slow) and implementing e.g. the linear-time algorithm of [86].

Thresholding and Stability Range The maximum particle intensity of the image I_{max} can be easily determined by actually looking at the brightest particles in the image. For best results, I_{max} is set to a rough estimate of the brightest particle intensity minus a small offset. This offset is due to the Gaussian kernel convolution which reduces the particle intensity. But the exact choice of this threshold does not matter: In a range of 255 intensities and a brightest particle of 200, a range of 150 to 255 yields almost the same results.
This step can probably also be automated in a future version of this algorithm. On the other hand this choice is highly intuitive and again has not much impact on the resulting quality.
The most important choice is the number of thresholds N together with the stability range R_{max}: Consider $I_{max} = 100$, $N = 10$ and an image with only one particle with this intensity and no noise. Thus, the stability range of this particle is $\frac{100}{10} = 10$ because the momentum remains at the same location for all $N = 10$ thresholds. Choosing $R_{max} = 11$ would cause the algorithm to never detect this perfectly visible particle. Hence, R_{max} should never be too large. In contrast, experiments showed that choosing R_{max} relatively small compared to the number of thresholds does not significantly decrease false positive rates. We assume that this is due to the fact that for noise or larger image structures occurring in particle clusters the momentum rapidly changes from one threshold to another. As a result, very few stable momentums suffice to reliably detect a particle image correctly. Thus, R_{max} should be chosen very small compared to the N. In all our experiments, we set $R_{max} = 5$ and yielded very good results for all evaluated data sets.
The choice of the number of thresholds N is related to that of R_{max}. It should be sufficiently high to capture all occurring intensities in the image. On the other hand, as only a limited number of actually differing intensities exist in the image, at some point

increasing N will not facilitate the particle detection any longer because the thresholded images look alike at various levels. In an extreme case this becomes disadvantageous because in identically looking thresholded images, all momentums remain constant. As mentioned above, we set $N = 500$ in our experiments. In a range of $N = [300..700]$ the results did not significantly change. As a rule of thumb in most cases (8 bit images with bright particles) this value should be around two times I_{max}. We assume that this parameter can be automatically selected in future implementations by counting the number of occurring intensities in the range $[0..I_{max}]$ before the Gaussian blur step.
Thus, this algorithm has basically three important parameters: I_{max}, N and R_{max}. The first two can be chosen easily, the latter requires some experimentation. All parameters are intuitive and can be adjusted by looking at single frames of the image sequences at hand.
A technical problem of the algorithm is that each frame is expanded to N images (one for each threshold) which can consume a lot of memory for large images and large N. As a solution, the binary images can be efficiently managed by storing 32 (or 64) boolean variables in a single integer value. This saves memory for large numbers of thresholds if necessary.

6.5. Evaluation Environment

As discussed in section 6.2, the PIV standard data set [74] cannot be used to fully describe the properties of the particle detection routine. Therefore, we implemented a new particle image generator (PG) for single images. A widely accepted way of creating a single particle image (cf. e.g. [77] for a motivation) is to approximate it with a two-dimensional Gaussian intensity distribution instead of using the Airy distribution. As particles are usually very small in real data sets, it is important that particle locations do not necessarily have to lie on the pixel grid. Other aspects are the brightness and the size of the particle images which can be controlled by the maximum and the variance of the Gaussian function. There are two ways to draw a Gaussian on a pixel grid: one can either sample the Gaussian at pixel center locations or integrate the Gaussian over the pixel area. Motivated by experiments carried out in [87] we sample the Gaussian function. Another interesting question is where to stop sampling. At least three options exist:

- Sampling is stopped as soon as the quantized intensity value of the Gaussian function becomes zero.
- As camera images are noisy, another option would be to stop sampling at the noise level.
- As it is known how much of the integral of a Gaussian is covered by a range of $n \cdot \sigma$ around its mean, we can stop sampling at e.g. 2σ so that 95% of the particle integral is covered.

Here, we stop sampling the Gaussian according to the first suggestion either at 10σ or when the rounded intensity value drops below the quantization error (depending on

which condition applies first). An appropriate notion of particle image diameters can be chosen by the reader as it can be directly deduced from noise level, quantization bits and Gaussian variance.
Another important consideration is how particle overlap is modeled. To the best of our knowledge, no experiments have been carried out to find an accurate model of this event. It is important to notice that not only the physical process of two occluding particles itself (including diffraction and other physical phenomena) has influence on this model but also the image acquisition including optical low passes and CCD or CMOS binning. The simplest model is the following: Assuming that a particle is much smaller than one pixel, we can consider it as Dirac impulse. Due to the optics, this Dirac impulse is low pass filtered and thus convolved with a Gaussian. This conforms with the assumption of Gaussian-shaped particles. Furthermore, they also appear larger than they actually are. (We observed this behavior in experiments with particles of $70\mu m$ diameter which appeared to be up to two times larger than they actually should be.) These assumptions are obviously not valid for larger particles and in fact the simulation of particle occlusion remains to be investigated more closely. Up to now, based on the above made assumptions we can simply add up particle intensities. Additionally, we avoid that particles fully occlude each other by defining a minimum distance between each two particles which can be passed to the PG.
A third important consideration is noise. In our case we use additive Gaussian noise with zero mean. Nonetheless, the PG allows to use any noise with individual variance and mean at each pixel location. Hence, camera properties can well be measured and integrated into particle detection experiments. The mean image can for example be interpreted as background image.
Once the whole image has been generated by randomly seeding particles (drawing from a uniform distribution) onto a black background, the intensity values are quantized to a given bit range (in our case always eight bit). Overshooting and values below zero (due to noise) are truncated to this range.
The PG as well as our algorithm are available as C++ cross-platform open source code and binary for windows and Linux upon email request. Furthermore, the images generated for the specific tests in this paper are made available.

6.6. Results and Discussion

In this section we not only test for the properties of our proposed algorithm. We also try to make a first suggestion for a common way to analyze particle detection routines for various applications. Important questions concern the minimum possible size of particles, influence of noise and background images and a study of the effect of occluding or clodding particles. Of special interest are extremes: what is the maximum noise, the minimum size or intensity and so on. Thus, we posed the following questions.

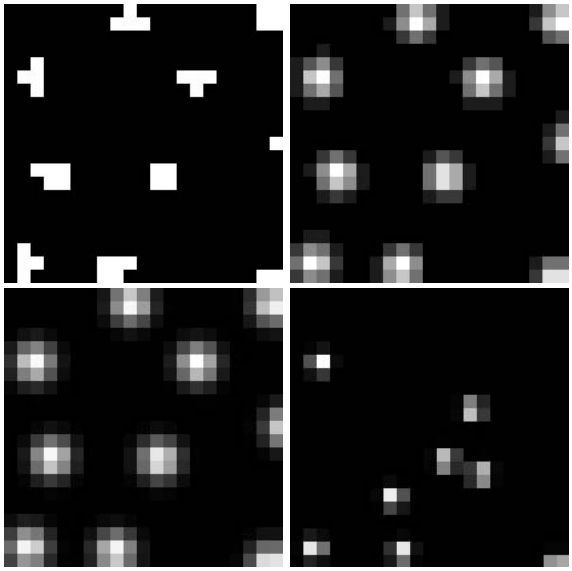

Figure 6.5.: Several particles with $\sigma = 1$ showing different diameters with intensity 2 (top left), 10 (top right) and 100 (bottom left). Bottom right: particles with $\sigma = 0.5$ and intensity 100.

6.6.1. Without noise, how can we still achieve optimal results?

It is important for applications to define an upper quality limit of the algorithm under ideal conditions. As we segment particles rather than searching for subpixel-accurate particle positions, we define an optimal result as the right answer to the question: Have all particles in an image been found without any false positives or false negatives? A particle is found if the true particle location differs at most one pixel from the estimated particle location (which is given by its momentum). Subpixel-accurate measurements can then still be applied as a the second building block of the whole PTV algorithm.
In our experiments with synthetic data we first created a non-overlapping particle image with 1000 particles and no noise. The size of the particles was set to $\sigma = 1.0$ and we quantized the image with eight bits. Under these circumstances, the number of visible pixels depends on the particle image intensity, which we varied. For an intensity of 10, the particle diameter is four and the particle area consists of 14 to 16 pixels. For a (very low) intensity of 2, the particle diameter is 3 and the area ranges between 4 and 6 pixels. Example particles are shown in Figure 6.5. Under these circumstances, any algorithm should be able to detect all particle images, which we want to prove four our algorithm.
The particle detection algorithm was configured with the following parameters: Minimum and maximum area for the connected components are set to 3 and 40 respectively, whereas the maximum area has no influence on this experiment. (This is due to the fact that no connected components will be computed that do not belong to actual particle images and that the size of all particle images is known and much smaller than 40.)

The maximum distance allowed for momentum differences is set to one and the stability threshold to five. As the image has no noise and particles do not overlap, the parameters for convexity and noise analysis are arbitrary. The image is blurred with a Gaussian of $\sigma_b = 0.3$ just to convert the quantized image into a floating point image. The two important parameters of this synthetic experiment are the number of thresholds $N = 500$ and the maximum occurring intensity I_{max}. As in real-world applications the latter can easily be roughly evaluated by looking at the images (up to noise uncertainties), we set I_{max} to the correct value (i.e. 10 and 2 respectively).
As expected, in both cases of intensity 2 and 10, the algorithm finds all particles. Nevertheless, the number of particles which are between a half and one pixel distance from the true location, increases: For an intensity of 10 only 50 of 1000 particle centers are off by more than 0.5 pixel, whereas for an intensity of 2, 580 particle centers are off by more than 0.5 pixel. This is due to the fact that for very small particles and intensities the subpixel center has more influence on the fact which locations receive which value (only a single intensity is used for all particle images as can be seen in Figure 6.5). However, as the particles do not overlap and have at least three pixels diameter, being off by one pixel for the location estimate can be uniquely ascribed to this phenomenon.

6.6.2. How do variable particle image intensities affect the outcome?

As the number of thresholds together with the stability threshold and the maximum occurring intensity determine the intensity range in which particles can still be detected, it is important to investigate this property of our algorithm. Therefore, we modified the above described experiment, by randomly varying the intensity range (drawing from a uniform distribution). The parameters of the detector remain the same, except that we set I_{max} to 255 and the minimum connected component area to one. Each range is between i_0 and 255 with $i_0 \in [10, 7, 5, 3]$. No false positives are found by the algorithm and as can be seen in Figure 6.6, for an extreme range of [3..255], 3.6% of all particles are missed, all of which are at the lower intensity limit. Thus, particles with intensities around the noise level are neglected. (What is considered as noise level depends on the number of thresholds and the stability range.)

6.6.3. How do variable particle image sizes affect the outcome?

In this experiment, instead of changing the intensity of the particle images, we vary the particle size. Based on the first experiment we set the particle intensity to 10 were optimal results could still be achieved. The particle size ranges where chosen to be $[\sigma_{min}..\sigma_{max}] = \{[1.0..2.5], [1.0..4.0], [0.5..1.0]\}$ from which we sampled randomly.
Again, the parameters of the PG remain the same, except that I_{max} is set to 10.
The results show that the algorithm is rather invariant with respect to particle sizes. For the first two ranges no false positives or false negatives have been detected and all particles are found correctly. In the first range 9.6% of all particles are off by one pixel, whereas in the second range this value is 12.6%. This is due to the fact that quite large particles around $\sigma = 4.0$ contain more than one pixel with maximum intensity so that

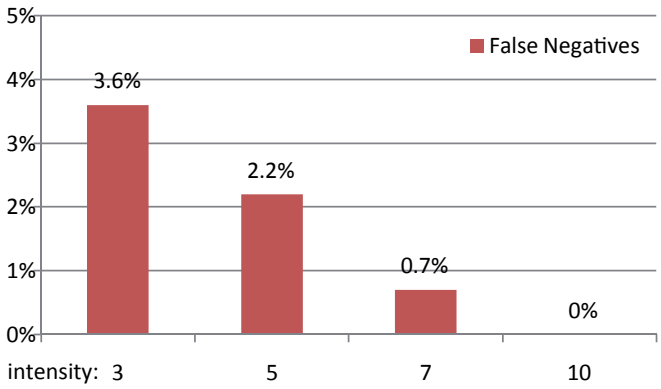

Figure 6.6.: False negative rates for an image containing 1000 random particle images in the intensity range $[i_0..255]$ with $i_0 = [3, 5, 7, 10]$. Even for very small intensities, particles are still detected accurately. (cf. Section 6.6

the center cannot be estimated as reliable as with smaller pixels.
For the third range, 2.3% of the particles (4.5% off by one) have not been detected, showing that particle sizes around $\sigma = 0.5$ are too small to be found reliably.

6.6.4. How close are particles allowed to be to still achieve a good result?

Whenever particles overlap, the particle detection becomes more difficult. The main problem for our algorithm is here (as illustrated in Figure 6.7), that the momentum only is stable in the intensity range of the particle peak and the closest local minimum. (As an example consider two close and added 1D-Gaussians: the local minimum of the valley between the two mean values of the Gaussian is the intensity at which the two momentums of the connected components start to remain constant; compared to two Gaussians which are far away from each other this causes a significantly decreased stability range.) Thus, the smaller and closer particles are, the more difficult the detection becomes.
In order to get an idea how much particles are allowed to overlap we carried out an experiment where we specified the minimum allowed distance between particles in pixels. As in our simulation overlapping particles add up their intensities, we set the intensity of all particles to 128 to avoid any values above the quantization range of 8 bits. In the first experiment the particle size was $\sigma = 1.5$ (a realistic value for many applications) and results are also shown in Figure 6.7: For a distance of 6 pixels only 0.1% were not detected, whereas the number of missed particles rapidly increases up to 29.3% for 1.5 pixels.
In a second experiment, we shrank the particles to a very small size of $\sigma = 1.0$. For a distance of 4.5 pixels 7.6% of all particles are missed and for 1.5 pixels already 44.8% cannot be found.
This shows that our algorithm neglect particle clusters caused by either lumped particles or inherently three-dimensional overlaps of freely moving particles. Depending on the

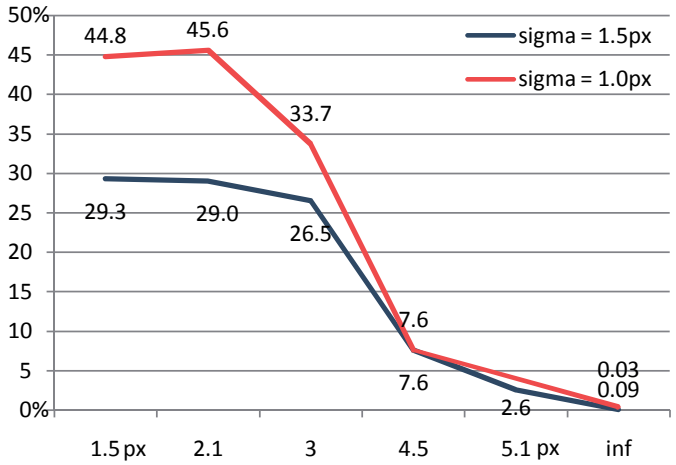

Figure 6.7.: False negative rates for various minimum allowed distances between particle images for the experiment described in Section 6.6. For small distances overlapping particles are not detected. Nevertheless, a postprocessing of such particle chunks could be used to improve rates for PTV-algorithms that cannot deal with this property.

application this bias towards false negative rates can be exploited as for example with the cross-gap tracking strategy of Li et al. [88]. Furthermore, most of these particle clusters are rejected due to their non-convexity. Hence, for such cases an additional step could be included in the algorithm that tries to seperate particle lumps from overlaps in order to reconstruct several particle centers from clusters caused by overlaps. This is a matter of future research.

