Inaugural-Dissertation

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

vorgelegt von
Dipl.–Inf. Florian Becker
aus Bad Soden am Taunus

Tag der mündlichen Prüfung: 30. April 2009

Variational Correlation and Decomposition Methods for Particle Image Velocimetry

Gutachter:	**Prof. Dr. Christoph Schnörr**
Zweitgutachter:	**Prof. Dr. Bernd Jähne**

Zusammenfassung

Particle Image Velocimetry (PIV, etwa: Geschwindigkeitsmessung basierend auf Partikelbildern) ist ein berührungsloses, optisches Messverfahren für Strömungen in Fluiden, das in der Industrie verwendet wird. Dabei werden kleine Partikel in das Gas oder die Flüssigkeit gegeben und dienen so als Indikator für die Bewegung der untersuchten Substanz an Hindernissen oder in Mischbereichen. In der zweidimensionalen Variante des Verfahrens wird eine dünne Ebene von einem Laser beleuchtet, so dass die sich dort befindlichen Partikel sichtbar werden. Eine Kamera zeichnet davon in kurzem zeitlichen Abstand Bilder auf. Deren Analyse erlaubt es, die Bewegung der Partikel, und somit die Geschwindigkeit, Verwirbelung sowie weitere abgeleitete physikalische Eigenschaften des Fluids, zu bestimmen.

Moderne Analysemethoden suchen Korrespondenzen zwischen Regionen zweier aufeinanderfolgender Bilder, indem sie die Kreuzkorrelation als Ähnlichkeitsmaß verwenden. In der Praxis hat sich dieses Maß als robust gegen Störungen erwiesen, wie sie typischerweise in PIV Daten vorkommen. Normalerweise wird eine erschöpfende Suche über eine diskrete Menge von Bewegungsvektoren durchgeführt, um denjenigen zu bestimmen, der die Bilddaten am besten erklärt. In dieser Arbeit jedoch formulieren wir diese Aufgabe als ein Variationsproblem. Motiviert wird dies durch die umfangreichen Ergebnisse auf dem Gebiet des optischen Flusses. Vorwissen über die physikalischen Eigenschaften des untersuchten Stoffes können durch die Formulierung mit Hilfe der Variationsrechnung miteinbezogen werden. Desweiteren ersetzen wird das üblicherweise quadratische Korrelationsfenster, welches die Bildregionen definiert, deren Korrespondenz untersucht wird, durch eine Gewichtung mit einer Gaussfunktion. Diese Wahl erhöht die Anpassungsfähigkeit des Messverfahrens an die Eigenschaften der Bilddaten deutlich. Wir definieren ein Kriterium, um die Grösse und Form der Fensterfunktion anzupassen, welches direkt darauf abzielt, die Genauigkeit der Bewegungsmessung zu verbessern. Die Anpassung der Fensterform und Geschwindigkeitsmessung werden in einem einzigen Optimierungsproblem vereint. Wir wenden Methoden aus der kontinuierlichen Optimierung an, um eine Lösung dieses nicht-linearen und nicht-konvexen Problems zu bestimmen.

Im experimentellen Teil demonstrieren wir die Fähigkeit unseres Ansatzes, synthetische als auch reale Daten mit hoher Genauigkeit zu verarbeiten, und vergleichen ihn mit anderen aktuellen Verfahren. Desweiteren zeigen wir, dass die vorgeschlagene Fensteranpassung die Messgenauigkeit erhöht. Insbesondere können starke Gradienten im Vektorfelder so besser aufgelöst werden.

Im zweiten Teil unserer Arbeit entwickeln wir einen Ansatz, um sehr große konvexe Optimierungsprobleme zu lösen. Diese Untersuchung wird zum einen dadurch motiviert, dass Variationsansätze es auf einfache Art und Weise erlauben, Vorwissen über Daten und Variablen einzubringen, und so die Qualität der Lösung zu verbessern. Zudem treten konvexe Probleme oftmals als Unterprogramme von Lösern für nicht-konvexe Optimierungsaufgaben auf, wie dies im Ansatz im ersten Teil der Arbeit der Fall ist. Die Erweiterung von zweidimensionalen Problemen aus der digitalen Bildverarbeitung auf 3D, oder auf die Zeitachse, sowie die stets höher werdenden Sensorauflösungen lassen jedoch die Anzahl der involvierten Variablen förmlich explodieren. Für viele interessante

Anwendungen, z.B. in der medizinischen Bildverarbeitung oder der Strömungsmechanik, überschreiten so die Problembeschreibungen leicht die Speichergrenzen aktueller nicht-paralleler Rechner.

Aus diesem Grunde untersuchen wir ein Zerlegungsverfahren für die Klasse der konvexen, quadratischen Optimierungsprobleme ohne Nebenbedingungen. Unser Ansatz basiert auf der dualen Formulierung des Problems, und unterteilt es in eine Reihe kleinerer Optimierungsaufgaben, die auf parallel arbeitende Hardware verteilt werden können. Jedes Teilproblem wiederum ist quadratisch und konvex und kann daher effizient mit Hilfe von Standardmethoden gelöst werden. Der Abhängigkeit der Optimierungsprobleme untereinander wird Rechnung getragen, um sicher zu stellen, dass wir wirklich das Originalproblem lösen. Weiterhin schlagen wir eine Erweiterung des Verfahrens vor, die es erlaubt, die numerischen Eigenschaften der Teilprobleme, und damit deren Konvergenzrate, zu verbessern. Der theoretische Teil wird durch eine Analyse der Konvergenzbedingungen und -rate abgeschlossen.

Zum Abschluss demonstrieren wir die Funktionsweise unseres Ansatzes an Hand dreier Variationsansätze aus der Bildverarbeitung. Die Genauigkeit der Ergebnisse wird im Vergleich zu Lösungen des nicht-zerlegten Problems gemessen.

Abstract

Particle Image Velocimetry (PIV) is a non-intrusive optical measurement technique for industrial fluid flow questions. Small particles are introduced into liquids or gases and act as indicators for the movement of the investigated substance around obstacles or in regions where fluids mix. For the two-dimensional variant of the PIV method, a thin plane is illuminated by laser light rendering the particles therein visible. A high speed camera system records an image sequence of the highlighted area. The analysis of this data allows to determine the movement of the particles, and in this way to measure the speed, turbulence or other derived physical properties of the fluid.

In state-of-the-art implementations, correspondences between regions of two subsequent image frames are determined using cross-correlation as similarity measurement. In practice it has proven to be robust against disturbances typically found in PIV data. Usually, an exhaustive search over a discrete set of velocity vectors is performed to find the one which describes the data best. In our work we consider a variational formulation of this problem, motivated by the extensive work on variational optical flow methods which allows to incorporate physical priors on the fluid. Furthermore, we replace the usually square shaped correlation window, which defines the image regions whose correspondence is investigated, by a Gaussian function. This design drastically increases the flexibility of the process to adjust to features in the experimental data. A sound criterion is proposed to adapt the size and shape of the correlation window, which directly formulates the aim to improve the measurement accuracy. The velocity measurement and window adaption are formulated as an interdependent variational problem. We apply continuous optimisation methods to determine a solution to this non-linear and non-convex problem.

In the experimental section, we demonstrate that our approach can handle both synthetic and real data with high accuracy and compare its performance to state-of-the-art methods. Furthermore, we show that the proposed window adaption scheme increases the measurement accuracy. In particular, high gradients in motion fields are resolved well.

In the second part of our work, we investigate an approach for solving very large convex optimisation problems. This is motivated by the fact that a variational formulation on the one hand allows to easily incorporate prior knowledge on data and variables to improve the quality of the solution. Furthermore, convex problems often occur as subprograms of solvers for non-convex optimisation tasks, as it is the case in the first part of this work. However, the extension of two-dimensional approaches to 3D, or to the time axis, as well as the ever increasing resolution of sensors, let the number of variables virtually explode. For many interesting applications, e.g. in medical imaging or fluid mechanics, the problem description easily exceeds the memory limits of available, single computational nodes.

Thus, we investigate a decomposition method for the class of unconstrained, convex and quadratic optimisation problems. Our approach is based on the idea of Dual Decomposition, or Lagrangian Relaxation, and splits up the problem into a couple of smaller tasks, which can be distributed to parallel hardware. Each subproblem is again quadratic

and convex and thus can be solved efficiently using standard methods. Their interconnection is respected to ensure that we find a solution to the original, non-decomposed problem. Furthermore we propose a framework to modify the numerical properties of the subproblems, which enables us to improve their convergence rates. The theoretical part is completed by the analysis of convergence conditions and rate.

Finally, we demonstrate our approach by means of three relevant variational problems from image processing. Error measurements in comparison to single-domain solutions are presented to assess the accuracy of the decomposition.

Danksagung

In meiner Zeit als Doktorand wurde ich durch viele Menschen unterstützt, ohne die mein Promotionsvorhaben wohl kaum in dieser Form möglich gewesen wäre.

Mein Doktorvater Christoph Schnörr führte mich schon während meines Studiums an das faszinierende Gebiet der digitalen Bildverarbeitung heran und gab mir schließlich die Möglichkeit, in seiner Gruppe zu promovieren. Besonders beeindruckt haben mich seine detaillierten Kenntnisse auf sein Forschungsgebiet und darüber hinaus, aber auch seinem Überblick auf einer sehr abstrakten Ebene. Sein Wissen und seine menschlichen Seiten machten die Zusammenarbeit mit ihm zu einer wertvollen, positiven Erfahrung. Ich danke ihm vielmals für diese Zeit und werde sie sicher in guter Erinnerung behalten.

Bernhard Wieneke ließ mich als Projektpartner dankenswerterweise an seinem umfangreichen Erfahrungsschatz auf dem Gebiet der Strömungsmechanik teilhaben. Durch die Zusammenarbeit mit ihm bekam ich wertvolle Resonanz zu meiner Arbeit auf dem Gebiet der adaptiven Korrelation.

Weiterhin geht mein Dank an all meine ehemaligen und gegenwärtigen Kolleginnen und Kollegen an den Universitäten Mannheim und Heidelberg für die angenehme Arbeitsathmossphäre. Speziell meine Mitstreiter in der IPA-Gruppe, ehemals CVGPR-Gruppe, boten mir viele fruchtbare Diskussionen, aber auch Freunschaft und schöne gemeinsame Erlebnisse in und außerhalb der Universität. Dies sind namentlich Martin Bergtholdt, Dirk Breitenreicher, Christian Gosch, Matthias Heiler, Jörg Kappes, Rezaul Karim, Timo Kohlberger, Jan Lellmann, Stefania Petra, Paul Ruhnau, Christian Schellewald, Stefan Schmidt, Thomas Schüle, Annette Stahl, Andriy Vlasenko, Stefan Weber und Jing Yuan. Ganz besonders hervorheben möchte ich Dirk, Jan, Jörg, Stefan Schmidt und Stefania, die diese Arbeit ganz oder in Teilen korrekturgelesen haben und mir wertvolle Hinweise gaben.

Außerdem möchte ich die Gelegenheit nutzen, mich ganz herzlich bei Rita Schieker und Barbara Werner zu bedanken, die mir nicht nur jederzeit in administrativen Dingen zur Seite standen, sondern auch moralische Unterstützung boten.

Schließlich danke ich meiner Familie, die stets hinter meinen Studienplänen stand und mir in arbeitsreichen Zeiten den Rücken freihielt.

Contents

1 Introduction

1.1 Overview and Organisation

Our work is sectioned into two parts: In Chap. 2 we introduce the reader to Particle Image Velocimetry (PIV), which is a non-intrusive optical measurement technique for industrial fluid flow questions. Basically, a camera system records an image sequence of the particles which act as indicators for the movement of the fluid. The analysis of the gained image sequence allows to measure the speed, turbulence or other derived physical properties of the fluid. In state-of-the-art implementations, correspondences between regions of two subsequent image frames are determined using cross-correlation as similarity measure. In practice, it has proven to be robust against disturbances typically found in PIV data. Usually, an exhaustive search over a discrete set of velocity vectors is performed to find the one which describes the data best.

In Chap. 3 we propose a variational approach to motion estimation for PIV image pairs, based on the cross-correlation measure. Continuous optimisation methods are used to determine the optimal displacements. Furthermore, we introduce a very flexible window shape, which is responsible for the selection of the image regions whose correspondence is investigated. This design drastically increases the flexibility of the process to adjust to features in the experimental data. A sound criterion is proposed to adapt the size and shape of the correlation window, which directly formulates the aim to improve the measurement accuracy. The velocity measurement and window adaption are formulated as a joint variational problem. We apply methods from continuous optimisation to find a solution to this non-linear and non-convex problem. In the experimental section, we evaluate our approach on synthetic and real data, and compare its performance to state-of-the-art methods.

The second part of our work is concerned with the decomposition of convex optimisation problems. Our motivation stems from the beneficial properties of variational methods in image processing, namely the possibility to incorporate spatial and/or temporal dependencies easily. While this is a clear advantage in the modelling phase, it circumvents a straightforward problem decomposition. Applications with 3D data, temporal sequences or high sensor resolution then lead to problem sizes that cannot be solved on a single computer anymore due to memory limitations. Besides feasibility, problem decomposition provides a further benefit by reducing the computation time significantly.

Chapter 4 introduces the theory of the employed methods. We summarise the idea of Dual Decomposition and give an overview over related approaches. Subsequently, we define the considered class of convex and unconstrained quadratic optimisation problems. Although the approach proposed in Chap. 3 does not fit in this pattern, convex optimisation often occurs as subroutine of iterative numerical solvers, e.g. for solving the linear system in a Newton step method. Furthermore, Dual Decomposition can be applied to any convex optimisation problem, and thus our results may provide interesting insights for further investigations.

Based on this method, we elaborate a decomposition approach for the considered problem class in Chap. 5. Furthermore, we propose a framework to modify the numerical properties of the subproblems, and in this way to improve their convergence rate. In every step of the decomposition procedure, it is ensured that the initial problem is solved. The theoretical part is completed by the analysis of convergence conditions and rate. Finally, we demonstrate our approach by means of three relevant variational problems from image processing. Two optical flow methods are included, which have already been successfully applied to motion estimation in PIV data.

The major results of the adaptive correlation approach and the decomposition method for quadratic problems are summarised in Chap. 6.

1.2 Notation and Definitions

The notation in our work basically follows the one used in [1] and [2]. For convenience we summarise them in this section.

$\mathbb{R}$	set of real numbers
$\mathbb{C}$	set of complex numbers
$\mathbb{Z} := \{\dots, -2, -1, 0, +1, +2, \dots\}$	set of integers
$\mathbb{N} := \{i \in \mathbb{Z} \,\vert\, i \geq 0\}$	set of non-negative integers
$\mathbb{N}_{++} := \{i \in \mathbb{Z} \,\vert\, i > 0\}$	set of strictly positive integers
$\vert\mathcal{S}\vert$	cardinality of a set
x_i	component i of vector x
$A_{i,j}$	element at row i and column j of matrix A
$x^\top, A^\top$	transposed vector, transposed matrix
A^{-1}	inverse of a (regular) matrix
$A \otimes B$	Kronecker product of matrices A and B
$\det(A)$	determinant of matrix A
$\operatorname{tr}(A)$	trace of matrix A
$\operatorname{diag}(x)$	diagonal matrix with the components of x on the main diagonal and zero elsewhere
$\operatorname{diag}(A_1, \dots, A_d)$	square, block-diagonal matrix constructed from square matrices $A_1, \dots, A_d$
I	unit matrix
e	all-one vector
e_i	vector with a single one in element i and zeros elsewhere
$\lambda_{\min}(A)$, $\lambda_{\max}(A)$	smallest and largest eigenvalue of square matrix A
$\kappa(A) := \sqrt{\frac{\lambda_{\max}(A^\top A)}{\lambda_{\min}(A^\top A)}}$	condition number of square matrix A

$\lvert\alpha\rvert$	absolute value of a scalar α
$\lVert x\rVert_2 := \sqrt{\sum_{i=1}^{n} \lvert x_i\rvert^2}$	Euclidean (L^2) vector norm of $x \in \mathbb{R}^n$
$\lVert x\rVert_1 := \sum_{i=1}^{n} \lvert x_i\rvert$	L^1 vector norm of $x \in \mathbb{R}^n$
$\lVert A\rVert_F := \sqrt{\operatorname{tr}(A^\top A)}$	Frobenius matrix norm
$\lVert A\rVert_2 := \sqrt{\lambda_{\max}(A^\top A)}$	spectral matrix norm
$S \succ T$	matrix $(S - T)$ is positive definite, i.e. $x^\top (S-T)x > 0$, $\forall x \in \mathbb{R}^n$
$S \succeq T$	matrix $(S - T)$ is positive semidefinite, i.e. $x^\top (S-T)x \geq 0$, $\forall x \in \mathbb{R}^n$
$\mathcal{S}^n := \left\{ S \in \mathbb{R}^{n\times n} \,\middle\vert\, S = S^\top \right\}$	set of real, symmetric $n \times n$-matrices
$\mathcal{S}^n_{++} := \{ S \in \mathcal{S}^n \,\vert S \succ 0 \}$	set of real, symmetric, positive definite $n \times n$-matrices
$\nabla_x f(x)$	gradient of a scalar-valued function f in the variables x
$\mathrm{H}_x\, f(x)$	Hessian of a scalar-valued function f in the variables x
$\mathrm{J}_x\, f(x)$	Jacobian matrix of a vector-valued function f in the variables x
$f * g$	convolution operator between functions f and g
$\delta_{i,j} := \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$	Kronecker delta function
$\chi_{\mathcal{S}}(x) := \begin{cases} 1 & \text{if } x \in \mathcal{S} \\ 0 & \text{if } x \notin \mathcal{S} \end{cases}$	characteristic function for the set $\mathcal{S}$
$\delta_{\mathcal{S}}(x) := \begin{cases} 0 & \text{if } x \in \mathcal{S} \\ +\infty & \text{if } x \notin \mathcal{S} \end{cases}$	indicator function for the set $\mathcal{S}$

Furthermore we use

$$\mathrm{G}(\mu, \Sigma)(x) := \frac{1}{\sqrt{(2\pi)^n \det(\Sigma)}} \exp\left(-\frac{1}{2} (x-\mu)^\top \Sigma^{-1} (x - \mu) \right)$$

to denote the multivariate Gaussian function for $x \in \mathbb{R}^n$, mean value $\mu \in \mathbb{R}^n$ and covariance matrix $\Sigma \in \mathcal{S}^n_{++}$.

In our work we use the following definition of the Fourier transformation [3] of a complex-valued, integrable function $f : \mathbb{R}^n \mapsto \mathbb{C}$:

$$\mathcal{F}_x \left\{ f(x) \right\}(\omega) := \int_{\mathbb{R}^2} f(x) \exp\left(-i\, \omega^\top x \right) \,\mathrm{d}x \ ,$$

with the spatial variable x, angular frequency ω and $i := \sqrt{-1}$. Its inverse is denoted by

$$\mathcal{F}^{-1}_\omega \left\{ F(\omega) \right\}(x) := \frac{1}{(2\pi)^n} \int_{\mathbb{R}^2} F(\omega) \exp\left(i\, \omega^\top x \right) \,\mathrm{d}\omega \ .$$

When it is clear from context, we will omit the specification of the variable of the originating domain, x and ω, for the Fourier transform and its inverse, respectively.

2 Cross-Correlation in Particle Image Velocimetry

2.1 Overview

2.1.1 Introduction and Motivation

Particle image velocimetry (PIV, [4]) is a non-intrusive optical measurement technique for industrial fluid flow questions. Small particles are introduced into liquids or gases and act as indicators for the movement of the investigated substance around obstacles or in regions where fluids mix. For the two-dimensional variant of the PIV method a thin plane (sheet) is illuminated by laser light rendering the particles therein visible. A high speed camera system records an image sequence of the highlighted area. We refer to Fig. 2.1 for an illustration of the method and Fig. 2.2(a) for a sample of real PIV image data. The analysis of the obtained image sequence allows to determine the movement of particles, and in this way to measure the speed, turbulence or other derived physical properties of the fluid. Figure 2.2(b) depicts a typical representation of the displacement field derived from a PIV image pair.

Cross-correlation has developed as the state-of-the-art method for motion estimation in PIV and benefits from its robustness against disturbances typical for fluid flow experiments: The brightness of identical particles may differ between two images as laser output is not constant over time and space. Although 2D-PIV experiments are designed to minimise fluid movements perpendicular to the observed plane, particles moving into or out of the illuminated area can always be observed in real data and lead to *unpaired particles*. Short exposure times and the inevitable noise in the camera circuits introduce random disturbances into the image data.

In this work we formulate a variational approach to estimating fluid flow through cross-correlation and solve it using continuous optimisation techniques. The usual rectangular shaped window is replaced by a Gaussian one. In addition, we propose a sound criterion to adapt the size and shape of the correlation window, which directly formulates the aim to improve the measurement accuracy.

2.1.2 Related Work and Contribution

A vast body of literature exists on all aspects of applications and implementation of cross-correlation for PIV, here we only refer to [5] as an excellent overview. Typically, an exhaustive search over the integer displacements is performed to find the highest correlation peak which corresponds to the most probable displacements in this region. The correlation function is interpolated to obtain sub-pixel accuracy. In contrast, we

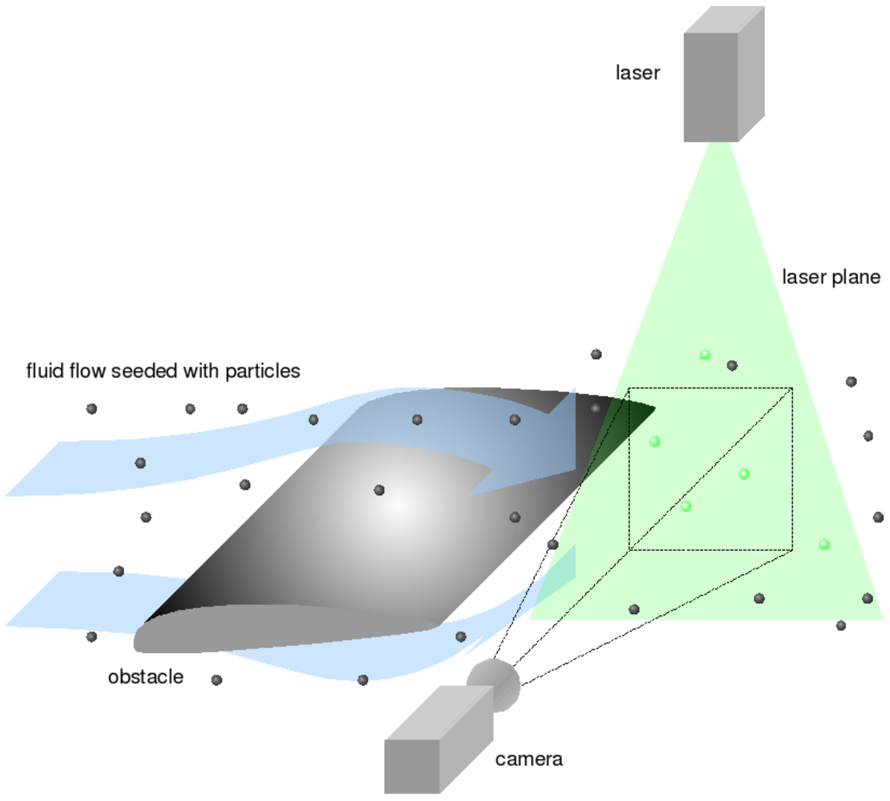


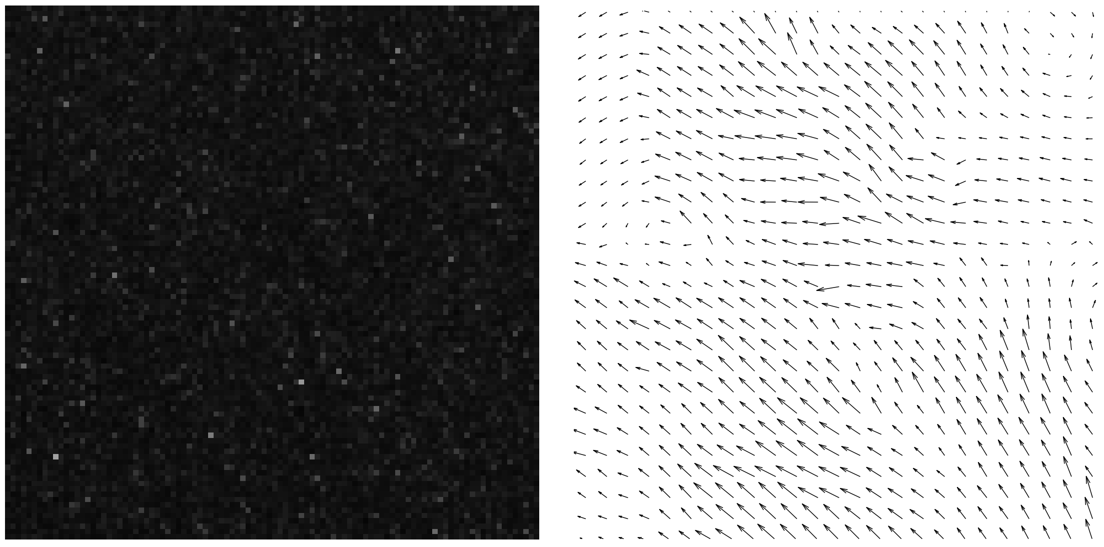
Figure 2.1: Particle Image Velocimetry (PIV) experimental setup: The flow of a fluid behind an obstacle shall be examined. To this end, it is seeded with small particles. A laser projects a thin two-dimensional plane into the region of interest and illuminates the particles therein. Fluid motion can be measured by analysis of the image sequence recorded by a camera system.

present a *variational* approach to motion estimation based on *continuously* maximising the cross-correlation measurement between two images.

Since the introduction of the 2D correlation method for measuring fluid motion, there have been enormous efforts to extend it to three dimensions, including *Dual Plane PIV* [6, 7], *Stereo PIV* [8] and recently *Tomographic PIV* [9]. However, we only consider two-dimensional image data here, although an extension to 3D-data is straightforward, see for example [10].

Particle Tracking Velocimetry (PTV) [11, 12] uses a different approach to obtain a displacement measurement from a seeded flow. Instead of matching image patches based on their grey-values, first particle coordinates are identified in the images. Then particle correspondences over adjacent image frames are identified. In [13] a variational approach to PTV is proposed.

Variational approaches using the optical flow constraint, originating from the original work of Horn and Schunck [14], have also been applied to measure motion in PIV [15].



(a) recorded frame (b) reconstructed displacement field

Figure 2.2: Particle Image Velocimetry (PIV): (a) Detail of an image frame recorded in a real fluid experiment (see Sect. 3.5.5), showing particles (polyglycol diluted in water) in an air flow. The image is 100px $\times$ 100px in size, corresponding to 7.5mm $\times$ 7.5mm in the illuminated plane. The particles have a diameter of less than 10µm. (b) Example of a reconstructed displacement field. Each arrow describes the motion vector estimated at its origin.

The dense vector field representation allows to incorporate prior knowledge on the spatial coherence of the vector field, such as incompressibility, see e.g. [16, 17]. However, the implied brightness constancy assumption often does not hold in real PIV data as indicated above. This is taken into account in [18] by modifying the data term accordingly.

A couple of variational approaches exist which make use of the robustness of the correlation measurement: Three statistical dissimilarity measurements between regions in an image pair are considered in [19] with cross-correlation being one of them. Based on this, a variational approach for image registration with spatial regularisation on the displacements is set up and solved using methods for partial differential equations.

In [20] Heitz et al. present a two-scale approach which combines the advantages of optical flow based approaches and cross-correlation: Basically, an optical flow approach with physically sound regularisation terms, which penalise large variations in the rotation and divergence of the flow, is endowed with an additional data term. Similarity to a coarse vector field, which is calculated beforehand using a local correlation approach, is enforced. It effects the coarse components of the flow, while the variational approach provides a dense and physically consistent flow measurement. The approach is extended to account for *temporal* consistence with respect to physical laws in [21].

The displacement field estimated by a local correlation approach is used to *initialise* a variational optical flow approach in [10].

The variational method in [22] accepts a possibly sparse vector field obtained from an arbitrary algorithm and creates a dense vector field that complies with physical laws. In contrast to the previously mentioned approaches, it does not involve any image data.

In the recent years, much effort has been put into improving the spatial resolution of cross-correlation methods [23, 24] by replacing the fixed square interrogation windows by appropriate alternatives. The authors of [25] consider a class of cone-shaped weighting functions and optimise the shape parameters by means of the frequency-response, however not with respect to a specific set of image data. The authors of [26] use square windows and locally adapt their size to the signal quality (seeding density) and spatial fluctuations in the flow. Window adaption is also used in [27] at interfaces to fixed objects in the scene. In [28] a Gaussian shaped weighting function is stretched and rotated along the measured mean displacement, steered by a set of update rules. In our work, the correlation window is also described by a "soft" Gaussian weighting function. This idea is used both in a local [29] and global context [30] for smoothing the optical flow constraint, however with isotropic windows of fixed size common for all positions. In contrast, we formulate a sound criterion for the location-dependent choice of the window shape parameters (size, orientation, anisotropy) by means of an error model function. The window adaption consists of finding the window shape which minimises this function.

Our contribution is a variational formulation of a correlation-based approach for measuring motion in PIV image pairs. A Gaussian weighting function is used to control the region considered in the displacement estimation. The shape of the window is controlled by means of a function which approximates the expected measurement error. Minimisation gives the optimal window shape with respect to this error model. The displacement measurement and window adaption are formulated as a pair of interdependent optimisation problems. We then solve them jointly via a multiscale gradient-based algorithm. We test our approach with synthetic and real particle images to demonstrate its ability to robustly determine displacements and show that window shape adaption can improve accuracy significantly. Some results of our work were published in [31] with the focus on image processing. In [32] we investigate our approach from the fluid mechanic point of view.

2.1.3 Organisation

In this chapter we give an overview over cross-correlation methods for displacement estimation. We start with the problem statement in Sect. 2.2 and formulate the displacement estimation as an optimisation problem in Sect. 2.3. The usual approach for solving, discrete optimisation, is summarised in Sect. 2.4. The error caused by an implicit linearisation of the particle trajectory in PIV methods is discussed in Sect. 2.5. Our contribution, a variational formulation of the cross-correlation problem with window adaption, is examined in Chap. 3.

2.2 Problem Statement

Two grey-valued images, $g_1, g_2 : \Omega \mapsto \mathbb{R}$ defined on $\Omega \subset \mathbb{R}^2$ and recorded with a (small) time difference $\Delta t := t_2 - t_1$ represent the input for the motion measurement task. The aim is to determine the displacement $\Delta x \in \mathbb{R}^2$ of visible structures between the first and second image. In PIV, this tracing structures are created by illuminated particles. When we divide the displacement by Δt, we obtain an estimate of the velocity vector

$$u := M \frac{\Delta x}{\Delta t} ,$$

which describes the movement of the patterns and thus by assumption the movement of the fluid. Here, M is the magnification factor of the camera. Without loss of generality, we will set $\Delta t = 1$ and $M = 1$ throughout the work, so $u = \Delta x$, and we use the terms displacement and velocity interchangeably.

This velocity measurement is performed for several points of interest within the scene to obtain a vector field which is dense enough to describe the fluid behaviour well.

2.3 Cross-Correlation for Displacement Measurement

For a start, we assume that the observed motion is rigid, i.e. there is a single displacement u that describes the movement of all particles between two images. Furthermore, to simplify notation, we extend the definition of the images to the whole two-dimensional plane, i.e. $g_1, g_2 : \mathbb{R}^2 \mapsto \mathbb{R}$ by defining the function values outside Ω to be zero. Then a generic approach to determine the displacements is to “move around” the images until they fit best to some criterion. If the two-dimensional cross-correlation [3] is chosen as dissimilarity measure, this can be written as the following optimisation problem:

$$u^* \in \arg\max_{u \in \mathbb{R}^2} \int_{\mathbb{R}^2} g_1(x)\, g_2(x+u)\, \mathrm{d}x \tag{2.1}$$

The value u^* which maximises the criteria is directly used as measurement for the displacement. Note, that this function usually has many local maxima as Fig. 2.3 demonstrates.

However, the assumption that an uniform displacement is valid for the whole image is far from conditions found in real fluid experiments. In fact, velocity gradients and turbulences dominate real flows and the motion estimator must be able to resolve these details. For this reason correlation is limited to a neighbourhood of the position of interest $x_0 \in \mathbb{R}^2$, where the assumption of an uniform movement holds at least approximately. In the simplest case, the evaluation is limited to a neighbourhood T (in g_1) around x_0. In addition, the search space for u might be limited to a reasonable range U.

$$\max_{u \in U} \int_T g_1(x) g_2(x+u)\, \mathrm{d}x \tag{2.2}$$

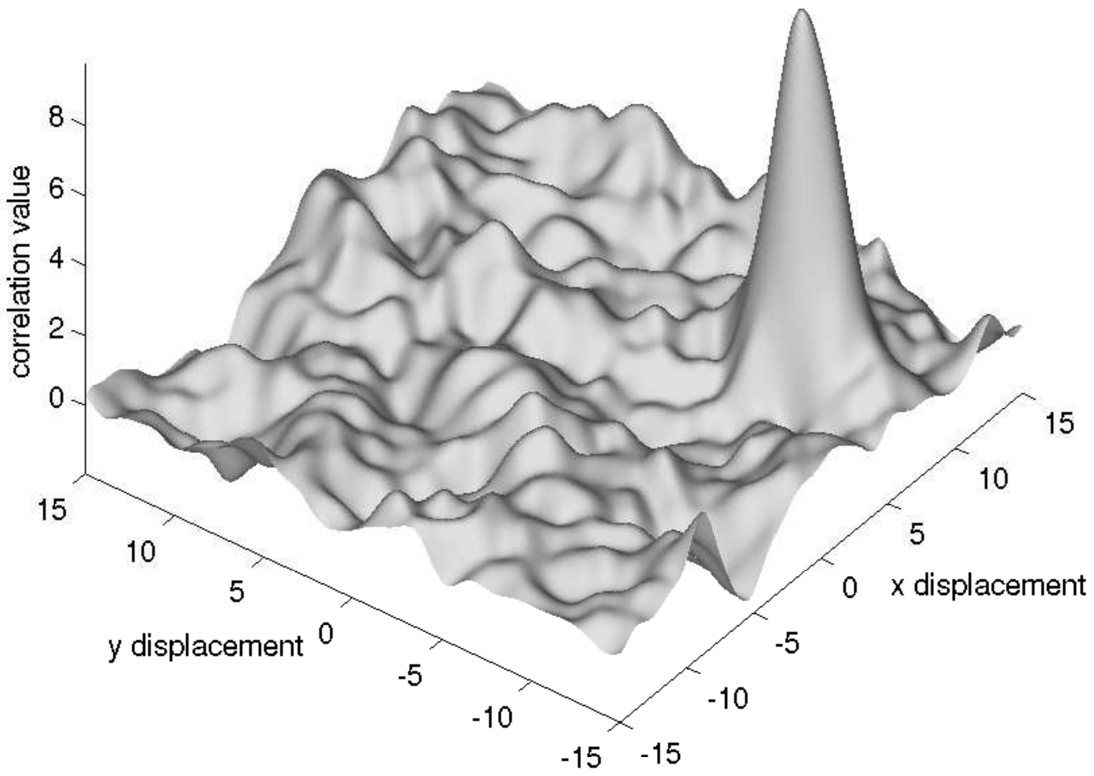


Figure 2.3: Value of the cross-correlation-function between two particle images. The global maxima, denoted as *correlation peak*, indicates the displacement with the best fit between the two frames, here $u^* = (8, -8)$.

The region T within the first image is denoted as *template*, while the examined part in the second image,

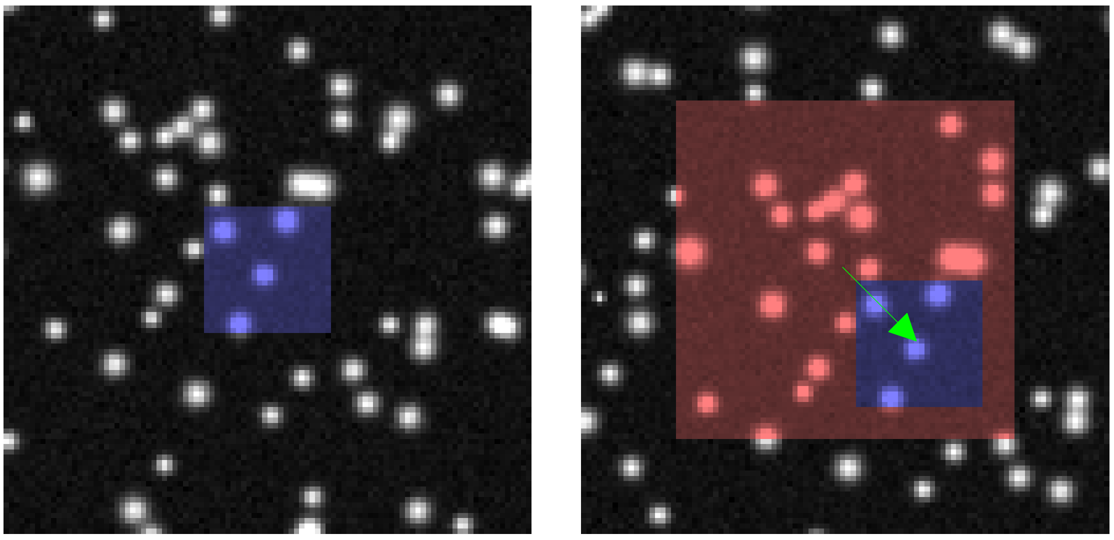
$$I := T + U = \{x_0 + u \,|x \in T, u \in U\}$$

is denoted as *interrogation area* or *spot* [33], see Fig. 2.4 for a simple example. Usually a square window with sharp boundaries is employed for T, but more complex shapes are possible and also reasonable as we will demonstrate in Chap. 3.