6.6.5. What noise levels and background images are allowed?

To evaluate the noise sensitivity of the algorithm we created relatively small particles of size $\sigma = 1.0$ but with a constant intensity of 255. The latter is motivated by the notion of signal to noise ratio: The darker particles actually are, the closer they are to the noise level. The same effect can be achieved by scaling up the noise i.e. only the relation between signal and noise plays a role for the detection rate (up to the quantization error). As described above, Gaussian noise is added to the image and values above 255 and below 0 are truncated.
The variance of the Gaussian noise was set to 200%, 400% and 800% of the maximum intensity (6, 12, and 18dB SNR accordingly). As it turned out 200% noise does not affect the detection ratio at all. For 400% noise, 12.6% of the particles are missed, but none are hallucinated. Only at 800% noise 13.3% particles are missing, 26.2% of the particles are spurious but still 86.7% of all particles are found correctly.
To further increase the detection difficulty, we added a background image from a real experiment containing a strong intensity gradient and smears on a transparent surface.

The final result (with added noise) is shown in Figure 6.1. The detection rate dropped by 0.1% to 86.6%.

6.7. Conclusions

Similar to the discussion in Chapter 3 we suggest to analyze and validate each algorithm module separately to successfully improve each algorithmic element of an PTV algorithm. To this end, we propose an evaluation framework for this specific algorithmic building block of PTV techniques and a new particle detection algorithm. The framework is based on a particle image generator which can be parameterized to create both realistic and ideal particle images. Three simple parameters can be used to adapt the algorithm output to specific types of image sequences. Other intuitive parameters can be used to fine-tune the results. Experiments showed that our algorithm is highly robust with respect to Gaussian noise, even with spatially varying mean values. For noise levels as high as 800% of the actual particle intensity we still achieve a true positive rate of more than 86% whereas the algorithm only generated 12% of false positives. Under more realistic assumptions, with 200% noise, 100% sensitivity could be obtained without any false positives. Occluding particles are regarded as lumped cluster and therefore rejected as soon as the distance of the particle centers becomes smaller than 4.5 pixels. Considering such large amounts of noise given the high detection rates, our algorithm is an ideal extension for a variety of PTV methods that can deal with particles missing due to occlusions. Hence, subsequent PTV algorithms could either deal with this problem by exploiting the fact that very little false positives exist in order to faciliate this problem (cf. [88]). As such particle occlusions can be detected quite easily, another possibility would be to include an additional step to seperate occluding particles - a matter of future research.
To contribute to the algorithm requirement of accessibility as described in Chapter 3, all algorithms and results are made publicly available to facilitate future research and extensions of our proposed evaluation framework.

As the proposed particle detection routine (or estimatibility measure for PTV image data) is just a single module of Charon we did not fully discuss the algorithm requirements for optical flow algorithms. Instead, in the following chapter, we will introduce a full motion estimation algorithm specialized on PTV data which will be tested against all requirements defined in Chapter 3.

7. Trajectory Ensembles for Fluid Flow Estimation

> The most exciting phrase to hear in science, the one that heralds the most discoveries, is not "Eureka!" but "That's funny..."
>
> *(Isaac Asimov)*

Until now we have concentrated on finding out those locations of an image sequence acquired for PTV where the motion of individual particles can be estimated at all. This chapter tackles the actual estimation of the motion at these locations. As emerging emerging Fluid Dynamics applications often deal with highly three-dimensional flows we will focus on this topic here. Through the projective nature of the imaging process complex and often occluding motions of the tracer particles are visualized. Despite the complex motion, optical flow estimators usually concentrate on spatial regularization techniques, ignoring the rich temporal structure of such long sequences. We, therefore, propose to bundle spatially nearby trajectories into an ensemble, parameterized by linear subspace estimation. This accounts for defining a new neighborhood module together with a motion model module in Charon. Thereby, we obtain an adaptable representation of the abundant temporal information and account for diverging and converging pixel neighborhoods. Using this model, more accurate motion estimates can be extracted without requiring spatial smoothness or piecewise constancy. Experiments on relevant real and synthetic data sets show the superior performance of this approach compared to two traditional approaches.

7.1. Introduction

Traditional approaches to optical flow estimation in computer vision utilize a very short time window of at least two frames, assuming that both the brightness of the structures contained in these images and the motion between each two frames remains constant [41]. To improve such methods, most research in this field concentrates on modeling prior knowledge on typical flow fields for spatial regularization techniques [43]. In Fluid Dynamics, this is the field of so-called particle image velocimetry (PIV).
Nonetheless, in many applications spatial regularization is problematic. This is due to the fact that highly three-dimensional motions of tracer-particles seeded into fluids are projected onto the image plane. Hence, in a small neighborhood, often a large number of completely different particle motions can be observed, violating spatial smoothness

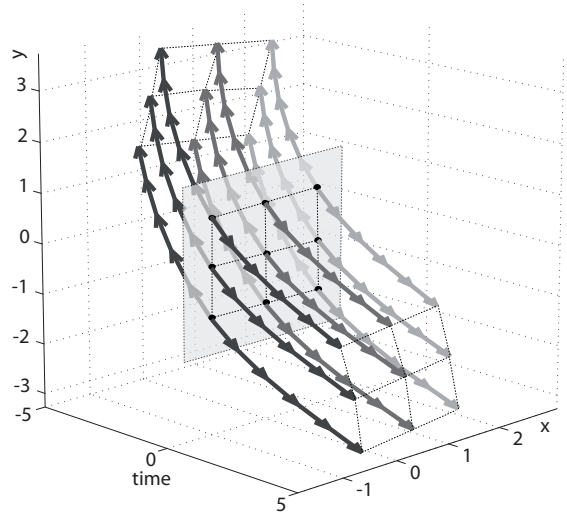

Figure 7.1.: Example for a trajectory ensemble with $l = 5$ and $w = 1$ of a diverging (c.f. black trajectories in front) and rotating flow modeling the training data depicted in Figure 7.8.

assumptions. With the emergence of more transient and turbulent flows currently being researched (e.g. [89]) the inherent three-dimensionality of particle motion can no longer be ignored.

Image sequences obtained by such experiments are usually of very high frame rates resulting in temporally smooth particle trajectories. A natural model would represent the motion of the same particle over time, not the motion of several objects in some spatial pixel neighborhood. Thus, applying spatial regularization is counter-intuitive, whereas inherent temporal image information is currently not exploited adequately. In Fluid Dynamics, one resorts to so-called particle tracking velocimetry (PTV) where individual particles are first located and then tracked. In contrast to most computer vision tracking algorithms, PTV methods have to be highly accurate. State-of-the-art algorithms are able to achieve around 0.1 pixels in accuracy [90].

Our contribution is the regularization of fluid flow estimation by means of trajectory ensemble models: Based on [**?**], we extend the concept of purely temporal trajectories by bundling a set of trajectories of a local neighborhood into an ensemble, or set of coupled trajectories (Fig. 7.1). This alleviates the problem of ambiguous intensity structures without posing many motion model assumptions on the spatial domain.

In our approach we obtain trajectory ensembles from computational fluid dynamics (CFD) simulations and use them as input data for a principal component analysis. The resulting parameter set jointly describes the motion and the interaction of trajectories of the whole ensemble. This enables us to allow diverging and converging pixel neighborhoods without the need for enforcing any kind of spatial motion constancy or smoothness. One estimated parameter set of our method describes the whole trajectory ensemble. Hence, this information can be explicitly exploited, as for example to initialize

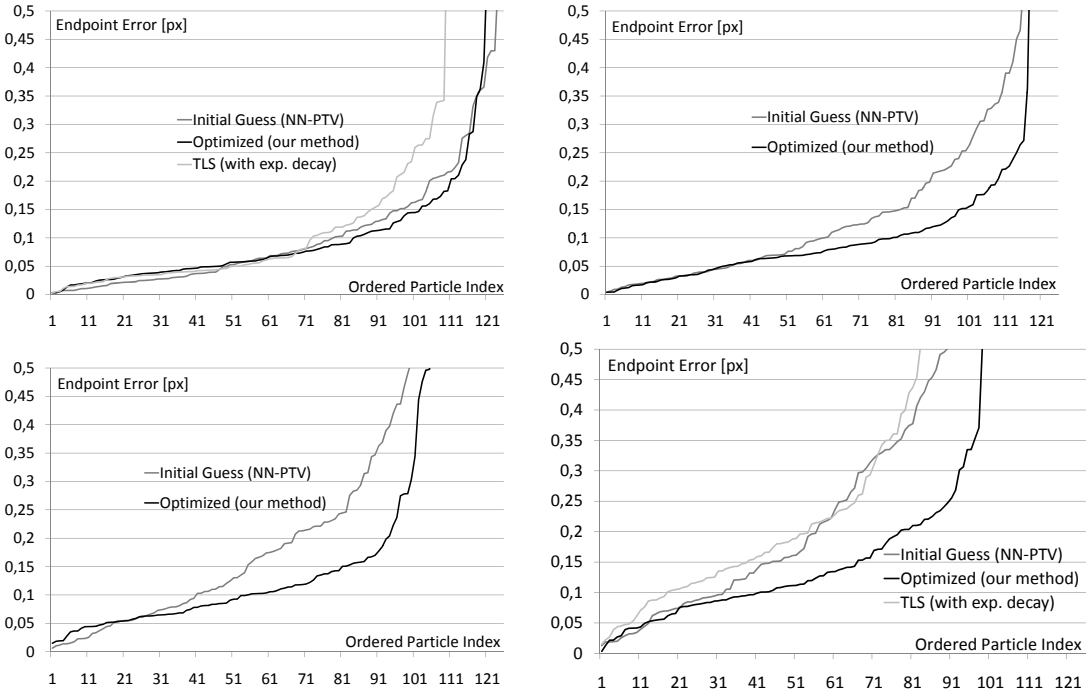

Figure 7.2.: Sub-pixel errors for noise levels of 1.0 (top left), 2.0 (top right), 4.0 (bottom left) and 6.0 (bottom right) gray values.

the optimization routine in the next frame or to extract other motion properties such as curvature or acceleration of the motion. This intuitive approach maintains a highly adaptable description of motion while utilizing a higher amount of temporal image information. We tackle the problem of the resulting nonlinear optimization problem by means of a randomized gradient descent, showing good results on relevant test sequences. As explained in Sections 7.4 and 7.6.3, this approach works well without the need for image derivative filters and linearizations, reducing typical problems with small particle sizes.
In the remainder of this paper first a rather detailed overview will be given of both optical flow and fluid flow estimation in order to familiarize readers from both worlds with the most important concepts. We then formalize our approach to fluid flow estimation and discuss experiments and results. Finally, possible uses of our concept of trajectory ensembles in a broader computer vision context are extrapolated.

7.2. Related Work

As this work focuses on two large fields of research, namely Fluid Dynamics and Computer Vision, we divide this section into two parts. The first parts discusses related work in the field of computer vision. The second part deals with publications in Fluid Dynamics.

7.2.1. Optical Flow Estimation with Temporal Constraints

The application of global optical flow methods in Fluid Dynamics recently became very popular. Since Corpetti et al. [91] stated in 2006: "The design of alternative approaches [of optical flow estimation] dedicated to fluid motion thus remains a widely open research domain", many authors contributed to this topic, as for example most recently [92, 93, 94].
Other local methods based on the structure tensor approach of [37] exist, which are especially suited for physical models of brightness changes and have originally been derived by Haussecker and Fleet [13]. To this end, the estimation of the full three-dimensional velocity field of particle image sequences recently became possible by means of PIV techniques [95]. Generally, in optical flow estimation, few approaches using mainly temporal regularization are available. Both the structure tensor [37] and the orientation tensor [7] approach incorporating temporal information by means of temporal derivative filters and temporal integration scales for orientation estimation. Our work is inspired by [96], where temporal coherence of motion is enforced by assuming a constant acceleration of moving objects, whereas in [97], spatio-temporal blocks of motion are analyzed in order to detect motion events. Specifically, in the field of Fluid Dynamics, analytical spatio-temporal regularization based on physics has been studied in [92], yielding very good results on mainly two-dimensional flows. Other ideas on the use of temporal information in global methods are presented in [98, 99, 100]. The use of motion trajectories was firstly mentioned by Agarwal and Sklansky [101]. Their idea was to search within a quantized set of possible flow vectors in a three-step algorithm. The last step consists of a refinement of the motion estimates based on intensity values in a spatial and temporal neighborhood of pixels along the motion trajectory. Chaudhury and Mehrotra [102] use trajectory information, too and, in their introduction, refer to psychological evidences [103, 104], that this information can more easily be extracted by the human visual system than mainly local motion estimates. More recent psychological research further supports this idea: for example, Verghese et al. [105] found that humans can more easily detect long instead of short trajectories. In [102] a global approach similar to that of [41] is formulated using two regularizers that use the Euclidean norm for spatial smoothness and the local curvature of a three-frame-trajectory for temporal smoothness of the flow field. Gibson and Spann [106] (by extending the work of [107]) propose to first estimate the trajectories parameterized by the physical laws of constant acceleration along some interest points (as is commonly done in tracking) and then to interpolate the results to a dense motion field based on piecewise affine motion models.

Another aspect of motion model driven regularization is the creation of models based on unsupervised learning. Our approach to learn a model is inspired by the work of Black and Fleet [108, 109] whose ideas have proved very successful. They learn, for example, spatial parametric models of rectangular blocks of flow fields for the classification of motion events with applications to human motion tracking. Yacoob et al. [110, 111] formulated a model of single trajectories and inserted it into an energy function for optical flow estimation. They linearized the energy based on the assumption that the local intensity distribution is sufficiently smooth and the motion accordingly small so

that the brightness constancy constrained equation (derived by [41]) holds true at each location. Furthermore, based on the same assumptions the flow vectors are applied to image positions along the trajectory itself. Instead, the flow of each point of the trajectory is inserted at the same location as the center pixel (c.f. equation 2 in [111]) in order to make linear least squares estimates possible.

7.2.2. PIV and PTV

The approach proposed in this paper comprises components from both PIV and PTV. We shortly summarize these approaches to familiarize the reader with the most important concepts and parallels to computer vision. Please note that this review is not intended to be comprehensive.

Particle Image Velocimetry In PIV the fundamental technique is the cross correlation of so-called interrogation windows with a second frame. This was first carried out algorithmically by [112]. (Prior to this paper mechanical/optical methods were applied for velocimetry.) This method is very similar to the block-matching approach of Anandan et al. [36]. However, the cross correlation technique has been further refined over the past two decades so that PIV can today achieve average accuracies around 0.1 pixels as mean endpoint error (cf. [30]). A few milestones in development are the following:

- Sub-pixel displacement estimation became possible by fitting an analytical function to the correlation plane and determining its maximum. This was first suggested by Willert and Gharib [113].

- Multipass/Multigrid interrogation is a technique similar to image pyramids for optical flow. Here the window size is successively reduced from a large to a small scale [114]. An image warping technique similar to many registration methods was suggested by Jambunathan et al. [115].

- As PIV essentially is a block-matching technique, window deformation techniques and window shifting techniques have been proposed, e.g. in [116, 114]

- Second order correlation [117] is based on the idea to find flows on a fictive third image lying exactly in the center between the first and the second image. This approach is very similar to the one of Alvarez et al. [93] in computer vision. There, the concept is called symmetrical optical flow.

A common data set for all methods has been established in three so-called PIV-challenges [118, 119, 120] where many groups contributed their flow estimates to facilitate the comparison of state-of-the-art methods. A similar approach has been undertaken by Baker et al. in the compute vision community [30]. A comprehensive overview on PIV methods can be found in [23].

Particle Tracking Velocimetry PTV is another well established approach to the fluid flow estimation problem. This approach is typically used when particle densities are small. Additionally, Pereira et al. [121] argue that for 3D velocimetry, particle tracking would the best method because the actual 3D volume being visualized contained few particles as well and the projection onto a 2D image plane violated fundamental assumptions of PIV techniques. PTV is typically divided into the following steps:

- Preprocessing: E.g. image sensor noise is reduced and the background is subtracted.
- Particle Detection: For each pixel a decision is made, whether it belongs to a particle or not.
- Particle Localization: The center of each particle is located with sub-pixel-accuracy.
- Particle Tracking: The correspondences of the particles between image frames are determined.
- Postprocessing: E.g. interpolation or confidence estimation of the sparse motion estimates is performed.

The first PTV algorithm was suggested by Adamczyk and Rimai [122] in 1988. (Prior to this paper particle tracking was carried out manually by e.g. clicking on particle image pairs.) The particle detection and particle tracking steps are the most crucial steps to find correct correspondences: Whenever a spurious particle is detected or one is missed (e.g. due to occluding particles or sensor noise), no correspondence can be found. The tracking step quality suffers from ambiguities caused by too fast flows, and particle detection errors. Typical approaches for particle segmentation are global thresholding ([75]), dynamic thresholding ([76, 77, 79]), correlation-based methods ([80, 81]), region-growing methods ([82, 95]) and learning-based methods ([85, 25]). Citations in brackets are exemplary publications, showing that this problem is not fully solved until today. A recent paper suggests a new method for a validation and comparison framework based on synthetic particle images [3].
The methods for tracking particles are as numerous as particle detection algorithms. Usually, only two to four frames of the image sequence are utilized for tracking to reduce the combinatorial complexity of the matching-problem to a manageable amount. Typical approaches are multiframe tracking heuristics [124, 121], combinatorial [125], statistical [126] or variational optimization [127]. An interesting fact is that in the PTV community a lot of references to the tracking community in computer vision can be found (cf. the PTV chapter in [24]). As our approach uses a simple nearest-neighbor particle tracking algorithm as an initial guess for a subsequent PIV-like matching algorithm, we will not go into detail here.

7.2.3. Relations Between Fluid Flow Research and Computer Vision

With this subsection we wish to emphasize the similarities between both fields in order to motivate a broader research perspective that could help solve problems in both ap-

plication domains.
Motion estimation in computer vision can be divided into three subsets: In registration, usually a dense displacement field between two often multidimensional images (sometimes of differing modality) is sought. In tracking, high level information as for example the three-dimensional transformation of a whole object has to be computed from a long image sequence. In between lies the task of optical flow estimation. In most cases, a short time window of longer unimodal image sequences is examined to calculate a dense flow field between each two frames. It often constitutes one step in a list of low-level algorithms being used as input for high level computer vision systems. These fields developed separately from each other since around 1980 [41].
Very similarly, fluid flow researchers started to use computers for motion estimation in the eighties [112, 122]. In both fields of research block-matching algorithms were developed around 1986 [112] which led to a broad acceptance of motion estimation algorithms. Local methods based on least squares methods have been applied to both computer vision and fluid flow problems [17, 95]. The same is true for global variational methods which recently have been adapted to fluid flow problems [41, 127]. These examples show that computer vision and fluid flow research have been closely related to each other since the very beginning of their research.
Despite, fluid mechanics has been largely ignored by the computer vision community: According to Google Scholar, in the past two decades only around 10 papers containing the terms PIV or PTV have been published in four large journals (namely "International Journal on Computer Vision", "Transactions in Image Processing", "Pattern Analysis and Machine Intelligence" and "Computer Vision and Image Understanding"). In contrast, in fluid flow research, a recent burst of optical-flow-related papers can be observed (e.g. [120]), signaling the interest of the researchers in alternative motion estimation techniques.
As the estimation of fluid flow has to be highly accurate and therefore the estimation algorithms have to be carefully validated, an abundance of related papers can be found. Validation techniques range from synthetic image sequences (e.g. [78, 127, 121]), real sequences with known flow characteristics (e.g. [128, 114, 95]), confidence measures (e.g. [129]), lower accuracy limit discussions [90], detailed error analysis (such as error histograms instead of mean and variance only, cf. e.g. [116]) and theoretical investigations (e.g. [130, 131]). As a result, the algorithms in fluid flow estimation are well understood, well established and regularly employed in commercial applications. Independently, in computer vision similar approaches have been investigated, but have not found such a broad acceptance. A short review on confidence measures can be found in [26].
We believe that for example the ground truth generation of complex scenes by means of computational fluid dynamics or the experience in experimental setup design for the generation of real world sequences with ground truth are just two interesting aspects of which computer vision could profit. Vice versa, fluid flow estimation algorithms that utilize optical flow techniques have already shown to be helpful for the recovery of physical flows.

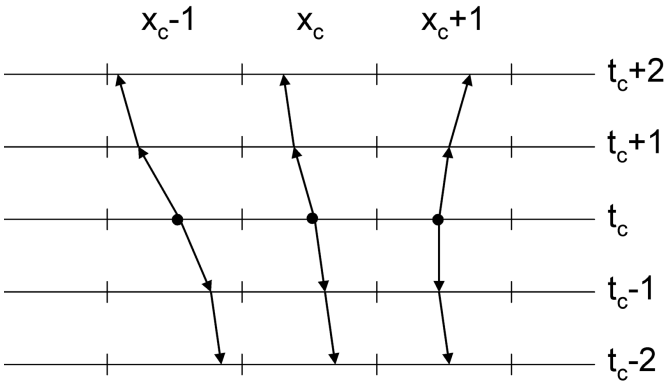

Figure 7.3.: A one-dimensional example for a trajectory ensemble in a three-neighborhood around $(x_c, 0, t_c)$. Each trajectory consists of four flow vectors relative to the position of the trajectory at time frame t. By the application of PCA to a large set of training ensembles, this representation can finally be described by a few parameters.

7.3. Trajectory Ensembles for Optical Flow

The idea of using trajectory ensembles as described in Section 7.1 can be formalized by first defining a trajectory ensemble as a set of trajectories in a spatial neighborhood. Let $\mathcal{I} \subset \mathbb{N}^3$ denote the domain of a given image sequence and $I : \mathcal{I} \to \mathbb{R}$ a map to the image intensities. Furthermore, let $\vec{x} = (x_c, y_c, t_c) \in \mathcal{I}$ be the location at which the flow is to be estimated. Then we define the (frame-independent) pixel-grid neighborhood of $\vec{x}$ of width and height $2w$ as:

$$\mathcal{N}_{\vec{x}} := [x_c - w, x_c + w] \times [y_c - w, y_c + w], w \in \mathbb{N} \tag{7.1}$$

A single trajectory of length $2l$ at this location can be defined as a parametric function:

$$T_{\vec{x}} : [t_c - l, t_c + l] \to \mathcal{I},\ l \in \mathbb{N} \text{ with } T_{\vec{x}}(t_c) = (x_c, y_c, t_c) \tag{7.2}$$

This definition ensures that a trajectory always crosses the image plane in frame t_c at the predefined location (x_c, y_c, t_c). Furthermore, let $\vec{d} = (x_d, y_d, 0) \in \mathcal{N}_{\vec{x}}$ be a spatial displacement vector. Then we define a trajectory ensemble as:

$$T_{\mathcal{N}_{\vec{x}}} := \{T_{\vec{x}+\vec{d}}(t) | \vec{x} + \vec{d} \in \mathcal{N}_{\vec{x}}\} \tag{7.3}$$

A two-dimensional example for this description is given in Figure 7.3. The resulting trajectory ensemble model now consists of a discrete set of flow vectors. Techniques from unsupervised learning can now be used to parameterize the flow within a neighborhood of fixed size. We applied the standard technique of principal component analysis [132]. (This technique is commonly known as proper orthogonal decomposition in the Fluid Dynamics community.) To increase the performance in the presence of outliers

for example robust PCA [133] can be applied instead. Based on the training data, these subspace estimation methods yield a map from a set of k parameters to an actual trajectory ensemble: $M : \mathbb{R}^k \rightarrow T_{\mathcal{N}_{\vec{x}}}$. The inverse mapping $M^{-1}(T_{\mathcal{N}_{\vec{x}}})$ can be used to find the best matching parameters describing a given trajectory ensemble, e.g. in order to transform an initial guess flow field into parameter vectors p.

Several options for the choice of training data for learning M exist. The simplest possibility is to heuristically generate synthetic trajectory ensembles (similar to [108]) that approximate a representation of the set of possible ensembles actually occurring in the image sequence to be estimated. Another method is to learn a model based on ground truth data [66]. The main problem of the latter is, that for a specific application valid ground truth data might not be readily at hand. A good tradeoff turned out to be the use of motion estimates of other algorithms. In this case, however, the problem of outliers and underrepresentation of ensembles which cannot be estimated by a given algorithm have to be carefully addressed. As mentioned above, in the field of Fluid Dynamics yet another method is feasible: Many physical fluid flows can be simulated by Computational Fluid Dynamics, allowing for highly application specific training data. We utilized such CFD data as is described in Section 7.6.

7.4. Optimization

For the optical flow estimation proposed in this paper, we define an energy function based on the discrete set of flow vectors of $T_{N_{\vec{x}}}$ together with a similarity measure $\mathcal{S}(\vec{x}, \vec{d}, I)$:

$$E(p, \vec{x}) := \sum_{\vec{d} \in M(p) = T_{N_{\vec{x}}}} \mathcal{S}(\vec{x}, \vec{d}, I) \tag{7.4}$$

For the similarity measure $\mathcal{S}$, many choices are possible. One possibility used by [111] is to insert the flow into the brightness constancy constrained equation (BCCE). But as the BCCE is simply a linearized version of the squared differences (SD) of intensities along a given trajectory [41] and our energy function is nonlinear due to the parameterization anyway, we directly apply SD as the similarity measure:

$$\mathcal{S}(\vec{x}, \vec{d}, I) := (I(\vec{x}) - I(\vec{x} + \vec{d}))^2 \tag{7.5}$$

In future research, other similarity measures should be examined as for example statistically robust measures proposed by [5]. In order to minimize the nonlinear energy function $E(p, \vec{x}(t))$ with SD as similarity measure, many optimization techniques can be applied. The parameter vector p consists of a relatively small number of values (usually $k \leq 8$). As it turns out, the optimization problem can be regarded as continuous and bounded. This becomes apparent by noting that the PCA model contains the variance for each parameter and by projection of all training samples into the parameter space we can obtain estimates of minimum p_{min} and maximum p_{max} values for each parameter.

To solve the resulting optimization task, a first idea would be to discretize this search space as is done in correlation-based methods [36]. Anyhow, this requires the invention

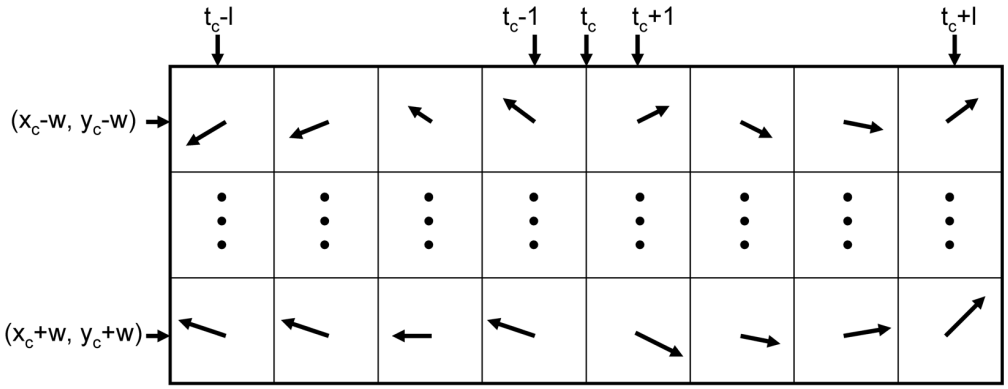

Figure 7.4.: The data structure of a trajectory ensemble consists of one trajectory per row. Each row consists of two lists of flow vectors: The first list describes the trajectory moving forward in time, the second list contains the backward-part.

of a number of heuristics, exploiting the special properties of our energy. A more general method based on fewer assumptions would be particle filtering, as for example used in [134]. It is a good method for keeping the parameter set continuous but reducing the number of parameter evaluations by means of statistical sampling. Finally, as nonlinear optimization is yet another major field of research, more elaborate methods for energy minimization exist, e.g. line search and trust-region methods [135].

To allow for a simple reimplementation of our method, we apply a trivial gradient descent method with one minor modification: in order to increase the probability of finding a global energy minimum, similar to particle filtering, we do not start our search at a single initial guess. Instead, we seed a number of random parameters by adding noise to the initial guess. This noise is uniformly distributed in the range $[-(p_{min} - p_{max})\sigma, (p_{min} - p_{max})\sigma]$ with $\sigma \approx 0.1$.