An interesting fact shall be mentioned in this context: The convolution theorem [3] states, that the convolution $f * g$ of two functions f, g can be performed in the frequency domain,

$$(f * g)(x) = \int_{\mathbb{R}^2} f(y)g(x-y)\,\mathrm{d}y = \mathcal{F}_\omega^{-1}\left\{\hat{f}(\omega)\hat{g}(\omega)\right\}(x) \ ,$$

with $\hat{f}$, $\hat{g}$ being the Fourier-transformed functions f and g, respectively. For readability, we introduce $g_{1,T} := \chi_T(x)g_1(x)$. The cross-correlation of the image functions in (2.2)



(a) frame 1 (g_1) (b) frame 2 (g_2)

Figure 2.4: Cross-correlation for motion estimation: two grey-valued particle image frames (synthetic) with square template and interrogation areas: (a) template T (blue) taken from frame 1, (b) interrogation area $I := T + U$ (red) in frame 2, with shifted template; the position of the highest correlation indicates the displacement u (green arrow).

can easily be written as convolution, and thus can be evaluated in frequency space.

$$\begin{aligned}\int_T g_1(x)g_2(x+u)\,\mathrm{d}x &= \int_{\mathbb{R}^2} \chi_T(x)g_1(x)g_2(x+u)\,\mathrm{d}x \\ &= \left(g_{1,T}(x) * g_2(-x)\right)(u) = \mathcal{F}^{-1}\left\{\hat{g}_{1,T}(\omega)\overline{\hat{g}_2(\omega)}\right\}(u)\end{aligned}$$

This important result allows to speed up the evaluation significantly in the discrete case.

2.4 Discrete Optimisation

State-of-the-art cross-correlation methods basically perform an exhaustive search over a range of integer displacements. However, here we can only give an outline over the techniques used. A huge number of publications exist on modifications and extensions with the aim to improve accuracy and computational efficiency. For an exhausting overview we refer to [5].

Let $g_1(x)$ and $g_2(x)$ be two images defined on integer coordinates $x \in \mathbb{Z}$, as it is the case in practice. Usually the templates and interrogation areas are defined as squared regions with integer parameters n and m,

$$T_n = \left\{ x \in \mathbb{Z}^2 \,\middle|\, \|x - x_0\|_\infty \leq \frac{n}{2} \right\} ,$$

and

$$U_m = \left\{ u \in \mathbb{Z}^2 \,\middle|\, \|u - u_0\|_\infty \leq \frac{m}{2} \right\} ,$$

where u_0 can be used to adapt the interrogation area if some prediction for the displacement exists, e.g. from a previous estimation on a different scale. Then the discrete formulation of (2.2) is

$$u^* \in \arg\max_{u \in U_m} C(u) , \qquad C(u) := \sum_{x \in T_n} g_1(x) g_2(x+u) .$$

Figure 2.4 visualises this situation.

The brute-force evaluation of the correlation function for each displacement $u \in U_m$ has a total computational complexity of $O(m^2 n^2)$ as for $|U_m|$ possible displacements, $|T_n|$ grey-value pairs have to be multiplied and their results added. Alternatively, the template T_n in g_1 can be extracted and zero-padded to the size of the interrogation area I in g_2, which is $(n+m) \times (n+m)$. Subsequently, we can apply the convolution theorem for the discrete case and evaluate the correlation using Fast Fourier Transformation (FFT) [34, 35]. The complexity then reduces to $O((n+m)^2 \log(n+m))$ and thus this is a common improvement in practical implementations.

After evaluating $C(u)$ within U_m, a simple search for the position of the maximum value is performed, which indicates the most probable displacement. The correlation peak can be determined with a resolution of U_m, which is typically one pixel (px) of the image data. However, this is insufficient in most applications. For obtaining a displacement u at sub-pixel resolution, a concave function, typically Gaussian or parabolic, is fit into the estimated correlation function in the close vicinity of u^*, usually 3px $\times$ 3px. From this continuous representation, the peak position can be determined analytically.

2.5 Accuracy of the Particle Trajectory Linearisation

Let us assume that a structure, e.g. a particle, moving along the trajectory $x(t)$ was observed in the two images at time t_1 and t_2 at positions $x(t_1)$ and $x(t_2)$, respectively. Furthermore, a reasonable presumption is, that the trajectory is a continuous function in t. Then the usual implication of motion estimation approaches is, that the considered image pattern moves with speed $u = \frac{\Delta x}{\Delta t}$ straight from $x(t_1)$ to $x(t_2)$ between the two images. This, however, is only an approximation of the actual track. Figure 2.5 illustrates this situation. In order to estimate the error caused by this assumption we represent the actual trajectory by a Taylor series about t,

$$x(\tau) = x(t) + \dot{x}(t) \cdot (\tau - t) + \sum_{k=2}^{\infty} x^{(k)}(t) \frac{(\tau - t)^k}{k!} ,$$

and rearrange it as

$$\dot{x}(t) \cdot (\tau - t) = x(\tau) - x(t) - \sum_{k=2}^{\infty} x^{(k)}(t) \frac{(\tau - t)^k}{k!} .$$

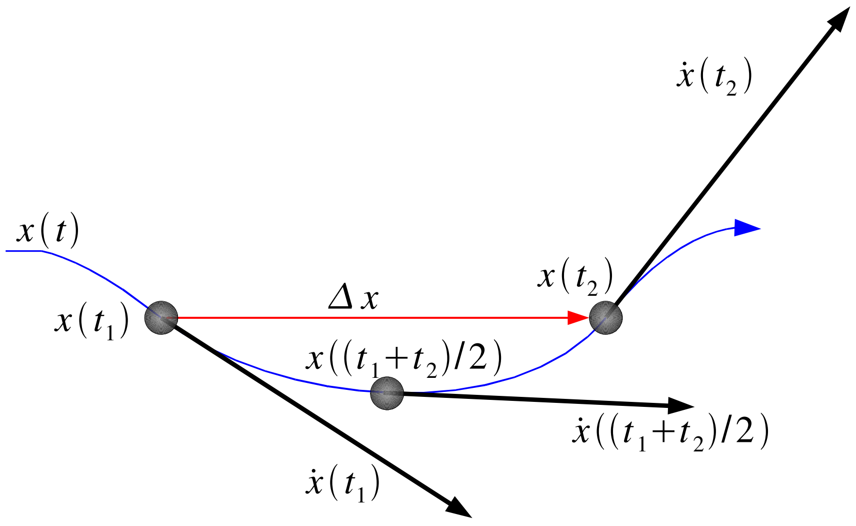


Figure 2.5: Movement of a particle $x(t)$ in time (blue), sampled at time t_1 and t_2 yielding positions $x(t_1)$ and $x(t_2)$, respectively. The movement $u = \frac{\Delta x}{\Delta t}$ derived from displacement Δx (red) is considered as an approximation of $\dot{x}(t)$ at some time $t \in [t_1, t_2]$. Depending on the choice of t, the estimate is first-order accurate ($t = t_1$ or $t = t_2$) or second-order accurate ($t = \frac{1}{2}(t_1 + t_2)$).

Here, $x^{(k)}(t)$ denotes the k-th derivative of $x(t)$ in t, and $\dot{x}(t) = x^{(1)}(t)$. Now we can express the actual speed at t in words of the two measurements at $\tau = t_1$ and $\tau = t_2$.

$$\begin{aligned}
\dot{x}(t) = \frac{\dot{x}(t)\cdot(t_2-t_1)}{t_2-t_1} &= \frac{\dot{x}(t)\cdot(\tau-t)|_{\tau=t_2} - \dot{x}(t)\cdot(\tau-t)|_{\tau=t_1}}{t_2-t_1} \\
&= \underbrace{\frac{x(t_2)-x(t_1)}{t_2-t_1}}_{u} + \underbrace{\sum_{k=2}^{\infty} x^{(k)}(t)\frac{(t_1-t)^k-(t_2-t)^k}{k!(t_2-t_1)}}_{\epsilon_{\mathrm{u}}(t)} \\
&= \quad u \quad + \quad \epsilon_{\mathrm{u}}(t)
\end{aligned}$$

The remaining higher-order terms represent the error $\epsilon_{\mathrm{u}}(t)$ between the actual speed $\dot{x}(t)$ and the estimation u if we assume that the measurement is dated at t. Although t can be chosen freely, three values are typically used in literature ([36, 37]):

Forward/Backward Differencing Interrogation: This method sets $t = t_1$ and $t = t_2$, respectively. Then the error in the velocity estimation is

$$\epsilon_{\mathrm{u}}(t_1) = \epsilon_{\mathrm{u}}(t_2) = -\sum_{k=2}^{\infty} x^{(k)}(t)\frac{(\Delta t)^{k-1}}{k!} = -x^{(2)}(t)\frac{(\Delta t)}{2} - \cdots .$$

The lowest order of the error term is two.

Central Differencing Interrogation: In this approach, the estimation is assumed to be right in between the two location observations, i.e. $t = \frac{1}{2}(t_1 + t_2)$.

$$\begin{aligned}\epsilon_{\mathrm{u}}\left(\frac{1}{2}(t_1+t_2)\right) &= \sum_{k=2}^{\infty} x^{(k)}(t)\frac{(-\frac{1}{2}\Delta t)^k - (\frac{1}{2}\Delta t)^k}{k!\Delta t} = -\sum_{k=1}^{\infty} x^{(2k+1)}(t)\frac{(\frac{1}{2}\Delta t)^{2k}}{(2k+1)!}\\ &= -x^{(3)}(t)\frac{(\Delta t)^2}{24} - \cdots\end{aligned}$$

Note, that now all terms with even order vanish, including the order of two. Thus, u is a second-order accurate estimation of the velocity in contrast to forward and backward differencing interrogation.

With the choice of t, however, also some implication about the *location* p of the measurement is made. While forward and backward differencing interrogation derive the position $x(t)$ directly from the observation and thus make no error at all, the central method errs in this respect by assuming $p = \frac{1}{2}(x(t_1) + x(t_2))$. Denoting the Taylor approximations about t as $x_t(\tau)$, we can write the correct position of the observed image structure as

$$\begin{aligned}x\left(\frac{1}{2}(t_1+t_2)\right) &= x_{t_1}\left(\frac{1}{2}(t_1+t_2)\right) = x_{t_2}\left(\frac{1}{2}(t_1+t_2)\right)\\ &= \frac{1}{2}\left(x_{t_1}\left(\frac{1}{2}(t_1+t_2)\right) + x_{t_2}\left(\frac{1}{2}(t_1+t_2)\right)\right)\\ &= \frac{1}{2}\left(x(t_1) + \sum_{k=1}^{\infty} x^{(k)}(t_1)\frac{(\frac{1}{2}\Delta t)^k}{k!}\right) + \frac{1}{2}\left(x(t_2) + \sum_{k=1}^{\infty} x^{(k)}(t_2)\frac{(-\frac{1}{2}\Delta t)^k}{k!}\right)\\ &= \underbrace{\frac{x(t_1)+x(t_2)}{2}}_{} + \underbrace{\frac{1}{2}\sum_{k=1}^{\infty}\left(x^{(k)}(t_1) + (-1)^k x^{(k)}(t_2)\right)\frac{(\frac{1}{2}\Delta t)^k}{k!}}_{}.\\ &= \qquad p \qquad + \qquad \epsilon_{\mathrm{p}}\end{aligned}$$

Hence, the error made by the assumption $x(\frac{1}{2}(t_1 + t_2)) = p$ is described by ϵ_{p}. Despite this drawback, in practice this choice has proven beneficial [24]. Thus we will employ central differencing interrogation in the definition of our approach in Chap. 3.

2.6 Summary

In this chapter we gave an overview over the two-dimensional version of PIV and the cross-correlation approach on which state-of-the-art motion measurement methods are based on. Furthermore, we discussed the influence of the linearisation of the particle trajectories on the accuracy. In the following chapter we propose a cross-correlation-based variational approach for motion estimation and extend it by a window adaption scheme.

3 Adaptive Correlation for Motion Estimation

In this work we consider the problem of motion estimation via cross-correlation in its natural form, as a continuous optimisation problem. In contrast to the approach presented in Sect. 2.4, we do not discretise the search space of u but directly determine the integer and fractional part of the displacement vector as a whole. In Sect. 3.1 we define our variational formulation of a correlation approach. In addition, we introduce the employed steerable Gaussian weighting function and discuss its properties.

In Sect. 3.2 we propose an approach for adapting the window shapes used for displacement estimation. Is is based on a sound error model which directly formulates the aim to minimise the error of the velocity measurement.

The displacement estimation and window adaption is combined into a joint optimisation problem in Sect. 3.3. Section 3.4 is dedicated to the discretisation and solving of the optimisation problem. Finally, in Sect. 3.5 we verify the design of the approach by means of synthetic and real PIV data and conclude in Sect. 3.6.

3.1 Variational Formulation for Cross-Correlation

3.1.1 Definition

In this work, the displacement estimation is based on the cross-correlation measurement. However, we slightly reformulate it as a minimisation problem. In addition, we use central differencing to obtain second-order accuracy in velocity estimation as demonstrated in Sect. 2.5. Due to the infinite integration domain, the following problem is equivalent to (2.1).

$$\min_{u\in\mathbb{R}^2} \; -\int_{\mathbb{R}^2} g_1\left(x-\frac{1}{2}u\right) g_2\left(x+\frac{1}{2}u\right) \mathrm{d}x$$

Instead of limiting the evaluation of this measurement to the interrogation area and template window, we introduce a function

$$w(x,\Sigma) : \mathbb{R}^2 \times S \mapsto [0,1]$$

which weights the similarity measurement. The parameter $\Sigma \in S$, with S being the set of allowed values, combines the description of shape and size of w. The choice of this function is discussed in Sect. 3.1.2.

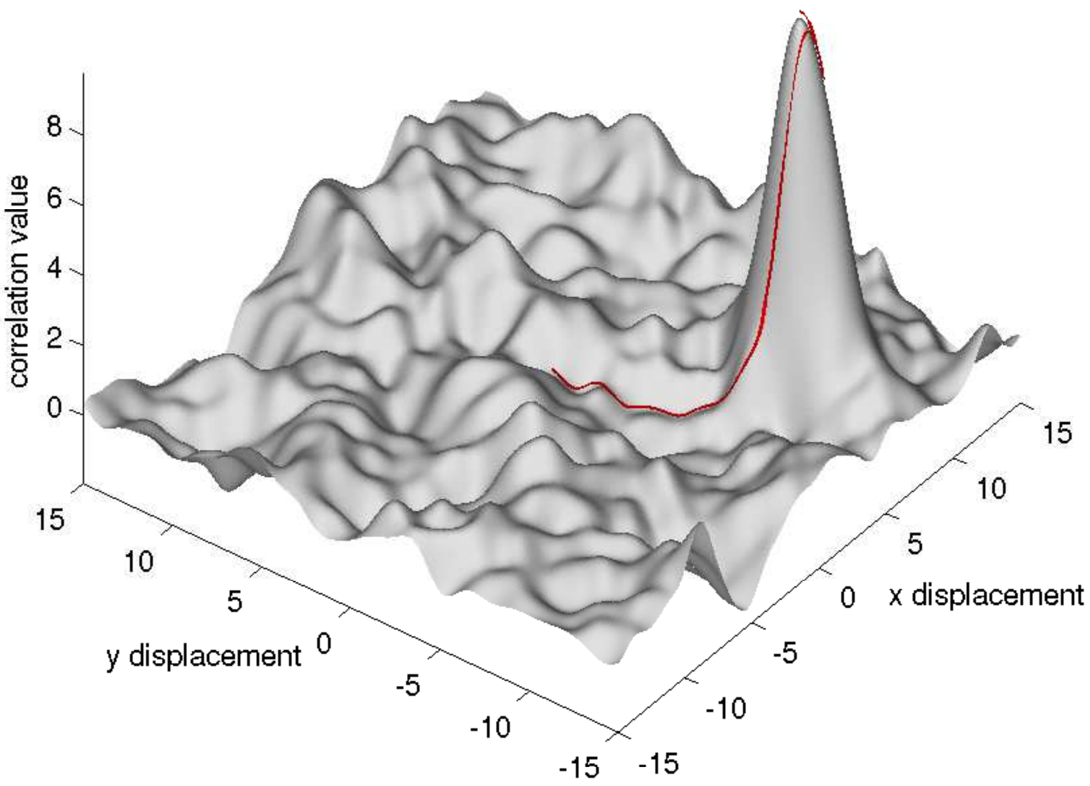


Figure 3.1: Value of the cross-correlation-function, $-C(u, \Sigma, x)$ between two particle images and a path of a gradient-based optimisation approach (red line). Multiscale techniques are employed to avoid that the process stops in a local optimum and to find the correlation peak in $(8, -8)$.

Then we define the estimation of the displacement between g_1 and g_2 at position x, using the fixed shape parameter Σ, by the following optimisation problem:

$$\begin{aligned}&\min_{u\in\mathbb{R}^2} C(u,\Sigma,x)\\ &\text{with } C(u,\Sigma,x) := -\int_{\mathbb{R}^2} w(y-x,\Sigma)\, g_1\left(y-\frac{1}{2}u\right) g_2\left(y+\frac{1}{2}u\right) \mathrm{d}y\ .\end{aligned} \tag{3.1}$$

The objective function is highly non-convex and has many local optima, which have to be overcome by a numerical approach, see Fig. 3.1. However, we are not only interested in the displacement at a single position but search for a global vector field estimation on the whole image or parts of it, denoted as Ω_V. We will represent the displacement field by

$$\mathbf{u} \in \mathcal{U} := \left\{ \mathbf{u} : \Omega_\mathrm{V} \mapsto \mathbb{R}^2 \,\middle|\, \int_{\Omega_\mathrm{V}} \|\mathbf{u}(x)\|_2^2 \,\mathrm{d}x < \infty \right\}\ ,$$

and only require it to be square integrable. Furthermore, we define the window parameters likewise:

$$\mathbf{\Sigma} \in \mathcal{S} := \left\{ \mathbf{\Sigma} : \Omega_\mathrm{V} \mapsto S \,\middle|\, \int_{\Omega_\mathrm{V}} \|\mathbf{\Sigma}(x)\|^2_{\mathbf{\Sigma}} \,\mathrm{d}x < \infty \right\} , \tag{3.2}$$

where S is the set of allowed window shapes and $\|\cdot\|_{\mathbf{\Sigma}}$ is some appropriate norm, depending on the parametrisation of w. Let us assume for a moment, that the window shapes $\mathbf{\Sigma}$ are known and fixed during the global optimisation for $\mathbf{u}$:

$$\begin{aligned} &\min_{\mathbf{u}\in\mathcal{U}} C(\mathbf{u}, \mathbf{\Sigma}) \\ &\text{with } C(\mathbf{u}, \mathbf{\Sigma}) := \int_{\Omega_\mathrm{V}} C(\mathbf{u}(x), \mathbf{\Sigma}(x), x) \,\mathrm{d}x \end{aligned}$$

In this formulation, displacements can be determined *independently* for each position $x \in \Omega_\mathrm{V}$ due to the lack of spatial dependencies. However, this form has the advantage that prior knowledge on the vector field can directly be incorporated as regularisation terms. This includes physical constraints on the fluid such as incompressibility as described in [16]. Nevertheless, in this work the integration over the correlation window will be the only spatial regularisation.

3.1.2 Weighting Function

In the previously defined variational formulation we employed a weighting function w to control the region on which the displacement estimation is based on. A simple choice in the spirit of the template region T used in Sect. 2.2 would be the characteristic function of some region T, i.e.

$$w(x, T) = \chi_T(x) := \begin{cases} 1 & \text{if } x \in T \\ 0 & \text{if } x \notin T \end{cases} ,$$

which includes square windows as used in Sect. 2.4.

However, in this work we employ a Gaussian weighting function for this purpose. Its size, anisotropy and orientation can be freely controlled by only few parameters but it is flexible enough to adapt to situations with high motion gradients of any orientation.

Definition

In our approach we define the window function to be a non-normalised Gaussian function,

$$\begin{aligned} &w : \mathbb{R}^2 \times \mathcal{S}^2_{++} \mapsto (0, 1] , \\ &w(x, \Sigma) := \exp\left(-\frac{1}{2} x^\top \Sigma^{-1} x\right) = A \cdot \mathrm{G}(0, \Sigma)(x) , \\ &\text{with } A := \int_{\mathbb{R}^2} \mathrm{G}(0, \Sigma)(x) \,\mathrm{d}x = 2\pi\sqrt{\det(\Sigma)} . \end{aligned} \tag{3.3}$$

Its shape is steered by a positive definite symmetric 2×2-matrix Σ. Thus, the set of allowed parameters S is a subset of $\mathcal{S}^2_{++}$. A few possible variants are depicted in Fig. 3.2. The associated norm used in (3.2) is the Frobenius norm, i.e. $\|\Sigma\|_{\mathbf{\Sigma}} = \|\Sigma\|_\mathrm{F}$.

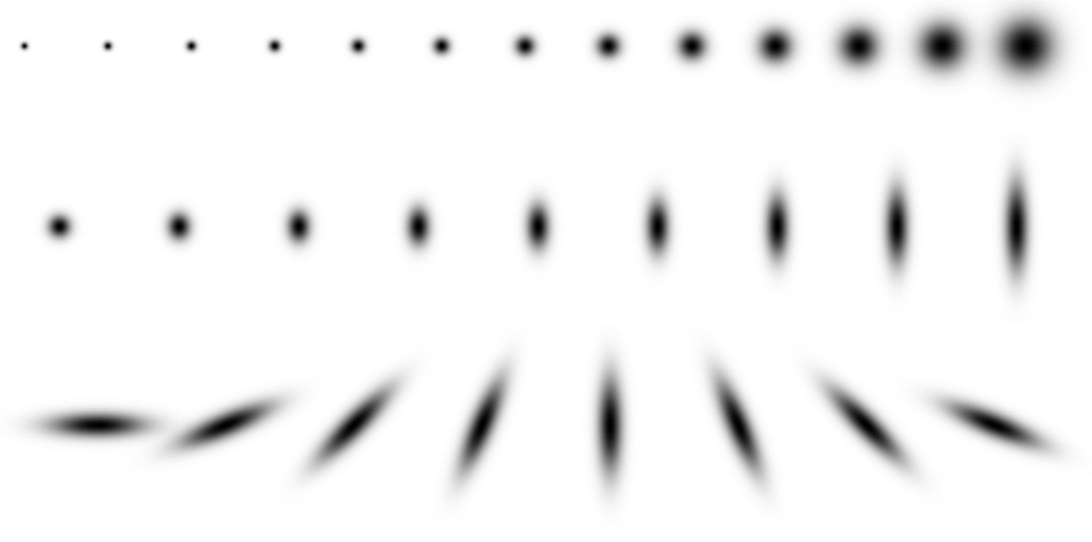


Figure 3.2: Some possible shapes of the Gaussian weighting function as grey-value plot (white=0, black=1), from top to bottom row: varying size, anisotropy (eccentricity) and orientation

Properties

The weighting function reaches its maximum value of one for $x = 0$, where also its centre of gravity,

$$\int_{\mathbb{R}^2} x \; w(x, \Sigma) \, \mathrm{d}x = 0 \; ,$$

is located. As $w(x, \Sigma)$ does not vanish for any x, the weighting function has an infinite support and we cannot define a clear boundary such as it is possible for square or round windows, see Fig. 3.3. However, for visualisation we consider the contour line for height $L \in (0, 1]$:

$$\mathcal{C}_L(\Sigma) := \left\{ x \in \mathbb{R}^2 \, \middle| w(x, \Sigma) = L \right\} = \left\{ x \in \mathbb{R}^2 \, \middle| x^\top \Sigma^{-1} x = -2 \log L \right\} \tag{3.4}$$

We refer to Fig. 3.4 for an illustration. In order to derive an explicit description of $\mathcal{C}_L(\Sigma)$, we represent the window parameter using spectral decomposition:

$$\Sigma = QDQ^\top \tag{3.5}$$

with $Q = \big(q_1, q_2\big) \in \mathbb{R}^{2\times 2}$ and orthogonal, i.e. $Q^\top Q = QQ^\top = I$, composed of the eigenvectors q_1, q_2 of Σ, and the eigenvalues $0 < \lambda_1 \leq \lambda_2$ arranged as $D = \operatorname{diag}(\lambda_1, \lambda_2)$. Then the contour is an ellipse described by

$$\mathcal{C}_L(\Sigma) = \{ v_1 \cos(\phi) + v_2 \sin(\phi) \, | \phi \in [0, 2\pi) \}$$

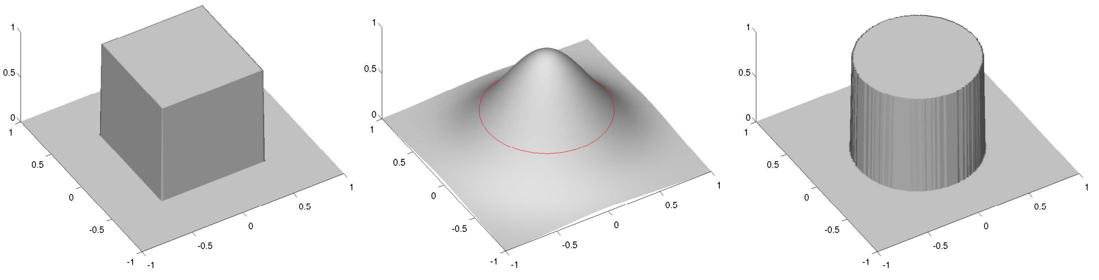

(a) square window (b) Gaussian window (c) round window

Figure 3.3: Three weighting functions with identical integral value $\int_{\mathbb{R}^2} w(x)\,\mathrm{d}x{=}1$: (a) Square window, as usually used in correlation techniques. (b) Gaussian window with infinite support, as used by our approach; the contour line $\mathcal{C}_L(\Sigma)$ (in red) with $L = \exp(-1)$ defines the boundary of a (c) round window with radius $\pi^{-1/2} \approx 0.564$.

where the semiminor axis v_1 and semimajor axis v_2 are defined as $v_i := q_i\sqrt{-2\lambda_i \log L}$. For this curve we can define some more descriptive shape measurements:

$$\text{semiminor } (i=1) \text{ and semimajor } (i=2) \text{ radii } r_i(\Sigma) := \|v_i\|_2 = \sqrt{-2\lambda_i \log L}$$

$$\text{anisotropy (eccentricity) } \varepsilon(\Sigma) := \sqrt{1-\frac{r_1^2}{r_2^2}} = \sqrt{1-\frac{\lambda_1}{\lambda_2}}$$

$$\text{orientation of the semimajor axis } \alpha(\Sigma) := \tan^{-1}\left(\frac{q_{2,2}}{q_{2,1}}\right)$$

$$\text{bounded area } A(\Sigma) := \pi r_1 r_2 = 2\pi\sqrt{\lambda_1\lambda_2}\log L$$

The measurements are also illustrated in Fig. 3.5.

Vice versa, a window shape parameter can be constructed from this parameters, e.g.

$$\begin{aligned} \Sigma(r,\varepsilon,\alpha) &:= QDQ^\top \\ \text{with } D &= \frac{r^2}{-2\log L}\operatorname{diag}\left(\frac{1}{1-\varepsilon^2}, 1\right) \\ \text{and } Q &= \begin{pmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{pmatrix} \end{aligned} \tag{3.6}$$

Whenever we refer to geometric properties of the weighting function $w(x,\Sigma)$ in the remaining work, we implicitly mean the properties of the associated contour function. In most cases we will omit the contour level L and assume $L = \exp(-1)$. Then the area bounded by $\mathcal{C}_L(\Sigma)$ equals the integral of $w(x,\Sigma)$ on $\mathbb{R}^2$:

$$A(\Sigma) = 2\pi\sqrt{\lambda_1\lambda_2} = 2\pi\sqrt{\det\Sigma} = \int_{\mathbb{R}^2} w(x,\Sigma)\,\mathrm{d}x$$

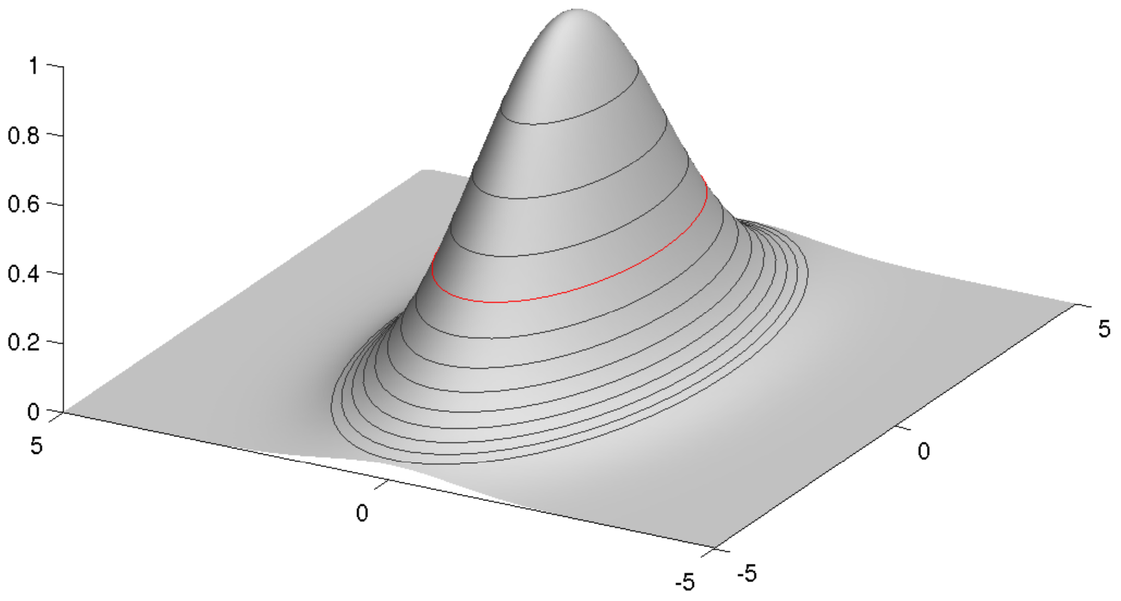


Figure 3.4: Weighting function $w(x, \Sigma)$ with $\Sigma = \operatorname{diag}(4, 1)$ and contour lines $\mathcal{C}_L(\Sigma)$ at level $L = \exp(-k/4)$, $k \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$.

Bounding Boxes

Due to the infinite support of $w(x, \Sigma)$, integrals involving this weighting function have to be evaluated on the whole domain. In order to reduce the computational time, we will use an approximation which reduces evaluation to a region $\mathcal{B}_L(\Sigma)$, which is defined to contain *at least* those positions where w weights the data with L or more, i.e.

$$\mathcal{W}_L(\Sigma) := \left\{x \in \mathbb{R}^2 \,\middle|\, w(x, \Sigma) \geq L\right\} \subset \mathcal{B}_L(\Sigma)$$

In this work we will use $L = 10^{-3}$ for this purpose. The smallest set fulfilling this relation is given by $\mathcal{C}_L(\Sigma)$ merged with its interior. However, for computational simplicity, here we choose the smallest rectangular shaped box which contains $\mathcal{W}_L(\Sigma)$, is centred at $x = 0$ and aligned with the x- and y-axes.

For an arbitrary fixed box orientation, we can derive an explicit expression for its size: We parametrise the bounding box by its orthonormal semiaxes e_1 and e_2, i.e. $\|e_1\|_2 = \|e_2\|_2 = 1$ and $e_1^\top e_2 = 0$, and side lengths $l_1, l_2 > 0$. The bounded area is defined as

$$\mathcal{B}_L^\square(\Sigma) := \left\{x \in \mathbb{R}^2 \,\middle|\, \left|x^\top e_i\right| \leq \frac{1}{2} l_i, i \in \{1, 2\}\right\} . \tag{3.7}$$

For briefness, we also introduce the translated version:

$$\mathcal{B}_L^\square(x, \Sigma) := \{x + y \,|y \in \mathcal{B}_L(\Sigma)\}$$

Figure 3.5 illustrates the interrelation between this bounding box and the contour set.

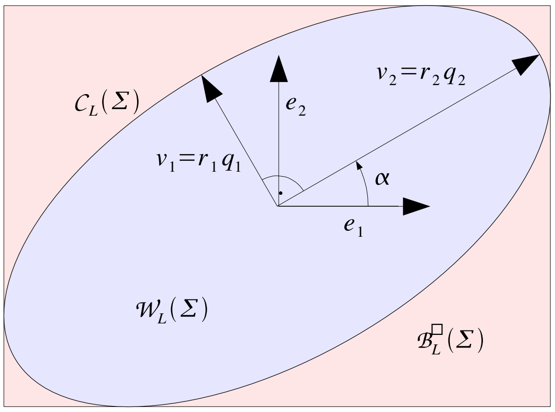


Figure 3.5: Contour line $\mathcal{C}_L(\Sigma)$ with height L of the weighting function $w(x, \Sigma)$, which limits $\mathcal{W}_L(\Sigma)$ (blue), and is described by the semiminor and - major axes v_1, v_2 and orientation α. It is enclosed by the bounding box $\mathcal{B}^{\square}_L(\Sigma)$ (red), with its semiaxes aligned on e_1 and e_2.

As $w(x, \Sigma)$ is continuous and strictly decreasing in x around 0 for Σ positive definite, it is sufficient to ensure that $\mathcal{B}^{\square}_L$ encloses all points on $\mathcal{C}_L(\Sigma)$. Then we can reformulate the search for the smallest window containing $\mathcal{W}_L(\Sigma)$ as an optimisation problem which maximises the shape size with the constraint that the edges of the bounding box intersect with the contour. Due to the orthogonality of e_1 and e_2, the side lengths can be determined independently:

$$l_i = 2 \max_{x \in \mathcal{C}_L(\Sigma)} e_i^\top x \,, \qquad i \in \{1, 2\} \tag{3.8}$$

For obtaining an explicit form, we reformulate the constraint using the spectral decomposition (3.5).

$$x \in \mathcal{C}_L(\Sigma) \quad \Leftrightarrow \quad \left\| \Sigma^{-\frac{1}{2}} x \right\|_2 = \sqrt{-2 \log L} \quad \Leftrightarrow \quad \left\| \frac{1}{\sqrt{-2 \log L}} D^{-\frac{1}{2}} Q^\top x \right\|_2 = 1$$

Due to the non-singularity of Q and D, we can rewrite problem (3.8) as

$$\max_{x \in \mathbb{R}^2} e_i^\top Q D^{\frac{1}{2}} D^{-\frac{1}{2}} Q^\top x \,, \qquad \text{s.t.} \left\| \frac{1}{\sqrt{-2 \log L}} D^{-\frac{1}{2}} Q^\top x \right\|_2 = 1 \,.$$

Substituting $y := \frac{1}{\sqrt{-2 \log L}} D^{-\frac{1}{2}} Q^\top x$ reduces the problem to

$$l_i = 2\sqrt{-2 \log L} \max_{y \in \mathbb{R}^2, \|y\|_2 = 1} e_i^\top Q D^{\frac{1}{2}} y = \sqrt{-8 \log L} \left\| D^{\frac{1}{2}} Q^\top e_i \right\|_2$$

where in the last step we used the fact that $\|x\|_2 = \max_{y \in \mathbb{R}^2, \|y\|_2 = 1} x^\top y$.

3.1.3 Evaluating the Weighted Correlation in Frequency Space

Due to the additional factor $w(x, \Sigma)$ in the objective function (3.1), the convolution theorem cannot be applied directly. However with some modifications it is possible to evaluate the weighted correlation in frequency space.