7.5. Implementation

For the implementation, a data format for trajectory ensembles and a discretization scheme for the gradient descent are necessary. An additional problem is to create trajectory ensembles which are guaranteed to pierce time frame t of the image sequence at a regular grid centered around $\vec{x}(t)$ (c.f. Figure 7.3). To this end, we let each trajectory of the ensemble start on frame t at location $\vec{x}(t) + \vec{d}$ and create flow vectors simultaneously forward and backward in time. Then, in a second array, we store inverse flow vectors between frames t and $t - l$. As a result, each trajectory consists of $2l$ flow vectors, "spreading" from frame t forward and backward in time. Hence, a five-frame trajectory in a neighborhood of $w = 1$ has 72 values. Reducing the number of values by PCA often results in as few as three or four parameters, containing 99 per cent of the total energy of the eigenvalues. Nevertheless, this data format leads to one drawback: Ground truth

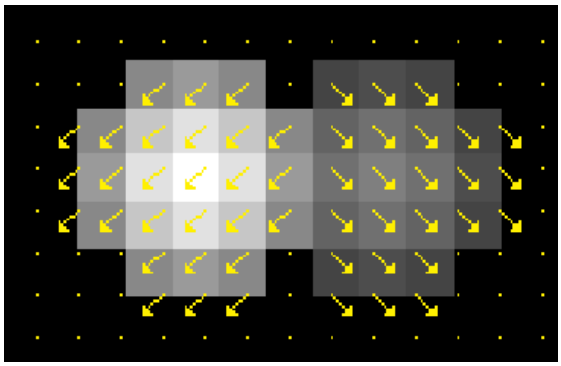

Figure 7.5.: Two particles moving diagonally through the image at different constant depths (hence one particle is brighter). They cross in the center with a complete overlap. The ground truth flow vectors are scaled for visualization.

data of common test sequences does not contain the inverse motion vectors. This is a problem, because the inverse for diverging or converging flow fields is not defined. As an approximation, we map each exact inverse flow onto a non-discrete triangular mesh and interpolate this mesh at pixel grid locations. The mesh is created by calculating $\vec{x}(t)+\vec{u}$ for each $\vec{u}$ and associating $-\vec{u}$ with it. The mesh of the resulting two-dimensional point cloud is computed by means of Delaunay Triangulation. Flow values at pixel grid locations are extracted via trilinear interpolation of the flow values associated with the points of the enclosing triangle.

For the finite difference approximation of the gradient we add a small increment h to each dimension of p in turn, evaluate the energy and subtract it from $E(p, \vec{x})$. To alleviate the problem of local minima, we calculate the initial guess by another motion estimation algorithm (cf. Section 7.6). As a convergence criterion we evaluate the amount of energy change after each step. For convergence, we apply infinite iterations until the energy change is around two to ten magnitudes smaller than the initial energy. If no convergence is achieved, no result is saved.

The resulting algorithm is carried out on each pixel location: Firstly, an initial guess is read and projected into parameter space by $p = M^{-1}(T_{N_{\vec{x}}})$. Then p is optimized as described above, yielding p'. On convergence, the result is converted back into a trajectory ensemble by $T_{N_{\vec{x}}} = M(p)$ and the flow vector is saved as the solution.

7.6. Experiments and Results

As described above, the model of trajectory ensembles is motivated by the presence of highly three-dimensional motion of fluid flow experiments, rendering spatial regularization problematic. Hence, in experiments, we evaluate the performance of the algorithm on a synthetic three-dimensional sphere as depicted in Figure 7.6. We then analyzed the behavior with respect to different noise levels. Finally, we applied our algorithm to real data with known ground truth to validate the results observed in the synthetic sequence.

7.6.1. Test Sequences

In real-world fluid experiments, a light-reflecting tracer is often added to the fluid and recorded by high-speed video cameras. It is difficult and costly to derive highly accurate ground truth for real ground truth sequences. Therefore, to simulate the image acquisition process, we carried out the following procedure: The radius of the sphere was adjusted so that its minimum and maximum depths yield the brightest, respectively darkest, particle image possible according to Beer-Lambert's Law [95]. According to [77], the particle shape can be approximated by a 2D-Gaussian function, which we sample with a variance σ of 2 px for each particle location. We cropped the sampling process at 3σ, where it reaches the precision of our 8-bit camera. 200 particles are randomly and uniformly distributed within the sphere. The radius of the sphere is set to 150 px. Some particles are partially or totally occluded, so that around 120 particle images can be recognized by the particle detection routine, depending on the noise level. To approximate the Poisson process of light collection on the chip, we simulated Gaussian iid noise of increasing variances between 1 and 6 levels of intensity for each experiment [136]. The noise is added to the scene and values above and below the camera's intensity range were truncated. Finally, the intensities are quantized to 8 bit, i.e. to the values between 0 and 255. As visualized in Fig. 7.6, the sphere is rotated with a constant angular speed of one degree in y-direction and 0.2 degrees in x-direction. The maximum speed of a particle was approximately 2.6 px per frame.

Furthermore, to study the effect of ideal data, we created simple sequence with two particles that diagonally cross each other in one-pixel steps in both x- and y-direction. This helps to find an accuracy limit of our method and to study the effect of occlusions.

Finally, we created a real sequence of an ascending particle with known ground truth. For this purpose, single particles are moved to defined measurement points, where they are recorded by the wall-PIV setup described in [137]. To ensure the exact 3D-movement of a particle, a high precision, three-dimensional traverse system is required. Using the camera Fastcam Super10K and an inversely mounted Nikon Nikkor 50 mm 1:1.8 lens, our setup had a fixed focal distance of 143 mm and a resolution of 68 px/mm. Aiming for a targeted accuracy of up to 1/100 px for the reference data, a precise movement of less than a micrometer had to be assured. This precision is reached by a high precision milling cutter (at the Department of Precision Engineering and Micro Technology (MFG), TU Berlin), which allows a movement accuracy up to 100 nm in all three directions. Our optical setup was placed on this milling cutter so that single particle measurements with a three-dimensional movement of the particle close to a transparent surface could be realized. A particle P is mounted at the head of the milling cutter t where it can be traversed in all three dimensions. We use the milling cutter as 3D traverse and a chromatic sensor (Fries Research & Technology – FRT GmbH, Germany) with an accuracy of 20 nm, measuring the particles distance to the glass surface. A visualization of the resulting test sequence is shown in Figure 7.7

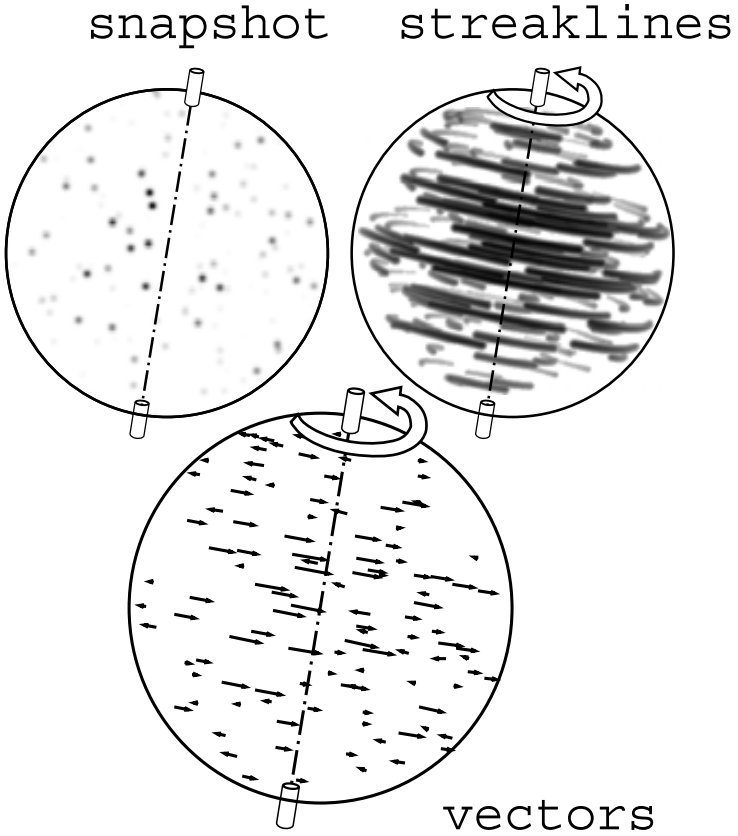

Figure 7.6.: Test sequence of a three-dimensional sphere rotating in space. The gray values are inverted to optimize the representation in this paper. On the upper left, the dots in the sphere are shown for a given position in time. Intensities of the particles are adjusted according to their depth. On the upper right, streak lines of the particles for a rotating sphere are shown. The corresponding calculated vectors are shown below.

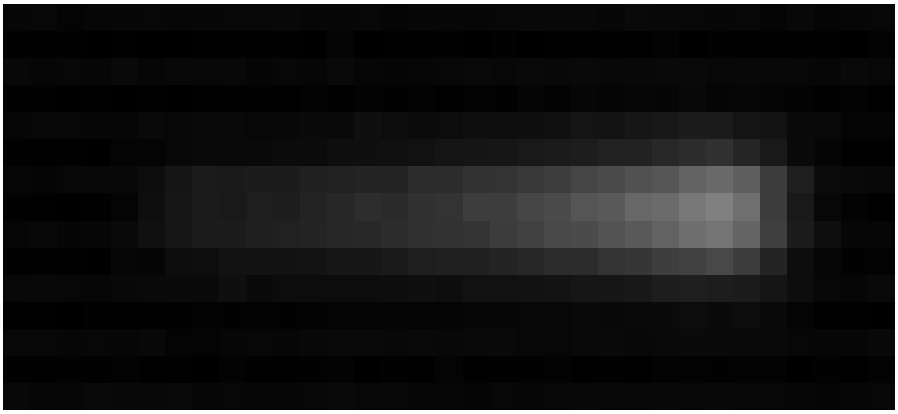

Figure 7.7.: The traversal of a real ascending particle image with known ground truth visualized as pixel-wise maximum image over all frames.

7.6.2. Training Data

To create realistic training data for our application, learning is based on the flow simulation inside a simplified model of a blood pump currently being developed. The computation of the steady laminar flow in the blood pump was done using the FLUENT6 solver [1]. An unstructured mesh of $1,600,000$ mixed cells was generated using the preprocessor Gambit (Fluent Inc.), whereas the simulation was carried out using a second order upwind discretization scheme for the convective terms of the Navier-Stokes equations (Reynolds number $Re = 600$). For the pressure-velocity correction we employed the SIMPLEC method with a no-slip boundary condition at all solid boundaries. We used a Newtonian blood model with kinematic viscosity $\mu = 3.5 \cdot 10^{-6}$ with a plug velocity profile at the inlet.
Thus, on the one hand, we are able to exploit application specific information on the flow field, but on the other hand effectively parameterize this knowledge to generalize over devices of similar type.
Please note that we learned from these realistic flows but applied the final matching algorithm to the simple sphere sequence. This was due to limited resources. In further experiments more CFD simulations are needed.

7.6.3. Experiments

The comparison of our approach with those commonly applied in the computer vision community is not an easy task. Firstly, as mentioned before, spatial regularization is not appropriate for images of highly transient and turbulent fluid flows: In the scale of a few pixels, robust regularizers [5] (as for example those used in the nonlinear CLG method [12]) often fail, because regions of transition between two areas of consistent motion

[1] (ANSYS-Fluent Inc., Lebanon, USA)

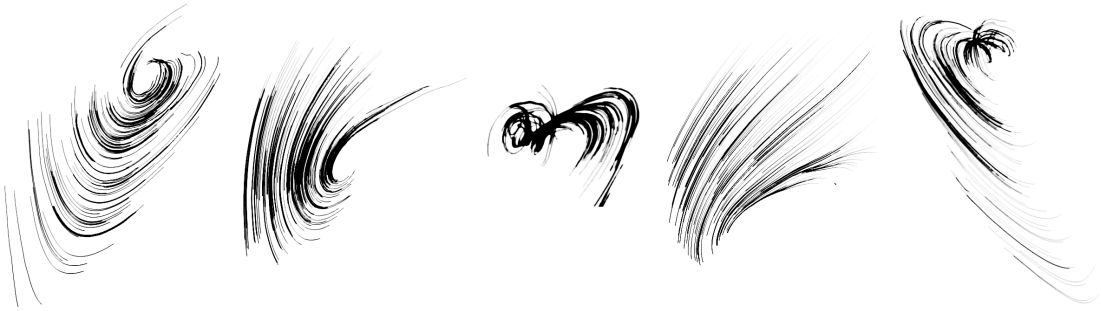

Figure 7.8.: Five sequences of varying complexity within the same physical device where generated. The result of the Computational Fluid Dynamics simulations is encoded by a trajectory of varying brightness: The darker a given point is the more time has elapsed since the beginning of the simulation.

often are as large as the images of the tracer particles themselves. Furthermore, regions of coherent motion become as numerous as the occlusions. On the other hand, simple smoothing regularizers [41] only work well if the distance between two moving particles is large enough to accommodate a smooth (yet physically meaningless) transition of the flow field. Hence the regularization strength depends on the distance of particles with differing motion: If the strength is too large, the motion is averaged too much - if the strength is too small, smoothing has little effect, causing errors due to intensity ambiguities on a very small scale. These effects can for example be observed in our synthetic sequences were as many particles moving right can be found moving into the opposite direction. Another problem is that these methods need to evaluate image derivatives: Usual images of tracer particles are very small (around three to five pixels in diameter). As soon as particles are crossing or moving close to each other but in different directions, isotropy-optimized derivative filters average over too large areas, resulting in erroneous estimates. Finally, the application of multiresolution approaches commonly employed for increased accuracy in global optical flow techniques does not always work for particle image velocimetry, because as soon as particle density and particle size are decreased too much, erroneous estimates of smaller pyramid levels (caused by disappearing or locally accumulating particle images) have a strong impact on final results.
For the evaluation of our approach, we examined the total-least-squares method of [95] that is able to accommodate the exponential brightness change of particles which is caused by their three-dimensional motion. We used the same generalized brightness change constrained equation of [13] and inserted it into the 2D linear combined local/global (CLG) method [12]. The latter is one of the numerous global techniques with spatial regularization. It is well known as a very accurate motion estimator in the computer vision community and, compared to other methods, yields acceptable results on our test sequences. Together with our extensions to 3D flows, it is able to capture smooth 3D flows reliably. Furthermore, this technique improves on the structure tensor method which is regularly employed in PIV literature (e.g. [95]). Thus, we applied a local and a global optical flow technique specifically adapted to the inherent brightness

changes to our test sequences. As initial guess for the trajectory ensemble method, we either use a very simple nearest-neighbor-based PTV algorithm or the same CLG flows. We only measured the accuracy at particle centers. These were obtained by the particle detection algorithm described in [3]. In all experiments, we compare the endpoint-error defined by [30] between the flow fields resulting from the flow estimation methods and the ground truth flow field. It is defined as the Euclidean distance between estimated flow vector and ground truth flow vector.
For the structure tensor in the CLG method we used the $3 \times 3 \times 3$ isotropy-optimized filters of Scharr [40] together with an integration scale of $\sigma = 4.0$ (using Gaussian filters, cropped and renormalized at 3σ). We also applied the optimized Sobel-filters described in [33].

Furthermore, we always trained our trajectory model on the CFD data (Fig. 7.8). Preliminary experiments showed that the quality of the outcome of our algorithm is robust with respect to different training data as long as the flow to be measured does not behave completely different from that in the training set. (Yet, investigations into the effect of different training data sets remain of interest.)

Crossing Particles We estimated the motion of two diagonally crossing particles (c.f. Fig. 7.5) in order to analyze the behavior of our algorithm in the case of occlusions. Additionally, we were able to estimate a lower accuracy limit for this type of date in the non-occluding part of the sequence. Hence, we can learn about the appropriateness of the energy term and the optimization technique.

Since the displacement of the ground truth is one pixel per frame in both directions, the maximum particle centers lie exactly on integer-locations. Hence, the nearest-neighbor-based PTV algorithm yields perfect results (zero endpoint-errors) whenever the particle detection routine localizes the two particles correctly. This is not the case for the particle being occluded in the middle of the sequence. Therefore, the flow estimates at these locations are wrong.

For the standard local least squares method with the Scharr-filters we achieved a constant endpoint-error of 0.083 pixels per non-occluding frame. This shows that a linearization of the problem (which is implied by the brightness change constrained equation of the structure tensor) prior to the actual optimization has some impact on the results and should be circumvented or specifically dealt with in our application if possible.

For the standard CLG method, we found almost constant endpoint-errors of 0.751 pixels in the non-occluding frames, depending on the type of derivative filter used. In the occluding-frames, the motion of the occluded particle was not estimated and the returned results were roughly those of the occluding particle image. This is due to the regularization that averages out contradicting directions of motion. As the error of a zero motion estimate would yield an error of $\sqrt{2}$, the CLG method essentially only halves this error. The problem of these high errors is caused due to the averaging effect of the spatial regularization even for very small regularization strength parameters.

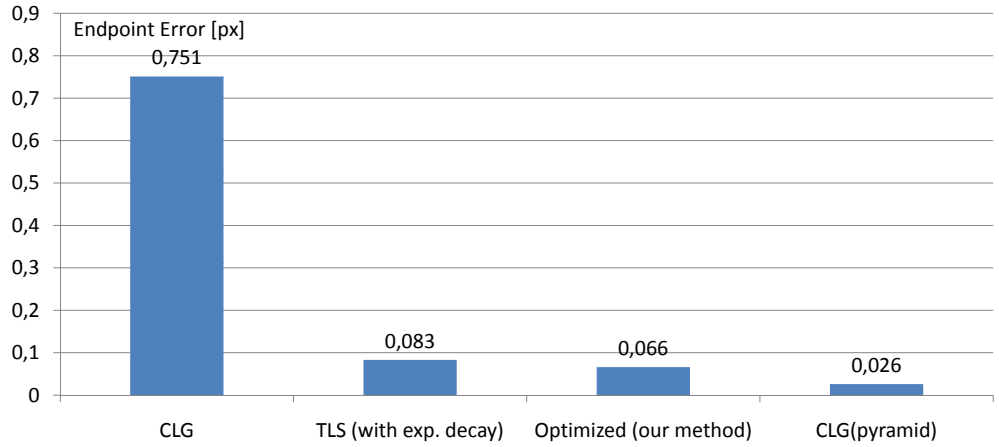

Figure 7.9.: The statistics of the endpoint-errors of the crossing particles sequence without occluding particles.

To yield better results with this method, we applied a five-level, warping-based pyramid scheme similar to that described in [14] together with the optimized Sobel-filters in space and forward differences in time. On the one hand this resulted in a time-consuming optimization process, but on the other hand the results reduced to around 0.01 to 0.07 pixels per frame (minimum and maximum error of all non-occluding frames; mean 0.026). The averaging effect is reduced due to the iterative warping which successively minimizes the energy.

Finally, we applied our trajectory ensemble optimization to this sequence. We used a ensemble-size of 2 and a trajectory length of 3, resulting in 25 trajectories of length 7. We used two eigenvectors of the trained model. These do not describe constant motions, as one might suspect. Instead they describe the two major motions (slight rotations and divergences/convergences) occurring in our training data. By using the perfect initial guess from the PTV estimate, we are able to estimate a lower accuracy limit for this data. The errors range from 0.04 to 0.07 pixels per frame (minimum and maximum error of all non-occluding frames; mean 0.066), showing that with a good initial guess, very good results can be achieved. (In the experiments shown on the sphere we will show that imperfect initial guesses can be improved by our method as well.)
For a zero initial guess (resulting in the mean trajectory ensemble of the training data) we obtained similar results. This shows that the defined energy is relatively well defined, so that the nonlinear optimization is not problematic even for rough initial guesses. The improved results seem to be due to the mainly temporal integration of investigated pixels: a larger number of meaningful locations is used for the optimization process without any noise-increasing image derivative estimation. As will be shown later this is especially useful for noise image data.

Rotating Sphere with Noise The rotating sphere sequence contains particles with various intensities between 10 and 255 and speeds with a mean of 0.94 pixels per frame and

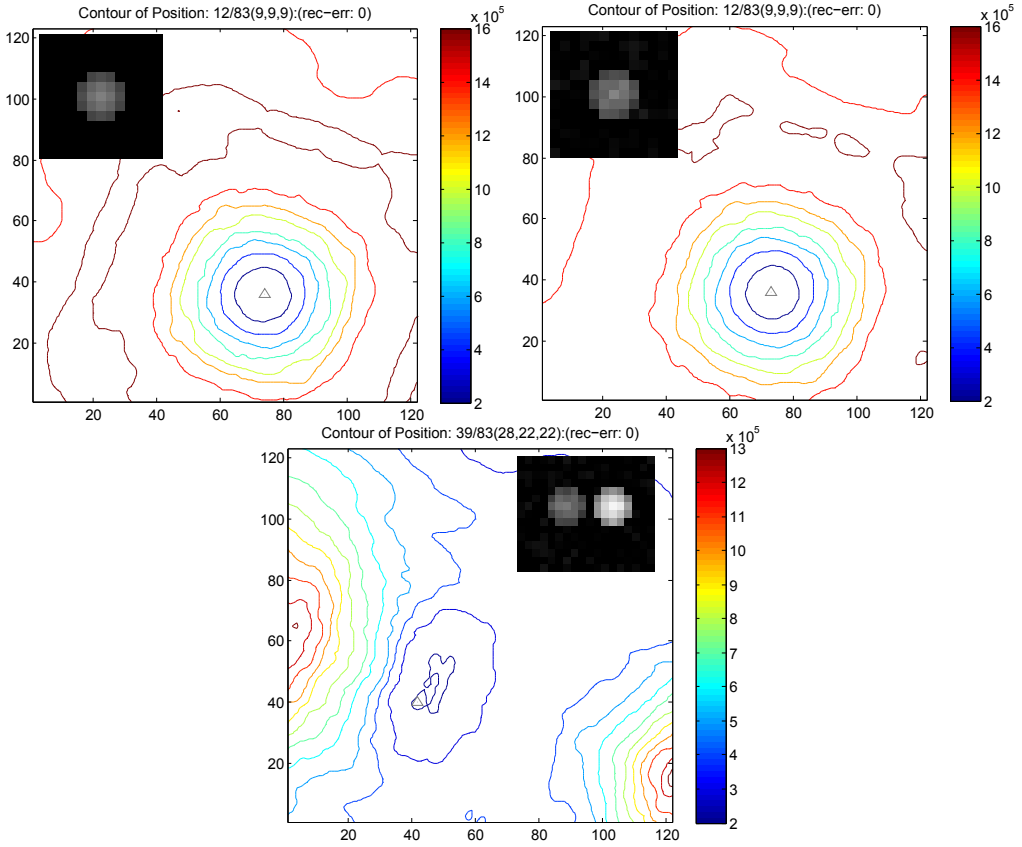

Figure 7.10.: Energies for the two-dimensional search space for the crossing particle sequence. Top: Example for the energy for non-occluding locations. Middle: The same as top with added noise of variance 128 intensities. Bottom: Example energy for occluding particle.

	Scharr-Filter	Sobel-Filter
Exp. BCCE	0.05 ± 0.06	0.2 ± 0.25
Const. BCCE	0.09 ± 0.1	0.22 ± 0.25

Table 7.1.: Average endpoint-errors for TLS variants applied to the artificial sphere sequence without noise.

a maximum speed of 2.5 pixels per frame both in dark and bright particle images. With a zero flow the average endpoint error of the detected particles is 0.95 ± 0.67. This value can be used as a reference on the relative improvement caused by the motion estimation. Using the CLG multiresolution approach described in Section 7.6.3 but modified with the BCCE for exponential decay and the optimized Sobel-filters yields an error of 0.54 ± 0.67. Without the multiresolution approach the error is slightly better (0.51 ± 0.5). This fact supports the thesis that too small particles or those being too close to each other cannot always be treated correctly by such an approach. For example, close particles are warped by the flow of the brighter, more dominant particle in the first step so that successive pyramid levels are based on an even worse initial guess than without using a pyramid.
Together with the BCCE that does not deal with exponential decay, the error is slightly higher (0.52 ± 0.54) showing that incorporating brightness models can principally improve the motion estimates.
To yield even better results with the CLG method we used the optimized Scharr derivative filters but the results deteriorated to an error of 0.7 ± 0.4. We believe that the reason is that the spatio-temporal neighborhood which does not follow the assumption of parameter constancy interferes with the global smoothness constraint. Even though all results are not very convincing we achieved the reported quality of results on the Yosemite sequence [12], indicating that the implementation is correct.
For comparison, we also applied the local method with a total least squares (TLS) solver with and without modified BCCE to the same data. We also used both derivative filters (optimized Sobel and Scharr). Results are shown in Table 7.1: As expected, the TLS result with exponential decay and the optimized Scharr-filters yields the best results (0.05 ± 0.06) which is significantly better than the CLG result. Without brightness model and with Sobel filters the results were still twice as accurate as those of the CLG algorithm (0.22 ± 0.25). Therefore, we believe that global methods have to be carefully evaluated and better understood when applied to particle image data in Fluid Dynamics.

In a next step we applied the nearest-neighbor PTV algorithm as initial guess for the trajectory ensemble approach. We also added Gaussian iid noise levels with variances of $1.0, 2.0, 4.0$ and 6.0 intensities. We used the same parameters for all noise levels: A single trajectory with a $5x5$ local neighborhood in the similarity term, two eigenvectors from the training date described above and a temporal support of five frames. For the small noise level of 1.0 the average results of the best TLS method are very similar to those of the initial guess (PTV) and our trajectory approach: For TLS we obtained an error of 0.1 ± 0.01, for PTV and our method yielded 0.14 ± 0.08. From this perspective our method performs worst. But, as all three methods are local, outliers can have a

significant impact on the mean value. Thus, we analyzed the whole error histogram for a more detailed evaluation. Results are shown in figure 7.2 and indicate that for about 80% of the most accurate flow estimates, our method yields the best resuls, followed by the PTV and then the TLS algorithm. Especially for higher noise levels this behaviour becomes more significant.
As expected, the method is more robust with respect to noise due to the larger temporal neighborhood, even though our approach does not model brightness variations explicitly. Please note that further increasing the spatial neighborhood would not yield more accurate results since the motion-information is only contained in the particle image and not its surroundings.

Real Data Finally we evaluated the accuracy of the trajectory ensemble estimator on the real test sequence of an ascending particle described in Section 7.6.3. The sequence has 530 frames. In this time, the particle moved 19 pixels. This value was estimated by comparing the particle image centers of the first and the last frame. As the particle moves horizontally, this results in a motion of 0.038 pixels per frame. An error of this estimate of one pixel would result in a difference smaller than 0.002. Assuming that the positioning system works as accurate as the manufacturer's specifications state, this is an sufficiently accurate ground truth for this sequence.

For comparison we have used the pyramid-based CLG method with modified BCCE, the TLS method with modified BCCE, and the nearest-neighbor PTV algorithm. To test how the ensemble-based method compares to the single trajectory based method described in [5], we used a single trajectory with a constant pixel-neighborhood of 49 pixels. Then, we applied the same method with the same trajectory length of 11 but with an ensemble of 49 trajectories. As shown in figure 7.7. The ensemble based method yields the best results of an average endpoint error over all 530 frames of $0,003 \pm 3,2e-6$. Again the effect of a larger neighborhood for averaging out noise along the trajectory of the particle yields superior results. This supports the results of the synthetic sequence described above.

7.7. Algorithm Requirements

In the previous section we have analyzed the accuracy of the newly proposed PTV technique. As discussed in Chapter 3, we will now also investigate other properties of the algorithm.

7.7.1. Estimatibility and Confidence

In Chapter 6, we have designed and carefully tested an estimatibility measure for this algorithm. Due to the nature of PTV image data the technique of detecting particle images in a first step arises naturally. The design of a confidence measure is more intricate and not discussed in this work. Usually, confidence measures are based either on the estimated flow, or both flow and image data. In our experiments we found that

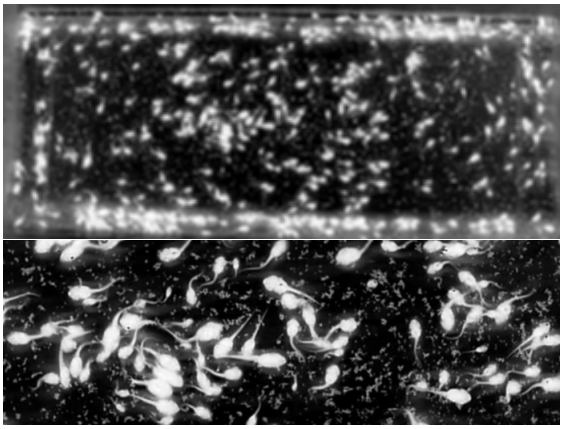

Figure 7.11.: Top: View of tadpoles swimming in a bassin, similar to tracer particles in a fluid. Bottom: Zoom to the center of the bassin.

the energy defined by our approach is relatively smooth for most cases. One idea for confidence measure would be the investigate the energy more closely. Nonetheless, the energy would have to be sampled which is time-consuming. Another choice would be to relate the flow vectors with respect to each other based on some model (as is done in [7]). Here, the problem is that the motion field is sparse and such a model cannot be learned easily. First steps towards this direction have been carried out in [139].
As there seems to be no intuitive way of defining a confidence measure, we leave this point open as future research.

7.7.2. Range of Applications

The concept of learning models of trajectory ensembles for a nonlinear energy minimization directly applied to the image sequence was specifically designed for the application in fluid dynamics. However, we believe that this idea is interesting for a broader range of problems in computer vision.