First note that for *any* $d \in \mathbb{R}^2$ the weighting function can be rewritten as

$$\begin{aligned} w(x, \Sigma) &= \exp\left(-\frac{1}{2} x^\top \Sigma^{-1} x\right) \\ &= \exp\left(-\frac{1}{4}(x-d)^\top \Sigma^{-1}(x-d) - \frac{1}{4}(x+d)^\top \Sigma^{-1}(x+d) + \frac{1}{2} d^\top \Sigma^{-1} d\right) \\ &= w(x-d, 2\Sigma)\, w(x+d, 2\Sigma)\, w(d, -\Sigma)\ . \end{aligned}$$

Using this equivalence (with $d = \frac{1}{2}u$), the weighted correlation can be performed in the frequency domain. Without loss of generality, we set $x = 0$ and get

$$\begin{aligned} C(u, \Sigma, 0) &= -\int_{\mathbb{R}^2} w(y, \Sigma)\, g_1\left(y - \frac{1}{2}u\right) g_2\left(y + \frac{1}{2}u\right) \mathrm{d}y \\ &= -w\left(\frac{1}{2}u, -\Sigma\right) \int_{\mathbb{R}^2} w\left(y - \frac{1}{2}u, 2\Sigma\right) g_1\left(y - \frac{1}{2}u\right) \\ &\quad \cdot w\left(y + \frac{1}{2}u, 2\Sigma\right) g_2\left(y + \frac{1}{2}u\right) \mathrm{d}y\ . \end{aligned}$$

For conciseness we introduce $h_i(x) := w(x, 2\Sigma)\, g_i(x)$ and their Fourier-transformed $\hat{h}_i(\omega) := \mathcal{F}\{h_i\}(\omega)$. Now it becomes clear, that the Gaussian-weighted correlation function can be evaluated in the frequency space:

$$\begin{aligned} C(u, \Sigma, 0) &= w\left(\frac{1}{2}u, -\Sigma\right) \int_{\mathbb{R}^2} h_1\left(y - \frac{1}{2}u\right) h_2\left(y + \frac{1}{2}u\right) \mathrm{d}y \\ &= w\left(\frac{1}{2}u, -\Sigma\right) \int_{\mathbb{R}^2} h_1(-(u-y))\, h_2(y)\, \mathrm{d}y \\ &= w\left(\frac{1}{2}u, -\Sigma\right) \mathcal{F}_\omega^{-1}\left\{\overline{\hat{h}_1(\omega)}\hat{h}_2(\omega)\right\}(u) \end{aligned}$$

However,

$$w\left(\frac{1}{2}u, -\Sigma\right) = \exp\left(\frac{1}{8} u^\top \Sigma^{-1} u\right)$$

is growing exponentially in u and exceeds the finite range of the usual double precision numeric quickly. As it is not clear how to circumnavigate this problem, we do not make use of this relation in this work.

3.2 Window Adaption

When cross-correlation is employed for estimating motion, it is implicitly assumed that the displacements within the considered window are constant. However, this only holds true in very simple cases and leads to estimation errors in areas of larger motion gradients. Then the vector field is smoothed out, because the position of the correlation peak rather represents the average motion within the correlation window, see [5, p. 145] and [24]. In the presence of sinus-shaped vector fields, the peak may even split up [38]. This effects can be avoided by reducing the window size, however at the costs of a smaller supporting area and number of particles and thus a higher influence of image noise [5, Chap. 5.5.6]

In order to improve accuracy by adapting the window shape we find a trade-off between these factors. For this purpose we introduce an error function that approximates the accuracy in the displacement estimation and optimise it with respect to the window shape.

3.2.1 Definition

For any position $x \in \Omega_\mathrm{V}$ we define the function $E(\Sigma, x)$ that models the expected square error of the correlation displacement measurement approach, if a weighting function described by the parameter Σ is used. Then the window adaption scheme consists of finding the shape parameter Σ that minimises this error function:

$$\Sigma(x) \in \arg\min_{\Sigma \in S} \; E(\Sigma, x) \;, \tag{3.9}$$

where S is the set of allowed values, e.g. $S = \mathcal{S}^2_{++}$. Let us assume that a fixed vector field $\mathbf{u} \in \mathcal{U}$ is known a priori. We include it in the definition of an extended version of the error function,

$$E\left(\Sigma, \mathbf{u}, x\right) := E_\mathrm{homog}\left(\Sigma, \mathbf{u}, x\right) + E_\mathrm{noise}\left(\Sigma\right) \;. \tag{3.10}$$

The two terms basically describe the error caused by inhomogeneities and gradients in the vector field (E_homog) and the influence of image noise (E_noise). The two functions are defined and discussed below.

For every position in $x \in \Omega_\mathrm{V}$ the window shape is chosen according to the error measurement. For compactness we define a global optimisation problem in the variable $\boldsymbol{\Sigma} \in \mathcal{S}$ as

$$\begin{aligned} &\min_{\boldsymbol{\Sigma} \in \mathcal{S}} E(\boldsymbol{\Sigma}, \mathbf{u}) \\ &\text{with } E(\boldsymbol{\Sigma}, \mathbf{u}) := \int_{\Omega_\mathrm{V}} E(\boldsymbol{\Sigma}(x), \mathbf{u}, x)\,\mathrm{d}x \;. \end{aligned}$$

This formulation opens up the opportunity to incorporate additional spatial regularisation terms (e.g. smoothness) on the window shape field, however none are used here.

In Fig. 3.6 a simple vector field and some adapted windows are depicted to illustrate the approach.

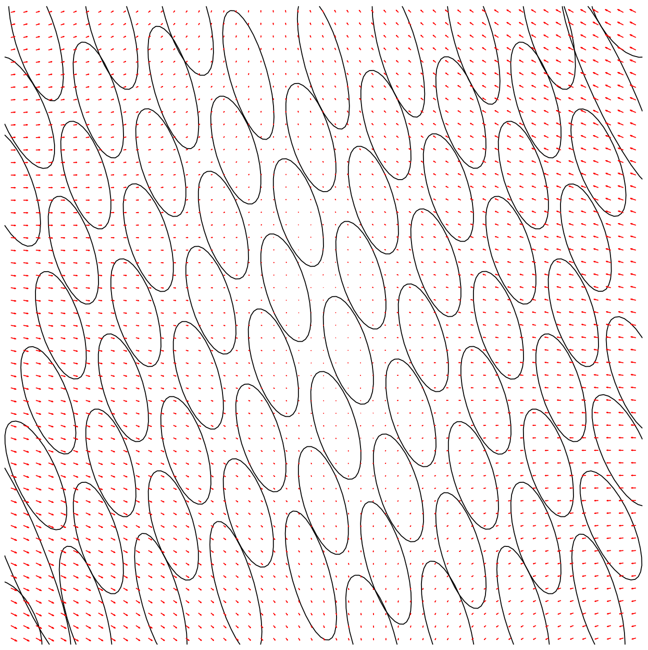


Figure 3.6: A synthetic displacement field (arrows, in red) and some of the adapted windows (represented by their contours $\mathcal{C}(\Sigma)$, in black).

Homogeneity Term

The homogeneity term measures the error caused by the deviation from the assumption that the motion within the window is homogeneous, i.e. $\mathbf{u} = \text{const}$.

$$E_{\text{homog}}(\Sigma, \mathbf{u}, x) := \int_{\mathbb{R}^2} w(y - x, \Sigma)\, e(x, y, \mathbf{u})\, \mathrm{d}y$$

$$e(x, y, \mathbf{u}) := \begin{cases} \|\mathbf{u}(y) - \mathbf{u}(x)\|_2^2 & \text{if } y \in \Omega_{\text{V}} \\ e_{\text{outside}}^2 & \text{otherwise} \end{cases}$$

For positions outside the definition range of $\mathbf{u}$, we assume a constant error e_{outside}. Thus, the window adaption approach will avoid incorporating areas outside Ω_{V}. By choosing the domain accordingly, it is possible to automatically adapt the windows to obstacles and image boundaries where no image data is available.

Noise Term

The noise term describes the influence of image noise and unpaired particles on the estimation error. It is derived from the assumption that the displacement u results from a weighted least-square fit of independent measurements $\mathbf{u}(x)$ in a square neighbourhood $A \subset \mathbb{R}^2$ around 0, e.g. a bounding box as described in Sect. 3.1.2:

$$\begin{aligned} u &:= \arg\min_{u \in \mathbb{R}^2} \int_A w(x,\Sigma)\, \|u - \mathbf{u}(x)\|_2^2 \, \mathrm{d}x \\ &= \frac{\int_A w(x,\Sigma)\mathbf{u}(x)\,\mathrm{d}x}{\int_A w(x,\Sigma)\,\mathrm{d}x} = \frac{\int_A \mathrm{G}(0,\Sigma)(x)\,\mathbf{u}(x)\,\mathrm{d}x}{\int_A \mathrm{G}(0,\Sigma)(x)\,\mathrm{d}x} \end{aligned} \tag{3.11}$$

The noise term shall only describe the effect of disturbances in the image data, but not the error caused by gradients in the vector field. Hence, we assume the measured displacements to represent the true value u^* but disturbed by Gaussian noise, i.e. $\mathbf{u}(x) \sim \mathcal{N}\left(u^*, \sigma_{\mathbf{u}}^2 I\right)$. Then we define the noise term to be the expected square deviation from the true solution, i.e.

$$\mathcal{E}\left\{\|u - u^*\|_2^2\right\} . \tag{3.12}$$

In the following, we will derive a closed form approximation of this term. To this end we describe the least-square estimation (3.11) as a Riemann integral:

$$\begin{aligned} u &= \frac{\int_A \mathrm{G}(0,\Sigma)(x)\,\mathbf{u}(x)\,\mathrm{d}x}{\int_A \mathrm{G}(0,\Sigma)(x)\,\mathrm{d}x} = \lim_{n\to\infty} u_n \\ \text{with } u_n &:= \frac{\sum_{i=1}^{N_n} |A_{ni}| \mathrm{G}(0,\Sigma)(x_{ni})\,\mathbf{u}(x_{ni})}{\sum_{i=1}^{N_n} |A_{ni}| \mathrm{G}(0,\Sigma)(x_{ni})} = \frac{\sum_{i=1}^{N_n} w_{ni}\mathbf{u}(x_{ni})}{\sum_{i=1}^{N_n} w_{ni}} \end{aligned}$$

The integration domain was decomposed into $N_n := n^2$ disjoint sets, so $A = \bigcup_{i=1}^{N_n} A_{ni}$. For simplicity, we can choose a uniform size, i.e. $|A_{ni}| = \frac{|A|}{N_n}$, e.g. by using square sub-regions. Furthermore, the sampling points are chosen as $x_{ni} \in A_{ni}$. For readability we introduce $w_{ni} := |A_{ni}|\mathrm{G}(0,\Sigma)(x_{ni})$.

Now the estimated displacement u_n is a linear combination of normally distributed variables and thus is normally distributed as well, more precisely with $u_n \sim \mathcal{N}(\mu_{u_n}, \Sigma_{u_n})$. Using e.g. [39, (333)], we can state the distribution parameters explicitly:

$$\begin{aligned} \mu_{u_n} &:= \mathcal{E}\{u_n\} = \frac{\sum_{i=1}^{N_n} w_{ni} u^*}{\sum_{i=1}^{N_n} w_{ni}} = u^* \\ \Sigma_{u_n} &:= \mathcal{E}\left\{(u_n - \mu_{u_n})(u_n - \mu_{u_n})^\top\right\} = \frac{\sum_{i=1}^{N_n} w_{ni}^2}{\left(\sum_{i=1}^{N_n} w_{ni}\right)^2} \sigma_{\mathbf{u}}^2 I \end{aligned} \tag{3.13}$$

For any required accuracy, the discretisation can be chosen fine enough, i.e. $a^2 := \frac{|A|}{N_n}$ small enough, such that the sum expressions in (3.13) approximate the corresponding integrals sufficiently well. Furthermore, the integral domain A can always be chosen as large as necessary to approximate the integral $\int_{\mathbb{R}^2} \mathrm{G}(0,\Sigma)(x)\,\mathrm{d}x$ with a predefined (non-zero) residual error. Then the denominator of (3.13) simplifies to

$$\left(\sum_{i=1}^{N_n} w_{ni}\right)^2 = \left(\sum_{i=1}^{N_n} |A_{ni}|\mathrm{G}(0,\Sigma)(x_{ni})\right)^2 \approx \left(\int_A \mathrm{G}\Big(0, \frac{1}{2}\Sigma\Big)(x)\,\mathrm{d}x\right)^2 \approx 1 .$$

Likewise, and using $(\mathrm{G}(0,\Sigma)(x))^2 = \frac{1}{4\pi\sqrt{\det\Sigma}}\mathrm{G}\big(0,\frac{1}{2}\Sigma\big)(x)$, the enumerator can be approximated by

$$\begin{aligned}\sum_{i=1}^{N_n} w_{ni}^2 = \sum_{i=1}^{N_n} (|A_{ni}|\mathrm{G}(0,\Sigma)(x_{ni}))^2 &= \frac{1}{4\pi\sqrt{\det\Sigma}}\frac{|A|}{N_n}\sum_{i=1}^{N_n}\frac{|A|}{N_n}\mathrm{G}\left(0,\frac{1}{2}\Sigma\right)(x_{ni}) \\ &\approx \frac{a^2}{4\pi\sqrt{\det\Sigma}}\int_A \mathrm{G}\left(0,\frac{1}{2}\Sigma\right)(x)\,\mathrm{d}x \approx \frac{a^2}{4\pi\sqrt{\det\Sigma}}\ ,\end{aligned}$$

and we get

$$\Sigma_{u_n} \approx \frac{a^2\sigma_{\mathbf{u}}^2}{4\pi\sqrt{\det\Sigma}}I = \frac{\sigma^2}{4\pi\sqrt{\det\Sigma}}I\ .$$

For convenience, we combine the discretisation parameter a and the noise level $\sigma_{\mathbf{u}}$ into $\sigma := a\sigma_{\mathbf{u}}$. With this result we can approximate the expected estimation error (3.12) by

$$\mathcal{E}\left\{\|u-u^*\|_2^2\right\} \approx \mathcal{E}\left\{\|u_n-u^*\|_2^2\right\} = \operatorname{tr}(\Sigma_{u_n}) = \operatorname{tr}\left(\frac{\sigma^2}{4\pi\sqrt{\det\Sigma}}I\right) = \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}}\ ,$$

and finally use this expression to define the noise term as

$$E_{\text{noise}}(\Sigma) := \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}}\ .$$

The denominator is exactly the support of the window function (3.3), and thus the term increases with smaller window sizes. σ is the only parameter and describes the influence of experiment conditions and the actual discretisation on the expected error.

3.2.2 Window Adaption in Presence of Affine Flows

Real displacement fields are complex and the optimal window shapes with respect to the defined error model cannot be written in closed form. However, for vector fields that can be described by an affine function in the coordinates, i.e.

$$\mathbf{u}(x) = Ax + b\ , \qquad \text{with } A \in \mathbb{R}^{2\times 2},\ b \in \mathbb{R}^2$$

and A non-singular, we can give an explicit description for the window shape resulting from the optimisation problem (3.9). Nevertheless, this special case can be considered as approximations of parts of real vector fields and we can obtain some insights how the window adaption will work in practice.

In addition to the affine form we assume $\Omega_{\mathrm{V}} = \mathbb{R}^2$ which is a good approximation of the general case when the window centre x is sufficiently far from the boundaries of Ω_{V}.

Then the error function simplifies to

$$\begin{aligned}
E(\Sigma, \mathbf{u}, x) &= \int_{\mathbb{R}^2} w\left(y - x, \Sigma\right) \cdot \|\mathbf{u}(y) - \mathbf{u}(x)\|_2^2 \,\mathrm{d}y + \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}} \\
&= \int_{\mathbb{R}^2} w\left(y - x, \Sigma\right) \cdot \|A(y - x)\|_2^2 \,\mathrm{d}y + \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}} \\
&= \int_{\mathbb{R}^2} w\left(y, \Sigma\right) \cdot \|Ay\|_2^2 \,\mathrm{d}y + \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}} \\
&= 2\pi\sqrt{\det\Sigma} \int_{\mathbb{R}^2} \mathrm{G}(0, \Sigma)(y) \cdot \|Ay\|_2^2 \,\mathrm{d}y + \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}} \\
&= 2\pi\sqrt{\det\Sigma} \int_{\mathbb{R}^2} \mathrm{G}\left(0, A\Sigma A^\top\right)(y) \cdot \|y\|_2^2 \,\mathrm{d}y + \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}} \\
&= 2\pi\sqrt{\det\Sigma}\,\mathrm{tr}\left(A\Sigma A^\top\right) + \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}} \;.
\end{aligned}$$

Note, that in the case of affine flows, the error function does not depend on the window centre coordinates x nor on the constant part b of $\mathbf{u}(x)$, but on A.

The gradient in Σ vanishes if

$$\begin{aligned}
\nabla_\Sigma E(\Sigma, \mathbf{u}, x) &= 2\pi\left(\frac{1}{2}\sqrt{\det\Sigma}\,\mathrm{tr}\left(A\Sigma A^\top\right)\Sigma^{-1} + \sqrt{\det\Sigma}\left(A^\top A\right)\right) - \frac{1}{2}\frac{\sigma^2}{2\pi\sqrt{\det\Sigma}}\Sigma^{-1} \\
&= 2\pi\sqrt{\det\Sigma}\left(A^\top A\right) + \frac{1}{2}\left(2\pi\sqrt{\det\Sigma}\,\mathrm{tr}\left(A\Sigma A^\top\right) - \frac{\sigma^2}{2\pi\sqrt{\det\Sigma}}\right)\Sigma^{-1} = 0 \\
\Leftrightarrow &\frac{1}{2}\left(\frac{\sigma^2}{(2\pi)^2\det\Sigma} - \mathrm{tr}\left(A\Sigma A^\top\right)\right)\Sigma^{-1} = A^\top A \;.
\end{aligned}$$

As A and Σ are non-singular by assumption, it is sufficient to set $\Sigma = \left(\alpha A^\top A\right)^{-1}$, and to find $\alpha \in \mathbb{R}$, such that the equation is fulfilled:

$$\begin{aligned}
0 &= \frac{1}{2}\left(\frac{\sigma^2}{(2\pi)^2\det\Sigma} - \mathrm{tr}\left(A\Sigma A^\top\right)\right)\Sigma^{-1} - A^\top A \\
&= \frac{1}{2}\left(\alpha^3\frac{\sigma^2}{(2\pi)^2}\det\left(A^\top A\right) - 4\right)\left(A^\top A\right)
\end{aligned}$$

Thus, the scalar term has to vanish:

$$\alpha^3\frac{\sigma^2}{(2\pi)^2}\det\left(A^\top A\right) - 4 = 0 \quad\Leftrightarrow\quad \alpha = \left(\frac{\sigma^2}{(4\pi)^2}\det\left(A^\top A\right)\right)^{-\frac{1}{3}}$$

Finally, the window shape parameter, for which the error model function takes a local extreme point, is

$$\Sigma^*(A, \sigma) = \left(\frac{\sigma^2}{(4\pi)^2}\det\left(A^\top A\right)\right)^{\frac{1}{3}}\left(A^\top A\right)^{-1} \;,$$

which is always positive definite. Note, that $E(\Sigma, \mathbf{u}, x)$ is continuous and bounded below by 0. The noise term goes to infinity at the boundary of the set $\mathcal{S}$ where the eigenvalues approach zero. On the other hand, the homogeneity term grows without bounds with increasing eigenvalues of Σ. As Σ^* is the only local optimum, it is therefore a global optimal solution.

It appears that in the case of an affine vector field, the adapted window shapes only depend on σ and the *inner product* of the Jacobian $\mathrm{J}_x\,\mathbf{u}(x) = A$, i.e. $A^\top A$. It is invariant against constant terms b and orthogonal mappings $U \in \mathbb{R}^{2\times 2}$, $U^\top U = I$ (e.g. rotations) of the vectors, because $\Sigma^*(UA, \sigma) = \Sigma^*(A, \sigma)$.

We consider the special case where the vector field is defined as $A := \operatorname{diag}(a_1, a_2)$ (and arbitrary b). Then, the optimal window shape is

$$\Sigma^* := \Sigma^*(A,\sigma) = \left(\frac{\sigma^2}{(4\pi)^2} a_1^2 a_2^2\right)^{\frac{1}{3}} \operatorname{diag}\left(a_1^{-2}, a_2^{-2}\right) = \left(\frac{\sigma}{4\pi}\right)^{\frac{2}{3}} \operatorname{diag}\left(a_1^{-2} a_2, a_1 a_2^{-2}\right)^{\frac{2}{3}} \ ,$$

or expressed in radii:

$$\begin{aligned} r_1(\Sigma^*) &= r_0 \left(a_1^{-2} a_2\right)^{\frac{1}{3}} \ , \\ \text{and } r_2(\Sigma^*) &= r_0 \left(a_2^{-2} a_1\right)^{\frac{1}{3}} \ , \\ \text{with } r_0 &:= \left(-2 \left(\frac{\sigma}{4\pi}\right)^{\frac{2}{3}} \log L\right)^{\frac{1}{2}} \ . \end{aligned}$$

For very homogeneous vector fields, the window grows infinitely large,

$$\lim_{a_1,a_2\to 0} r_1 = \lim_{a_1,a_2\to 0} r_2 = \infty$$

while in regions of high vector gradients we have infinite small windows,

$$\lim_{a_1,a_2\to \infty} r_1 = \lim_{a_1,a_2\to \infty} r_2 = 0 \ .$$

The vector field in Fig. 3.6 is affine, and the optimal windows are the same everywhere except for the boundaries. In the experiments presented in Sect. 3.5.3 we consider further simple synthetic vector fields. The resulting adapted window shapes can be explained by means of the results of this consideration.

3.2.3 Extensions

As discussed in the previous section, the optimisation criteria may lead to windows with extreme shapes. However, very large windows are computational expensive and do not increase accuracy significantly. In contrast, very small windows may not include enough particles and thus lead to inaccurate estimations. For this reason we allow to limit the window size by the bounding radii $r_{\min}$ and $r_{\max}$, and add constraints to the window adaption, such that

$$0 < r_{\min} \leq r_1(\Sigma), r_2(\Sigma) \leq r_{\max}$$

holds. It is sufficient to replace the constraint set S in the optimisation problem (3.9) by

$$S = \left\{\Sigma \in \mathcal{S}^2 \,\middle|\, \lambda_{\min} I \leq \Sigma \leq \lambda_{\max} I \right\} \tag{3.14}$$

with eigenvalue bounds

$$\lambda_{\min} := -\frac{1}{2\log L} r_{\min}^2 \qquad \text{and } \lambda_{\max} := -\frac{1}{2\log L} r_{\max}^2 \, .$$

The set S is convex and – due to $0 < r_{\min}$ – consists only of positive definite matrices, i.e. $S \subset \mathcal{S}^2_{++}$.

The error model (3.10) considered in this work is based on some basic observations in the context of window adaption methods for correlation approaches. However, it can easily be extended and refined to involve further knowledge about the influence of other (measurable) experimental parameters on the expected measurement error. This might be models based on physical considerations as well as statistics from real or simulated experiments. For example, it is possible to incorporate non-homogeneous noise levels or particle seeding density into our approach by making the parameter σ coordinate-dependent.

3.3 Joint Motion Estimation and Window Adaption

In Sect. 3.1 we described a motion estimation approach but assumed that the window shape parameters are fixed. Then, in Sect. 3.2, we proposed a criterion for the choice of the window shape which, however, relies on the knowledge of a complete vector field. Let us describe this chicken-and-egg-dependencies in words of a mathematical tractable formulation:

$$\mathbf{u} \in \arg\min_{\mathbf{u}\in\mathcal{U}} C(\mathbf{u}, \mathbf{\Sigma}) \tag{3.15}$$

$$\text{and } \mathbf{\Sigma} \in \arg\min_{\mathbf{\Sigma}\in\mathcal{S}} E(\mathbf{\Sigma}, \mathbf{u}) \, . \tag{3.16}$$

The top-level optimisation estimates the displacements $\mathbf{u}$ at all positions x in the variable domain by maximising the correlation terms. The window shapes are adapted in the underlying optimisation problems which constrain each $\mathbf{\Sigma}(x)$ to an optimum of the error model function E and again depend on $\mathbf{u}$. Thus, we have two strongly connected minimisation problems which have to be solved jointly. Both subproblems have non-linear and non-convex objective functions.

Note that (3.15) optimises the energy function only with respect to $\mathbf{u}$, but not to $\mathbf{\Sigma}$. The reason is, that the choice of the window shapes should only be steered by the error function and *not* by the correlation measurement between the image patches. If required, data-based information on the expected accuracy should be incorporated into the error model function instead.

3.4 Discretisation and Optimisation

The defined optimisation problem (3.15)-(3.16) is

$$\mathbf{u} \in \arg\min_{\mathbf{u}\in\mathcal{U}} C(\mathbf{u},\mathbf{\Sigma}) \qquad \text{and } \mathbf{\Sigma} \in \arg\min_{\mathbf{\Sigma}\in\mathcal{S}} E(\mathbf{\Sigma},\mathbf{u}) \; .$$

A number of carefully chosen approximations and relaxations are applied with the aim to make the optimisation problem tractable.

In Sect. 3.4.1 we start with the description of the variable discretisation which is based on piecewise linear finite elements. Moreover, bounding boxes for the weighting function are introduced and a numerical integration scheme for the involved integrals is defined.

The problem is converted into an unconstrained optimisation problem in Sect. 3.4.2. The minimality criteria are then replaced by stationary conditions. For solving the resulting equalities, we employ a Newton step method with respect to all variables. A multiscale framework improves the quality of the solution by circumnavigating many of the local optima.

3.4.1 Discretisation

Variable Representation

Let $f : \Omega \mapsto \mathbb{R}^d$ be a continuously defined function. Then a finite element approximation [40] of f is defined by

$$f(x) \approx \sum_{x_i \in X} \varphi_i(x)\eta_i \; ,$$

where $x_i \in X$ are the locations of the control points, $\varphi_i(x)$ and η_i are the corresponding basis functions and weights for position x_i, respectively. In our work, we use piecewise linear finite elements, defined on a regular grid with spacing a and origin x_0.

$$X(a) := \left\{ x_i \,\middle|\, i \in \mathbb{Z}^2 \right\} \; , \qquad \text{with } x_i := x_0 + ai$$

$$\text{and basis functions } \varphi_i(y) := \begin{cases} \left(1 - \frac{|y_1 - x_{i,1}|}{a}\right)\left(1 - \frac{|y_2 - x_{i,2}|}{a}\right) & \text{if } y \in [x_i - a, x_i + a] \\ 0 & \text{otherwise} \end{cases}$$

Due to

$$\varphi_i(x_j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

the weightings are simplify given by $\eta_i = f(x_i)$.

In addition, we denote a grid $X(a)$ restricted to some domain A as

$$X(a, A) := X(a) \cap A \; .$$

The functions $\mathbf{u}$ and $\mathbf{\Sigma}$ are discretised component-wise on a regular grid with spacing a_{V}, typically chosen coarser than the discretisation grid of the image data. In addition it is restricted to Ω_V and abbreviated as $X_\mathrm{V} := X(a_\mathrm{V}, \Omega_\mathrm{V})$. Then the functions are approximated as

$$\begin{aligned} \mathbf{u}(x) &\approx \sum_{x_i \in X_\mathrm{V}} \varphi_i(x)\, \mathbf{u}_i \;, \\ \text{and } \boldsymbol{\Sigma}(x) &\approx \sum_{x_i \in X_\mathrm{V}} \varphi_i(x)\, \boldsymbol{\Sigma}_i \;, \end{aligned}$$

where $\mathbf{u}_i := \mathbf{u}(x_i)$ and $\boldsymbol{\Sigma}_i := \boldsymbol{\Sigma}(x_i)$ are the corresponding discrete variables. Note, that it is possible to choose arbitrary, e.g. irregular, grids that adapt to the seeding density as it is used in [41].

Image Data Representation

The discrete input images are normalised separately so their pixel-wise mean is zero and the standard deviation is one. We store the grey-value information in a cubic spline representation [42] and assume zero values outside the original definition range. The spline representation yields two continuously defined functions,

$$g_1, g_2 : \mathbb{R}^2 \mapsto \mathbb{R} \;,$$

which are two times continuously derivable everywhere. The spline bases are arranged on a regular grid with spacing a_D.

An efficient implementation [43] of the spline interpolation allows to evaluate the function value g_i, its gradient ∇g_i and second derivatives $\mathrm{H}\, g_i = (\nabla \nabla^\top) g_i$ everywhere on $\mathbb{R}^2$.

Numerical Integration

The integrals within the continuous definitions of the correlation and error model functional are approximated using the Newton-Cotes-formula (trapezoid rule). This is equivalent to approximating the integrand using finite elements as introduced in Sect. 3.4.1:

$$\int_\Omega f(y)\,\mathrm{d}y \approx \int_{\mathbb{R}^2} \sum_{x_i \in X(a) \cap \Omega} f(x_i) \varphi_i(y)\,\mathrm{d}y = \sum_{x_i \in X(a,\Omega)} A_i(a,\Omega) f(x_i)$$

The constant weights are defined as

$$A_i(a,\Omega) := \int_\Omega \varphi_i(y)\,\mathrm{d}y$$

for variable i, domain Ω and a regular grid with spacing a. Note that $A_i(a,\Omega)$ equals a^2 almost everywhere, except for the regions near the boundaries of Ω.

For the integrals involved in C and E the control points are chosen appropriately to the underlying variable discretisation, i.e. on X_V.

$$C(\mathbf{u},\mathbf{\Sigma}) = \int_{\Omega_\mathrm{V}} C(\mathbf{u}(x),\mathbf{\Sigma}(x),x)\,\mathrm{d}x \approx \sum_{x_i\in X_\mathrm{V}} A_i(a_\mathrm{V},\Omega_\mathrm{V}) C(\mathbf{u}_i,\mathbf{\Sigma}_i,x_i)\ ,$$

$$\text{and } E(\mathbf{\Sigma},\mathbf{u}) = \int_{\Omega_\mathrm{V}} E(\mathbf{\Sigma}(x),\mathbf{u},x)\,\mathrm{d}x \approx \sum_{x_i\in X_\mathrm{V}} A_i(a_\mathrm{V},\Omega_\mathrm{V}) E(\mathbf{\Sigma}_i,\mathbf{u},x_i)$$

The term $E_\mathrm{homog}(\Sigma,\mathbf{u},x)$ (within $E(\Sigma,\mathbf{u},x)$) in turn contains an integral over whole $\mathbb{R}^2$:

$$\begin{aligned} E_\mathrm{homog}(\Sigma,\mathbf{u},x) &= \int_{\mathbb{R}^2} w(y-x,\Sigma)e(x,y,\mathbf{u})\,\mathrm{d}y \\ &\approx \sum_{y_i\in X(a_\mathrm{V},\mathbb{R}^2)} A_i(a_\mathrm{V},\mathbb{R}^2) w(y_i-x,\Sigma)e(x,y_i,\mathbf{u}) \\ &= a_\mathrm{V}^2 \sum_{y_i\in X(a_\mathrm{V})} w(y_i-x,\Sigma)e(x,y_i,\mathbf{u}) \end{aligned}$$

Furthermore, for window sizes that are reasonable in practice, only few samples have a relevant impact on the result. Thus, we introduce a further approximation by limiting the evaluation to regions where the window function $w(w,\Sigma)$ has a reasonable minimum weight

$$E_\mathrm{homog}(\Sigma,\mathbf{u},x) \approx \sum_{y_i\in X(a_\mathrm{V},\mathcal{B}_L(x,\Sigma))} A_i(a_\mathrm{V},\mathcal{B}_L(x,\Sigma)) w(y_i-x,\Sigma)e(x,y_i,\mathbf{u})\ ,$$

where $\mathcal{B}_L(x,\Sigma)$ is a bounding box that contains at least those points y, where $w(y-x,\Sigma)\geq L$, see Sect. 3.1.2 for details.

In the same way we approximate the term $C(u,\Sigma,x)$ which occurs within $C(\mathbf{u},\mathbf{\Sigma})$. However, here we use the (usually finer) grid $X_\mathrm{D} := X(a_\mathrm{D})$ on which the image data is defined, and also employ a bounding box:

$$\begin{aligned} C(u,\Sigma,x) &= -\int_{\mathbb{R}^2} w(y-x,\Sigma)\, g_1\left(y-\frac{1}{2}u\right) g_2\left(y+\frac{1}{2}u\right)\,\mathrm{d}y \\ &\approx -\sum_{y_i\in X(a_\mathrm{D},\mathbb{R}^2)} A_i(a_\mathrm{D},\mathbb{R}^2) w(y_i-x,\Sigma)\, g_1\left(y_i-\frac{1}{2}u\right) g_2\left(y_i+\frac{1}{2}u\right) \\ &\approx -\sum_{y_i\in X(a_\mathrm{D},\mathcal{B}_L(x,\Sigma))} A_i(a_\mathrm{D},\mathcal{B}_L(x,\Sigma)) w(y_i-x,\Sigma)\, g_1\left(y_i-\frac{1}{2}u\right) g_2\left(y_i+\frac{1}{2}u\right) \end{aligned}$$

In the remaining work we only consider the discretised version of the problem and – for readability – denote the involved variables, sets and functions with the same symbols as their original, non-discretised counterparts.

For brevity, we will omit the restriction of the integral evaluation to the bounding boxes. This, however, camouflages the fact that the discretised integrals in C and E

depend non-smoothly on $\mathbf{\Sigma}$. The reason is that the approximating sums change abruptly when points of the data or variable grid leave or join the sets $X_\mathrm{V} \cap \mathcal{B}_L(x,\Sigma)$ and $X_\mathrm{D} \cap \mathcal{B}_L(x,\Sigma)$. By definition of the bounding boxes, the impact of these terms is smaller or equal to the bound L and thus we disregard this error in the remaining description.

3.4.2 Optimisation

Single Scale

The constrained problem defined for the window adaption can be reformulated as an equivalent problem by incorporating the bounding constraints $\Sigma \in S$ as defined in (3.14) into the objective function:

$$E(\Sigma,\mathbf{u},x) + \delta_S(\Sigma) \ ,$$

where $\delta_S(\Sigma)$ is the set indicator function of the set S. However, in this work we approximate the characteristic function by a logarithmic barrier [44, Chapter 9.2],

$$B_S(\Sigma) := -\,\mu \log\det\left(\Sigma - \lambda_{\min} I\right) - \mu \log\det\left(\lambda_{\max} I - \Sigma\right) \ ,$$

with scaling $\mu > 0$, which we chose as $\mu = 10^{-2}$ throughout the work. So the modified window adaption objective reads in the discretised form

$$\begin{aligned} E_S(\Sigma,\mathbf{u},x) &:= E(\Sigma,\mathbf{u},x) + B_S(\Sigma) \ , \\ \text{and} \qquad E_S(\mathbf{\Sigma},\mathbf{u}) &:= \sum_{x_i \in X_\mathrm{V}} A_i(a_\mathrm{V},\Omega_\mathrm{V}) E_S(\mathbf{\Sigma}_i,\mathbf{u},x_i) \end{aligned}$$

Note that due to this approximation the window shapes can never take a value on or very close to the boundary of S, since $B_S(\Sigma)$ tends to infinity there.

Nevertheless, the advantage of introducing the barrier function is that we can now apply methods for continuous optimisation to the modified problem:

$$\mathbf{u} \in \arg\min_{\mathbf{u}\in\mathcal{U}} C(\mathbf{u},\mathbf{\Sigma}) \qquad \text{and } \mathbf{\Sigma} \in \arg\min_{\mathbf{\Sigma}\in\mathcal{S}} E_S(\mathbf{\Sigma},\mathbf{u}) \ . \tag{3.17}$$

A major simplification of the problem is performed by replacing both minimality objectives by the stationary conditions

$$\nabla_{\mathbf{u}_j} C(\mathbf{u},\mathbf{\Sigma}) = 0 \qquad \forall x_j \in X_\mathrm{V} \ , \tag{3.18}$$

$$\nabla_{\mathbf{\Sigma}_j} E_S(\mathbf{\Sigma},\mathbf{u}) = 0 \qquad \forall x_j \in X_\mathrm{V} \ . \tag{3.19}$$

For the solution of the equality system, we employ a Newton step method with respect to all displacement and window shape variables, respectively, to find a combination of variables $\mathbf{u} \in \mathcal{U}$ and $\mathbf{\Sigma} \in \mathcal{S}$ that satisfy these conditions.

However, not all solutions to (3.18)–(3.19) are necessarily a local minimum of the problem (3.17). Local maxima and saddle points also fulfil these conditions. Thus, we extend the Newton step method by a line search method which guarantees that in every step the objective value does not increase.