Tracking Currently, the energy minimization is carried out by directly measuring the intensity variations along the trajectory. In a more general way, a first step could be to compute a set of features from the image sequence. Then, the energy term could be applied to this feature space. As a result, more complex objects than particle images could be tracked. Furthermore, the trajectory ensemble not necessarily has to consist of flow vectors: It can reside in any parameter space such as the space of affine transformations. Such extensions would be similar to the tracking method of Isard and Blake [140] were a contour is parameterized by means of PCA. The difference would be that a set of neighboring parameters would be modeled jointly as an ensemble.

Swarm Modeling In biological computer vision often the behavior of animals like birds or insects is studied. One example is the tracking of tadpoles were hundreds of exemplars

are swimming inside a basin (Figure 7.11). By detecting the heads of the tadpoles with an appropriate algorithm, our method could be directly applied to this problem. Other applications could be the tracking of cells in microscopy, the application of trajectory ensembles as regularization in landmark based registration or similar fields of research.

Flow Representation One distinct feature of our approach is that we do not search for single flow vectors. The solution of the optimization problem at one location directly yields a local representation of a whole set of trajectories. These trajectories are temporally centered around the location so the flow can be studied forward and backward in time without the need for the flow inversions described in Section 7.5. Furthermore, these trajectory ensembles overlap so that confidence measures can be developed based on the consistency in overlapping regions.

7.7.3. Execution Speed

The most computationally demanding part of our algorithm is the evaluation of the energy with a given set of parameters. To compute the derivative of the energy numerically, for each dimension of the parameter vector, the energy is evaluated once. Actually, this step is very fast, as only a relatively small number of pixel locations is investigated. But this step also is carried out during each iteration of the optimization routine. To guarantee convergence, we applied a small step size and a large number of iterations. If the initial guess is already good, the algorithm converges fast and few steps are needed. The overall speed of our algorithm therefore depends on the two optimization parameters, the initial guess and the number of detected particles in the scene. On a standard machine (Intel 2GHz CPU, 2GB RAM) we observed computation times between a few milliseconds and a few seconds per trajectory. The results therefore strongly vary and cannot be easily predicted. Typically, on our test sequences and conservative iteration parameters, the algorithm needed a few hours per frame.
Improvements of the computation time can be achieved by improving the optimization routine: The fewer energy evaluations it needs and the faster it converges, the lower the computation time will become. A more refined optimization such as the iterated conjugate gradients method will thus improve the execution speed. Furthermore, as no global regularization term is employed, the each trajectory location can be optimized individually. The parallelization of the algorithm can therefore be carried out by simply distributing all particle locations over all available CPU cores.

7.7.4. Modularity

As discussed above, the algorithm consists of an estimatibility measure, a learned motion model, a similarity measure and a nonlinear optimization strategy. Furthermore, the model an be learned by various techniques. We implemented these modules in Charon to allow for experiments with other techniques. Each of these modules can be improved separately. Therefore, our algorithm is as modular as those algorithms discussed in Chapter 4,

7.7.5. Accessibility of Implementations

On the one hand our algorithm exhibits a few parameters which are easy to understand. On the other hand, the technique is based on learning a model. Although the same learned model was appropriate for all our experiments carried out in the previous section, the selection of training data might be more difficult in other fields of application. Therefore, we believe that more experiments with respect to those parameters related to learning should be carried out in future research in order to better understand the influence of this part of the algorithm.
To facilitate future research based on this algorithm we offer the code as open source to the research community within the framework of Charon.

7.8. Conclusion

In this chapter we have presented a new motion model for optical flow estimation based on trajectory ensembles. We implemented an optical flow estimator based on the proposed model by means of a standard similarity measure and optimization technique, showing promising results on relevant test sequences in the field of Fluid Dynamics.
Any energy functional based on trajectory ensembles becomes nonlinear. Hence, more sophisticated optimization techniques such as the iterated conjugate gradients method should be examined in future research. Furthermore, the energy term currently applied does not model occlusions explicitly. We believe that a robust similarity measures would help dealing with occluding particle images.
The low-dimensional representation of trajectory ensembles is advantageous due to several reasons:

- It allows for spatially unconnected temporal neighborhoods which are naturally and easily interpretable.
- The mainly temporal structures of fluid flows are explicitly exploited.
- The model can be adapted to special kinds of flows by incorporating prior knowledge without the need for exact physical models.
- PCA effectively reduces the number of parameters of flow field information. This greatly alleviates the optimization problem by significantly decreasing the dimensionality of the search space.
- The use of ensembles instead of single trajectories notably reduces the risk of local minima in the optimization problem.
- No image derivatives are needed, avoiding associated problems (c.f. Sec. 7.6.3).

Experiments indicate that the application of our model to Fluid Dynamics data yields good results, even though a standard energy term and optimization technique were used. To this end, we applied our method to a realistic, synthetic image sequence showing that

the exploited temporal information helps in obtaining more accurate results. We also verified this with a simple real sequence yielding results 15 times more accurate than the other evaluated methods.

More refined approaches will lead to a further increase of accuracy. We, therefore, propose to put more effort in the research of mainly temporal regularization methods for motion estimators.

Finally, we noted that the fields of Computer Vision and Fluid Dynamics research share a highly correlated history of image processing techniques. We believe that more interdisciplinary research in motion estimation might be fruitful for both areas. Therefore, we offer our source code within the framework of Charon to facilitate future research in both areas.

8. Conclusion and Future Research

> Science is always wrong. It never solves a problem without creating ten more.
>
> *(George Bernard Shaw)*

In the first part of this work we have carefully motivated the concept of modular optical flow estimation. Based on the theoretical considerations we have implemented a software library named Charon, which eases the implementation of new algorithms as well as experimenting with existing techniques.
In the second part we used this implementation and the concept of modularity to derive a new algorithm for particle tracking velocimetry which outperforms existing methods with respect to accuracy.
We will now give a detailed conclusion of our work and discuss future research topics.

8.1. Modular Optical Flow

The range of possible applications for optical flow techniques is broad. Furthermore, the number of developed algorithms addressing this task is huge. In chapters 2 and 5, we have shown that a general solution of the flow problem is not only very unlikely to be found; it can even be counterproductive in some circumstances. Since no existing method solves all problems, one main challenge is to choose the most appropriate method for a given type of image sequences.
To be able to choose, the characteristics of the algorithms have to be known in great detail. Therefore, in Chapter 3, we suggested a basic set of requirements which we believe should be used to describe an optical flow algorithm. The requirements are:

- Accuracy
- Existence of estimatibility and confidence measures
- Range of applications
- Algorithm complexity, computation speeds and possibilities for their improvement
- Modularity
- Accessibility of source code and ease of use

We argued that all of these points have not sufficiently been addressed in prior research and that more attention should be paid to the thorough analysis of existing methods. To contribute to these aims we start with the last two points: modularity and accessibility. Optical flow algorithms become more and more complex systems of combined techniques. Hence, we identified the most prominent modules of these techniques in Chapter 4, which are:

- Brightness Variation Models
- Neighborhood Selection Methods
- Motion Models
- Regularization Terms
- Robustness Functions
- Image Derivative Estimation
- Image Interpolation
- Multiresolution Techniques
- Optimization Methods

More modules are likely to be found with the advance of research in this field, such as training data selection and training regime. Yet, this first classification already shows that a huge number of optical flow algorithms can be derived simply by combining any of these modules. To contribute to the accessibility, we have developed Charon, a software library that implements many existing techniques in a modular way. Charon is designed with both researches and engineers in mind, allowing for easy experimentation, parallelization and automatization of common workflows such as the evaluation of motion estimates based on ground truth (cf. Chapter 5).

Designing Charon with modularity and accessibility in mind not only helps in experimenting with existing optical flow techniques. Further advantages are:

- The possibility of designing new algorithms by simply plugging together existing modules.
- The implementation of new modules can easily be carried out by implementing the appropriate interfaces.
- Algorithm requirements can be analyzed, such as the influence of free parameters.
- Methods can be compared by only modifying a single module whereas the rest remains exactly the same.
- A database of existing algorithms becomes available in a single framework.

We have shown exemplary results computed with Charon. On the one hand, this shows the versatility of the library; on the other hand we used these experiments to practically substantiate the hypothesis that general-purpose optical flow algorithms are not feasible to design: We have shown that derivative filters can significantly improve the accuracy on small flows with noise, but not in cases of larger flows. Furthermore, we have experimented with pyramids and found that (in our examples) more accurate results can be achieved without downsampling. Finally, we have evaluated the theoretical accuracy limits on three given sequences with ground truth and found that the optimal parameters yield significantly worse results if applied to other sequences. All these facts show that none of the tested algorithms is general (even in simple cases of small flows) and that a high amount of expert knowledge is necessary to optimally choose parameters for a given image sequence.

8.2. Particle Tracking Velocimetry

In the second part of this work we applied the concepts described above to the field of fluid dynamics. From this point of view we developed an estimatibility measure for PTV image data (which is in fact a particle detection algorithm) and a new motion model based on trajectory ensembles. The application to PTV data shows the versatility of Charon and describes a case were standard optical flow methods often fail to achieve good results.

The particle detection method is based on the stability of the center of gravity (first momentum) of connected components obtained at various threshold levels. A few intuitive parameters such as the maximum intensity of all particle images and the minimum area are needed. To analyze and validate the method, we developed an evaluation framework which comprises particle simulation and accuracy evaluation code.
The results are promising: Our new technique is highly robust with respect to Gaussian noise and can even accommodate spatially varying mean values. A very low false positive rate helps subsequent modules to deal with false negatives.
To contribute to the aim of accessibility, we offer our source code to the research community.

To measure not only the locations of particle images, but also their motion, we implemented a new PTV algorithm around the new motion model. To validate the idea of trajectory ensembles we applied only simple modules: For learning, we applied PCA to training data generated by CFD simulations. As similarity measure we used the squared difference of intensities along the trajectories. For optimization we applied a gradient descent scheme in model parameter space.
Although these other models are simple, we were able to show promising results with accuracy gains of up to 15 times the accuracy of standard methods. These results are based on both realistic synthetic sequences containing noise and intensity variations and real sequences generated by a newly developed validation technique.
Beside the practical advantages we identified a number of theoretical advantages of the representation of motion via trajectory ensembles. These are for example the in-

tuitiveness of the parameters, the exploitation of mainly temporal structures and the adaptability to various applications.

8.3. Future Research

As we addressed some very general problems in optical flow estimation, the possibilities for future research are vast. We will discuss a few starting points for requirements engineering and will then discuss future research topics more closely related to Charon and PTV.

8.3.1. Requirements Engineering

Although the average angular error (and its standard deviation) has regularly been used as error measure, the proposal of the endpoint error seems to be gladly embraced by the research community. This indicates that the best choice for a performance measure is still not fully accepted by everyone. We believe that one main problem is that usually only average and standard deviation are shown. A first step to tackle this problem is the Middlebury website which visualizes the error as image and furthermore offers more than one error measure. Thus, one important future research topic is how to evaluate the accuracy of an optical flow field in a way that is accepted by the whole optical flow community. Additionally, it is unclear if or when one can be satisfied with the accuracy of a method: As most presented methods today are only slightly more accurate on specific image data like the Yosemite sequence, it largely remains unclear whether there actually is a need for new algorithms. Obtaining knowledge about theoretical and fundamental limits of accuracy on a given sequence would therefore be another future research topic. Directly related is the generation of real ground truth data. The accuracy of the ground truth in existing real data is not known. In [30] the accuracy is not even discussed. As the technique (roughly) described there is highly similar to PIV techniques, it is likely that it has similar error rates. However, even if ground truth data can be generated with enough accuracy, it remains an open question which sequence is most general for a given field of application. Current sequences look very similar as shown in Chapter 2. It is unlikely that these sequences can be used as base for a performance measurement for all computer vision applications.
Furthermore, it is obvious that motion cannot been estimated everywhere. Dedicated research into estimatibility and confidence measures has just begun [7, 26, 8].

8.3.2. Choices of Appropriate Methods

As indicated in the exemplary results shown in Chapter 5, research into the automatic choice of the parameters (such as regularization strength) based on given image data is urgently needed for real-world applications with unknown ground truth. It would be even more interesting to find algorithms that are able to automatically choose the correct modules based on the image sequence.
First steps toward this direction could be the analysis of existing image data and the

optical flow results of existing methods. It might be possible to draw features from such data on which a learning-based approach could find the best answers.

8.3.3. Modular Optical Flow

The range of applications of most methods is unknown today. To be able to find further applications for existing methods, accessible implementations need to be more widely available. As we hope that more and more methods will be implemented in Charon, this might be a first step into this direction.
Up to now, the optimization method implies the use of specific classes for modeling. Hence, another goal for the development of Charon is to further abstract the existing modules so that they are fully independent from the optimization technique used.
Finally, we plan to improve on the accessibility by providing a thorough documentation integration through Tuchulcha and an improved visualization tool for flow estimates based on Argos.

8.3.4. Particle Tracking Velocimetry

We believe that future research steps into PTV should more carefully take into account the modularity of existing PTV algorithms. Our particle detection algorithm could be further improved by including an additional step that either identifies occluding particle images or even separates the found particles. The first idea might be not so difficult, but separating the particles probably involves the fitting of several particle image models to the image data. This could for example be carried out by expectation maximization.
For the actual motion estimation algorithms a number of possible extensions exist. One idea would be to create a model-based initial guess: Instead of choosing the nearest neighbor in the initial guess standard PTV approach we could first compute all possible particle paths up to a given maximum displacement over a fixed number of frames. The resulting trajectories could each be projected into our learned model space and then be backprojected into trajectory space. By comparing the difference between the original and the backprojected trajectory we can measure the "probability" that the trajectory can be explained by our model. Finally, we would choose the best fitting from all possible trajectories.
This idea is nice, because then the initial guess is also computed based on the trained model and might already be of sufficient accuracy for many applications. To further improve the results one could then apply the method described here, which searches for a more stable fit of the given trajectory based on the unsegmented image data.
Another future research topic is to improve the simple model of brightness variation and the optimization scheme. As the observed energies are already locally rather convex, an iterated conjugate gradient descent could be one way to improve the convergence rate and therefore the overall speed of the algorithm.
The brightness variation model is currently not robust with respect to particle occlusions. As our particle detection technique already removes occluding particles so that these locations are simply skipped and marked as not estimatible, this poses no great

problem to the current implementation, but it would be desirable to deal with such cases as well.
Finally, we believe that our method can be very easily be extended to measure the intensity variations along the found trajectories in order to reconstruct the three-dimensional motion based on a brightness variation model. Our method proved to be very robust to intensity variations so we did not explicitly model them. Instead, we could now employ a second step which models these variations separately: By learning from the same CFD simulations we can learn a one-dimensional model on how the depth of particles can vary along their trajectory. Then, we simply have to fit this model to the intensities measured along our trajectories. This can be done linearly for example with the total least squares method.

Bibliography

[1] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Real-time optic flow computation with variational methods. *IEEE Transactions in Image Processing*, 14(5):608–615, 2005.

[2] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision, Proceedings*, pages 25–36, 2004.

[3] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv- l 1 optical flow. In *29th Annual Symposium of the German Association for Pattern Recognition (DAGM'07)*, pages 214–223, 2007.

[4] Y. Mizukami and K. Tadamura. Optical flow computation on compute unified device architecture. In *14th International Conference on Image Analysis and Processing*, pages 179–184, 2007.

[5] M.J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, January 1996.

[6] Gunnar Farnebäck. *Spatial Domain Methods for Orientation and Velocity Estimation*. PhD thesis, Linköping University, Sweden, 1999.

[7] G. Farnebäck. Fast and accurate motion estimation using orientation tensors and parametric motion models. In *International Conference on Pattern Recognition, Proceedings*, volume 1, pages 135–139, Barcelona, Spain, September 2000.

[8] B. Galvin, B. McCane, K. Novins, D. Mason, and S. Mills. Recovering motion fields: An analysis of eight optical flow algorithms. In *Proceedings of the 1998 British Machine Vision Conference*, 1998.

[9] J. L. Barron, D. J. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[10] A. Mitiche and P. Bouthemy. Computation and analysis of image motion: A synopsis of current problems and methods. *International Journal of Computer Vision*, 19(1):29–55, 1996.

[11] Neil A. Thacker, Adrian F. Clark, John L. Barron, J. Ross Beveridge, Patrick Courtneye, William R. Crum, Visvanathan Ramesh, and Christine Clark. Performance characterization in computer vision: A guide to best practices. *Computer Vision and Image Understanding*, 109(3):305–334, March 2008.

[12] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, 2005.

[13] H. Haussecker and D. Fleet. Computing optical flow with physical models of brightness variation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(6):661–673, 2001.

[14] N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert. Highly accurate optic flow computation with theoretically justified warping. *International Journal of Computer Vision*, 67(2):141–158, 2006.

[15] R. Battiti, E. Amaldi, and C. Koch. Computing optical flow across multiple scales: An adaptive coarse-to-fine strategy. *International Journal of Computer Vision*, 6(2):133–145, June 1991.

[16] Luis Alvarez, Joachim Weickert, and Javier Sánchez. A scale-space approach to nonlocal optical flow calculations. In *In ScaleSpace 1999*, pages 235–246. Springer, 1999.

[17] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision (darpa). In *Proceedings of the 1981 DARPA Image Understanding Workshop*, pages 121–130, 1981.

[18] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47:7–42, 2001.

[19] Joseph V. Hajnal, David J. Hawkes, and Derek L. G. Hill. *Medical image registration.* CRC Press, 2001.

[20] G. Hermosillo, C. Chefd'hotel, and O.Faugeras. Variational methods for multimodal image matching. *International Journal of Computer Vision*, 50(3):329–343, 2002.

[21] J. P. Lewis. Fast normalized cross-correlation. In *Vision Interface*, pages 120–123. Canadian Image Processing and Pattern Recognition Society, 1995.

[22] Samuel Blackman and Robert Popoli. *Design and Analysis of Modern Tracking Systems.* Artech House, 1999.

[23] M. Raffel, C. E. Willert, S.T. Wereley, and J. Kompenhans. *Particle image velocimetry.* Springer, 2007.

[24] Cameron Tropea, Alexander L. Yarin, and John F. Foss. *Springer Handbook of Experimental Fluid Mechanics.* Springer, 2007.

[25] Nicholas T. Ouellette, Haitao Xu, and Eberhard Bodenschatz. A quantitative study of three-dimensional lagrangian particle tracking algorithms. *Experiments in Fluids*, 40(2):301–313, 2006.

[26] Claudia Kondermann, Rudolf Mester, and Christoph Garbe. A statistical confidence measure for optical flows. In *Computer Vision - ECCV 2008*, pages 290–301, October 2008.

[27] D. Heeger. Model for the extraction of image flow. *Journal of the Optical Society of America*, 4(8):1455–1471, 1987.

[28] B. McCane, K. Novins, D. Crannitch, and B. Galvin. On benchmarking optical flow. `http://of-eval.sourceforge.net/`, 2001.

[29] M. Otte and H. Nagel. Optical flow estimation: advances and comparisons. In *Proceedings of the European Conference on Computer Vision*, pages 51–60, 1994.

[30] S. Baker, S. Roth, D. Scharstein, M. Black, J. Lewis, and R. Szeliski. A database and evaluation methodology for optical flow. In *Proceedings of the International Conference on Computer Vision*, pages 1–8, 2007.

[31] D. J. Fleet and A. Jepson. Computation of component image velocity from local phase information. *International Journal on Computer Vision*, 5(1):77–104, 1990.

[32] H. Haussecker and H. Spies. Motion. In B. Jähne, H. Haussecker, and P. Geissler, editors, *Handbook of Computer Vision and Applications*, volume 2, chapter 13. Academic Press, 1999.

[33] B. Jähne, H. Haussecker, and P. (eds.) Geißler. *Handbook of Computer Vision and Application. Volume 2*. Academic Press, 1999.

[34] C. Kondermann, D. Kondermann, B. Jähne, and C. Garbe. An adaptive confidence measure for optical flows based on linear subspace projections. In *Pattern Recognition*, volume 4713 of *LNCS*, pages 132–141. Springer, 2007.

[35] C. Kondermann, D. Kondermann, and C. Garbe. Postprocessing of optical flows via surface measures and motion inpainting. In *Pattern Recognition*, volume 5096 of *LNCS*, pages 355–365. Springer, 2008.

[36] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–319, 1989.

[37] J. Bigün, G.H. Granlund, and J.Wiklund. Multidimensional orientation estimation with applications to texture analysis and optical flow. *IEEE Journal of Pattern Analysis and Machine Intelligence*, 13(8):775–790, 1991.

[38] Andres Bruhn, Joachim Weickert, Christian Feddern, Timo Kohlberger, and Christoph Schnörr. *Real-Time Optic Flow Computation with Variational Methods*, volume 2756/2003, pages 222–229. Springer Berlin Heidelberg, 2003.

[39] Tomer Amiaz, Eyal Lubetzky, and Nahum Kiryati. Coarse to over-fine optical flow estimation. *Pattern Recogn.*, 40(9):2496–2503, 2007.

[40] H. Scharr. Optimal filters for extended optical flow. In *Complex Motion, Lecture Notes in Computer Science*, volume 3417. Springer, 2004.

[41] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–204, 1981.

[42] H. H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(5):565–593, 1986.

[43] J. Weickert and C. Schnörr. A theoretical framework for convex regularizers in PDE-based computation of image motion. *International Journal of Computer Vision*, 45(3):245–264, 2001.

[44] D. Cremers and S. Soatto. Motion competition: A variational approach to piecewise parametric motion segmentation. *International Journal of Computer Vision*, 62(3):249–265, May 2005.

[45] N. Ohta. Optical flow detection by color images. In *Proceedings of IEEE International Conference on Image Processing*, pages 801–805. IEEE Computer Society, 19895.

[46] P. Golland and A. M. Bruckstein. Motion from color. *Comput. Vis. Image Underst.*, 68(3):346–362, 1997.

[47] Robert J. Andrews and Brian C. Lovell. Color optical flow. In *In Eds. Proceedings Workshop on Digital Image Computing*, pages 135–139, 2003.

[48] Michael A. Gennert and Shahriar Negahdaripour. Relaxing the brightness constancy assumption in computing optical flow. Technical report, Cambridge, MA, USA, 1987.

[49] C.S. Garbe. Fluid flow estimation through integration of physical flow configurations. In *29th Annual Symposium of the German Association for Pattern Recognition (DAGM'07)*, pages 92–101, 2007.

[50] Daniel Toth, Til Aach, and Volker Metzler. Illumination-invariant change detection. In *SSIAI '00: Proceedings of the 4th IEEE Southwest Symposium on Image Analysis and Interpretation*, page 3, Washington, DC, USA, 2000. IEEE Computer Society.

[51] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. 1984, pyramid methods in image processing. *RCA Engineer*, 29(6):33–41, 1984.

[52] S.H. Lai and B.C. Vemuri. Robust and efficient algorithms for optical flow computation. In *Computer Vision, 1995. Proceedings., International Symposium on*, pages 455–460, November 1995.

[53] M.J. Black and P. Anandan. A framework for the robust estimation of optical flow. In *Proceedings of the Fourth International Conference on Computer Vision (ICCV'93)*, pages 231–236, 1993.

[54] A. Bab-Hadiashar and D. Suter. Robust optic flow computation. *International Journal of Computer Vision*, 29(1):59–77, 1998.

[55] S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis*. U.S. Society for Industrial and Applied Mathematics, 1992.

[56] Tal Nir, Alfred M. Bruckstein, and Ron Kimmel. Over-parameterized variational optical flow. *International Journal of Computer Vision, Online First*, June 2007.

[57] James R. Bergen, P. Anandan, Keith J. Hanna, and Rajesh Hingorani. Hierarchical model-based motion estimation. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 237–252, London, UK, 1992. Springer-Verlag.

[58] G. Farnebäck. Very high accuracy velocity estimation using orientation tensors, parametric motion, and simultaneous segmentation of the motion field. In *International Conference on Computer Vision, Proceedings*, volume I, pages 171–177, Vancouver, Canada, July 2001.

[59] C. Stiller and J. Konrad. Estimating motion in image sequences. *IEEE Signal Processing Magazine*, 16(4):70–91, 1999.

[60] S. X. Ju, M. J. Black, and A. D. Jepson. Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, 1996.

[61] Masahiko Shizawa and Kenji Mase. Simultaneous multiple optical flow estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 274–278, 1990.

[62] C. Mota, I. Stuke, and E. Barth. Analytical solutions for multiple motions. In *Proceedings of the International Conference on Image Processing ICIP*, 2001.

[63] Erhard Barth, Ingo Stuke, Til Aach, and Cicero Mota. Spatio-temporal motion estimation for transparency and occlusions. In *Proceedings of the International Conference on Image Processing (ICIP)*, volume 3, pages 69–72, 2003.

[64] L. Alvarez, R. Deriche, and J. Papadopoulo, T. Sanchez. Symmetrical dense optical flow estimation with occlusions detection. In *European Conference on Computer Vision (ECCV)*, pages 721–735, 2002.

[65] S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis*. Society for Industrial and Applied Mathematics, 1991.

[66] S. Roth and M.J. Black. On the spatial statistics of optical flow. In *International Conference on Computer Vision, Proceedings*, volume 1, pages 42–49, 2005.

[67] Deqing Sun, Stefan Roth, J. P. Lewis, and Michael J. Black. Learning optical flow. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 83–97, Berlin, Heidelberg, 2008. Springer-Verlag.

[68] Philippe Thévenaz, Thierry Blu, and Michael Unser. Image interpolation and resampling. pages 393–420, 2000.

[69] Kai Krajsek and Rudolf Mester. Signal and noise adapted filters for differential motion estimation. In Walter G. Kropatsch, Robert Sablatnig, and Allan Hanbury, editors, *DAGM-Symposium*, volume 3663 of *Lecture Notes in Computer Science*, pages 476–484. Springer, 2005.

[70] H. Farid and E. P. Simoncelli. Differentiation of discrete multidimensional signals. *IEEE Transactions on Image Processing*, 13(4):496–508, 2004.

[71] M. Unser, A. Aldroubi, and M. Eden. B-Spline signal processing: Part I—Theory. *IEEE Transactions on Signal Processing*, 41(2):821–833, February 1993. IEEE Signal Processing Society's 1995 best paper award.

[72] M. Unser, A. Aldroubi, and M. Eden. B-Spline signal processing: Part II—Efficient design and applications. *IEEE Transactions on Signal Processing*, 41(2):834–848, February 1993.

[73] Luis Alvarez, Joachim Weickert, and Javier Sánchez. Reliable estimation of dense optical flow fields with large displacements. *International Journal of Computer Vision*, 39(1):41–56, 2000.

[74] K. Okamoto, S. Nishio, T. Saga, and T. Kobayashi. Standard images for particle-image velocimetry. *Measurement Science and Technology*, 11:685–691, 2000.

[75] Y. G. Guezennec and N. Kiritisis. Statistical investigation of errors in particle image velocimetry. *Experiments in Fluids*, 10(2-3):138–146, 1990.

[76] H. G. Maas, A Gruen, and D. Papantoniou. Particle tracking velocimetry in three-dimensional flows-part 1. photogrammetric determination of particle coordinates. *Experiments in Fluids*, 15(2):133–146, 1993.

[77] M. Marxen, P. E. Sullivan, M. R. Loewen, and B. Jähne. Comparison of gaussian particle center estimators and the achievable measurement density for particle tracking velocimetry. *Experiments in Fluids*, 29(2):145–153, 2000.

[78] K. Ohmi and H. Li. Particle-tracking velocimetry with new algorithms. *Measurement Science and Technology*, 11, 2000.

[79] A. V. Mikheev and V. M. Zubtsov. Enhanced particle-tracking velocimetry (EPTV) with a combined two-component pair-matching algorithm. *Measurement Science and Technology*, 19, 2008.

[80] K. Takehara and T. Etho. A study on particle identification in PTV particle mask correlation method. *Journal of Visualization*, 1(3):313–323, 1998.

[81] K. Ohmi and H. Li. Particle tracking velocimetry by combined use of the moravec operator and the relaxation algorithm. In *Proc. 2nd Pacific Symp. on Flow Visualization*, 1999.

[82] F. Hering, M. Merle, D. Wierzimok, and B. Jähne. A robust technique for tracking particles over long image sequences. In *Proc. of ISPRS Intercommission Workshop "From Pixels to Sequences", Intl Arch. of Photog. and Rem. Sens.*, 1995.

[83] F. Hering, C. Leue, D. Wierzimok, and B. Jähne. Particle tracking velocimetry beneath water waves. part i: visualization and tracking algorithms. *Experiments in Fluids*, 23(6):472–482, 1997.

[84] M. Jehle. Spatio-temporal analysis of flows close to free water surfaces, Dissertation. University of Heidelberg, 2006.

[85] F. Carosone, A. Cenedese, and G. Querzoli. Recognition of partially overlapped images using the Kohonen neural network. *Experiments in Fluids*, 19(4):225–232, 1995.

[86] Kenji Suzuki, Isao Horiba, and Noboru Sugie. Linear-time connected-component labeling based on sequential local operations. *Computer Vision and Image Understanding*, 89(1):1–23, 2003.