Note, that although each equality constraint is a non-linear and non-convex functions in all variables, they simplify strongly to

$$\begin{aligned}\nabla_{\mathbf{u}_j} C(\mathbf{u},\boldsymbol{\Sigma}) &= \nabla_{\mathbf{u}_j} \sum_{x_i \in X_\mathrm{V}} A_i(a_\mathrm{V},\Omega_\mathrm{V}) C(\mathbf{u}_i,\boldsymbol{\Sigma}_i,x_i) \\ &= A_j(a_\mathrm{V},\Omega_\mathrm{V}) \nabla_{\mathbf{u}_j} C(\mathbf{u}_j,\boldsymbol{\Sigma}_j,x_j) \ ,\end{aligned}$$

and

$$\begin{aligned}\nabla_{\boldsymbol{\Sigma}_j} E_S(\boldsymbol{\Sigma},\mathbf{u}) &= \nabla_{\boldsymbol{\Sigma}_j} \sum_{x_i \in X_\mathrm{V}} A_i(a_\mathrm{V},\Omega_\mathrm{V}) \left(E(\boldsymbol{\Sigma}_i,\mathbf{u},x_i) + B_S(\boldsymbol{\Sigma}_i)\right) \\ &= A_j(a_\mathrm{V},\Omega_\mathrm{V}) \nabla_{\boldsymbol{\Sigma}_j} \left(E(\boldsymbol{\Sigma}_j,\mathbf{u},x_j) + B_S(\boldsymbol{\Sigma}_j)\right) \ .\end{aligned}$$

From this result, it becomes apparent, that the displacements can be updated independently of each other. The same is true for the window shape parameters. The cause for this independency is that we did not incorporate any spatial regularisation terms for the variables in the definition of C and E. However, the interconnection between the displacement estimation and the window adaption remains.

The iterative update of the displacement and window shape variables is summarised in Algorithm 3.1. The variables associated with iteration k are denoted by $\mathbf{u}^{(k)}$ and $\boldsymbol{\Sigma}^{(k)}$. We measure the mean difference of the variables between two iteration steps and use an upper bound on the change of the components of $\mathbf{u}$ and $\boldsymbol{\Sigma}$ as convergence criterion.

Algorithm 3.2 describes the update of parts of the variable vector, e.g. $\mathbf{u}_i$ in $\mathbf{u}$, where the remaining component are kept fixed. During the calculation of the Newton step direction, the Hessian matrix is modified by adding a (positive) multiple of the identity matrix. This allows the computation of a Newton step even in cases where the Hessian is singular or close to. In addition, the higher the added value λ is, the more large steps are penalised. By choosing different values for the update of displacements ($\lambda_\mathbf{u}$) and window shape parameters ($\lambda_{\boldsymbol{\Sigma}}$), we are able to control the relative update speed of the two variable sets. The following line search scales the calculated Newton step to ensure that the function value does not increase.

Convergence

Our investigation on an existence proof for a fix point of this problem showed that even the relaxed version is still involved. Especially the correlation function is highly non-convex (see Fig. 3.1) and heavily depends on the image data. For the window shape optimisation, we could derive an explicit form of the solution in presence of an affine vector field, see Sect. 3.2.2. However, general vector fields are more complex and make the error model function non-convex.

In [23, 25], the stability of iterative variants of cross-correlation methods is investigated. The authors use an idealised, linear model of the process and analyse its stability by regarding the window weighting function as a filter on the components of the correct vector field. The filter output is used as displacement update and thus the frequency response to motion variations with specific wavelengths determines the convergence.

Algorithm 3.1: Single-scale, iterative update of the displacement and window shape variables. Both are updated independently using the function `variableUpdate` which is defined in Algorithm 3.2.

```
Algorithm: singlescaleSolution ($\mathbf{u}^{\text{init}}, \mathbf{\Sigma}^{\text{init}}$)
Input: initial displacements $\mathbf{u}^{\text{init}}$ and window shapes $\mathbf{\Sigma}^{\text{init}}$
Output: updated variables $\mathbf{u}, \mathbf{\Sigma}$

$k \leftarrow 0$
$\mathbf{u}^{(0)} \leftarrow \mathbf{u}^{\text{init}}$, $\mathbf{\Sigma}^{(0)} \leftarrow \mathbf{\Sigma}^{\text{init}}$
repeat
    /* update displacements */
    foreach $x_i \in X_{\text{V}}$ do
        $\mathbf{u}_i^{(k+1)} \leftarrow$ variableUpdate($C(\mathbf{u}, \mathbf{\Sigma}), \mathbf{u}_i, (\mathbf{u}^{(k)}, \mathbf{\Sigma}^{(k)})$)
    end
    /* update window shapes */
    foreach $x_i \in X_{\text{V}}$ do
        $\mathbf{\Sigma}_i^{(k+1)} \leftarrow$ variableUpdate($E(\mathbf{\Sigma}, \mathbf{u}), \mathbf{\Sigma}_i, (\mathbf{\Sigma}^{(k)}, \mathbf{u}^{(k)})$)
    end
    $k \leftarrow k + 1$
until until convergence
return $\mathbf{u}^{(k)}$, $\mathbf{\Sigma}^{(k)}$
```

They show that the spectrum of rectangular window shapes (see Fig. 3.3(a)) contains 180° phase changes which cause a divergence behaviour in this model. Furthermore, they define a weighting function which, just as the Gaussian function used in our work, does not show this property. This analysis does not respect the window adaption nor the non-linear components of our approach (e.g. line search), but substantiates the choice of the Gaussian weighting function.

The experimental results in Sect. 3.5 also indicated that convergence is not an issue in regions with sufficient image information. However, a theoretical analysis of the convergence of our approach remains as further work.

Multiscale Framework

The Algorithm 3.1 finds a solution, which might be considerably worse than the global optimum, because it gets stuck in a local minimum. Thus, we embed the previously described single scale search into a multiscale framework with the aim to circumnavigate most of the local optima.

The basic idea behind the multiscale framework is to first estimate the optimal variables on a coarse version of the image data that only contains information about the large displacements in the image. The result is then transferred to the next finer level of the resolution pyramid where it acts as initialisation for the refinement of the solution. For an elaborate investigation of multiscale methods for flow estimation, we refer to [45]

Algorithm 3.2: Takes a function $f(x)$ and updates variables y (which might be a subset of x), so the value of function $f(x)$ is reduced, or at least not increased. $x|_y$ denotes the subset of components y of a variable x. A Newton step is determined at $x = x_0$ with respect to y and scaled using line search.

```
Algorithm: variableUpdate ($f$,$y$,$x_0$)
Input: function $f(x)$, variable to update $y$, start point $x_0$
Input: parameters $\lambda, \alpha_{\min}, \alpha_{\max}, \beta$
Output: $x$

/* calculate a Newton step direction */
$g \leftarrow \nabla_y f(x_0)$
$H \leftarrow \mathrm{H}_y\, f(x_0)$
$\Delta y \leftarrow -(H + \lambda I)^{-1} g$

/* perform line search, so that the function value does not increase
   */
$\alpha^* \leftarrow 0$
$\alpha \leftarrow \alpha_{\max}$
$x_\alpha \leftarrow x_0$
while $\alpha \geq \alpha_{\min}$ do
   /* update variable $y$ in $x_0$ */
   $x_\alpha|_y \leftarrow x_0|_y + \alpha \Delta y$
   if $f(x_\alpha) < f(x_0)$ then
      $\alpha^* \leftarrow \alpha$
      break
   end
   $\alpha \leftarrow \beta\alpha$
end
return $x_0|_y + \alpha^* \Delta y$
```

and [46].

The multi-resolution pyramid for the image data g_1 and g_2 is created by recursively resampling the original image data (denoted as level 0) on a coarser level $l > 0$ (denoted as `resampleImage`). The data representation on resolution level l has a grid spacing of $a_\mathrm{D}^{[l]} = s^l a_\mathrm{D}^{[0]}$, where s is the scaling step factor, e.g. $s = 2$ for a dyadic image pyramid. We denote image data associated with scale level l as $g_i^{[l]}$ and use the same convention for all data and variables.

A cubic spline representation is used for interpolating an image defined on a grid $X_\mathrm{D}^{[l]}$ to the next coarser level on grid $X_\mathrm{D}^{[l+1]}$. Aliasing effects caused by the sub-sampling are avoided by applying a binomial low-pass filter [47, 48] in before.

In a similar way, the variables defined on grid X_V are transferred between two different resolutions (`resampleDisplacements`), so $a_\mathrm{V}^{[l]} = s^l a_\mathrm{V}^{[0]}$. The displacement variables $\mathbf{u}$ are interpolated component-wise using cubic spline interpolation. Aliasing is avoided by applying a small binomial low-pass filter in before.

The transfer of the window parameters (`resampleWindowShapes`) is more critical as the constraint $\mathbf{\Sigma} \in \mathcal{S}$ has to be conserved. However, for simple component-wise bilinear interpolation the interpolated value

$$\Sigma_{x,y} = (1-x)(1-y)\Sigma_{0,0} + (1-x)(y)\Sigma_{0,1} + (x)(1-y)\Sigma_{1,0} + (x)(y)\Sigma_{1,1}$$

is a convex combination of the supporting matrices $\Sigma_{0,0}, \Sigma_{0,1}, \Sigma_{1,0}, \Sigma_{1,1} \in S$ for any $x, y \in [0,1]$, because of

$$(1-x)(1-y) + (1-x)(y) + (x)(1-y) + (x)(y) = 1 \ .$$

Thus, the resulting matrix lies within in the convex hull of the reference matrices, which again is a subset of S, and the constraint $\mathbf{\Sigma} \in \mathcal{S}$ is conserved.

A similar argument holds for applying a linear low-pass filters during sub-sampling, as long as the filter coefficients are bounded by $[0,1]$ and sum up to one – which is always true for binomial filters.

The complete multiscale optimisation approach is summarised in Algorithm 3.3.

3.5 Experiments and Discussion

In this section we investigate the properties and performance of the proposed variational correlation approach and the window adaption scheme. Section 3.5.1 describes the choice of some of the algorithm parameters which are used commonly for all experiments. In addition, error measurements and visualisation issues are discussed.

The experiments in Sect. 3.5.2 are dedicated to the question whether the proposed window adaption can improve the accuracy of the correlation approach at all.

In Sect. 3.5.3 we apply the window adaption approach to some simple synthetic vector fields. These experiments give some valuable insights which can be transferred to more complex situations which we will meet in the remaining section.

The performance of the adaptive displacement estimation is verified in Sect. 3.5.4. Three synthetic scenarios are evaluated which were designed for the *PIV Challenge 2005* to compare the accuracy of the participating velocity measurement algorithms.

Finally, we apply our approach to an image pair obtained from a real PIV experiment. We compare our solution to the vector field measured by a commercial implementation of the cross-correlation approach. The results are presented in Sect. 3.5.5.

3.5.1 General Experiment Setup and Visualisation

Optimisation: In most cases, we used a scaling factor of $\sqrt{2}$, so the level scales are

$$\{1, 2^{1/2}, \dots, 2^{l_{\max}/2}\} \ ,$$

where $l_{\max}$ is the coarsest level.

A rectangular bounding box $\mathcal{B}_L(\Sigma)$ as defined in (3.7) with $L = 10^{-3}$ is used to limit the evaluation of the integrals.

The parameters for the variable update (see Algorithm 3.2) were chosen quite conservative as $\alpha_{\min} = 10^{-9}$, $\alpha_{\max} = 0.999$ and $\beta = 0.1$. The regularisation of the Hessian matrix during the calculation of the Newton step were set to $\lambda_{\mathbf{u}} = \lambda_{\boldsymbol{\Sigma}} = 100$.

Evaluation: If we have ground truth or a reference solution $\mathbf{v}$ for the displacement, we can give some numerical measurements for comparison. If necessary, the solution of interest $\mathbf{u}$ is re-sampled to the same grid as $\mathbf{v}$ using cubic spline interpolation. Based on the position-wise Euclidean distance $\varepsilon_i := \|\mathbf{u}_i - \mathbf{v}_i\|_2$ we define three measurements:

$$\begin{aligned} \text{mean error } \mu(\varepsilon) &:= \frac{1}{|X_{\mathrm{V}}|} \sum_{x_i \in X_{\mathrm{V}}} \varepsilon_i \\ \text{standard deviation } s(\varepsilon) &:= \sqrt{\frac{1}{|X_{\mathrm{V}}|} \sum_{x_i \in X_{\mathrm{V}}} |\varepsilon_i - \mu(\varepsilon)|_2^2} \\ \text{maximal error } \max(\varepsilon) &:= \max_{x_i \in X_{\mathrm{V}}} \varepsilon_i \end{aligned}$$

Visualisation: In addition to the numerical declaration of the error, we will also visualise its spatial distribution using an error map which represents ε point-wise using some colour scale.

Whenever we present vector field data we either represent each vector by an arrow, especially when a detailed view is of interest. However, this method is not applicable for large and complex vector fields and we use a colour encoding scheme which is described in Fig. 3.7.

Window shapes are represented by their contour $\mathcal{C}_{\Sigma}(L)$ as defined in (3.4) with their centre located at the according displacement measurement. If not mentioned otherwise, we use $L = \exp(-1)$. For the sake of clarity, we restrict ourselves to a few representative windows.

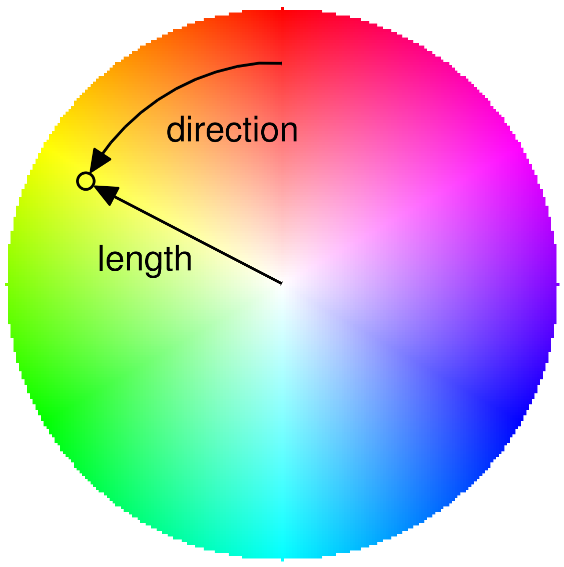


Figure 3.7: Colour encoding for vector fields: Each vector u is represented by a value within the depicted colour range: Hue for the vector direction and saturation for its length from white ($u = 0$) to the pure colour when $\|u\|_2 \geq v_{\max}$, where $v_{\max}$ is the maximum vector length chosen for the representation.

Algorithm 3.3: Multiscale framework for the joint displacement estimation and window adaption. The functions `resampleImage`(g, X), `resampleDisplacements`$(\mathbf{u}, X)$ and `resampleWindows`$(\boldsymbol{\Sigma}, X)$ interpolate image data, displacement fields and window-shapes, respectively, to a new grid X. Aliasing is avoided by applying a low-pass filter in before.

```
Algorithm: multiscaleSolution
Input: image data $g_1$, $g_2$, defined on grid $X_{\mathrm{D}}^{[0]}$
Input: initial displacements $\mathbf{u}^{\mathrm{init}}$, window shapes $\boldsymbol{\Sigma}^{\mathrm{init}}$, defined on grid $X_{\mathrm{V}}^{[0]}$
Input: parameters $l_{\max}$
Input: data and variable grids for each scale: $X_{\mathrm{D}}^{[0]}, \dots, X_{\mathrm{D}}^{[l_{\max}]}$, $X_{\mathrm{V}}^{[0]}, \dots, X_{\mathrm{V}}^{[l_{\max}]}$
Output: $\mathbf{u}^{[0]}$
$g_1^{[0]} \leftarrow g_1$, $g_2^{[0]} \leftarrow g_2$, $\mathbf{u}^{[0]} \leftarrow \mathbf{u}^{\mathrm{init}}$, $\boldsymbol{\Sigma}^{[0]} \leftarrow \boldsymbol{\Sigma}^{\mathrm{init}}$
/* fine to coarse: recursively create coarser versions of the
   images and variables initialisation */
for $l = 1, 2, \dots, l_{\max}$ do
    $g_i^{[l]} \leftarrow$ resampleImage $\left(g_i^{[l-1]}, X_{\mathrm{D}}^{[l]}\right)$      $i = 1, 2$
    $\mathbf{u}^{[l]} \leftarrow$ resampleDisplacements($\mathbf{u}^{[l-1]}, X_{\mathrm{V}}^{[l]}$))
    $\boldsymbol{\Sigma}^{[l]} \leftarrow$ resampleWindowShapes($\boldsymbol{\Sigma}^{[l-1]}, X_{\mathrm{V}}^{[l]}$)
end
/* coarse to fine: recursively refine the initial solution */
for $l = l_{\max}, l_{\max} - 1, \dots, 0$ do
    $\left(\mathbf{u}^{[l]}, \boldsymbol{\Sigma}^{[l]}\right) \leftarrow$ singlescaleSolution ($\mathbf{u}^{[l]}$, $\boldsymbol{\Sigma}^{[l]}$)
    if $l > 0$ then
        $\mathbf{u}^{[l-1]} \leftarrow$ resampleDisplacements($\mathbf{u}^{[l]}, X_{\mathrm{V}}^{[l-1]}$)
        $\boldsymbol{\Sigma}^{[l-1]} \leftarrow$ resampleWindowShapes($\boldsymbol{\Sigma}^{[l]}, X_{\mathrm{V}}^{[l-1]}$)
    end
end
return $\mathbf{u}^{[0]}$
```

3.5.2 Comparison of Window Selection Strategies

The following experiments are dedicated to the question whether our error model approach can improve the accuracy of the displacement estimation at all. Thus, we take care that errors that might be caused by the continuous optimisation of the error model function, e.g. getting stuck in local optima, are reduced to a minimum. The experiments are based on synthetic PIV image data which contains motion gradients of varying degree.

A set S of 975 window shapes was selected with varying radius, orientation and anisotropy,

$$S := \left\{ \Sigma(r, \sqrt{1-a^{-2}}, \alpha) \;\middle|\; \begin{array}{l} r \in \left\{ 2^{\frac{i}{2}} \,\middle|\, i \in \{-4, -3, \ldots, 10\} \right\}, \\ a \in \left\{ 2^{\frac{i}{4}} \,\middle|\, i \in \{0, 1, \ldots, 8\} \right\}, \\ \alpha \in \left\{ \frac{i}{8}\pi \,\middle|\, i \in \{0, 1, \ldots, 7\} \right\} \end{array} \right\},$$

where $\Sigma(r, \varepsilon, \alpha)$ is the parametrisation defined in (3.6). We fixed each of this window shapes and calculated the displacements at 160 positions $x \in X$ which are equally distributed over the image domain, using our proposed algorithm. The results were compared to the ground truth solution using the Euclidean norm, providing an error measurement $\varepsilon(x, \Sigma)$ for each position $x \in X$ and window shape $\Sigma \in S$. The ground truth vector field was used to initialise the displacements $\mathbf{u}$ with the aim to minimise the error caused by finding a non-global optimum of $C(u, \Sigma, x)$. Thus, the remaining displacement error is mainly caused by the sub-optimal choice of the window shape with respect to the motion and image disturbances.

Next, we define three window selection strategies. Each strategy $\Sigma_i(x)$, $i \in \{1, 2, 3\}$ describes how to select the window shape $\Sigma \in S$ at position $x \in X$. The quality of each strategy is measured by the mean error

$$\mu_i := \frac{1}{|X|} \sum_{x \in X} \varepsilon(x, \Sigma_i(x)) \ .$$

In addition, the standard deviation is defined by

$$s_i := \sqrt{\frac{1}{|X|} \sum_{x \in X} \left(\varepsilon(x, \Sigma_i(x)) - \mu_i\right)^2} \ .$$

The three considered window selection strategies are:

oracle: This hypothetical strategy “magically” knows the values of $\varepsilon(x, \Sigma)$ a priori. For every position it selects the optimal window shape from S, i.e.

$$\Sigma_1(x) := \arg\min_{\Sigma \in S} \varepsilon(x, \Sigma) \ .$$

Thus, the oracle gives a lower bound for the displacement error under this conditions. The error measurements are denoted as $\mu_1^* := \mu_1$ and $s_1^* := s_1$.

error model: The second strategy represents the proposed window adaption method. For each position the window shape is chosen such that it is optimal for to the defined error model function:

$$\Sigma_2(x) := \arg\min_{\Sigma \in S} \; E(\Sigma, \mathbf{u}, x)$$

This optimisation problem is solved by brute-force evaluation of the error model function for each of the 975 window shapes. In this context, the ground truth vector field is used for $\mathbf{u}$ to exclude influences caused by inaccuracies of the displacement estimation. The parameters of the function E are chosen as $e_{\text{outside}} = 0$, while the disturbance level estimation σ is a parameter to the strategy. Thus, the error measurements depend on σ and are denoted as $\mu_2(\sigma)$ and $s_2(\sigma)$.

fixed radius: In this strategy, a fixed radius r is chosen a priori and used uniformly at all positions. Using the relation (3.6) for the construction of the window shape parameters, the strategy is defined as

$$\Sigma_3(x) := \Sigma(r, 0, 0) = \frac{r^2}{-2\log L} I \; .$$

This is the most naive method considered here and is expected to provide an upper error bound. The associated quality measures are denoted by $\mu_3(r)$ and $s_3(r)$.

For the latter two strategies, we define the parameters σ^* and r^* as those, which minimise the mean error. For shortness, we denote $\mu_2^* := \mu_2(\sigma^*)$ and $s_2^* := s_2(\sigma^*)$ and accordingly $\mu_3^* := \mu_3(r^*)$ and $s_3^* := s_3(r^*)$.

The experiments were performed with two sets of synthetic particle image data. The regions *Sinusoids I* and *Sinusoids II* are part of the *Case A4* created for the *PIV Challenge 2005* data set and both have the same ground-truth vector field. However, the image data differs in the amount of disturbances. For details we refer to the evaluation report [49] and the PIV Challenge homepage [50]. The data is available at [51] and presented in Fig. 3.8. In Sect. 3.5.4 we further examine this data set.

Data without Disturbances

The first data set contains no image noise at all. In Fig. 3.9(a) the first frame of the image pair is depicted and gives an impression on the image quality.

The error measurements of the fixed window shape strategy with respect to the parameter r are plotted in Fig. 3.10. The results of the adaptive window strategy with respect to parameter σ can be found in Fig. 3.11. Table 3.2 lists the calculated error measurements numerically.

As expected, the oracle strategy constitutes a lower bound on the error of the other methods. When windows are selected by means of the error model, the error for the best parameter choice reduces to about 50% in comparison to the naive, non-adaptive method, while a gap to the optimal strategy remains. In addition, the choice of the parameter σ seems to be much less critical, since the values for σ in the range $[1, 100]$ influence the mean error only marginally.

Table 3.1: Comparison of window selection strategies, based on data without image noise (*Sinusoids I*): Optimal parameters and error measurements.

i	strategy	optimal parameter	min. mean error μ_i^*	standard deviation s_i^*
1	oracle	-	0.00627	0.0125
2	error model	$\sigma^* = 10$	0.0421	0.0484
3	fixed windows	$r^* = 2$	0.0796	0.0972

Table 3.2: Comparison of window selection strategies, based on data with image disturbances (*Sinusoids II*): Error measurements.

i	strategy	optimal parameter	min. mean error μ_i^*	standard deviation s_i^*
1	oracle	-	0.0255	0.0480
2	error model	$\sigma^* = 10^{1.75} \approx 56.2$	0.109	0.105
3	fixed windows	$r^* = 2^{2.5} \approx 5.66$	0.206	0.237

Data with Disturbances in the Image Data

For the second experiment, data with two kind of image disturbances were used: 3% of pixel noise was added and 20% of the particles are unpaired, i.e. their projection only occurs in one frame and cannot be matched in the other frame. Figure 3.9(c) gives an impression of the data quality.

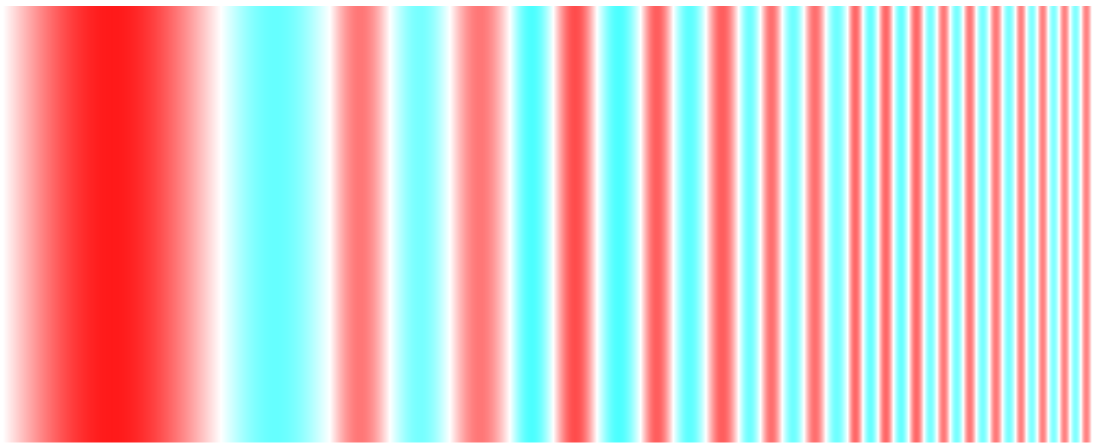
The error measurements for three proposed window selection strategies are compared visually in Fig. 3.12 and Fig. 3.13. The exact values are summarised in Table 3.2.

Basically, we have the same situation as in the previous experiments with undisturbed image data. The window adaption strategy improves the non-adaptive by 50%, but there is still room for improvement to the optimal strategy. Again, the results are not sensible to the choice of the error model parameter σ, but to the window radius r.

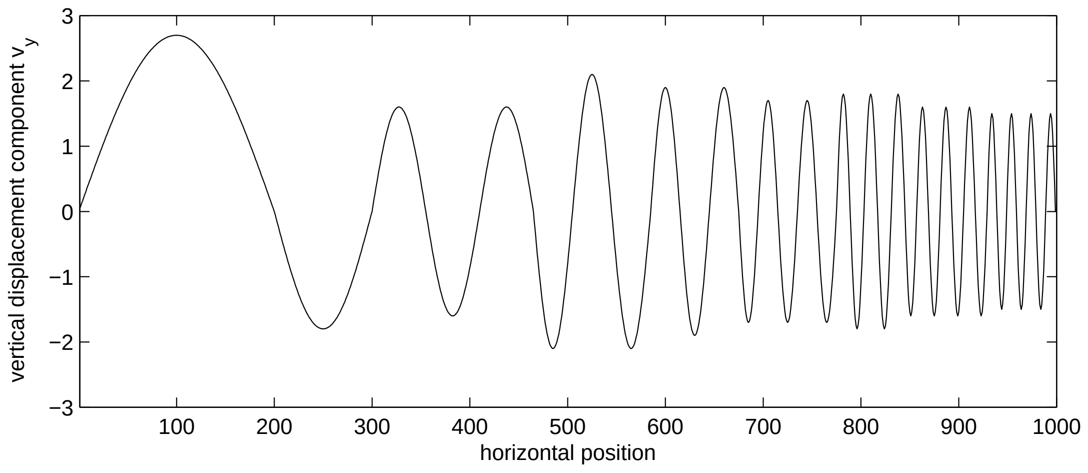
Summary and Conclusion

In this experiments we investigated whether the proposed window adaption strategy has the potential to improve displacement error measurement. For this purpose we compared an error model based window selection to a naive one and determined a lower bound on the error.

Results showed that selecting the window shapes based on the window error model improves the estimation by 50% compared to choosing the same window shape everywhere. In addition, it became apparent, that the choice of the model parameter σ is not critical.

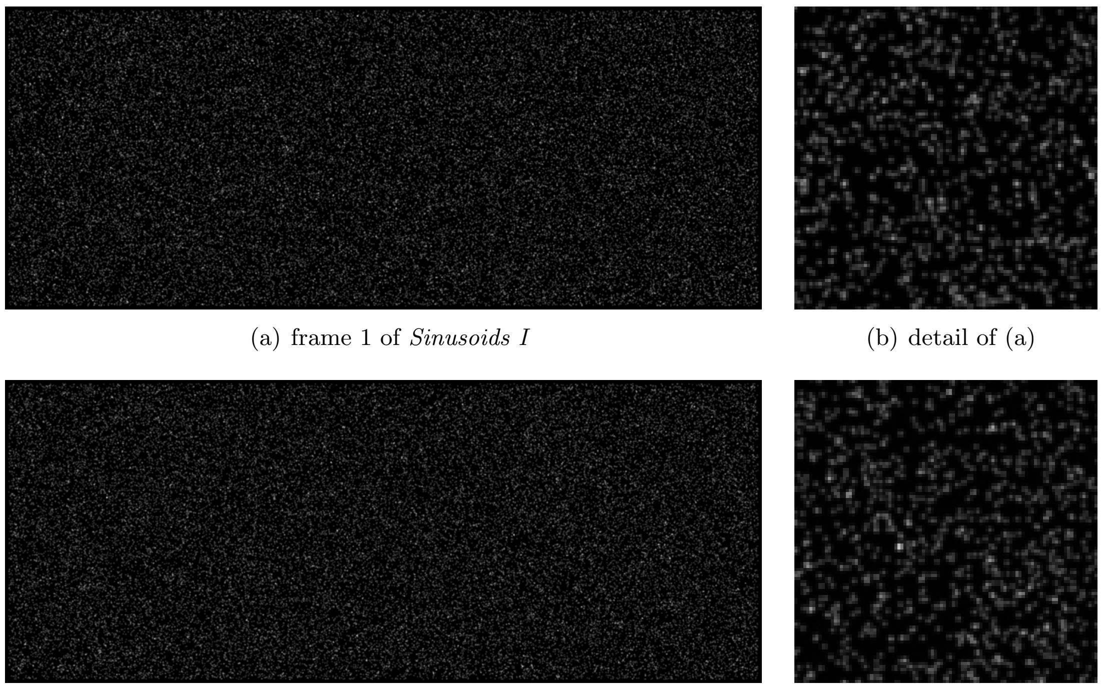


(a) displacement map



(b) profile of the vertical displacement component

Figure 3.8: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids I/II*: (a) Displacement map and (b) profile of the vertical component of the displacements common for both data sets, *Sinusoids I* and *Sinusoids II*. The vector field has a zero horizontal component, while the vertical component is piecewise described by sine functions with decreasing wavelength (from 400px to 20px) and varying amplitude (around 2px).



(a) frame 1 of *Sinusoids I*

(b) detail of (a)

(c) frame 1 of *Sinusoids II*

(d) detail of (c)

Figure 3.9: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids I/II*: First image frame (1000px × 400px) and detail (100px × 100px) from the data sets (a)-(b) *Sinusoids I* and (c)-(d) *Sinusoids II*. The upper one contains no disturbances while in the second 20% of the particles are unpaired between the two frames, and 3% pixel noise was added.

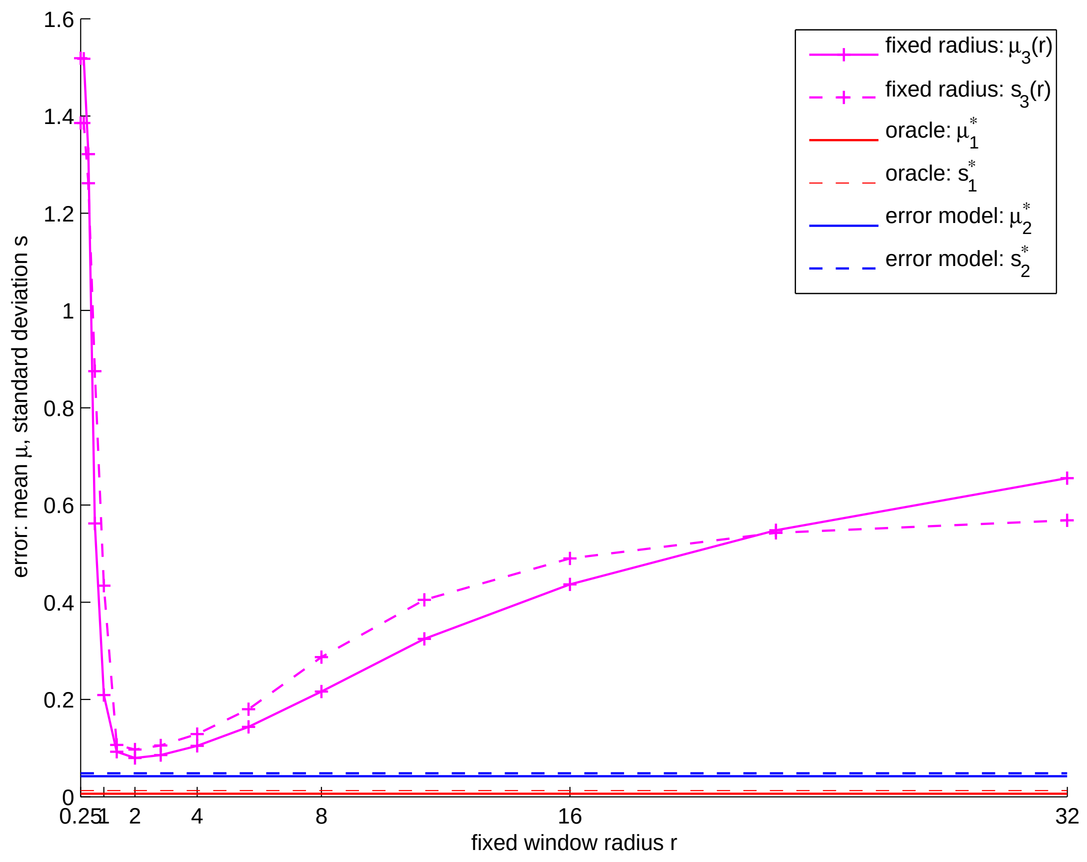


Figure 3.10: Comparison of window selection strategies, based on data without image disturbances (*Sinusoids I*): Results of the fixed radius strategy depending on the choice of window radius r. The smallest mean error is achieved for $r^* = 2$ with $\mu_3^* = 0.0796$ and $s_3^* = 0.0972$. For comparison, the results of the oracle strategy and the window selection strategy with $\sigma = \sigma^*$ are plotted.

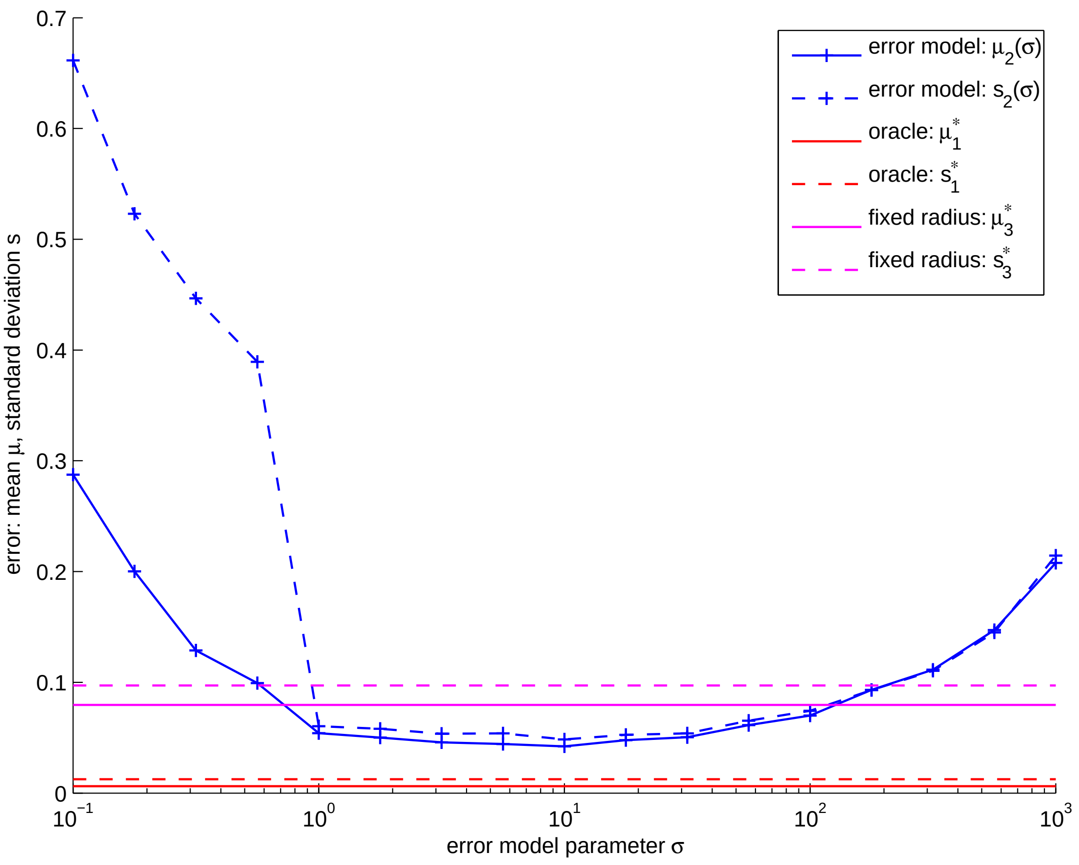


Figure 3.11: Comparison of window selection strategies, based on data without image disturbances (*Sinusoids I*): Results of the window adaption strategy depending on the choice of parameter σ. The smallest mean error is achieved for $\sigma^* = 10$ with $\mu_2^* = 0.0421$ and $s_2^* = 0.0484$. For comparison, the results of the oracle strategy and the fixed radius strategy with $r = r^*$ are plotted. The window selection strategy outperforms the naive selection by about 50%. In addition, the choice of σ is not critical, e.g. within the range $[1, 100]$ the error varies only slightly.