[87] Holger Nobach, Nils Damaschke, and Cam Tropea. High-precision sub-pixel interpolation in particle image velocimetry image processing. *Experiments in Fluids*, 39(2):299–304, 2005.

[88] Dongning Li, Yuanhui Zhang, Yigang Sun, and Wei Ya. A multi-frame particle tracking algorithm robust against input noise. *Measurement Science and Technology*, 19, 2008.

[89] M. Grigioni, C. Daniele, G D'Avenio, U. Morbiducci, C. Del Gaudio, M. Abbate, and D. Di Meo. Innovative technologies for the assessment of cardiovascular medical devices: state-of-the-art techniques for artificial heart valve testing. *Expert Rev Med Devices*, 1(1):81–93, September 2004.

[90] Holger Nobach and Eberhard Bodenschatz. Limitations of accuracy in PIV due to individual variations of particle image intensities. *Experiments in Fluids*, 2009.

[91] T. Corpetti, D. Heitz, G. Arroyo, E. Mémin, and A. Santa-Cruz. Fluid experimental flow estimation based on an optical-flow scheme. *Experiments in Fluids*, 40(1):80–97, January 2006.

[92] Paul Ruhnau, Annette Stahl, and Christoph Schnörr. Variational estimation of experimental fluid flows with physics-based spatio-temporal regularization. *Measurement Science and Technology*, 18(3):755–763, March 2007.

[93] Luis Alvarez, Carlos Castano, Miguel Garcia, Karl Krissian, Luis Mazorra, Agustin Salgado, and Javier Sanchez. A variational approach for 3d motion estimation of incompressible PIV flows. In *Scale Space and Variational Methods in Computer Vision*, pages 837–847. Springer Berlin/Heidelberg, 2007.

[94] Jing Yuan, Christoph Schnörr, and Gabriele Steidl. Simultaneous higher-order optical flow estimation and decomposition. *Journal on Scientific Computing*, 29(6):2283–2304, October 2007.

[95] M. Jehle and B. Jähne. A novel method for 3d3c analysis of flows close to free water surfaces. *Experiments in Fluids.*, January 2008.

[96] M.J. Black and P. Anandan. Robust dynamic motion estimation over time. In *IEEE Proc. Computer Vision and Pattern Recognition, CVPR'91*, pages 296–302, 1991.

[97] M.J. Black. Explaining optical flow events with parameterized spatio-temporal models. In *IEEE Proc. Computer Vision and Pattern Recognition, CVPR'99*, pages 326–332, 1999.

[98] H.-H. Nagel. Extending the ´oriented smoothness constraint´ into the temporal domain and the estimation of derivatives of optical flow. In *Computer Vision - ECCV ´90, Lecture Notes in Computer Science*, volume 427, pages 139–148. Springer, Berlin, 1990.

[99] J. Weickert and C. Schnörr. Variational optic flow computation with a spatio-temporal smoothness constraint. *Journal of Mathematical Imaging and Vision*, 14(3):245–255, May 2001.

[100] A. Salgado and J. Sanchez. A temporal regularizer for large optical flow estimation. In *Image Processing, 2006 IEEE International Conference on*, pages 1233–1236, October 2006.

[101] R. Agarwal and J. Sklansky. Estimating optical flow from clustered trajectories in velocity-time. In *International Conference on Pattern Recognition, Proceedings*, pages 215–219, August 1992.

[102] K. Chaudhury and R. Mehrotra. A trajectory-based computational model for optical flow estimation. *Transactions on Robotics and Automation*, 11(5):733–741, October 1995.

[103] J.T. Todd. Visual information about rigid and nonrigid motion: a geometric analysis. *Experimental Psychology, Human Perception Performance*, 8:238–252, 1982.

[104] V.S. Ramachandran and S.M. Antis. Extrapolation of motion path in human visual perception. *Vision Research*, 23:83–85, 1983.

[105] P. Verghese, S. N. J. Watamaniuk, S. P. McKee, and N. M. Grzywacz. Local motion detectors cannot account for the detectability of an extended trajectory in noise. *Vision Research*, 39:19–30, 1999.

[106] D. Gibson and M. Spann. Robust optical flow estimation based on a sparse motion trajectory set. *Transactions on Image Processing*, 12(4):431–445, April 2003.

[107] A. G. Bors and A. Pitas. Prediction and tracking of moving objects in image sequences. *Transactions on Image Processing*, 9(8):1441–1445, August 2000.

[108] M.J. Black, Y. Yacoob, A. Jepson, and D. Fleet. Learning parameterized models of image motion. In *Conference on Computer Vision and Pattern Recognition, Proceedings*, 1997.

[109] David J. Fleet, Michael J. Black, Yaser Yacoob, and Allan D. Jepson. Design and use of linear models for image motion analysis. *International Journal of Computer Vision*, 36(3):171–193, February 2000.

[110] Y. Yacoob and M.J. Black. Parameterized modeling and recognition of activities. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 120–127, January 1998.

[111] Yaser Yacoob and Larry S. Davis. Learned models for estimation of rigid and articulated human motion from stationary or moving camera. *International Journal of Computer Vision*, 36(1):5–30, January 2000.

[112] Ichiro Kimura and Toshi Takamori. Image processing of flow around a circular cylinder by using correlation technique. In *Flow visualization IV; Proceedings of the Fourth International Symposium, Paris, France*, pages 221–226, August 1986.

[113] C. E. Willert and M. Gharib. Digital particle image velocimetry. *Experiments in Fluids*, 10(4):181–193.

[114] Christian Willert. Stereoscopic digital particle image velocimetry for application in wind tunnel flows. *Measurement Science and Technology*, 8(12):1465–1479, 1997.

[115] K. Jambunathan, X. Y. Ju, B. N. Dobbins, and S. Ashforth-Frost. An improved cross correlation technique for particle image velocimetry. *Measurement Science and Technology*, 6:507–514.

[116] H. T. Huang, H. E. Fiedler, and J. J. Wang. Limitation and improvement of PIV. *Experiments in Fluids*, 15(4):263–273, September 1993.

[117] S. T. Wereley and C. D. Meinhart. Second-order accurate particle image velocimetry. *Experiments in Fluids*, 31(3):258–268, September 2001.

[118] M. Stanislas, K. Okamoto, and C. Kähler. Main results of the first international PIV challenge. *Measurement Science and Technology*, 14(10):R63–R89, 2003.

[119] M. Stanislas, Okamoto K., C. Kaehler, and J. Westerveel. Main results of the second international PIV challenge. *Experiments in Fluids*, 39:170–191, 2005.

[120] M. Stanislas, K. Okamoto, C. Kähler, J. Westerweel, and F. Scarano. Main results of the third international PIV challenge. *Experiments in Fluids*, 45(1):27–71, July 2008.

[121] Francisco Pereira, Heinrich Stuer, Emilio C. Graff, and Morteza Gharib. Two-frame 3d particle tracking. *Measurement Science and Technology*, 17(7):1680–1692, 2006.

[122] A. A. Adamczyk and L. Rimai. 2-dimensional particle tracking velocimetry (PTV): Technique and image processing algorithms. *Experiments in Fluids*, 6(6):373–380.

[123] Daniel Kondermann, André Berthe, Ulrich Kertzscher, and Christoph Garbe. Particle detection by momentum stability. *Experiments in Fluids, accepted.*

[124] S. J. Baek and S. J. Lee. A new two-frame particle tracking algorithm using match probability. *Experiments in Fluids*, 22(1):23–32, November 1996.

[125] Stuart B. Dalziel. Decay of rotating turbulence: some particle tracking experiments. *Applied Scientific Research*, 49(3):217–244, 1992.

[126] K. Takehara, R. J. Adrian, G. T. Etoh, and K. T. Christensen. A Kalman tracker for super-resolution PIV. *Experiments in Fluids*, 29(7):34–41, 2000.

[127] P. Ruhnau, T. Kohlberger, C. Schnörr, and H. Nobach. Variational optical flow estimation for particle image velocimetry. *Experiments in Fluids*, pages 21–32, 2005.

[128] Nobuhide Kasagi and Akio Matsunaga. Three-dimensional particle-tracking velocimetry measurement of turbulence statistics and energy budget in a backward-facing step flow. *International Journal of Heat and Fluid Flow*, 16(6):477–485, December 1995.

[129] D. P. Hart. PIV error correction. *Experiments in Fluids*, 29(1):13–22, July 2000.

[130] Haitao Xu, Nicholas T. Ouellette, and Eberhard Bodenschatz. Curvature of lagrangian trajectories in turbulence. *Phys. Rev. Lett.*, 98(5):050201–4, February 2007.

[131] E. C. Graff and M. Gharib. Performance prediction of point-based three-dimensional volumetric measurement systems. *Measurement Science and Technology*, 19(7):075403–, 2008.

[132] I. T. Jolliffe. *Principal Component Analysis*. Springer, 1986.

[133] F. De la Torre and M. J. Black. A framework for robust subspace learning. *International Journal on Computer Vision*, 54(1-3):117–142, 2003.

[134] M. J. Black and A. D. Jepson. A probabilistic framework for matching temporal trajectories: Condensation-based recognition of gestures and expressions. In H. Burkhardt and B. Neumann, editors, *European Conference on Computer Vision*, volume 1406 of *Lecture Notes in Computer Science*, pages 909–924, Freiburg, Germany, 1998. Springer-Verlag.

[135] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. Trust-region methods. SIAM Philadelphia, 2000.

[136] B. Jähne. *Digital Image Processing.* Springer Verlag, 2002.

[137] A. Berthe, D. Kondermann, C. Christensen, L. Goubergrits, and U. Kertzscher. Using single particles for the validation of a 3d-3c near wall measurement technique. *Experiments in Fluids, under revision.*, 2009.

[138] Daniel Kondermann, Claudia Kondermann, André Berthe, Ulrich Kertzscher, and Christoph Garbe. Motion estimation based on a temporal model of fluid flows. In *The 13th International Symposium on Flow Visualization*, Nice, France, 2008.

[139] Claudia Kondermann. Postprocessing and restoration of optical flows, dissertation. University of Heidelberg, 2009.

[140] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

List of Publications

[1] Daniel Kondermann and Christoph Garbe. New frontiers in optical flow estimation. *International Journal of Computer Vision, submitted.*

[2] Daniel Kondermann, André Berthe, Ulrich Kertzscher, and Christoph Garbe. Trajectory ensembles for fluid flow estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, submitted.*

[3] Daniel Kondermann, André Berthe, Ulrich Kertzscher, and Christoph Garbe. Particle detection by momentum stability. *Experiments in Fluids, under revision.*

[4] Scholz J., Wiersbinski T., Ruhnau P., Kondermann D., Garbe C.S., Hain R., and Beushausen V. Double-pulse planar-lif investigations using fluorescence motion analysis for mixture formation investigation. *Experiments in Fluids*, 2008.

[5] Daniel Kondermann, Claudia Kondermann, André Berthe, Ulrich Kertzscher, and Christoph Garbe. Motion estimation based on a temporal model of fluid flows. In *The 13th International Symposium on Flow Visualization*, Nice, France, 2008.

[6] B. Andres, C. Kondermann, D. Kondermann, U. Köthe, F. Hamprecht, and C. Garbe. On errors-in-variables regression with arbitrary covariance and its application to optical flow estimation. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition. CVPR 2008.*, 2008.

[7] C. Kondermann, D. Kondermann, B. Jähne, and C. Garbe. An adaptive confidence measure for optical flows based on linear subspace projections. In *Pattern Recognition*, volume 4713 of *LNCS*, pages 132–141. Springer, 2007.

[8] C. Kondermann, D. Kondermann, and C. Garbe. Postprocessing of optical flows via surface measures and motion inpainting. In *Pattern Recognition*, volume 5096 of *LNCS*, pages 355–365. Springer, 2008.

[9] André Berthe, Daniel Kondermann, Carolyn Christensen, Leonid Goubergrits, and Ulrich Kertzscher. Using single particles for the validation of a 3d-3c near wall measurement technique. In *Proceedings of the 7th International Symposium Particle Image Velocimetry*, Roma, Italy, 2007.

[10] André Berthe, Sarah Weber, Daniel Kondermann, Leonid Goubergrits, and Ulrich Kertzscher. Application of the wall-piv technique on cerebral aneurysm models. In *The 13th International Symposium on Flow Visualization*, Nice, France, 2008.

[11] André Berthe, Daniel Kondermann, Leonid Goubergrits, and Ulrich Kertzscher. Visualisierung einer wandnahen blutpumpenströmung mittels wand-piv. In *Tagungsband der 16. Fachtagung Lasermethoden in der Strömungsmesstechnik*, Karlsruhe, Germany, 2008.

[12] Sarah Weber, André Berthe, Daniel Kondermann, Jens Pöthke, Leonid Goubergrits, Ulrich Kertzscher, and Klaus Affeld. A cerebral aneurysm flow visualization using a new technique. In *35th European Society for Artificial Organs Congress*, Geneva, Switzerland, 2008.

[13] André Berthe, Daniel Kondermann, Carolyn Christensen, Leonid Goubergrits, Ulrich Kertzscher, and Klaus Affeld. A blood pump flow visualization using a wall-piv technique. In *35th European Society for Artificial Organs Congress*, Geneva, Switzerland, 2008.

[14] F. Rotter, J. Scholz, J. Müller, T. Wiersbinski, M. Röhl, P. Ruhnau, D. Kondermann, C. Garbe, and V. Beushausen. *Imaging Measurement Methods for Flow Analysis, Results of the DFG Priority Programme 1147 "Imaging Measurement Methods for Flow Analysis" 2003 - 2009*, volume 106 of *Notes on numerical Fluid Mechanics and Multidisciplinary Design*, chapter Simultaneous, Planar Determination of Fuel/Air Ratio and Velocity Field in Single Phase Mixture Formation Processes, pages 165 – 175. Springer Verlag, 2009.

[15] Christoph Garbe, Daniel Kondermann, Markus Jehle, and Bernd Jähne. *Imaging Measurement Methods for Flow Analysis, Results of the DFG Priority Programme 1147 "Imaging Measurement Methods for Flow Analysis" 2003 - 2009*, volume 106 of *Notes on numerical Fluid Mechanics and Multidisciplinary Design*, chapter Spatiotemporal Image Analysis for Fluid Flow Measurements, pages 289 – 305. Springer Verlag, 2009.

[16] André Berthe, Daniel Kondermann, Christoph Garbe, Klaus Affeld, Bernd Jähne, and Ulrich Kertzscher. *Imaging Measurement Methods for Flow Analysis, Results of the DFG Priority Programme 1147 "Imaging Measurement Methods for Flow Analysis" 2003 - 2009*, volume 106 of *Notes on numerical Fluid Mechanics and Multidisciplinary Design*, chapter The Wall-PIV Measurement Technique for Near Wall Flow Fields in Biofluid Mechanics, pages 11 – 20. Springer Verlag, 2009.