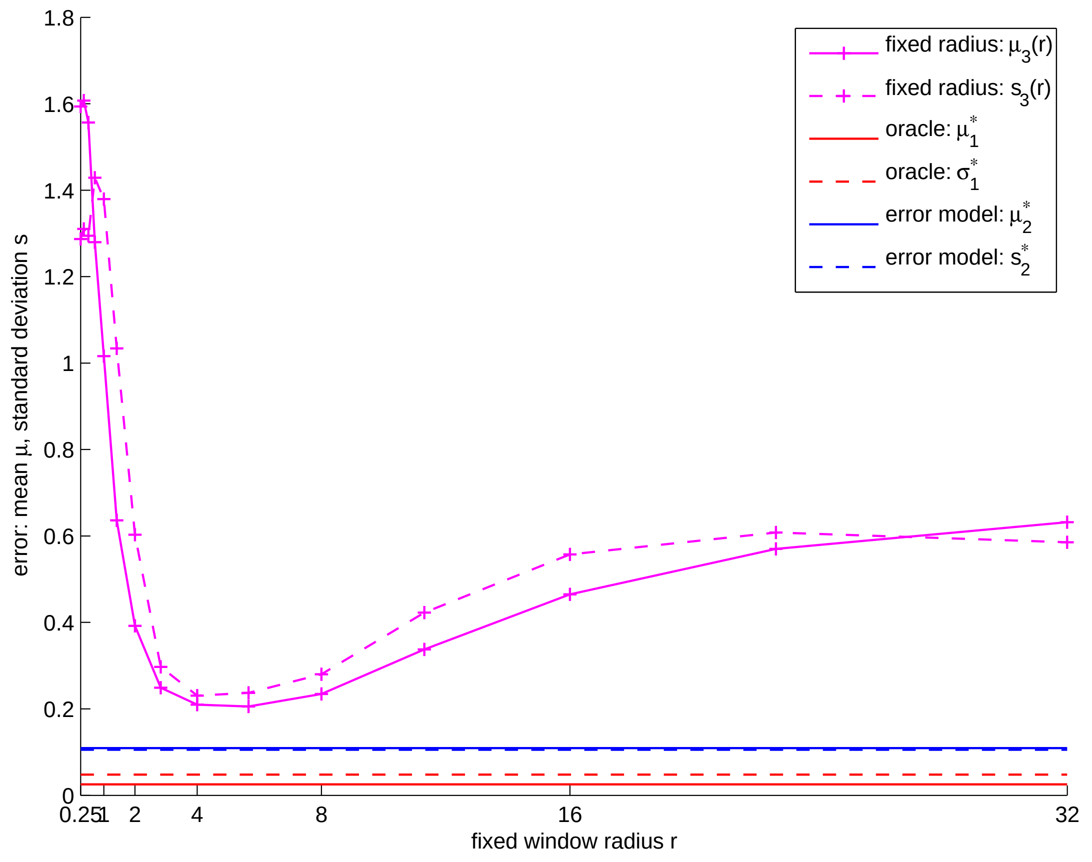


Figure 3.12: Comparison of window selection strategies, based on data with disturbances (*Sinusoids II*): Results of the fixed radius strategy depending on the choice of window radius r. The smallest mean error is achieved for $r^* = 2^{2.5}$ with $\mu_3^* = 0.206$ and $s_3^* = 0.237$. For comparison, the results of the oracle strategy and the window selection strategy with $\sigma = \sigma^*$ are plotted.

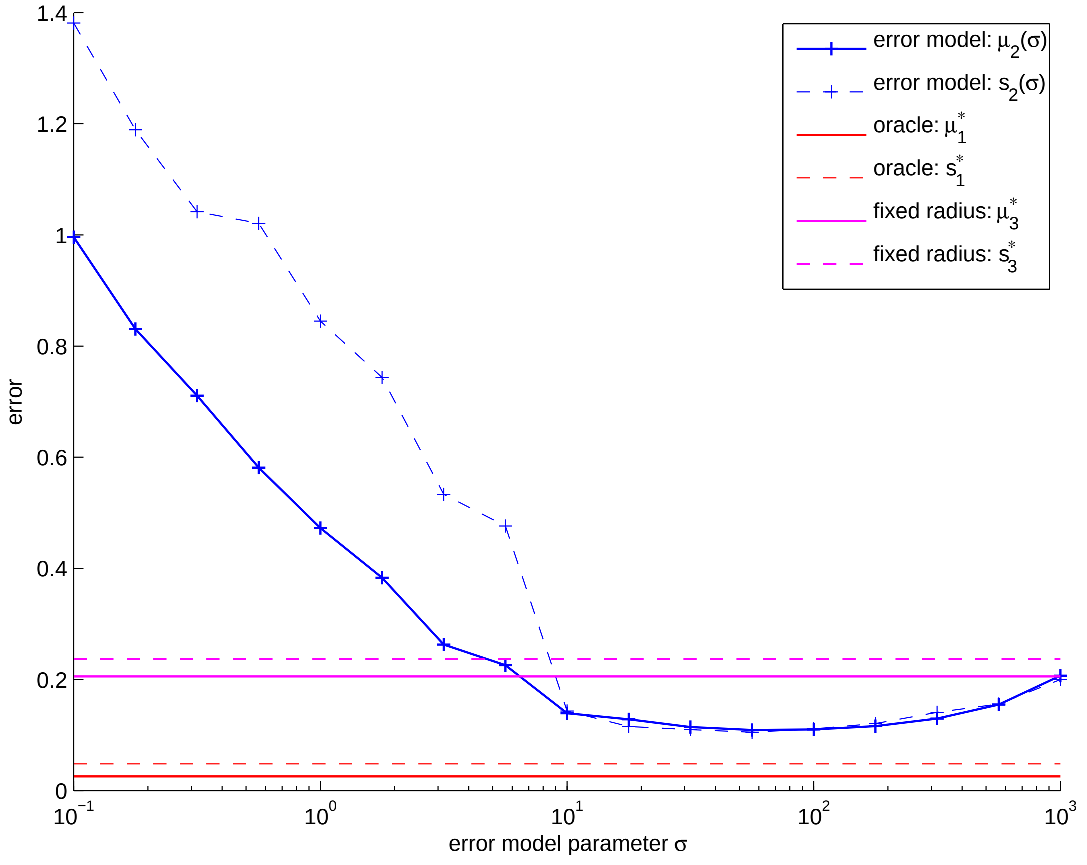


Figure 3.13: Comparison of window selection strategies, based on data with disturbances (*Sinusoids II*): Results of the window adaption strategy depending on the choice of parameter σ. The smallest mean error is achieved for $\sigma^* = 10^{1.75}$ with $\mu_2^* = 0.109$ and $s_2^* = 0.105$ For comparison, the results of the oracle strategy and the fixed radius strategy using $r = r^*$ are plotted.

3.5.3 Window Adaption on Synthetic Data

The optimal windows with respect to the defined error model can form complex structures for arbitrary displacement fields. Thus, we demonstrate the behaviour of the proposed window adaption method by means of some simple synthetic vector fields of size 51 by 51 vectors. In particular, we verify the theory for affine vector fields developed in Sect. 3.2.2.

The proposed algorithm adapted the window shapes $\mathbf{\Sigma}$, but the displacements $\mathbf{u}$ were not modified after initialisation from the experimental setup. In all experiments we started with round windows ($r = 5$) and used a single resolution scale. If not mentioned otherwise, we chose $\sigma = 1$, $e_{\text{outside}} = 0$ and an upper limit $r_{\max} = 6$ on the window size, but no lower bound $r_{\min}$.

Affine vector fields

In Fig. 3.14 four affine vector fields are examined. As the theory states, windows grow to infinite size in regions of constant vector fields, which, however, is limited here by the constraint $r \leq r_{\max} = 6$ (Fig. 3.14(a)). If we set the parameter $e_{\text{outside}} = 10$ the windows avoid areas where no image data is given (Fig. 3.14(b)). Note that they do not decrease in size in the axis parallel to the boundary as long as there is not additional influence as near image corners.

In affine vector fields with non-singular Jacobian $A = \mathrm{J}_x\, \mathbf{u}(x)$, the window shape and size tends to an unit shape and size (except for regions near boundaries), which depend only on σ and A. No upper size limit was enforced, i.e. $r_{\max} = \infty$. If $A^\top A$ is a scaled unit matrix, the weightings are isotropic (Fig. 3.14(c)), while for a random, regular A they are anisotropic (Fig. 3.14(d)).

Transition Zones

Next we consider situations with a gradient in the vector field in vertical direction. Constant regions in the upper and lower part enclose a transition area and are colour-encoded in Fig. 3.15. In the first three experiments different displacement fields with affine transition zones are examined. However, the adapted window fields are identical and demonstrate that the adaption scheme only depends on the properties of $A^\top A$, but not on A. The window shapes are invariant to a constant component added (Fig. 3.15(a) vs. 3.15(b)) and a rotation of the vectors (Fig. 3.15(a) vs. 3.15(c)).

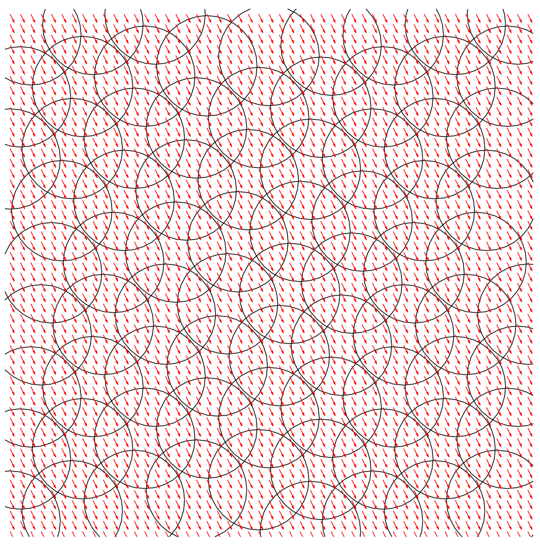
The transition zone in Fig. 3.15(d) is a sine-function in the range of $[0, \pi]$. Windows are smallest in vertical direction where the curve rises or falls and the gradient is largest. In contrast, at the centre the displacement change is small and thus the windows enlarge. Note, that their horizontal dimension of the shapes is not affected due to the lack of a gradient. Their size is only limited by the constraint $r \leq 6$.

Sharp motion boundaries

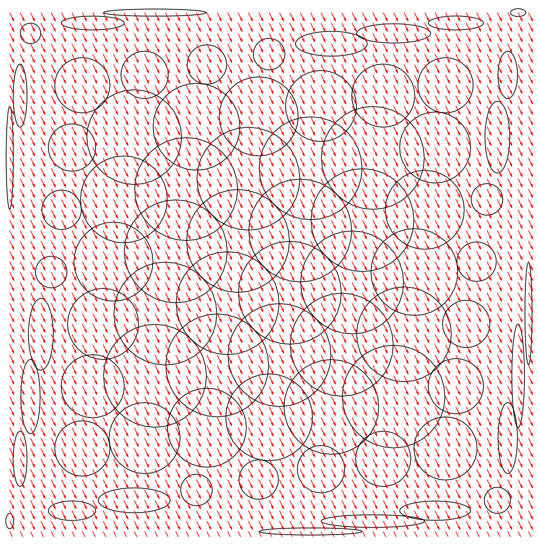
Finally, we examine situations where two affine displacement fields meet and form sharp motion boundaries. Here, it is of great importance for an accurate solution, that the correlation windows do not breach the discontinuities, which would cause the displacement estimation to smooth out the vector field. No matter whether the two touching vector fields are constant (Fig. 3.16(a) and 3.16(b)) or rotational (affine, Fig. 3.16(c) and 3.16(d)) each, the adapted windows shape respect the boundaries well.

Conclusion

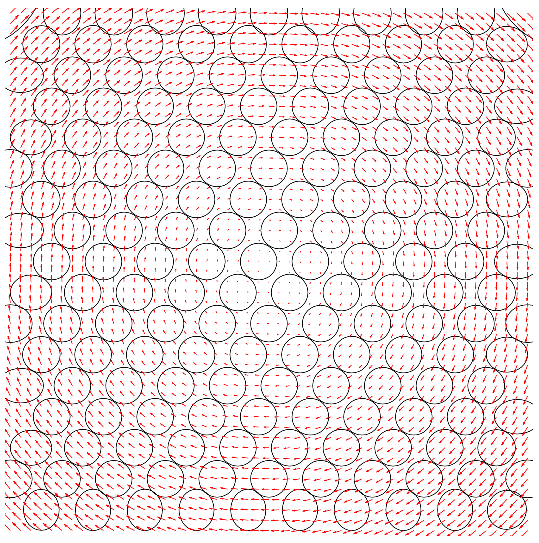
We tested the behaviour of the window adaption scheme by means of some simple, synthetic vector fields. The results confirmed the theoretical results for affine vector fields of Sect. 3.2.2. Moreover, the ability to adapt to motion gradients was demonstrated.



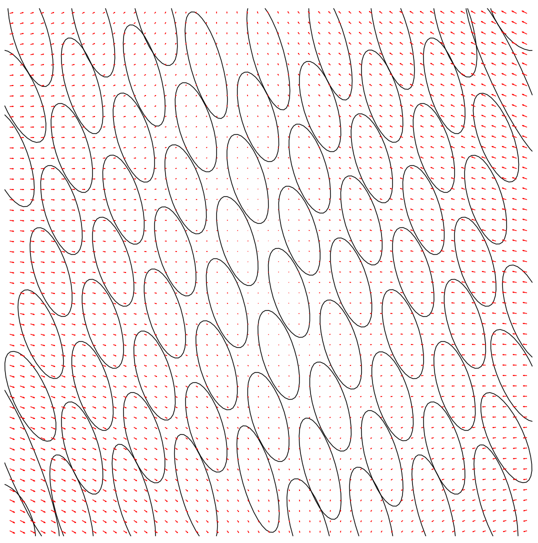
(a) constant vector field



(b) constant vector field, adaption to boundaries

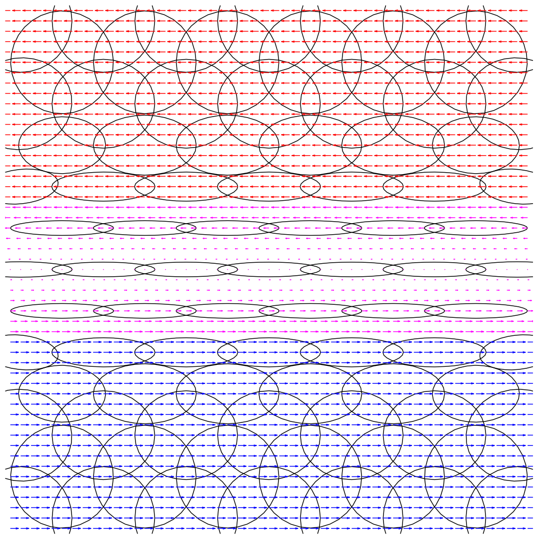


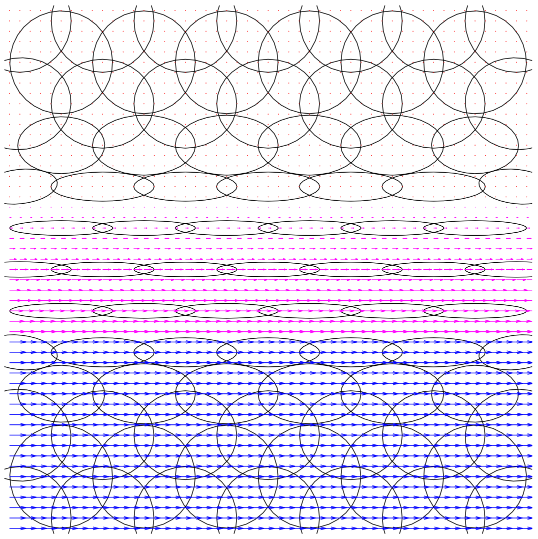
(c) affine vector field with isotropic gradient

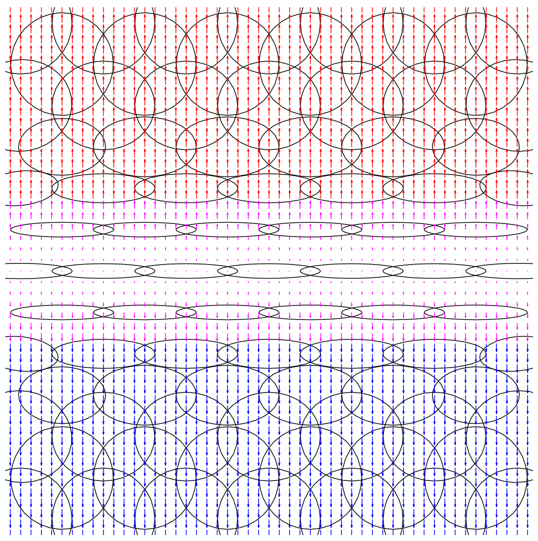


(d) affine vector field with anisotropic gradient

Figure 3.14: Synthetic displacement fields (arrows) and some of the adapted windows (represented by their contour $\mathcal{C}(\Sigma)$, in black). **Constant vector fields:** (a) The window size would approach infinity due to the lack of a gradient, but is limited by the constraint $r \leq r_{\max} = 6$. (b) The windows are additionally constrained to adapt to the boundaries of the image domain by setting $e_{\text{outside}} = 10$. **Affine vector fields:** (c) Rotational field with isotropic gradients leads to round windows. (d) Vector field with anisotropic gradients leads to ellipse-shaped windows.


(a) displacement gradient perpendicular to the flow


(b) same as (a) with a constant flow superimposed


(c) displacement gradient along the flow

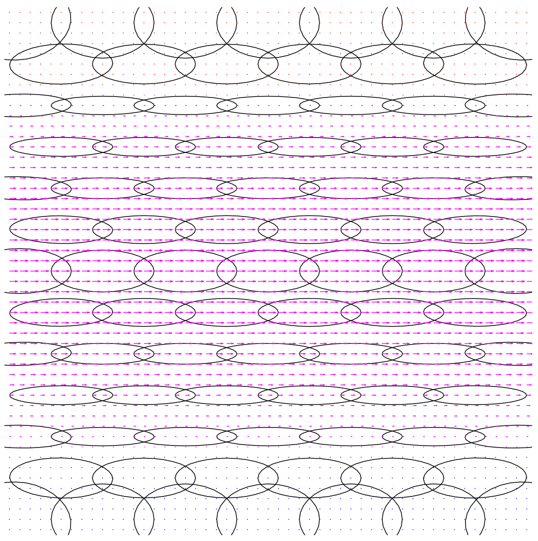
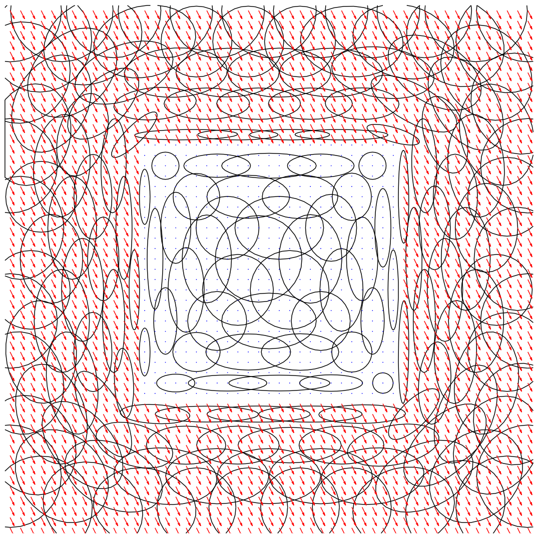
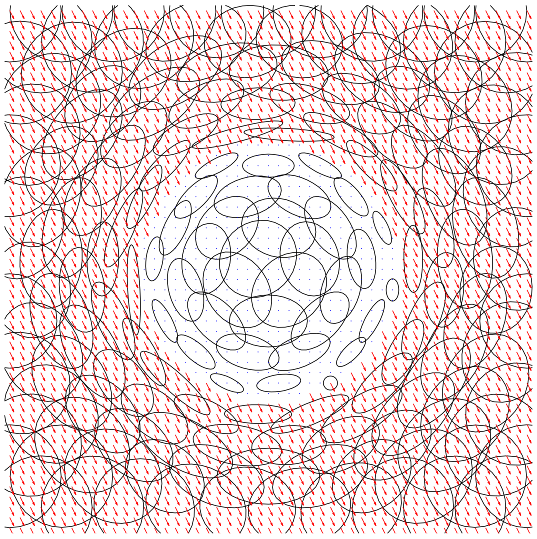
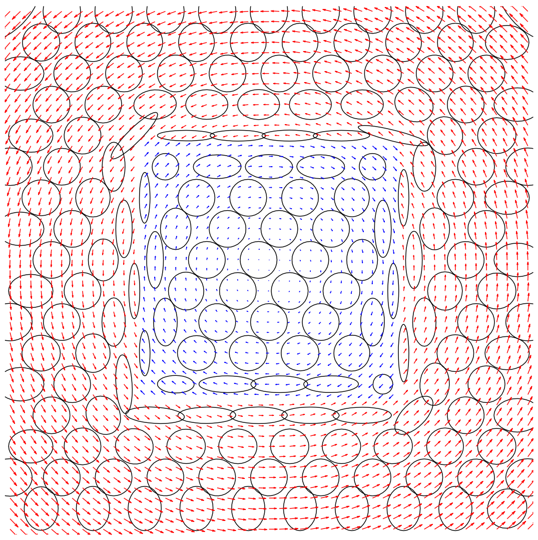

(d) sinusoid

Figure 3.15: Synthetic displacement fields (arrows) and some of the adapted windows (represented by their contour $\mathcal{C}(\Sigma)$, in black). Constant vector fields (red, blue) enclose a transition zone (magenta). All windows are constrained by $r \leq 6$. **Affine transition zone:** Identical window shapes for different vector fields: (a) displacement gradients perpendicular to the flow, (b) superimposed by a constant field, and (c) gradients parallel to flow. **Sinusoid:** (d) Transition zone is sinus-shaped ($[0, \pi]$). The adaption scheme aligns the windows perpendicular to the displacement gradient and reduces the size along the transition direction to avoid smoothing out the boundary.


(a) constant vector fields, square boundaries


(b) constant vector fields, round boundaries


(c) affine vector fields, square boundaries

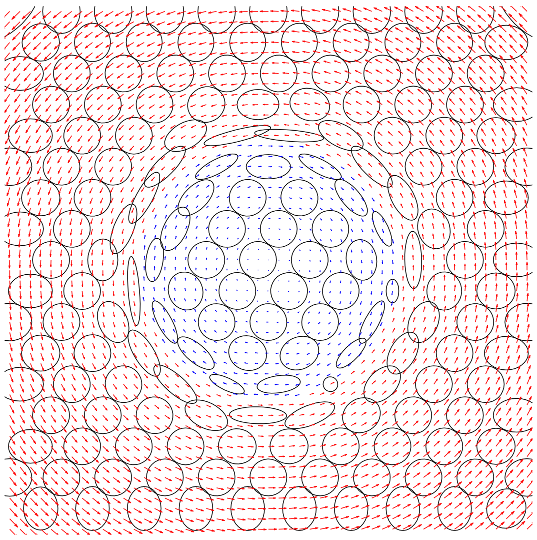

(d) affine vector fields, round boundaries

Figure 3.16: Synthetic displacement fields (arrows) and some of the adapted windows (represented by their contour $\mathcal{C}(\Sigma)$, in black), with sharp discontinuities between two motion regions (red, blue). All windows are constrained by $r \leq r_{\max} = 6$. **Constant flows:** Constant flow (red) interrupted by a zero flow (blue) with (a) square and (b) round inner region. **Rotational (affine) flows:** Two contrarily rotating (affine) flows with (a) square and (b) round inner region.
The adaption scheme reduce the window sizes near the region boundaries to avoid smoothing over motion discontinuities.

3.5.4 Evaluation with a PIV Challenge Data Set

Our algorithm is designed to avoid smoothing over vector gradients by adapting the window shape accordingly. We verify this aim by means of the data set introduced as *Case A4* in the *PIV Challenge 2005.* It consists of a synthetic particle image pair with ground truth and is divided into seven regions of which we consider the set *Boundary Layers* in Sect. 3.5.4 and *Sinusoids I* and *Sinusoids II* in Sect. 3.5.4. The data set was designed to measure spatial resolution of velocity measurement algorithms and was used to evaluate the performance of 19 PIV algorithms in [49].

Boundary Layers

The region *Boundary Layers* is 1000 $\times$ 1000 pixel in size. The motion field consists of three slightly tilted regions with straight boundaries which have gradients of different slope, see Fig. 3.17. The movement is aligned parallel to the borders (heading to the right) while in between the three regions the motion is zero. Basically, the images consists of particles but there is an important difference between the left and right half: In the left, regions with no movement do not contain grey value gradient while in the right half a constant pattern provides information for motion estimation algorithms.

First experiments showed that it is essential to have a good initialisation for the joint displacement estimation and window adaption. Thus, we started by calculating a vector field with *fixed* window shapes and initialised the joint approach with this result.

The initialisation was calculated using the correlation approach with Gaussian shaped windows as defined in this chapter. The window radii were fixed to $r = 10$ throughout the calculation. Seven multiscale levels were used. The variable grid is spaced with $a_V = 4$.

In Fig. 3.18 the resulting vector field is illustrated, the position-wise Euclidean error with respect to the ground truth solution is presented in Fig. 3.19. Not surprisingly, large errors occur near the boundaries of the three regions, because the gradients are smoothed out, while inside the regions motion is reconstructed well.

There are large errors in regions where no image information is available. The reason that the displacements are non-zero there is due to the fact that grey-value gradients exist on coarser levels of the multiscale framework due to the smoothing of the images. In addition, our approach is not able to identify regions with a small amount of information in the image data and to handle them accordingly, e.g. by enlarging the windows.

In the second step, this result was refined using the proposed joint displacement and window adaption approach. Only three multiscale levels were used with scaling factor $\sqrt{2}$. The variable grid was refined to $a_V = 2$. The error model parameters were chosen as $\sigma = 100$ and $e_{\text{outside}} = 20$. The initial window size was set to $r = 10$ and constrained by $r_{\max} = 40$ and $r_{\min} = 2$.

The results are presented in Fig. 3.20 and Fig. 3.21. Error measurements for the non-adaptive and adaptive approach are listed in Table 3.3. The gradients on the boundaries are resolved much better due to the adaptive approach, defining clear borders of the regions. Some of the adapted windows are depicted in Fig. 3.22, which shows that the algorithm behaves as predicted in Sect. 3.5.3. However, the large errors in the left half

Table 3.3: Error measurements for *Case A4* in the *PIV Challenge 2005*, region *Boundary Layers*. The error measurements are defined in Sect. 3.5.1.

region	method	$\mu(\varepsilon)$	$s(\varepsilon)$	$\max(\varepsilon)$
Boundary Layers	correlation only	0.388	1.05	14.0
	correlation with adaptive windows	0.257	0.923	13.5

could not be corrected and additional errors appeared close to the left and right image boundary.

For further evaluation we extracted the profiles of the motion perpendicular to the boundaries and compare them in Fig. 3.23. We focused only on the right half of the data, were image information is available everywhere.

The non-adaptive scheme smoothes out the boundaries to a high degree. In contrast, the adapted windows reconstructs the ground truth profile much better for the first and second profile, however gives no improvement on the third one. Presumably, the reason is that the initialisation is not good enough in this region.

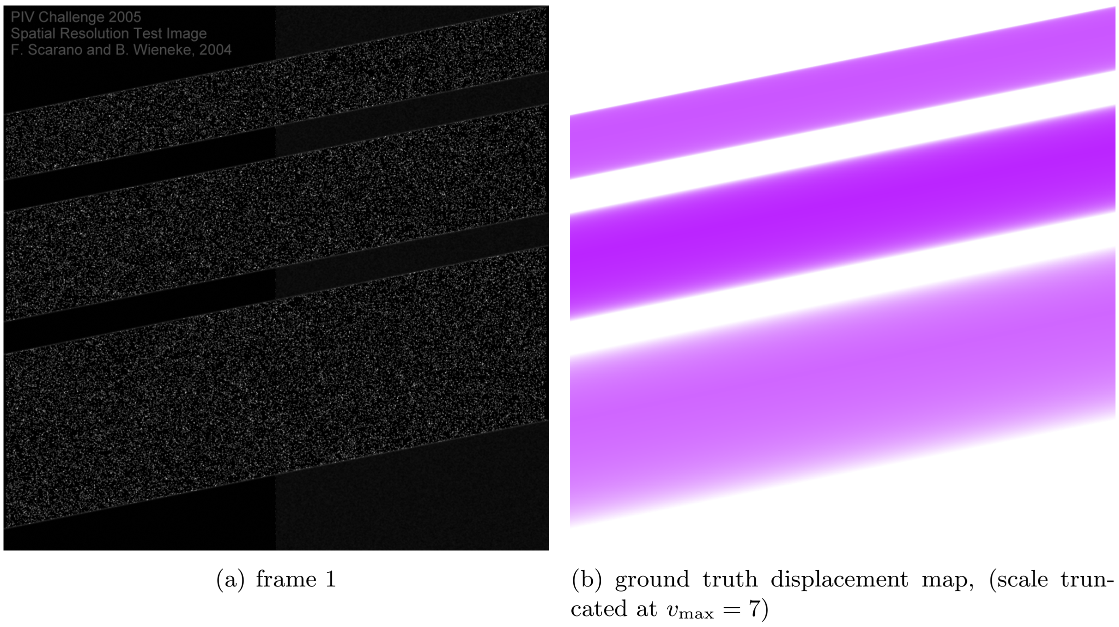
Sinusoids

Next, we considered the region *Sinusoids I* and *Sinusoids II* of *Case A4* in the *PIV Challenge 2005* data set. Both consist of a motion field with vertical components only. These vary sine-like with wavelength from 20 to 400 pixels, see Fig. 3.8. For further details on the data set, we refer to [49]. The two scenarios only differ in the amount of disturbances in the image data: While *Sinusoids I* has no noise at all, *Sinusoids II* contains pixel noise as well as unpaired particles. Figure 3.9 gives an impression of the image quality.

Sinusoids I: First we applied our correlation approach with fixed, round Gaussian window shapes with a size of $r = 8$ pixels everywhere. The variable grid was chosen as $a_{\mathrm{V}} = 2$ and seven resolution levels were used. The results in Fig. 3.24 show that the method is able to capture the main structures but fails to deliver accurate estimations for small wavelengths.

The previously calculated solution was used as initialisation for the combined displacement estimation and window adaption approach. The window adaption parameter were chosen as $\sigma = 20$ and $e_{\mathrm{outside}} = 20$. The windows were initialised with $r = 8$ and constrained by $r_{\max} = 40$ and $r_{\min} = 2$. The calculation was performed on three resolution levels and on the same variable grid as the initialisation.

The results in Fig. 3.25 show how our approach reconstructs the vector field with high precision even at the smallest wavelength and close to the right image boundary. This is also supported by the error measurements which are summarised in Table 3.4. As in previous experiments, the windows are compressed along the displacement gradients, see Fig. 3.26.



(a) frame 1

(b) ground truth displacement map, (scale truncated at $v_{\max} = 7$)

Figure 3.17: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Boundary layers*: (a) First frame of the image pair (1000px × 1000px). (b) Ground truth displacement field. Three regions with boundaries of varying high sharpness, and flows aligned in parallel. In the left half, image regions with no movement contain no grey-value patterns except for the label in the upper left, while in the right half a constant pattern provides movement information.

Sinusoids II: The previous experiments were repeated with the data set *Sinusoids II*, which is much more challenging due to the image disturbances in combination with the high velocity gradients near the right image border.

The number of multiscale levels for the calculation of the initialisation was reduced to five, but the scaling factor was increased to two, because the previously used settings caused the results to be too distorted.

The results, depicted in Fig. 3.27, attest the increased degree of disturbances in the measured displacements, which also influences the quality of the subsequent adaptive estimation.

For the adaptive approach, the noise level parameter was increased to $\sigma = 40$ to cope with the increased noise level but not to smooth over the smallest structures. All remaining parameters were chosen the same as for the region *Sinusoids I*.

From the displacement map and error visualisation in Fig. 3.28 it becomes apparent, that in general the adaptive approach could reconstruct the solution well. However, locally large errors occur which mostly coincide with disturbances in the initialisation. Again, the windows (Fig. 3.29) are aligned perpendicular to the displacement gradients, but also react to regions of high errors in the displacement field.

Table 3.4: Error measurements for *Case A4* in the *PIV Challenge 2005*, regions *Sinusoids I* and *Sinusoids II*. The measurements are defined in Sect. 3.5.1.

region	method	$\mu(\varepsilon)$	$s(\varepsilon)$	$\max(\varepsilon)$
Sinusoids I	correlation only	0.222	0.286	2.67
	correlation with adaptive windows	0.132	0.537	18.1
Sinusoids II	correlation only	0.276	0.350	3.84
	correlation with adaptive windows	0.190	0.472	19.1

The comparison by means of error measurements in Table 3.4 also shows the improvement of the solution with respect to the non-adaptive approach.

Furthermore, we compared the solution of our approach on this data set to the results published in [49]. For this purpose we evaluated our displacement field in the same way and merged the comparison with those of the other approaches. For each wavelength λ of the sinusoids, the amplitude ratio between the measured u and the ground truth u^* was calculated as

$$A := \frac{\int_{-\lambda/4}^{+\lambda/4} u(x)\,\mathrm{d}x}{\int_{-\lambda/4}^{+\lambda/4} u^*(x)\,\mathrm{d}x}\ .$$

For details we refer to the discussion in the evaluation paper. We created a copy of the corresponding figures in there and registered the coordinate system. This enabled us to accurately include the characteristic curve of our approach into the comparison plot of the methods investigated during the *PIV Challenge 2005.* The combined plots are depicted in Fig. 3.30 and Fig. 3.31. Our approach reconstructs the sinusoid amplitudes with high accuracy and outperforms most of the investigated methods.

Conclusion

The adaptive correlation approach was evaluated with synthetic data which was created for evaluating the spatial resolution of PIV measurement implementations during the *PIV Challenge 2005.* The results showed that it is capable to adapt the window shapes to gradients and thus to obtain a good solution even for small structures. It outperforms most of the 19 implementations for the most involved data set with image disturbances.

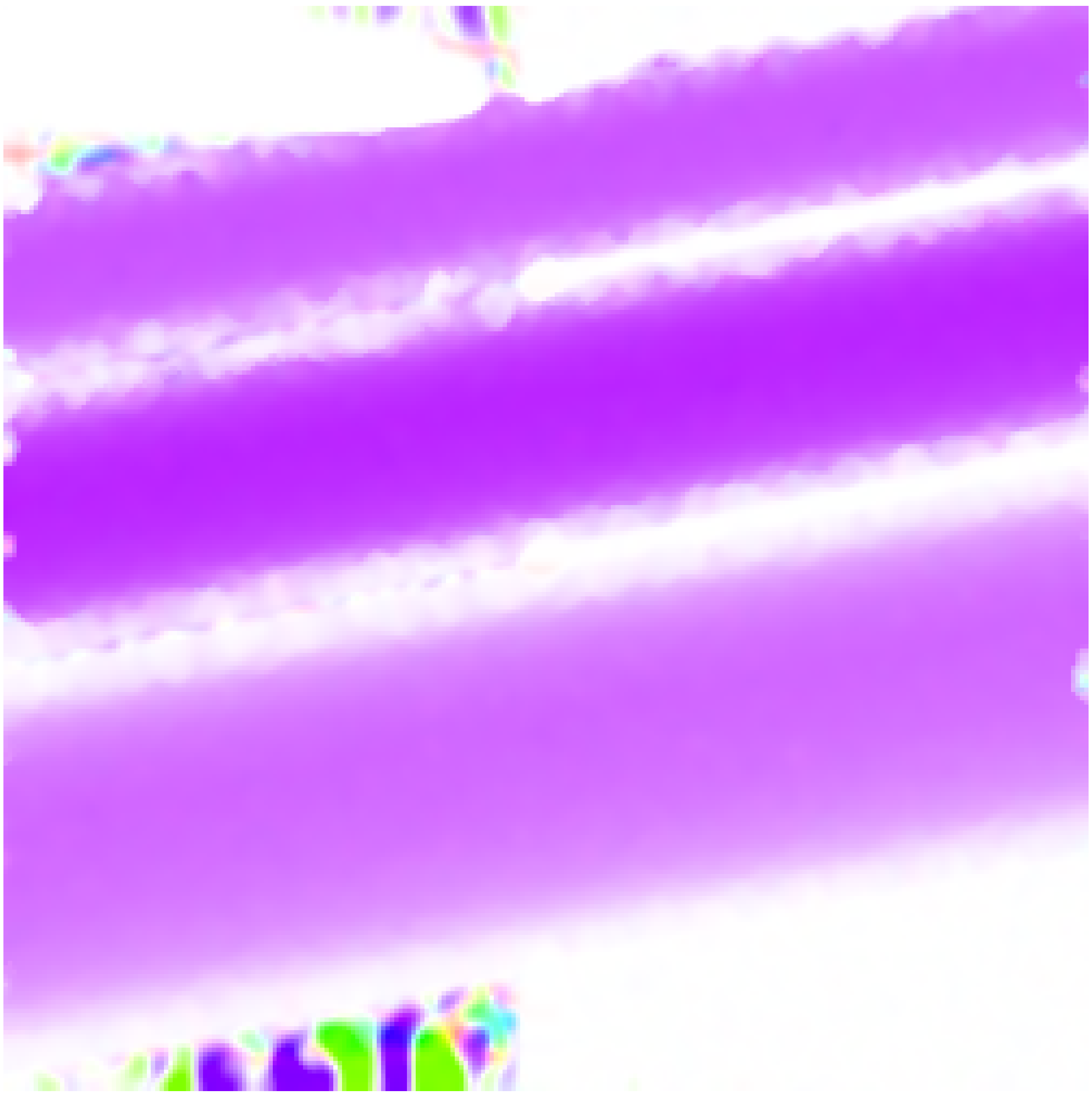


Figure 3.18: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Boundary layers*: Results of our approach using *fixed* windows with $r = 10$. The calculated displacement field with a grid spacing of 4px is visualised using the colour encoding defined in Fig. 3.7 (with $v_{\max} = 7$). The approach is able to reconstruct the main structure of the vector field, except for the boundaries of the flow regions which are not resolved well and smoothed out. Moreover, large error occurs where no image information is present.

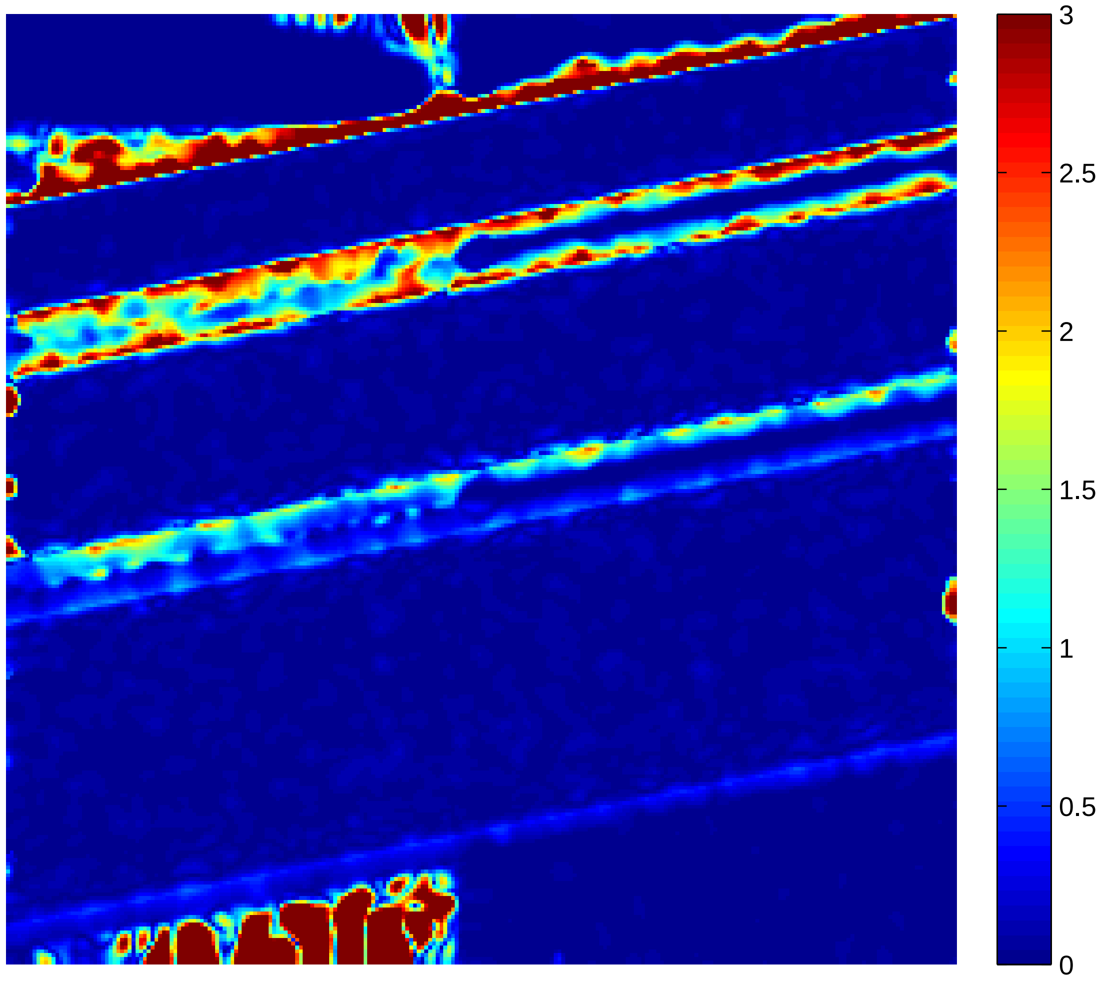


Figure 3.19: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Boundary layers*: Results of our approach using *fixed* windows with $r = 10$. The error map of the displacements in comparison to ground truth is illustrated with the error scale truncated at 3px.

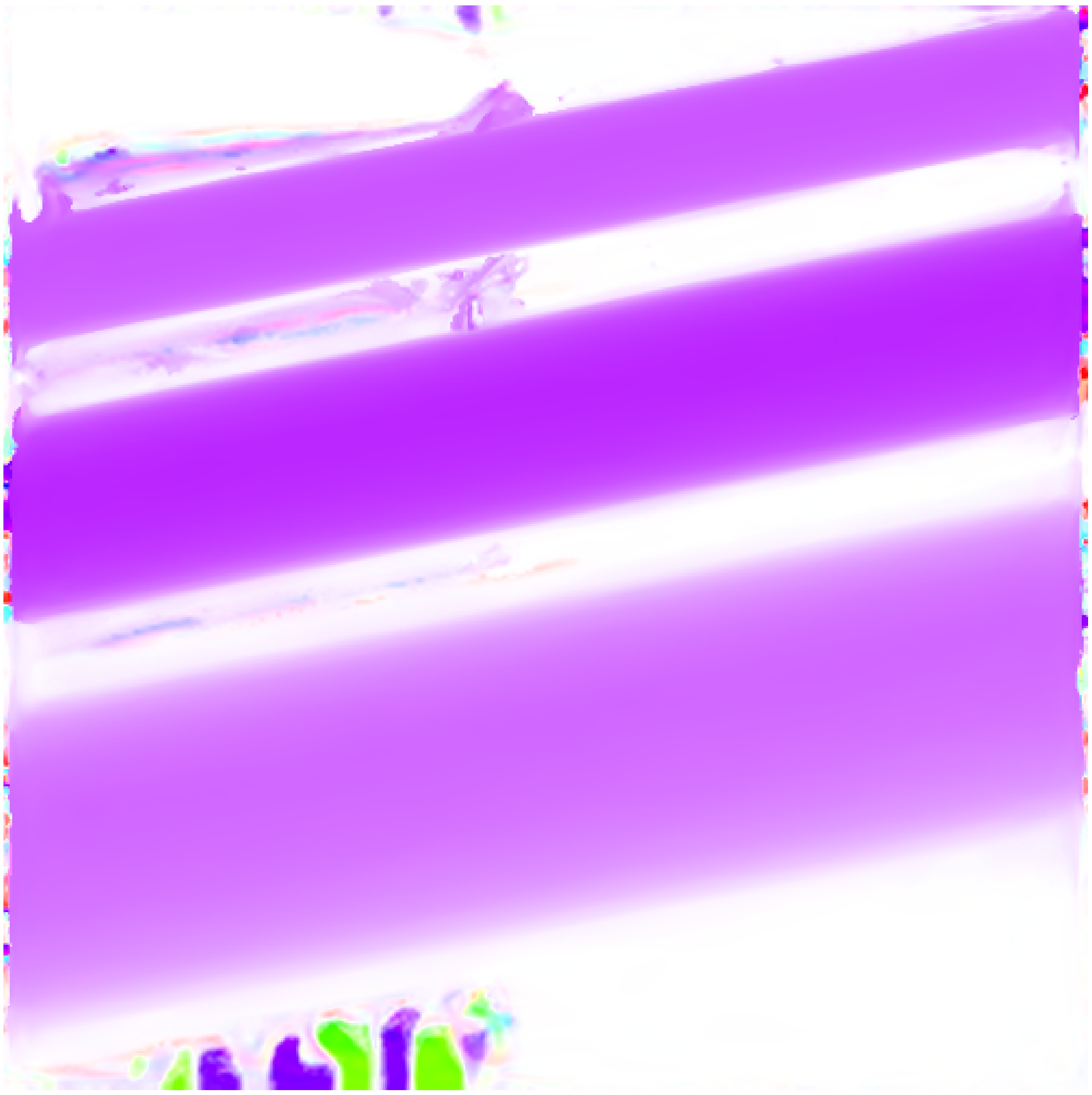


Figure 3.20: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Boundary layers*: Results of our approach with joint velocity estimation and window adaption, using the result from Fig. 3.18 as initialisation. The displacement map, estimated with grid spacing 2, is visualised with $v_{\max} = 7$. Velocity structures are more sharp in comparison to the results of the non-adaptive approach. Especially the gradients are better resolved due to the adaption, resulting in a clear boundary of the motion regions. Additional errors appear close to the image boundaries.

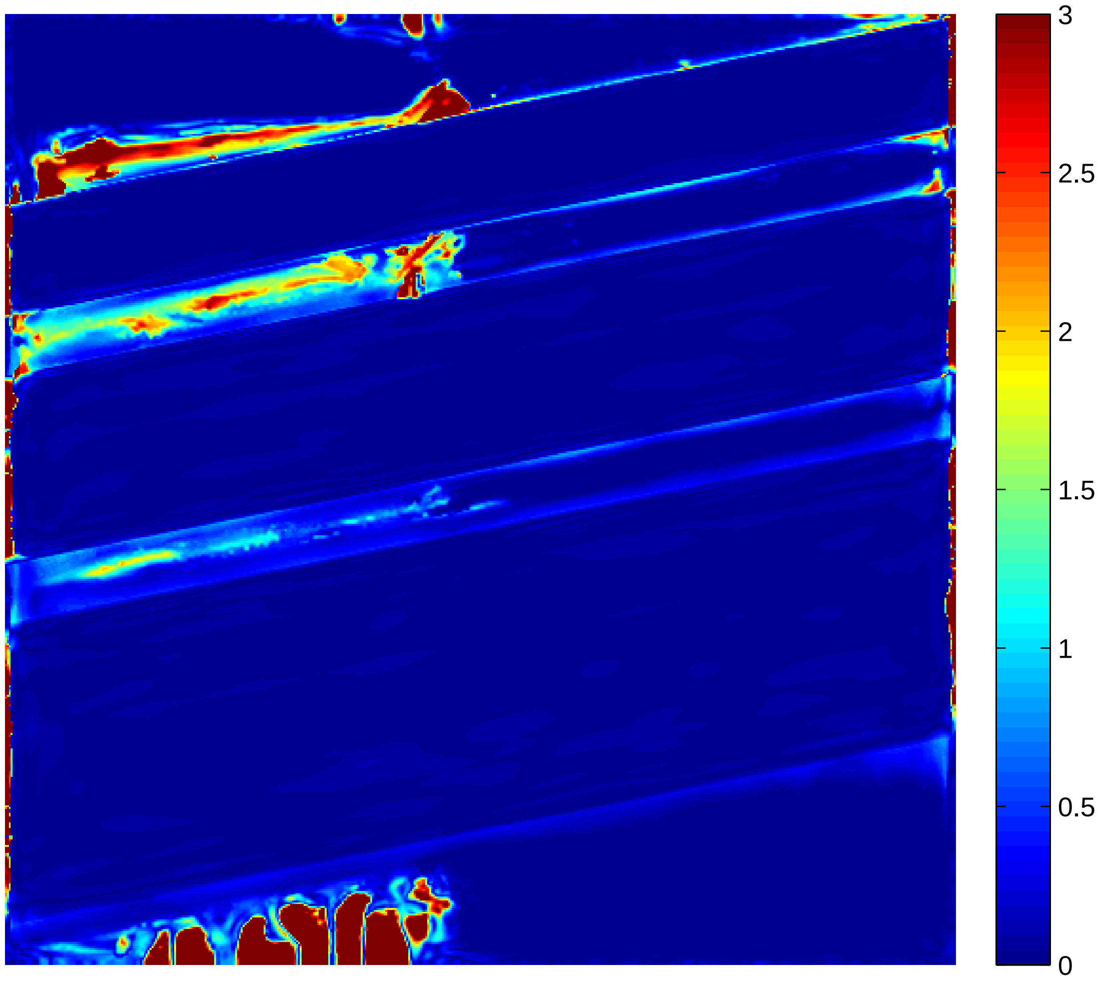


Figure 3.21: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Boundary layers*: Results of our approach with joint velocity estimation and window adaption, using the result from Fig. 3.18 as initialisation. The error map in comparison to ground truth is visualised with the error scale truncated at 3px.

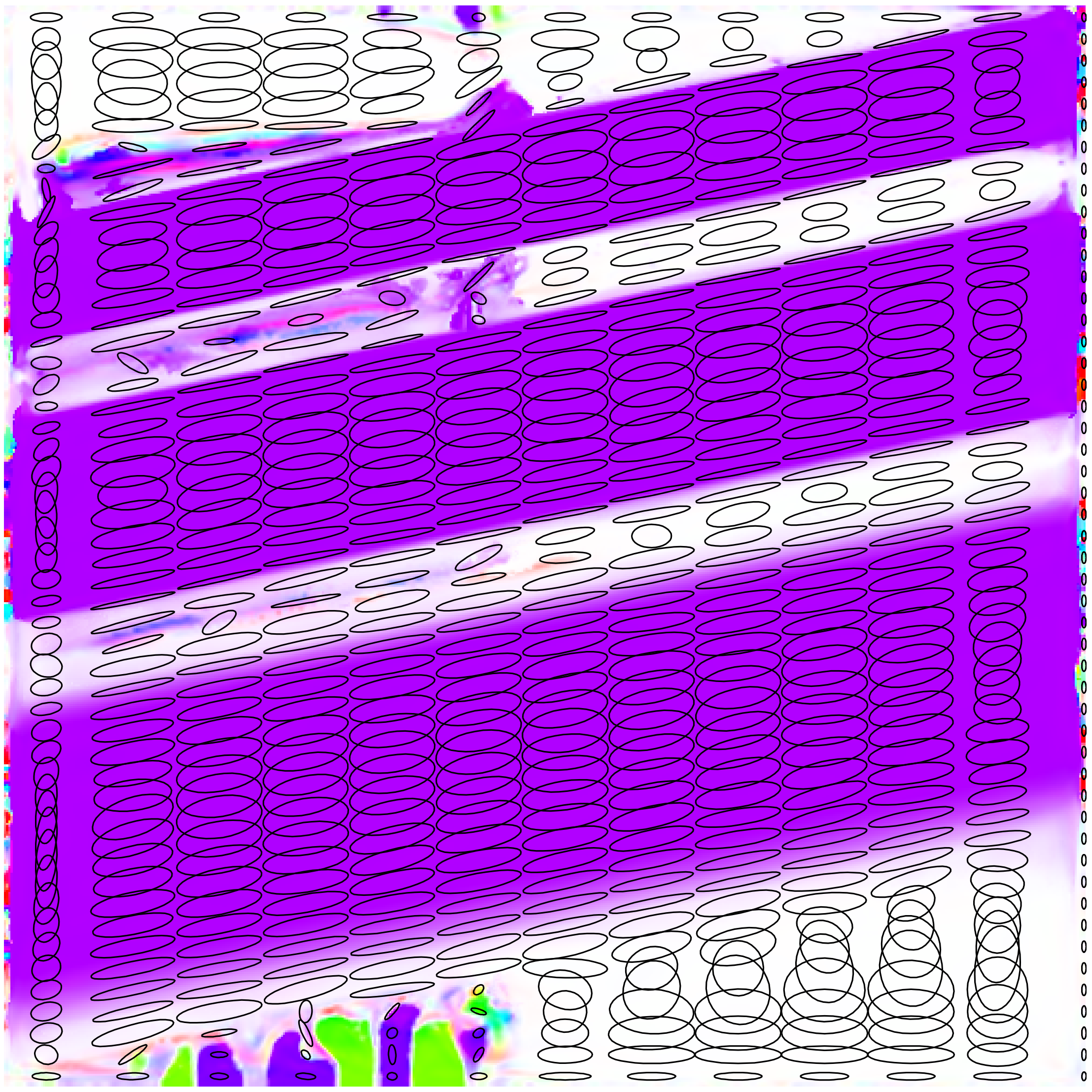
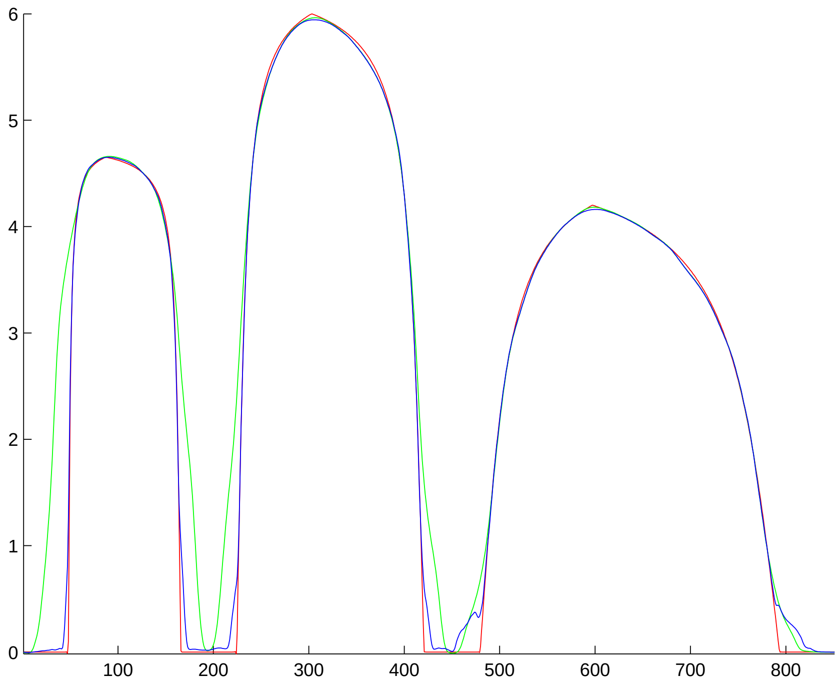
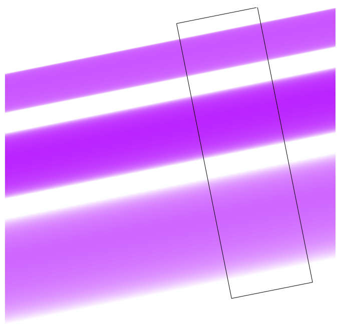


Figure 3.22: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Boundary layers*: Results of our approach with joint velocity estimation and window adaption, using the result from Fig. 3.18 as initialisation. The displacement field is overlaid with the contours of some of the window shapes. As expected, they align in parallel to the motion boundaries.

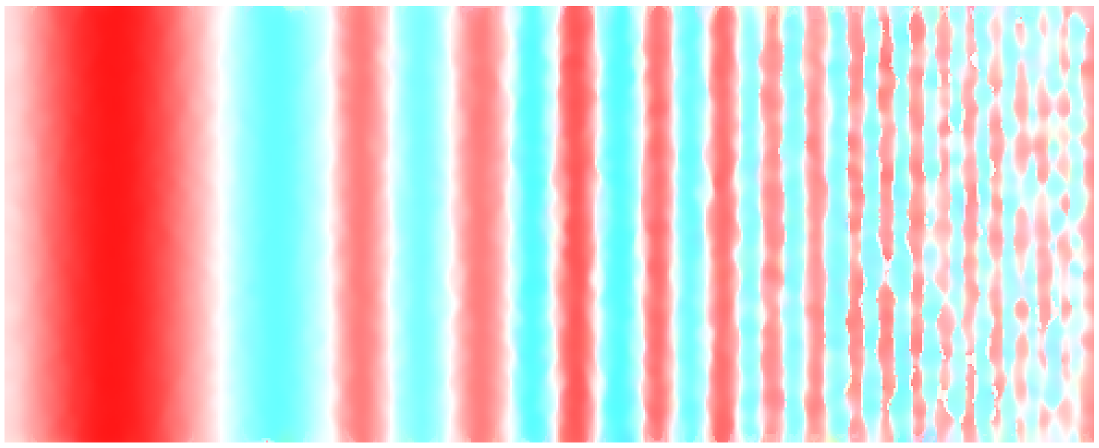


(a) displacement profiles

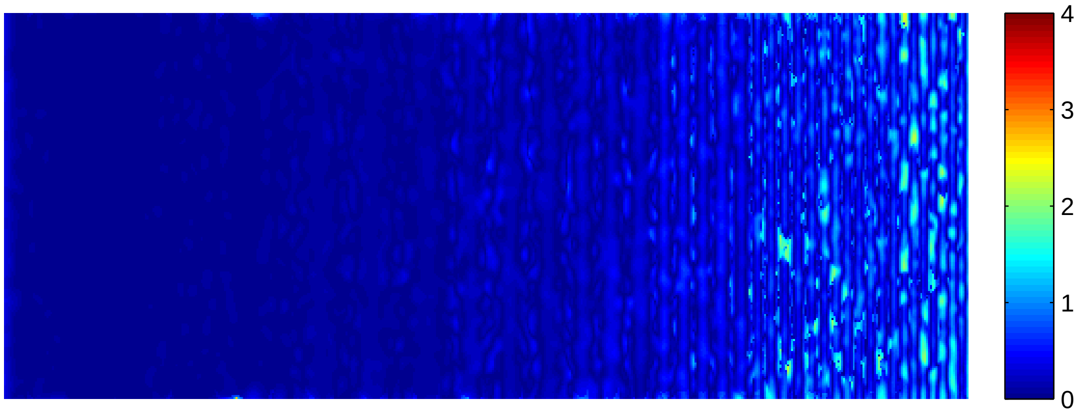


(b) location of the data used to create the profile

Figure 3.23: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Boundary layers*: Comparison of the reconstruction of the motion boundaries. (a) Profiles representing the displacement component parallel to the boundary, i.e. in the direction $(5, 1)$; Ground truth (red), correlation only (green) and with window adaption (blue). The displacements were extracted from the region highlighted in (b) and averaged in parallel to the boundary.
The approach with the fixed windows smoothes out the boundaries to a high degree. In contrast, the adapted windows reconstruct the ground truth profile much better for the left and middle profile, however give no improvement on the right one.

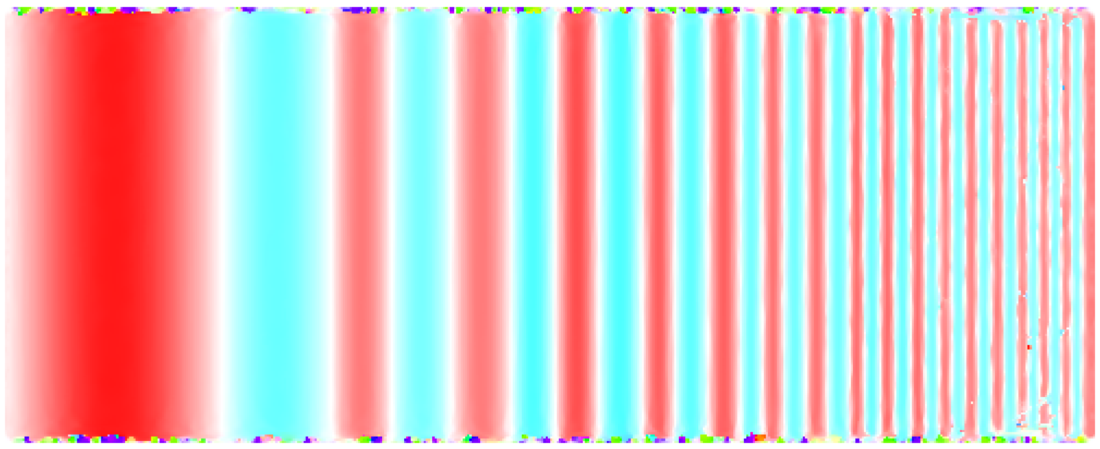


(a) displacement map (using $v_{\max} = 3$ for visualisation)

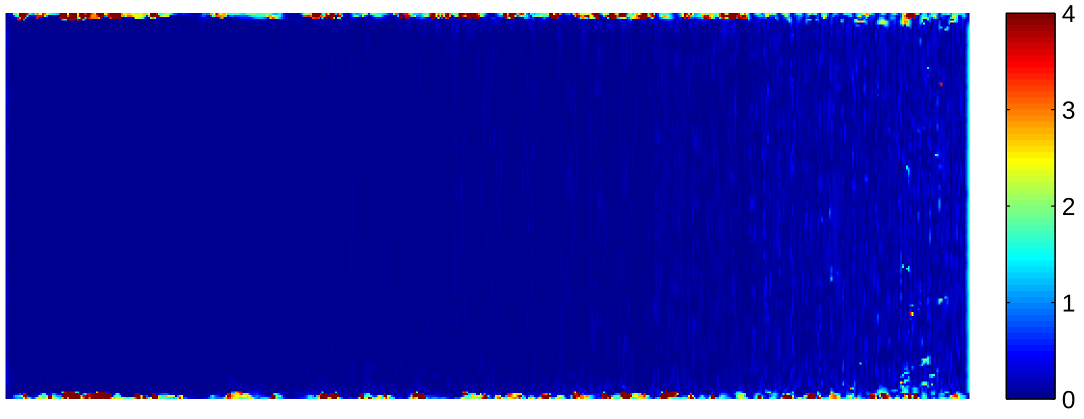


(b) error (scale truncated at 4px) compared to ground truth

Figure 3.24: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids I*: Results of our approach using *fixed* windows with $r = 8$. (a) The calculated displacement map, and (b) the error map with respect to the ground truth. Although the structure is reconstructed approximately, the non-adaptive method is unable to resolve the small structures.



(a) displacement map (scale truncated at $v_{\max} = 3$)



(b) error (scale truncated at 4px) compared to ground truth

Figure 3.25: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids I*: Results of the joint correlation and window adaption. (a) Displacement map, and (b) error map.

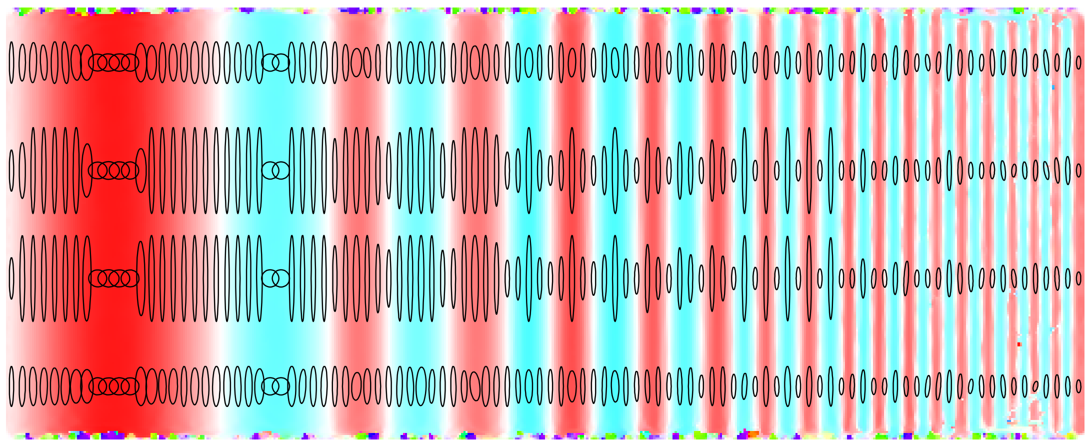
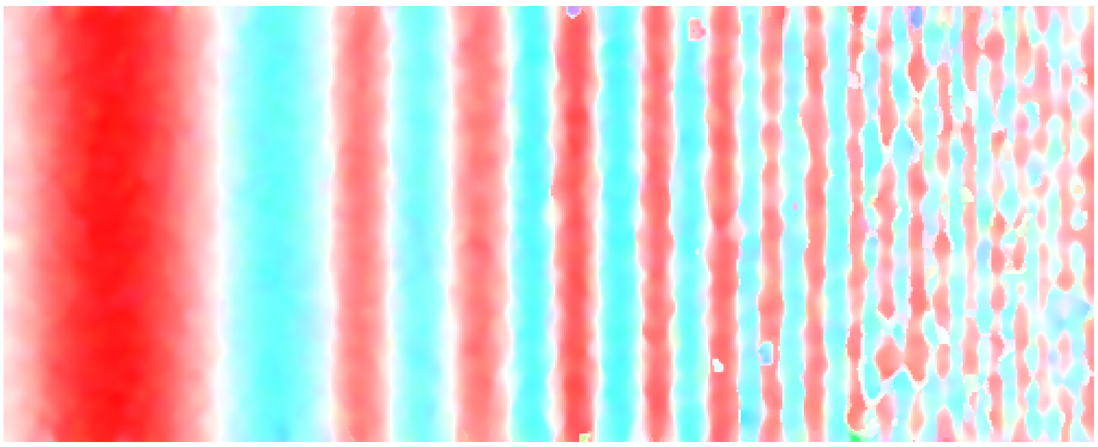
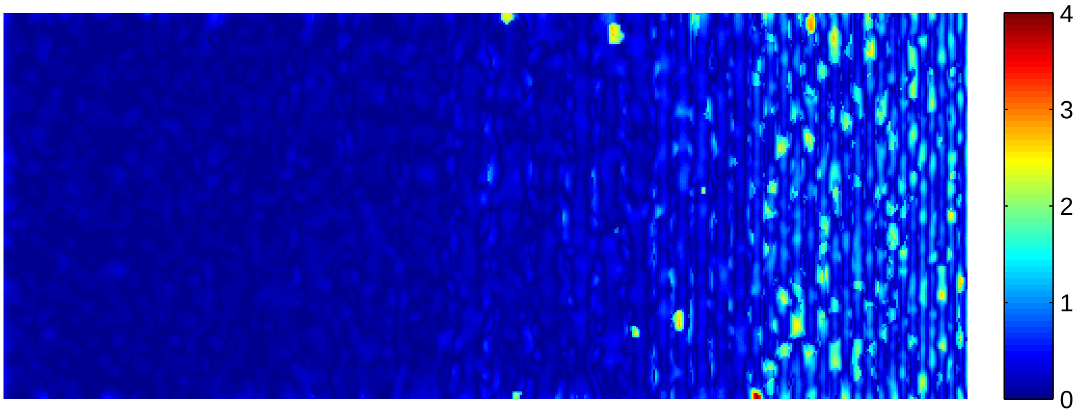


Figure 3.26: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids I*: Results of the joint correlation and window adaption. Displacement map with some representative window shapes superimposed.

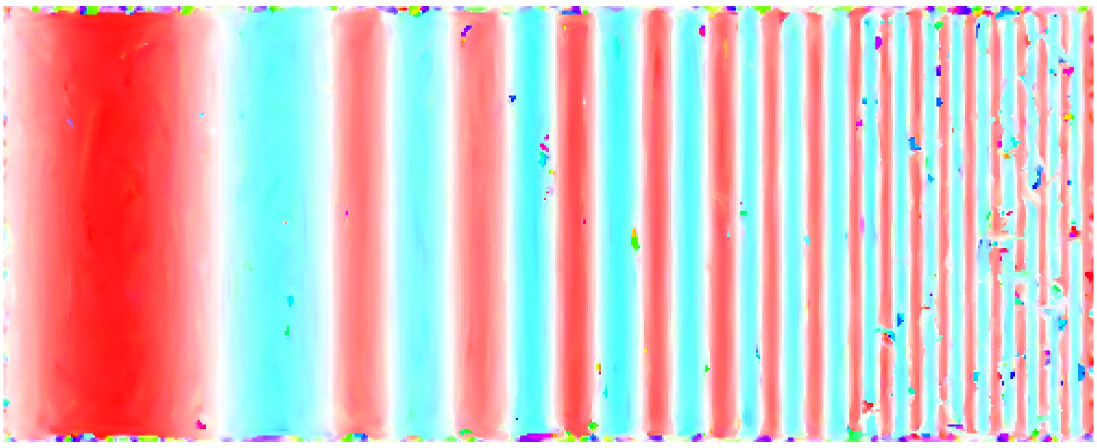


(a) displacement map (scale truncated at $v_{\max} = 3$)

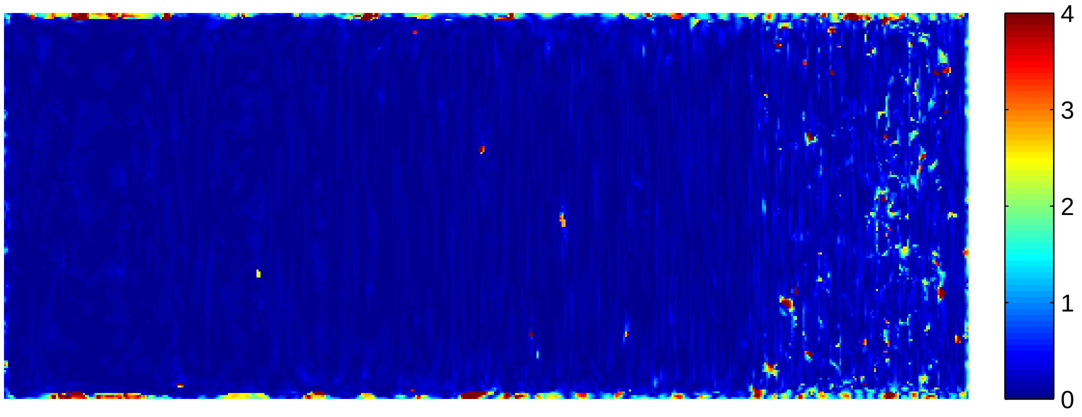


(b) error (scale truncated at 4px) compared to ground truth

Figure 3.27: Synthetic vector field, *PIV-Challenge 2005*, case A4, region *Sinusoids II*: Results of our approach using fixed windows with $r = 8$. (a) The calculated displacement field, and (b) the error map with respect to the ground truth. The amount of errors attest that this data set is more involved than *Sinusoids I* due to the supplemental image disturbances.



(a) displacement map (scale truncated at $v_{\max} = 3$)



(b) error (scale truncated at 4px) compared to ground truth

Figure 3.28: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids II*: Results of the joint correlation and window adaption. (a) Displacement map, and (b) error map.

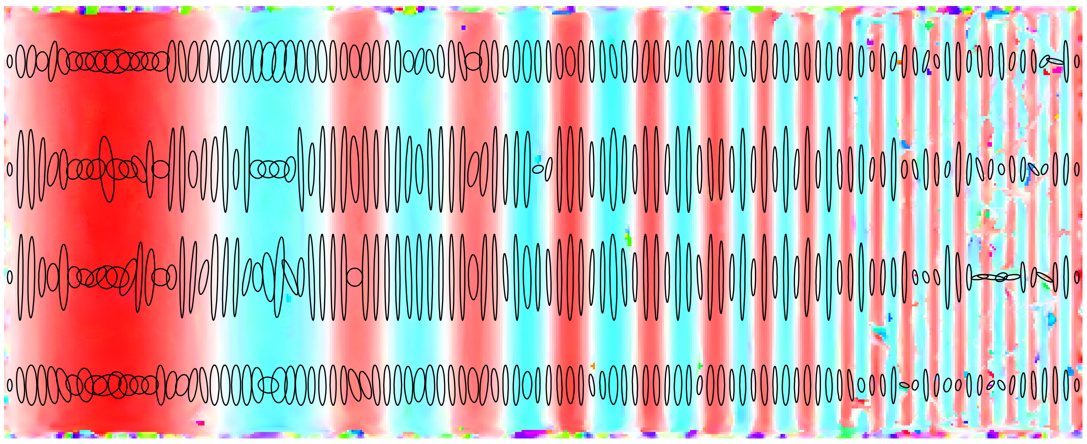


Figure 3.29: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids II*: Results of the joint correlation and window adaption. Displacement map with some representative window shaped superimposed.

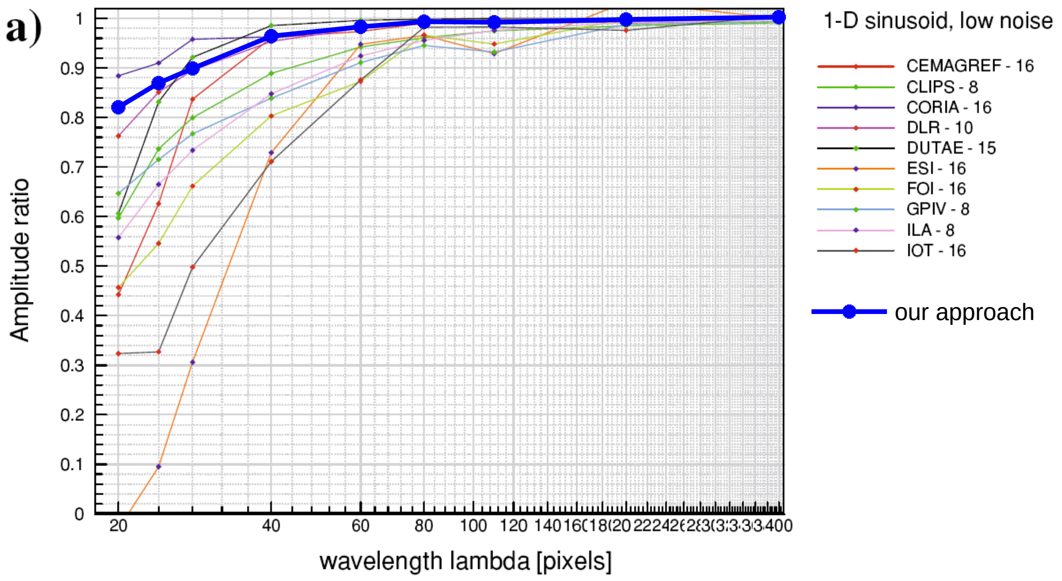


Figure 3.30: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids II*: Part 1 of the comparison of the amplitude response depending on the wavelength λ. We included the measurement of our approach into the plot taken from [49, Fig. 21a] where 19 implementations for PIV measurements were compared. The second part of the evaluation is presented in Fig. 3.31.

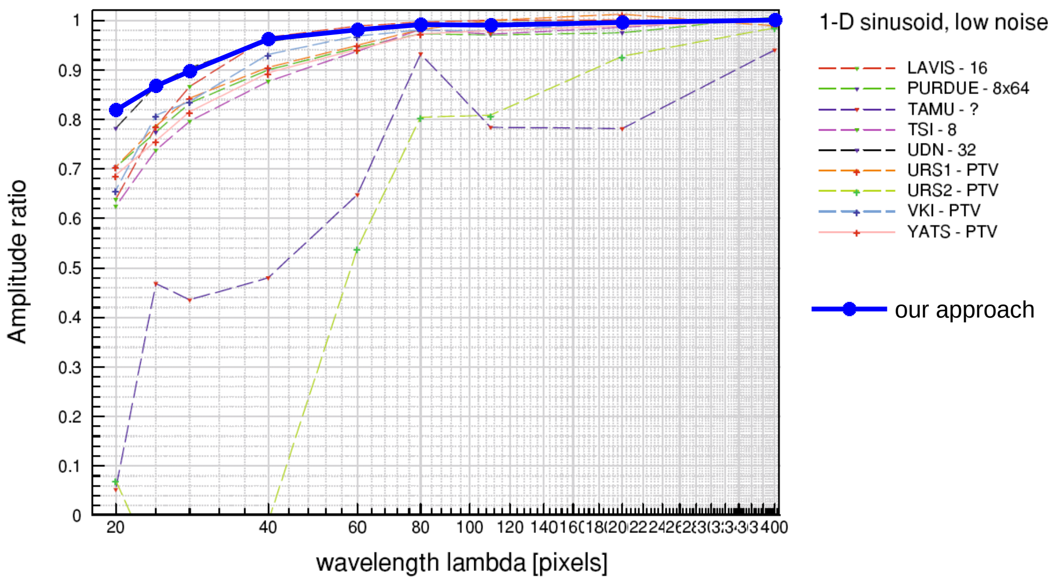


Figure 3.31: Synthetic vector field, *PIV-Challenge 2005*, *Case A4*, region *Sinusoids II*: Part 2 of the comparison of the amplitude response depending on the wavelength λ. We included the measurement of our approach into the drawings taken from [49, Fig. 21a] where 19 implementations for PIV measurements were compared in this way. The first part of the evaluation is presented in Fig. 3.30.

3.5.5 Experiments with Real Data

In the last experiment we apply our approach to a data set obtained from a real PIV experiment. The data was provided by Johan Carlier and shows the motion of water behind a cylinder. In Fig. 3.32 a sample of the image data is presented as well as an overview over the vector field. Each image is 1280px $\times$ 1024px in size and has a dynamics of 12bit. The time difference between two frames is 200 µs. For our experiments we used image pair number 600. Further details on the experimental setup can be found in [52]. Both the description and data are available for download at the *FLUID* project homepage [51].

Due to the lack of a ground truth displacement field we used a solution obtained from a commercial implementation of a correlation approach, the *Davis* software marketed by the *Lavision* company. The displacement field provided with a grid spacing of $a_{\mathrm{V}} = 8$ is depicted in Fig. 3.35.

As in the previous experiments, we calculated an initialisation for the adaptive approach by fixing the window shapes. The radii were set to $r \approx 19$px and five problem scales $\{1, 2, 4, 8, 16\}$ were used. This step was initialised by the reference solution to reduce the influence of inconsistent displacement estimations with the aim to focus on the effect on the window adaption in the next step. The resulting vector field is visualised in Fig. 3.33.

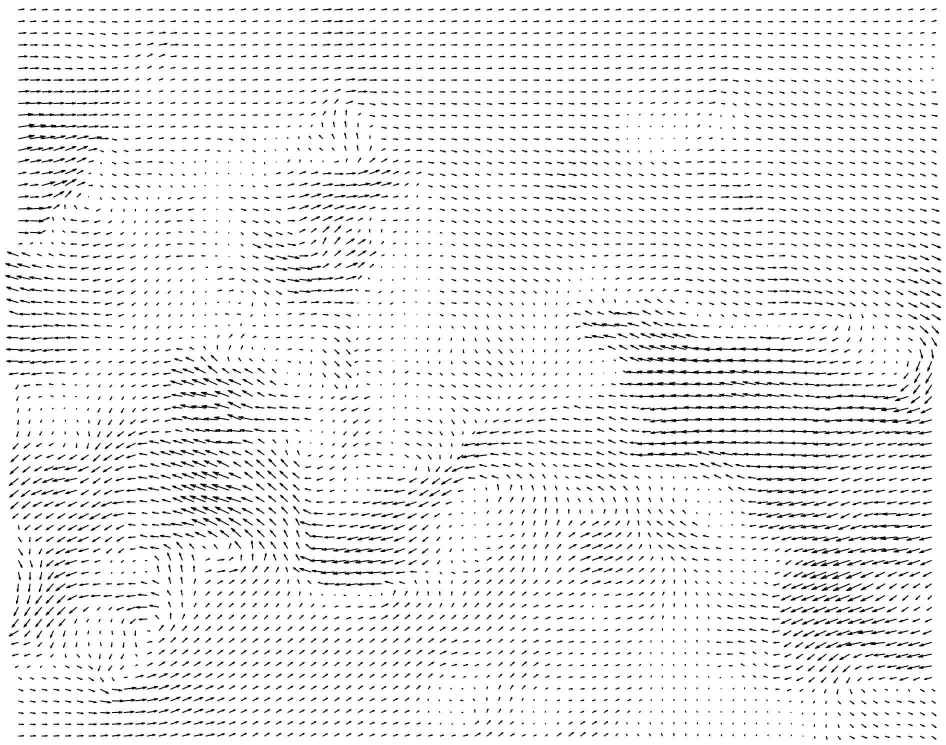
The obtained solution was refined by running the adaptive approach on this data set. The error model parameters were chosen as $\sigma = 100$ and $e_{\mathrm{outside}} = 0$. The window shapes were constrained by $r_{\min} = 3$ and $r_{\max} = 50$. Only the finest scale of the problem was used for calculation, i.e. $l_{\max} = 0$.

The resulting vector field can be found in Fig. 3.34. In addition, we compared the result of the adaptive approach to the reference solution in Fig. 3.36, using the similarity measurement defined in Sect. 3.5.1. The two solutions only differ considerably in few regions.

Finally, we present some detailed views on the calculated vector fields and visualise the window shapes. In Fig. 3.37 the location of the investigated regions are marked. Figure 3.38 illustrates how window sizes differ between homogeneous and turbulent areas. In Fig. 3.39 a situation is depicted, where the windows align perpendicular (and not in parallel) to the flow. A very complex vector field in presented in Fig. 3.40, where the advantage of freely rotatable window shapes become clear. In the region detailed in Fig. 3.41 our displacement field differs significantly from the reference solution.

It is the nature of real experimental data that we cannot give a clear statement, which of the results is “more correct”. Thus, we only discuss the differences between these two approaches: They mainly differ in regions of high gradients, where our approach seems to create significantly sharper structures which, however, do not need to be necessarily more correct. Interestingly, at the same time there are regions near the upper and lower image border, where we obtained more smooth solutions in regions of homogeneous motion. Thus, the approach seems to be capable to both prevail gradients in turbulent regions and to reduce noise in smooth regions by adaption the window shapes at the same time.

(a) detail of frame 1



(b) displacements

Figure 3.32: Real 2D PIV experiment: (a) Detail (100px × 100px) from the first of the two frames. (b) Overview over the fluid flow, determined by a correlation method. A mean vector field of about 12px to the right was subtracted everywhere.

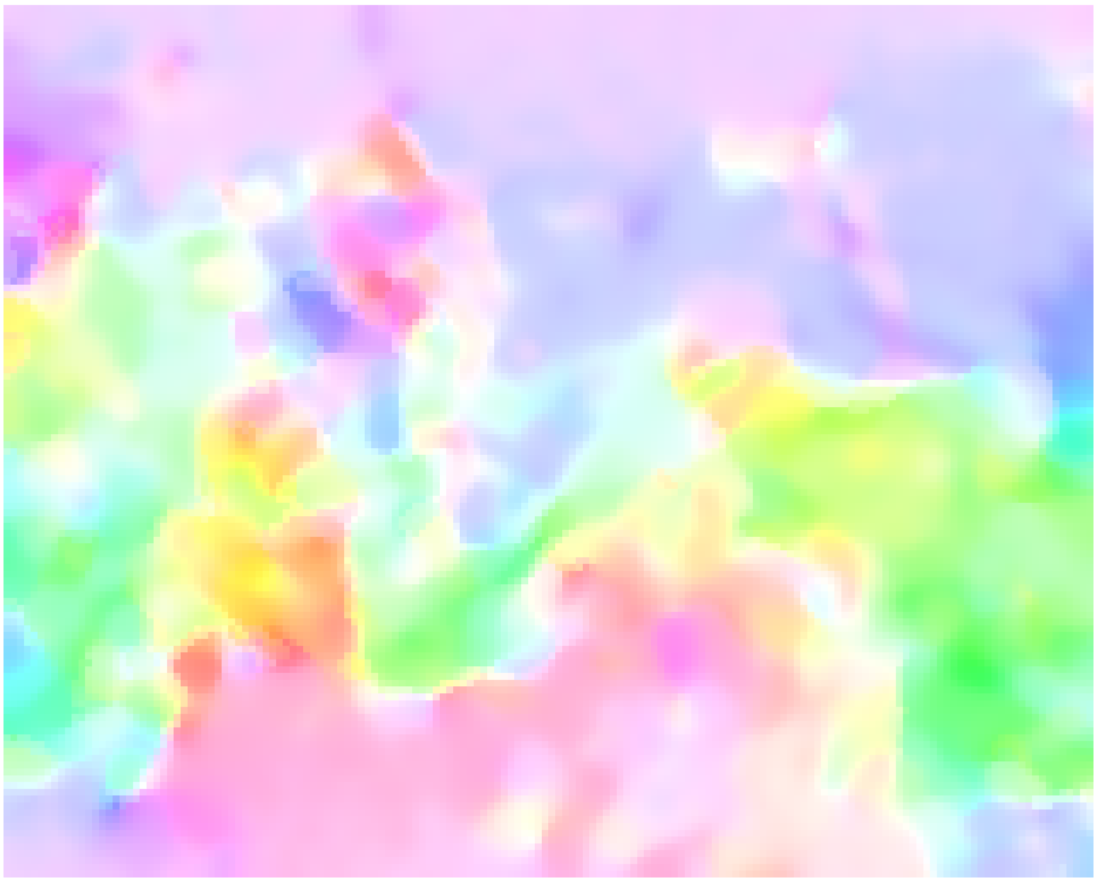


Figure 3.33: Real 2D PIV experiment: Results of applying our correlation approach with fixed window shapes ($r = 19.2px$). A mean vector field of about 12px to the right was subtracted everywhere. In the representation the displacements were limited to $v_{\max} = 5$.

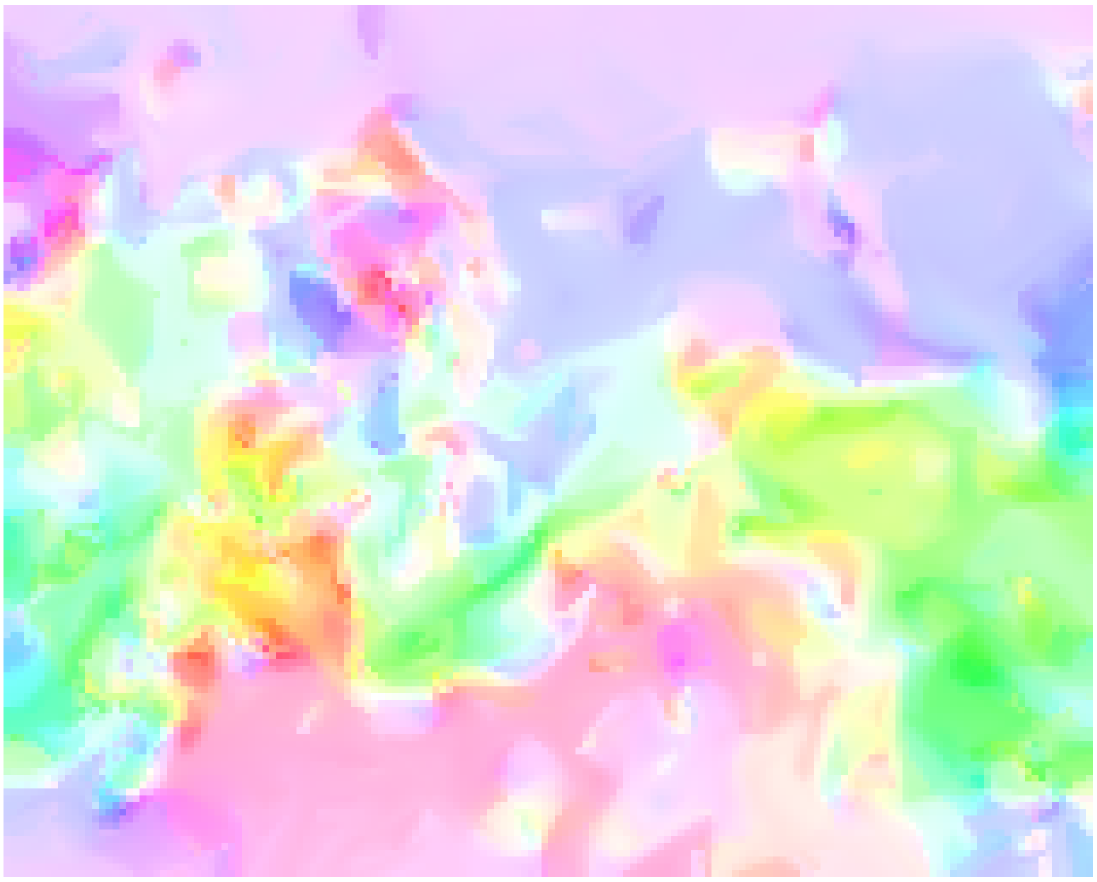


Figure 3.34: Real 2D PIV experiment: Displacement field calculated with our combined correlation approach with window adaption. A mean vector field of about 12px to the right was subtracted everywhere. In the representation the displacements were limited to $v_{\max} = 5$.

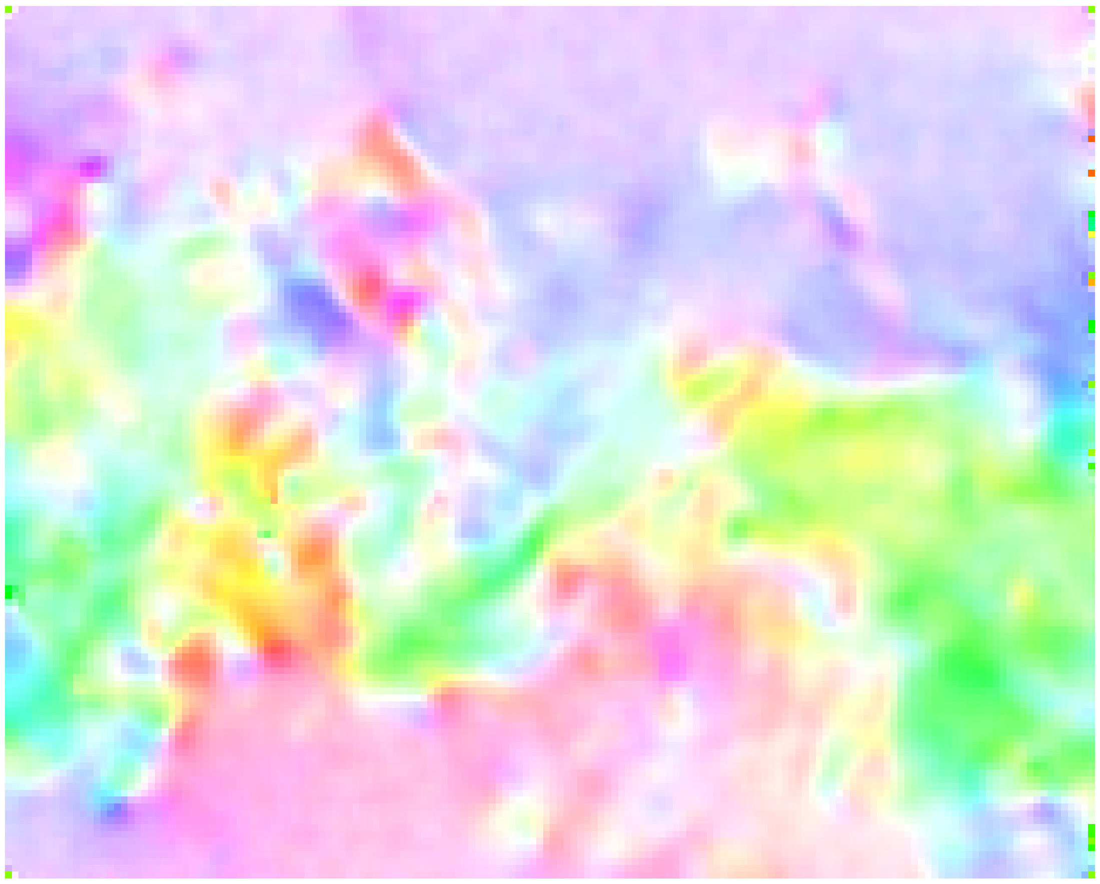
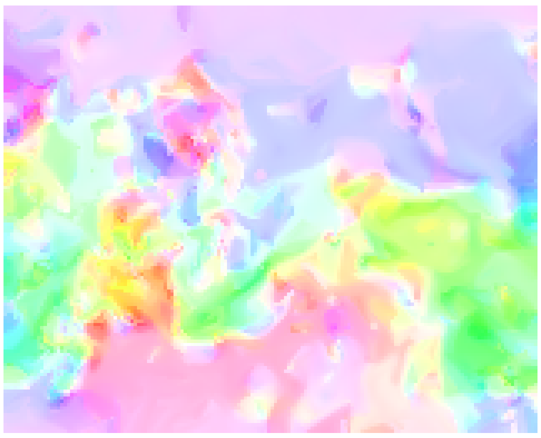
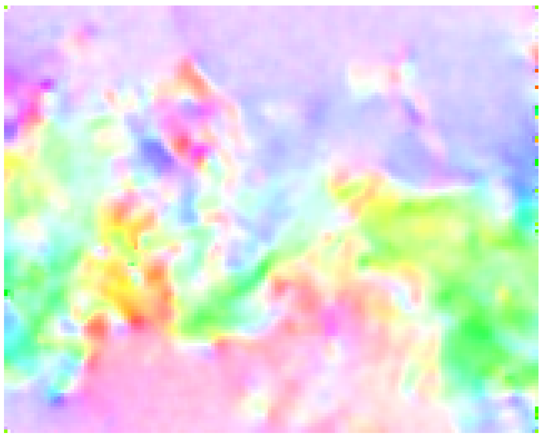
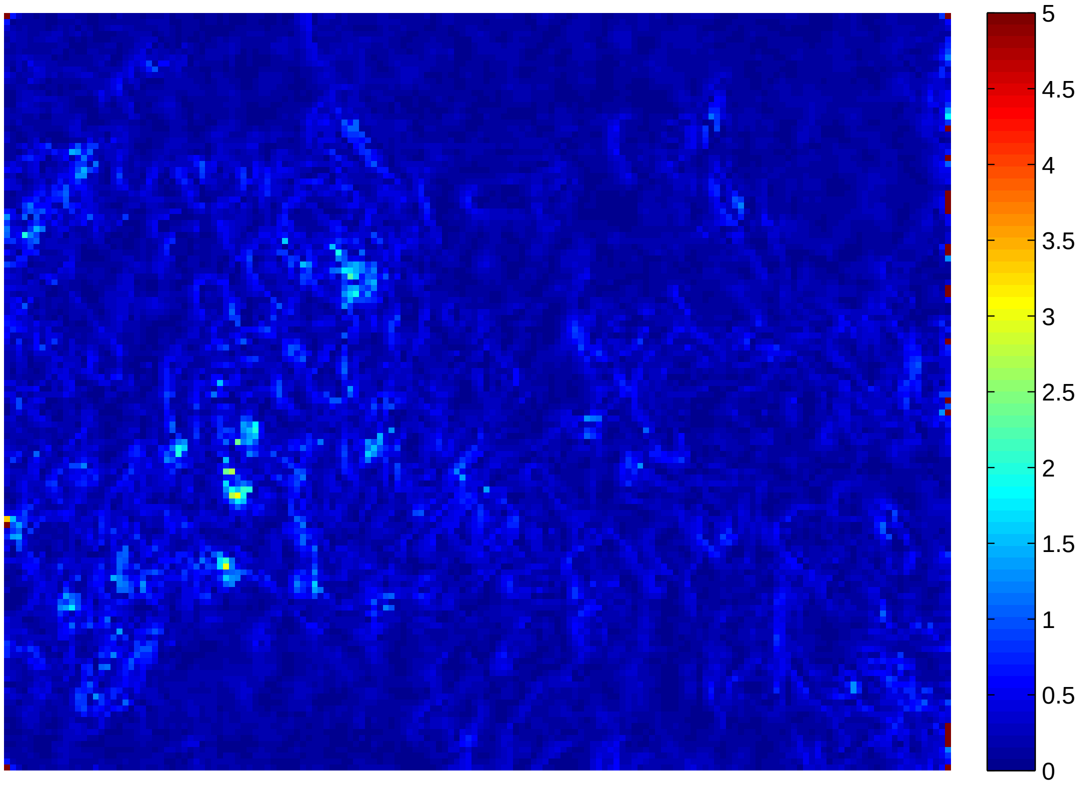


Figure 3.35: Real 2D PIV experiment: Reference displacement field calculated with a commercial PIV software. A mean vector field of about 12px to the right was subtracted everywhere. In the representation the displacements were limited to $v_{\max} = 5$.


(a) joint correlation and window adaption


(b) reference



(c) difference map between (a) and (b)

Figure 3.36: Real 2D PIV experiment: Comparison of the result of (a) our joint correlation and window adaption approach (same as Fig. 3.34) to the (b) reference (same es Fig. 3.35). (c) The difference measurement between these two solutions is visualised with the scale truncated at 5.

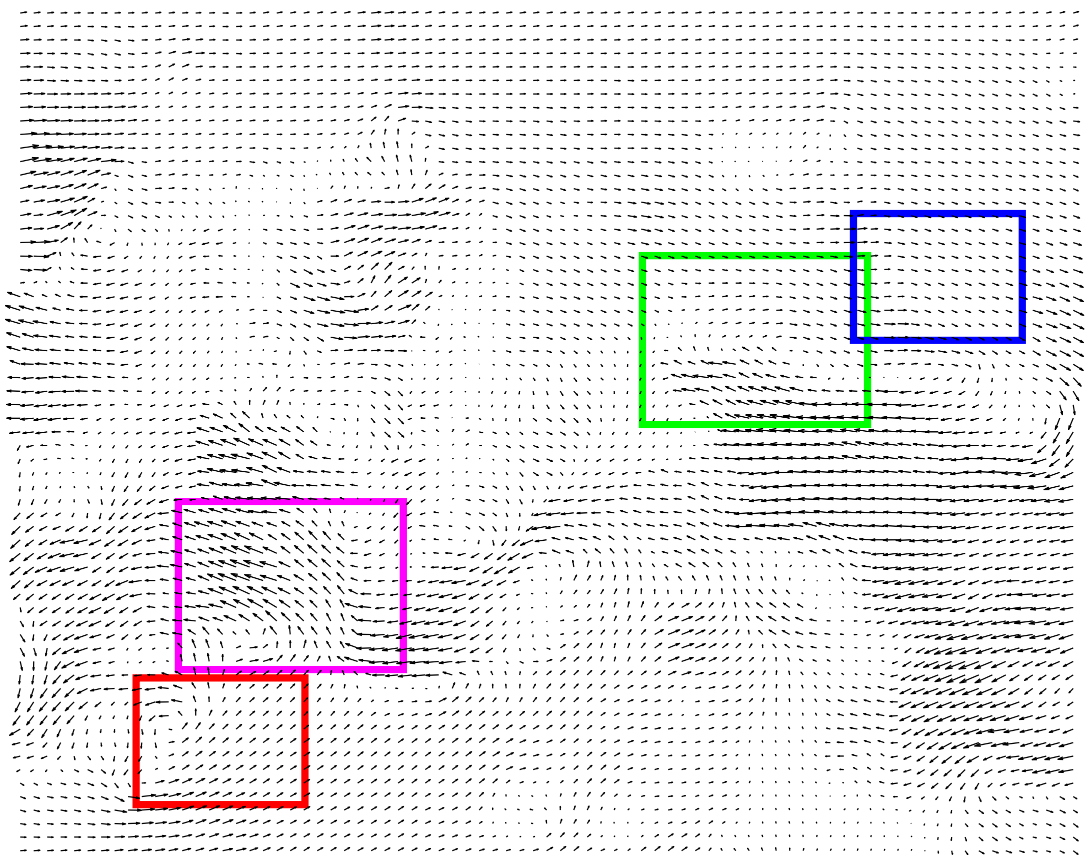


Figure 3.37: Real 2D PIV experiment: Overview over the displacement field with four regions marked which are studied in detail in Figures 3.38 (red), 3.39 (blue), 3.40 (green) and 3.41 (magenta).

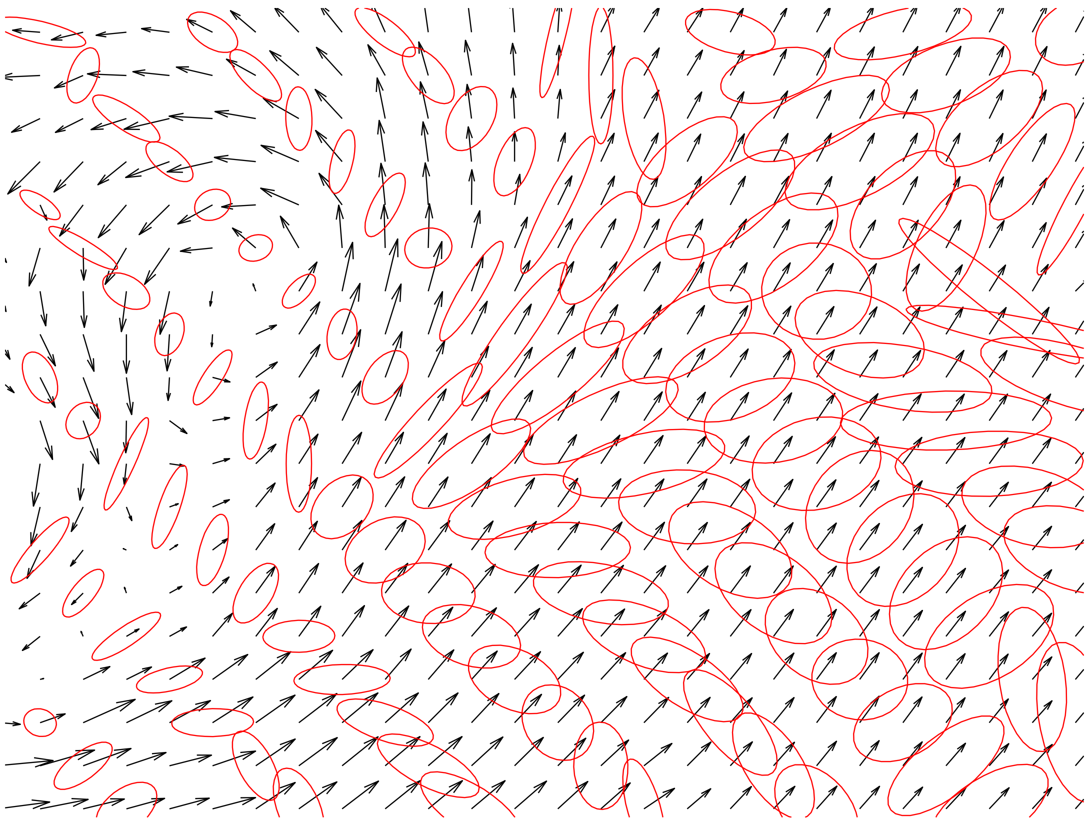


Figure 3.38: Real 2D PIV experiment: Detailed view of the jointly estimated displacements (after subtracting an average displacement) and some of the adapted correlation windows (scaled down by factor 2 for clarity). Window size increases in areas of homogeneous motion (middle), and decreases in presence of gradients (upper left).

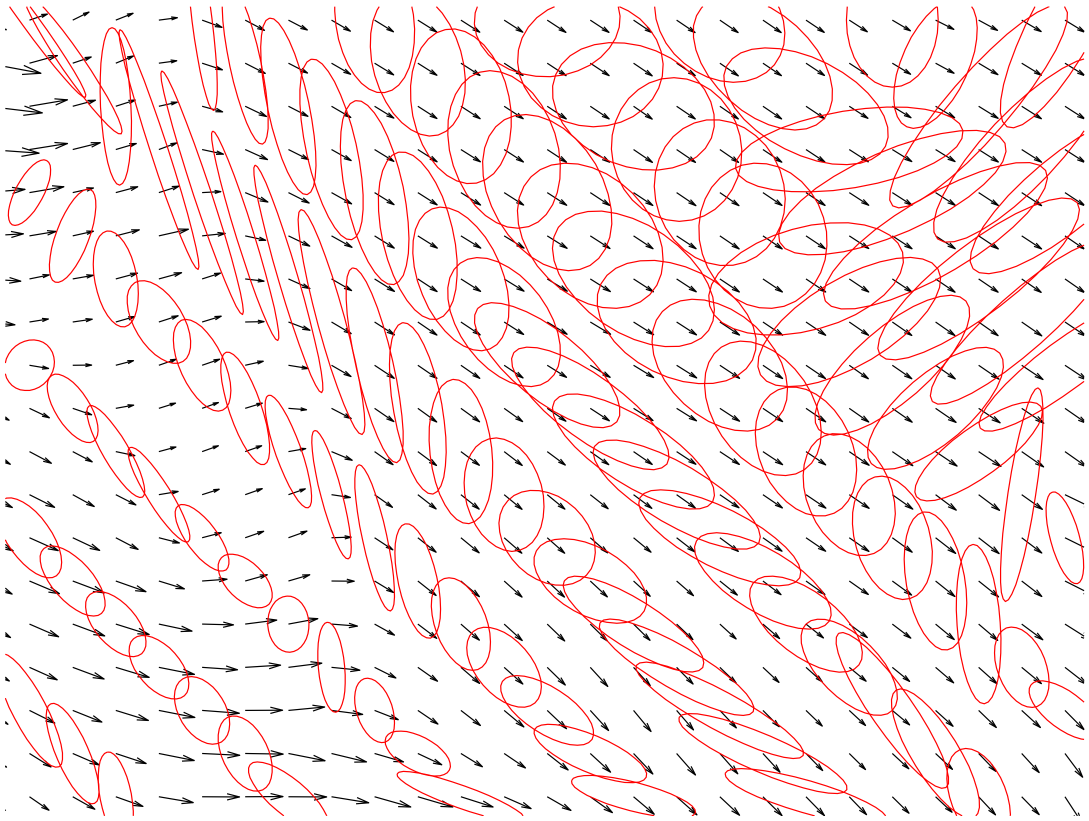


Figure 3.39: Real 2D PIV experiment: Detailed view of the jointly estimated displacements (after subtracting an average displacement) and some of the adapted correlation windows (scaled down by factor 2 for readability). Windows are stretched perpendicular to the flow direction (upper left, along the “wave”).

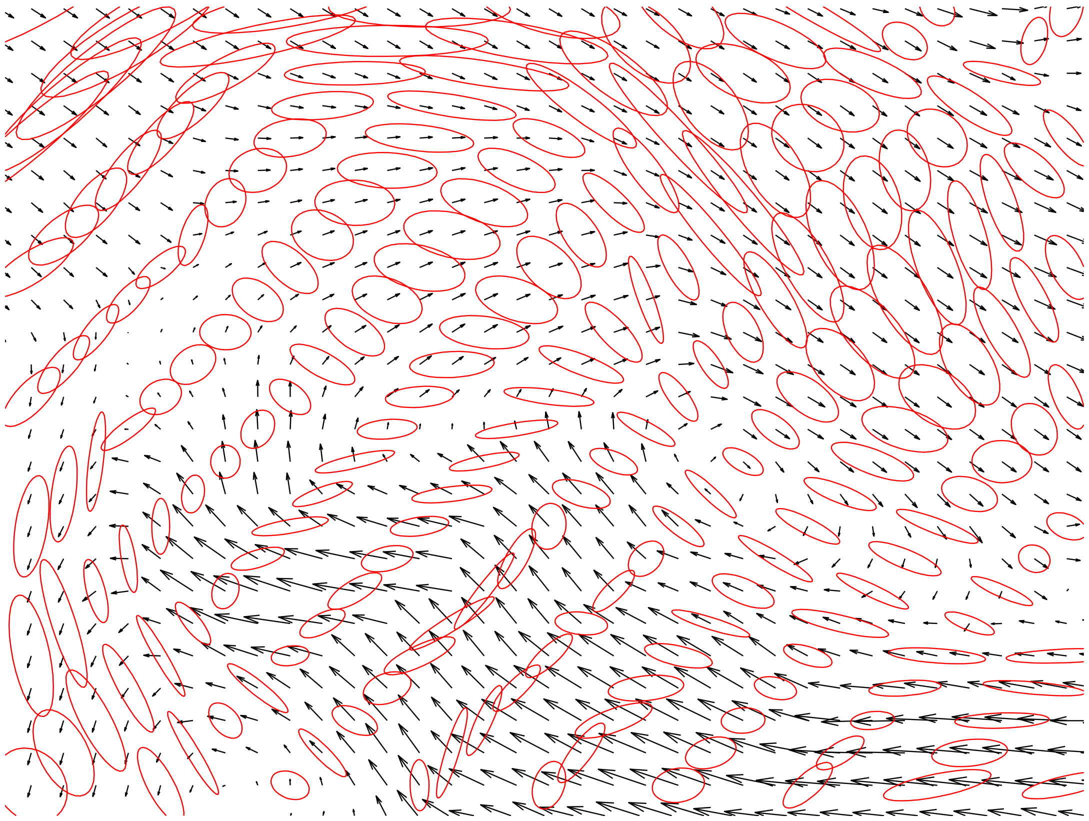
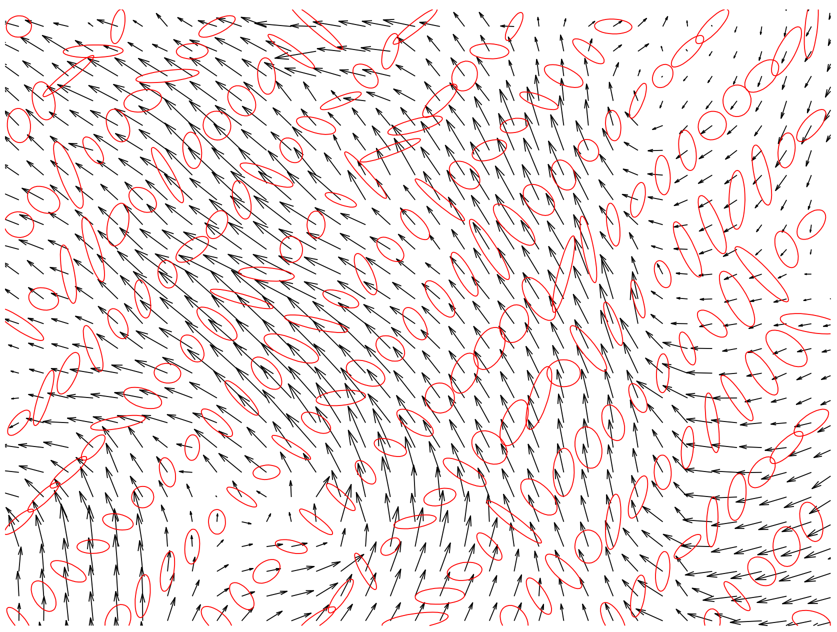


Figure 3.40: Real 2D PIV experiment: Detailed view of the jointly estimated displacements (after subtracting an average displacement) and some of the adapted correlation windows (scaled down by factor 2 for readability). Windows adaption in a complex vector field around a vortex.

(a)



(b)

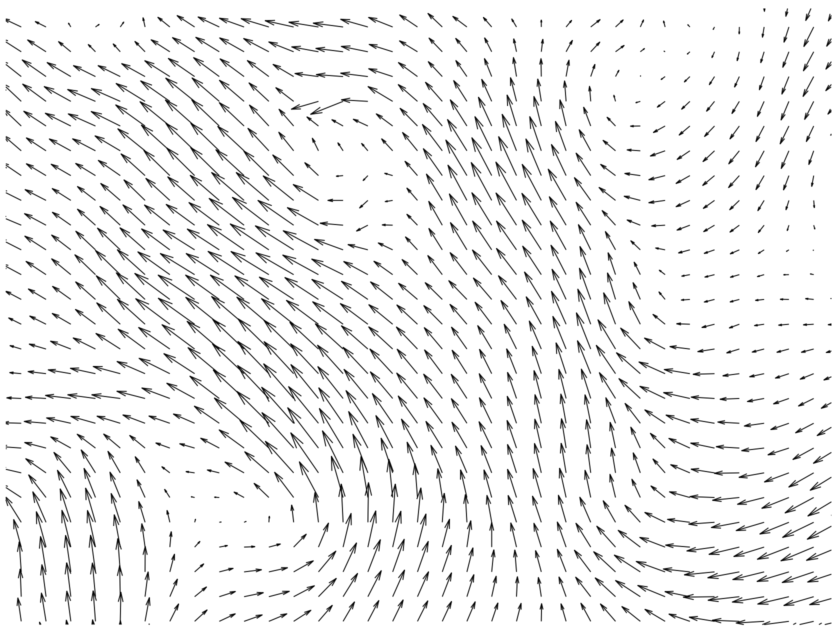


Figure 3.41: Real 2D PIV experiment: (a) Detailed view of the jointly estimated displacements (after subtracting an average displacement) and some of the adapted correlation windows (scaled down by factor 2 for readability). (b) Reference vector field. In the considered region the vector field differ considerably (see also Fig. 3.36): In the upper half (centre), the reference method locates an additional vortex.

3.6 Conclusion and Further Work

3.6.1 Summary and Conclusion

In this chapter we presented a variational approach to motion estimation for image data obtained from PIV experiments. We based the displacement estimation on the correlation measurement which has proven to be beneficial in PIV applications. However, we applied continuous optimisation instead of performing an exhaustive search for the optimal displacements vectors. Furthermore, we used a continuously parameterisable Gaussian window function with the aim to adapt to velocity gradients in the observed motion. A sound error model was defined to select the window shapes, which directly formulates the aim to increase the accuracy of the displacement measurement. The velocity estimation and window adaption were combined in a single optimisation problem, which was solved jointly using methods for non-linear and non-convex optimisation.

In the last section we presented results for experiments on real and synthetic PIV data. We demonstrated that the proposed window selection scheme adapts to velocity gradients in the vector field as it was intended in the design, and increase accuracy by resolving also small structures. Owing to the window adaption, our approach outperformed most of 19 other approaches in a direct comparison. At the same time, precision is also increased in homogeneous regions by increasing the window size.

3.6.2 Further Work

For both the displacement estimation as well as the window adaption we did not incorporate any spatial regularisation terms, although the variational formulations allows to do so. For example, physical priors on the fluid flow can be added to further improve accuracy. However, this makes the solution of the optimisation problem more involved.

The convergence of the method for this highly complex optimisation problem could only be showed experimentally, a thorough analysis remains an open issue. Furthermore, the computational speed of the current solver implementation is at a low level typical for a proof of concept. A couple of starting points for the improvement of the performance exist, including parameter tuning, initialisation of the displacements from a classical correlation method, and parallelisation.

The approach to formulate the choice of the window shape as the minimisation of an error model function has proven well. However, our definition of the objective function can probably be improved for example by incorporating further expert knowledge, such as local seeding density or image noise level which can be measured a priori. Furthermore, the error model definition and its parameters lack a physical interpretation. The error model can also be interpreted as a confidence measure (see e.g. [53]) of the estimated measurement which incorporates the variations of the velocities and provides feedback for the estimation process. Further refinements of the model could profit from this link.

The window adaption based on an error model can also be transferred to optical flow approaches such as that of Lucas and Kanade [29]. Furthermore, other, possibly more complex, window shapes can be defined.

4 Dual Decomposition for Convex Optimisation

4.1 Overview

4.1.1 Introduction and Motivation

Variational problems have proven beneficial in many image processing and analysis applications. These include motion estimation problems, of which we presented one instance in Chap. 3. In contrast to local methods, variational approaches allow to directly incorporate prior knowledge on spatial dependencies, such as physical constraints, however, with the drawback that parallelisation is not straightforward anymore.

The increasing sensor resolution used to obtain image data demands for adapted solving strategies to handle the huge amount of data. For example, in experimental fluid dynamics, methods such as *Tomographic PIV* aim for 3D data sizes of up to 4000 by 4000 by a few hundred voxels. The accompanying variational problem descriptions are too large to be handled by a single standard hardware node anymore. For this reason, the problem has to be split up and distributed to parallel hardware. Although the focus of this work lies on the feasibility of the solution of such large problems, it can be expected that decomposition also reduces computation time.

In this work we address the decomposition of the class of convex and unconstrained quadratic problems, which includes the discretised version of several important variational approaches. The basic strategy is to subdivide the originally intractable problem into a set of smaller, yet convex quadratic problems, which can be solved with standard methods on off-the-shelf hardware. Due to the inevitable dependencies, the subproblems have to communicate to compute a solution of the original, non-decomposed problem.

4.1.2 Related Work and Contribution

Performance improvements are just one advantage of problem decomposition if tasks are distributed to parallel hardware. However, the main motivation of our work is to permit the solution of large variational problems, which exceed the memory limits of recent hardware. Thus, our focus is on the accurate solution of the decomposed problem and we will not go into technical details on synchronisation between parallel computing nodes.

The use of variational domain decomposition [54] for motion estimation has been introduced by Kohlberger et al. [55]. However, this approach is not applicable to variational models involving higher-order regularisation, such as [17], because subproblems in inner domains become inherently singular, and the corresponding handling of boundary conditions becomes involved.

In this work, we therefore employ the idea of *Dual Decomposition*, or *Lagrangian Relaxation* [56, 57], for solving variational problems which can be formulated as a convex and unconstrained quadratic optimisation problem. Some of the related decomposition methods in literature are: The *Augmented Lagrangian Decomposition* [57], *Optimality Condition Decomposition* [58] and *Dantzig-Wolfe decomposition algorithm* [59, Chap. 8.2] are variations of the Dual Decomposition. While they have in common, that their subproblems are synchronised by updating the dual variables of the decomposed problem, parts of the primal variables take this role in the *Primal Decomposition* methods [60]. In Sect. 4.3.5 we detail on the interrelation of these methods.

Another approach to solve large linear systems, as they occur in quadratic optimisation problems, are *Row-Action Methods* [61], which consider only parts of the problem description at a time. A modification, the *variable-block Algebraic Reconstruction Techniques*, allows to process the problem block-wise and in parallel.

Furthermore, we propose a method for splitting up the objective function of the initial problem as required by the Dual Decomposition scheme. Our approach guarantees convexity of the subproblems, which makes them easy to solve, using established and efficient methods (see [62]). The initial problem can then be solved as several smaller, independent convex problems and computationally cheap synchronisation steps, without changing the overall objective. We describe an extension that allows to improve the numerical properties of the underlying subproblems, hence improving their convergence speed. General results on convergence rate and conditions of the overall problem with respect to the decomposition parameters (dual variable update, decomposition, subproblem regularisation) are presented. We demonstrate our approach by means of three relevant variational problems from image processing. Error measurements in comparison to single-domain solutions are presented. The results of the work presented here were partially published in [63].

4.1.3 Organisation

This chapter summarises the relevant theory behind our approach. In Sect. 4.2 we restate the definition of quadratic optimisation problems, and present the general idea of the Dual Decomposition method in Sect. 4.3.

In Chap. 5 we propose a decomposition method for convex and unconstrained quadratic problems and evaluate it by means of three exemplary variational approaches for image processing tasks.

4.2 Quadratic Optimisation Problems

4.2.1 Definition

A linearly constrained quadratic optimisation problem, or *quadratic program (QP)*, is defined [62, 2] as

$$(QP) \qquad p^* := \inf_{x\in\mathbb{R}^n} \frac{1}{2}x^\top Ax + b^\top x \;, \qquad (4.1)$$
$$\text{s.t. } Gx \le h \;,$$
$$Px = q \;,$$

with $A \in \mathcal{S}^n$, $b \in \mathbb{R}^n$, $G \in \mathbb{R}^{m\times n}$, $h \in \mathbb{R}^m$, $P \in \mathbb{R}^{p\times n}$, $q \in \mathbb{R}^p$ and optimal value p^*. If we restrict ourselves to positive semidefinite matrices A, i.e. $A \in \mathcal{S}^n_+$, the problem is convex and hence any local minimiser is also a global minimum. In addition, for positive definite matrices, i.e. $A \in \mathcal{S}^n_{++}$, the problem is strictly convex and has a unique minimiser.

4.2.2 Solving

From optimisation theory follows that for *convex* quadratic problems any solution that satisfies the so called *Karush-Kuhn-Tucker (KKT)* conditions is a global optimum. With additional variables $\lambda \in \mathbb{R}^m$ and $\mu \in \mathbb{R}^p$, the *dual variables*, the KKT conditions read

$$\begin{aligned}
&Gx - h \le 0 \;,\\
&Px = q \;,\\
&Ax + b + G^\top\lambda + P^\top\mu = 0 \;,\\
&\lambda \ge 0 \;,\\
&\lambda^\top(Gx - h) = 0 \;.
\end{aligned}$$

In this work we will only consider problems with positive semidefinite matrices A with inequality constraints. Furthermore we assume, that the optimal value is bounded below, i.e. $p^* > -\infty$. Then it is sufficient to solve the KKT conditions which simplify to the linear equality system

$$\begin{aligned}
Ax + P^\top\mu &= -b \;,\\
Px &= q \;.
\end{aligned}$$

The usual way to solve this equation system is to use conjugate gradient, interior point or active set methods. For details we refer to [62] and [2].

4.3 Dual Decomposition

4.3.1 Decomposition of the Objective Function

Given a convex optimisation problem,

$$\inf_{u\in\mathbb{R}^n} f(u) \;, \qquad (4.2)$$

with f convex. We assume that the objective function is separable into d *convex* functions f_l, so that

$$f(u) = \sum_{l=1}^{d} f_l(x_l, y) \ .$$

The variable vector u is split up into one set of *private variables* $x_l \in \mathbb{R}^{n_l}$ for each subfunction and a common set of *global* variables $y \in \mathbb{R}^{n_\mathrm{C}}$. The latter variables are involved in at least two subfunctions and are therefore also referred to as *complicating variables*, because they prevent that each subproblem can be solved independently. In contrast, the variables x_l are *internal* – also denoted as *local* – and only occur in f_l. For conciseness we will abbreviate the set of subfunction indices by $\mathcal{L} := \{1, \ldots, d\}$ and denote any set of subproblem-specific objects by $\{x_l\}_\mathcal{L} := \{x_l \,|\, l \in \mathcal{L}\}$. Then the decomposed problem reads

$$\inf_{\{x_l\}_\mathcal{L}, y} \ \sum_{l \in \mathcal{L}} f_l(x_l, y) \ . \tag{4.3}$$

In order to efficiently solve the problem using the decomposition algorithm described in this work, the number of complicating variables should be small compared to the local ones. In the trivial case of $n_\mathrm{C} = 0$, each subproblem can be solved independently and the results can be assembled easily to a solution of the original problem.

If we duplicate the complicating variables for each subfunction and enforce their identity by an equality constraint, we get

$$\inf_{\{x_l, y_l\}_\mathcal{L}} \ \sum_{l \in \mathcal{L}} f_l(x_l, y_l) \tag{4.4}$$

$$\text{s.t. } y_1 = y_2 = \ldots = y_d \ , \tag{4.5}$$

which is equivalent to (4.3). Note that now every function has its own set of variables. These, however, are linked via the so called *complicating* or *consistency constraints* which prevent independent treatment.

4.3.2 Dual Decomposition for Two Subfunctions

The authors of [60, 64] describe a method to solve convex optimisation problems that can be reformulated as (4.3). They make use of the *Lagrange dual problem* formulation, also referred to as *Lagrangian relaxation* in [57]. For demonstrating the basic idea, we restrict ourselves to two subfunctions for now, i.e. $\mathcal{L} = \{1, 2\}$ and $d = 2$:

$$\inf_{\{x_l, y_l\}_\mathcal{L}} \ f_1(x_1, y_1) + f_2(x_2, y_2) \tag{4.6}$$

$$\text{s.t. } y_1 = y_2 \tag{4.7}$$

In order to solve an equivalent *unconstrained* problem, we include the equality constraints into the objective function by adding a term – the set indicator function for the constraint set – that vanishes exactly if $y_1 = y_2$ and takes a positive infinite value otherwise:

$$\begin{aligned} p(\{x_l, y_l\}_{\mathcal{L}}) &:= f_1(x_1, y_1) + f_2(x_2, y_2) + \sup_{\lambda \in \mathbb{R}^{n_C}} \lambda^\top (y_1 - y_2) \\ &= \begin{cases} f_1(x_1, y_1) + f_2(x_2, y_2) & \text{if } y_1 = y_2 \\ +\infty & \text{if } y_1 \neq y_2 \end{cases} \\ &= \sup_{\lambda \in \mathbb{R}^{n_C}} L(\{x_l, y_l\}_{\mathcal{L}}, \lambda) , \end{aligned}$$

where L is the Lagrange function for problem (4.6),

$$L(\{x_l, y_l\}_{\mathcal{L}}, \lambda) = f_1(x_1, y_1) + f_2(x_2, y_2) + \lambda^\top (y_1 - y_2) ,$$

and $\lambda \in \mathbb{R}^{n_C}$ are the so called *dual variables*, or *Lagrange multipliers.* Then, problem (4.6)-(4.7) can be written as the following unconstrained optimisation problem:

$$\text{(P)} \qquad p^* := \inf_{\{x_l, y_l\}_{\mathcal{L}}} \sup_{\lambda} L(\{x_l, y_l\}_{\mathcal{L}}, \lambda)$$

The corresponding *Lagrange dual problem* is defined as

$$\text{(D)} \qquad d^* := \sup_{\lambda} \inf_{\{x_l, y_l\}_{\mathcal{L}}} L(\{x_l, y_l\}_{\mathcal{L}}, \lambda) , \tag{4.8}$$

which is a concave optimisation problem in λ, even for non-convex primal problems, see [2, Chap. 5]. In addition, the optimal values p^* and d^* of the primal and dual problem, respectively, are related by the *weak duality* inequality [57, Proposition 5.1.3],

$$p^* \geq d^* .$$

However, under certain mild circumstances, *strong duality* holds, i.e. the *duality gap* $p^* - d^*$ vanishes:

Theorem 4.3.1 (Strong duality for equally constrained optimisation problems)**.** *Consider an optimisation problem with equality constraints,*

$$\begin{aligned} f^* := \inf_{x \in \mathbb{R}^n} &\; f(x) \\ s.t.\ Ax &= b \end{aligned}$$

with convex objective function $f : \mathbb{R}^n \mapsto \mathbb{R}$ and $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$. If the optimal value f^* is finite and there exists a feasible $\overline{x} \in \mathbb{R}^n$, i.e. $A\overline{x} = b$, then there is no duality gap, so $p^* = d^*$.

Proof. see [57, Proposition 5.3.2] □

For problem (4.6)-(4.7), all conditions are fulfilled: The objective function $f(x)$ is convex and bounded below by assumption, and the equality constraints hold for $y_1 = \ldots = y_d$ (and any $\{x_l\}_{\mathcal{L}}$). Thus, strong duality holds:

$$\text{(P)} \qquad p^* = \inf_{\{x_l, y_l\}_{\mathcal{L}}} \sup_{\lambda} L(\{x_l, y_l\}_{\mathcal{L}}, \lambda) = \sup_{\lambda} \inf_{\{x_l, y_l\}_{\mathcal{L}}} L(\{x_l, y_l\}_{\mathcal{L}}, \lambda) = d^* \qquad \text{(D)}$$

The previous result allows us to solve the original problem via its dual formulation (4.8) where it decomposes into two subproblems for the primal variables, embedded into a master problem that chooses λ:

$$\underbrace{\sup_{\lambda} \underbrace{\inf_{x_1,y_1} \Big(f_1(x_1,y_1) + \lambda^\top y_1\Big)}_{\text{subproblem 1}} + \underbrace{\inf_{x_2,y_2} \Big(f_2(x_2,y_2) - \lambda^\top y_2\Big)}_{\text{subproblem 2}}}_{\text{master problem}} . \tag{4.9}$$

The convex subproblems can now be solved *independently*, for example on parallel hardware. The master problem "synchronises" them by updating the dual variables.

4.3.3 Generalisation to $d > 2$ Subproblems

First of all we define a compact representation for the consistency constraints. For this purpose we combine the internal and complicating variables of each subproblem into a single variable vector,

$$v_l := \begin{pmatrix} x_l \\ y_l \end{pmatrix} ,$$

and define the primal variable vector

$$v := \begin{pmatrix} v_1 \\ \vdots \\ v_d \end{pmatrix} ,$$

which, in contrast to u, stores a copy of the complicating variables for each subproblem.

Then we introduce a set of linear operators $\{C_l\}_{\mathcal{L}}$, which represent the complicating constraints:

$$(y_1 = y_2 = \ldots = y_d) \Leftrightarrow \sum_{l \in \mathcal{L}} C_l v_l = 0 , \tag{4.10}$$

$$\text{for example } \sum_{l \in \mathcal{L}} C_l v_l = \begin{pmatrix} y_1 - y_2 \\ y_2 - y_3 \\ \vdots \\ y_{d-1} - y_d \end{pmatrix} .$$

Note that although the formulation $C_l v_l$ involves internal variables x_l, the constraints only depend on complicating ones, i.e. $y_l = 0 \ \Rightarrow\ C_l \begin{pmatrix} x_l \\ y_l \end{pmatrix} = 0$ for all x_l.

Now the original problem can be written in its decomposed form with consistency constraints,

$$\inf_v \sum_{l \in \mathcal{L}} f_l(v_l) \qquad \text{s.t.} \sum_{l \in \mathcal{L}} C_l v_l = 0 .$$

With the corresponding Lagrange function,

$$L(v, \lambda) = \sum_{l \in \mathcal{L}} \Big(f_l(v_l) + \lambda^\top C_l v_l \Big) \ ,$$

we obtain the Lagrange dual problem,

$$\underbrace{\sup_{\lambda} \sum_{l \in \mathcal{L}} \underbrace{\Big(\inf_{v_l} f_l(v_l) + \lambda^\top C_l v_l \Big)}_{\text{subproblem } l}}_{\text{master problem}} \ . \tag{4.11}$$

4.3.4 Iterative Algorithm

If we manage to rewrite the original problem in the form (4.11) we can solve it via Algorithm 4.1 (see [60]). After initialisation, the primal and dual variables are updated alternately until a stopping criterion is fulfilled. We denote the primal and dual variables in iteration k with $v_l^{(k)}$ and $\lambda^{(k)}$, respectively. We give some general description of the individual components of the approach below and go into details in the following chapter.

Algorithm 4.1: Optimisation of a general convex minimsation problem in its decomposed form via its dual form.

```
Algorithm: Iterative solution of a decomposed convex problem.
Input: decomposed problem: ${f_l(v_l)}_{\mathcal{L}}$, ${C_l}_{\mathcal{L}}$
Output: solution ${v_l}_{\mathcal{L}}$
/* initialise variables */
$\lambda^{(0)} \leftarrow 0$
$v_l^{(0)} \leftarrow 0 \quad \forall l \in \mathcal{L}$
$k \leftarrow 0$
repeat
    /* update primal variables */
    solve subproblems $l \in \mathcal{L}$ in parallel:
        $v_l^{(k+1)} \leftarrow \arg\inf_{v_l} f_l(v_l) + {\lambda^{(k)}}^\top C_l v_l$
    /* update dual variables */
    $\lambda^{(k+1)} \leftarrow$ updated dual variables
    $k \leftarrow k + 1$
until convergence
```

Update of the Primal Variables. For a fixed $\lambda^{(k)}$ the subproblems can be solved independently, yielding updated primal variables $v_l^{(k+1)}$. By assumption the subproblems are convex and standard solvers can be used for this purpose.

Update of the Dual Variables. In literature, [60, 57] cutting plane, bundle, or trust-region methods are proposed beside subgradient based approaches, e.g.

$$\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \alpha^{(k)} \frac{s^{(k)}}{\left\| s^{(k)} \right\|_2}$$

where s is the gradient of the Lagrange function with respect to λ and is given by

$$s^{(k)} := \nabla_\lambda L \left(v^{(k+1)}, \lambda \right) = \nabla_\lambda \sum_{l \in \mathcal{L}} \left(f_l \left(v_l^{(k+1)} \right) + \lambda^\top C_l v_l^{(k+1)} \right) = \sum_{l \in \mathcal{L}} C_l v_l^{(k+1)} .$$

The authors of [56] propose to choose the scaling parameter $\alpha^{(k)} > 0$ as a sequence that converges to zero and is not summable, i.e. $\lim_{k \to 0} \alpha^{(k)} = 0$ and $\sum_{k=0}^{\infty} \alpha^{(k)} \to \infty$.

Solution of the Original Problem. After convergence we need to construct a solution u of the original problem from the results $\{v_l\}_{\mathcal{L}}$ of the subproblems. The internal values x_l can directly be copied to their position in the original vector u. Although the identity of the instances of the complicating variables y_l is enforced via the complicating constraints, suboptimal solutions and numerical inaccuracies may lead to small differences which have to be handled during reconstruction.

Stopping Criterion. Besides from stopping after a predefined number of iterations, the relative change of the dual variables can be used as stopping criteria, see [56].

$$\frac{\left\| \lambda^{(k+1)} - \lambda^{(k-1)} \right\|_2^2}{\left\| \lambda^{(k)} \right\|_2^2} .$$

4.3.5 Alternative Decomposition Methods

For the discussion on alternative decomposition methods, we restrict ourselves to the case of two subfunctions and denote for each subproblem $l \in \{1, 2\}$ the index of the opposing one by $\bar{l} := 3 - l$. Furthermore, we introduce a vector-valued function which represents the consistency constraints as $c(y_1, y_2) = 0$. Here, this is just $c(y_1, y_2) := y_2 - y_1$, but more general functions are possible. All discussed methods can be extended to more than two subdivisions as well as to include additional equality and inequality constraints.

With this notation the Lagrangian of the introduced Dual Decomposition, or Lagrangian Decomposition, reads

$$L_{\mathrm{DD}}(x_1, y_1, x_2, y_2, \lambda) = f_1(x_1, y_1) + f_2(x_2, y_2) + \lambda^\top c(y_1, y_2) .$$

The complicating variables of the opposing subproblem $y_{\bar{l}}$ are fixed while minimising $f_l(x_l, y_l) + \lambda^\top c(y_1, y_2)$ in (x_l, y_l). The dual variables λ is updated in a master problem. The Dantzig-Wolfe decomposition algorithm is designed for solving linear programs in decomposed manner and can be derived from Dual Decomposition [59, Chap. 8.2].

The first considered variation, Augmented Lagrangian Decomposition, extends the Lagrangian by a quadratic penalty term for the consistency constraints and endows it with good convexity properties [57]:

$$L_{\mathrm{ALD}}(x_1, y_1, x_2, y_2, \lambda) = f_1(x_1, y_1) + f_2(x_2, y_2) + \lambda^\top c(y_1, y_2) + \frac{\alpha}{2} \left\| c(y_1, y_2) \right\|_2^2$$

The penalty parameters α is increased in every iteration. Again the variables $y_{\bar{l}}$ are fixed in subproblem l.

The Optimality Condition Decomposition [58] splits up the constraint function into $c_1(y_1, y_2)$ and $c_2(y_1, y_2)$, so the Lagrangian reads

$$L_{\mathrm{OCD}}(x_1, y_1, x_2, y_2, \lambda) = f_1(x_1, y_1) + f_2(x_2, y_2) + \lambda_1^\top c_1(y_1, y_2) + \lambda_2^\top c_2(y_1, y_2) \ .$$

In each subproblem, an update step for (x_l, y_l) is calculated to reduce $f_l(x_l, y_l) + \lambda_l^\top c_l(y_1, y_2)$ with $y_{\bar{l}}$ fixed. In contrast to the previous methods, the subproblems also determine the dual variable steps. This maximises the interdependency and degrades the master problem to a distributor of variables.

All the mentioned methods have in common that the subproblems update the primal variables and notify the master problem which updates the dual ones. In contrast, the subproblems of the following class of methods are synchronised by updating parts of the *primal* variables.

Primal Decomposition methods [60] solve the subproblems in their internal variables and obtain independency by fixing the complicating ones to a current estimate y:

$$\phi_l(y) := \inf_{x_l, y_l} f_l(x_l, y_l) \ , \qquad \text{s.t. } y = y_l$$

The master problem then updates y with the aim to reduce $\phi(y) = \phi_1(y) + \phi_2(y)$, based on the (sub-)gradient of $\phi(y)$. A well known instance of primal methods is Bender's Decomposition [59, Chap. 8.4]. It basically minimises a piecewise linear approximation of $\phi(y)$ which is refined in every iteration.

For further details and a comparison between the mentioned methods we refer to [56, 59]. The authors of [65] highlight the duality between the presented primal and dual methods.

4.4 Summary

In this chapter we defined the theory on which the following analysis is based on and gave an overview over related methods. The Dual Decomposition method is applied to the class of convex and unconstrained quadratic optimisation problems in Chap. 5.

5 Decomposition for Convex Quadratic Optimisation Problems

In the previous chapter we outlined a general method for solving optimisation problems whose objective function can be decomposed into a sum of convex functions. In Sect. 5.1 we apply the Dual Decomposition approach to the class of convex and unconstrained quadratic optimisation problems. Furthermore, we propose an extension to modify the numerical properties of the subproblems in Sect. 5.2 and analyse convergence. The considered problem class includes several important problems from image processing. We exemplarily examine an image denoising approach as well as two optical flow estimation problems with regularisation terms of first and higher order in Sect. 5.3. Experimental results are presented in Sect. 5.4.

5.1 Decomposition of Quadratic Optimisation Problems

5.1.1 Considered Class of Problems

In this work, we consider the class of unconstrained, convex quadratic optimisation problems of the form

$$\inf_{u \in \mathbb{R}^n} \frac{1}{2} \|Du + c\|_2^2 \,, \qquad \text{with } D \in \mathbb{R}^{m \times n} \text{ and } c \in \mathbb{R}^m. \tag{5.1}$$

Any instance can be converted into the standard form for quadratic problems defined in Sect. 4.2:

$$p^* = \inf_{u \in \mathbb{R}^n} \frac{1}{2} u^\top A u + b^\top u + \text{const} \tag{5.2}$$

by setting $A = D^\top D$ and $b = D^\top c$.

Conversely, any unconstrained quadratic problem in form (5.2) can be brought into the required representation (5.1) up to some constant scalar, if it is strictly convex ($A \succ 0$). We just have to set $D = A^{\frac{1}{2}}$ and $c = A^{-\frac{1}{2}} b$:

$$\frac{1}{2} \left\| A^{\frac{1}{2}} u + A^{-\frac{1}{2}} b \right\|_2^2 = \frac{1}{2} u^\top A u + b^\top u + \frac{1}{2} b^\top A^{-1} b$$

However, $A^{\frac{1}{2}}$ and $A^{-\frac{1}{2}} b$ have to be evaluated for this purpose.

5.1.2 Decomposition of the Objective Functions

In our work we presume that the objective function is given in form (5.1), as then the proposed approach guarantees that the objective function can be decomposed into a sum of *convex* subfunctions, which is imperative for applying Dual Decomposition. This requirement on the structure, however, does not restrict the applicability to quadratic problems as shown in Sect. 5.1.1. In addition, this seems to be natural representation for many discretised variational problems including the case studies in Sect. 5.3.

The basic idea of the decomposition method proposed in this work is to bring the objective function into an equivalent form, that makes the independent structure explicit:

$$\begin{aligned} f(u) &= \frac{1}{2}\left\|Du + c\right\|_2^2 \\ &= \frac{1}{2}\left\| \begin{pmatrix} D_{1,\mathrm{I}} & 0 & \cdots & 0 & D_{1,\mathrm{C}} \\ 0 & D_{2,\mathrm{I}} & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & D_{d,\mathrm{I}} & D_{d,\mathrm{C}} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_d \\ y \end{pmatrix} + \begin{pmatrix} c_1 \\ \vdots \\ \vdots \\ c_d \end{pmatrix} \right\|_2^2 \end{aligned} \tag{5.3}$$

The variables are arranged into private variables $x_l \in \mathbb{R}^{n_l}$ and complicating ones $y \in \mathbb{R}^{n_\mathrm{C}}$. The matrices $D_{l,\mathrm{I}}$ and $D_{l,\mathrm{C}}$ describe the involvement of the private and complicating variables in subproblem l. The vectors c_l hold the corresponding constant terms. Then the objective function $f(u)$ decomposes into

$$\begin{aligned} &\sum_{l\in\mathcal{L}} \left(\frac{1}{2}\begin{pmatrix} x_l \\ y \end{pmatrix}^\top \underbrace{\begin{pmatrix} D_{l,\mathrm{I}} & D_{l,\mathrm{C}} \end{pmatrix}^\top \begin{pmatrix} D_{l,\mathrm{I}} & D_{l,\mathrm{C}} \end{pmatrix}}_{A_l:=} \begin{pmatrix} x_l \\ y \end{pmatrix} + \begin{pmatrix} x_l \\ y \end{pmatrix}^\top \underbrace{\begin{pmatrix} D_{l,\mathrm{I}} & D_{l,\mathrm{C}} \end{pmatrix}^\top c_l}_{-b_l:=} + \underbrace{\frac{1}{2} c_l^\top c_l}_{\text{const}} \right) \\ =&\sum_{l\in\mathcal{L}} \underbrace{\left(\frac{1}{2}\begin{pmatrix} x_l \\ y \end{pmatrix}^\top A_l \begin{pmatrix} x_l \\ y \end{pmatrix} - \begin{pmatrix} x_l \\ y \end{pmatrix}^\top b_l \right)}_{f_l(x_l,y):=} + \underbrace{\frac{1}{2}\sum_{l\in\mathcal{L}} c_l^\top c_l}_{\text{const}} = \sum_{l\in\mathcal{L}} f_l(x_l, y) + \text{const}\ , \end{aligned}$$

with d subfunction $f_1, \ldots, f_d$ and a constant part, which we will omit in the remaining description. Note that due to

$$x^\top A_l x = \left\| \begin{pmatrix} D_{l,\mathrm{I}} & D_{l,\mathrm{C}} \end{pmatrix} x \right\|_2^2 \geq 0,\ \forall x \in \mathbb{R}^{n_l + n_\mathrm{C}}$$

all A_l are positive semidefinite by construction. This beneficial property guarantees that each subproblem is convex in (x_l, y).

In the remaining part we detail on the method used to obtain the representation (5.3), and further simplify the subfunctions.

The Restriction Operator

The decomposition of the objective function requires the selection of a subset of variables as well as rows and columns from D and c. For this purpose we define two finite sets of indices $\mathcal{I} \subset \mathbb{Z}$ and $\mathcal{J} \subset \mathbb{Z}$. Both are associated with an arbitrary (but fixed) relation "$<$", so we can order the set elements by $\mathcal{I}_1 < \mathcal{I}_2 < \cdots < \mathcal{I}_{|\mathcal{I}|}$ and $\mathcal{J}_1 < \mathcal{J}_2 < \cdots < \mathcal{J}_{|\mathcal{J}|}$, respectively. For a compact notation of the analysis below, we introduce a linear operator,

$$\begin{aligned} R_{\mathcal{J}\to\mathcal{I}} :\ & \mathbb{R}^{|\mathcal{J}|} \mapsto \mathbb{R}^{|\mathcal{I}|} , \\ \text{defined in matrix form } R_{\mathcal{J}\to\mathcal{I}} &:= \left(\delta_{\mathcal{I}_i,\mathcal{J}_j}\right)_{i,j} \in \{0,1\}^{|\mathcal{I}|\times|\mathcal{J}|} , \\ \text{using the Kronecker delta function } \delta_{i,j} &:= \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} . \end{aligned} \tag{5.4}$$

Then $y = R_{\mathcal{J}\to\mathcal{I}}x$ copies the element x_j to y_i if there is a $j \in \{1, \dots, |\mathcal{J}|\}$, so $\mathcal{I}_i = \mathcal{J}_j$, and sets the remaining entries of y to zero. An example with $\mathcal{J} = \{2, 4\}$ and $\mathcal{I} = \{1, 2, 3\}$, and the usual order on integers shall demonstrate this property:

$$R_{\mathcal{J}\to\mathcal{I}} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ a \\ 0 \end{pmatrix}$$

The transposed version of the operator is related as

$$R^{\top}_{\mathcal{J}\to\mathcal{I}} = \left(\delta_{\mathcal{I}_j,\mathcal{J}_i}\right)_{i,j} = R_{\mathcal{I}\to\mathcal{J}} \in \{0,1\}^{|\mathcal{J}|\times|\mathcal{I}|} .$$

In the remaining work, we will always assume $\mathcal{J} \subseteq \mathcal{I}$. Then the transposed version of (5.4) can be regarded as a *restriction operator* that extracts a subset of rows $\mathcal{J} \subseteq \mathcal{I} := \{1, \dots, m\}$ from a matrix $A \in \mathbb{R}^{m\times n}$,

$$R^{\top}_{\mathcal{J}\to\mathcal{I}} A = A_{\mathcal{J},*} = \begin{pmatrix} A_{\mathcal{J}_1,*} \\ \vdots \\ A_{\mathcal{J}_{|\mathcal{J}|},*} \end{pmatrix} \in \mathbb{R}^{|\mathcal{J}|\times n} ,$$

and vertically stacks them together, where $A_{\mathcal{J},*}$ denotes the rows of A indicated by $\mathcal{J}$. Furthermore, the outer product of R is a diagonal matrix which contains the ordered values of the set characteristic function for $\mathcal{I}_i \in \mathcal{J}$, $i = 1, \dots, |\mathcal{I}|$.

$$R_{\mathcal{J}\to\mathcal{I}} R^{\top}_{\mathcal{J}\to\mathcal{I}} = \left(\sum_{k=1}^{|\mathcal{J}|} \delta_{\mathcal{I}_i,\mathcal{J}_k} \delta_{\mathcal{I}_j,\mathcal{J}_k} \right)_{i,j} = \operatorname{diag}\left(\left(\chi_{\mathcal{J}}(\mathcal{I}_i)\right)_{i=1,\dots,|\mathcal{I}|} \right) \in \{0,1\}^{|\mathcal{I}|\times|\mathcal{I}|} ,$$

where the characteristic function for a set $\mathcal{S}$ is defined as

$$\chi_{\mathcal{S}}(x) := \begin{cases} 1 & \text{if } x \in \mathcal{S} \\ 0 & \text{if } x \notin \mathcal{S} \end{cases} .$$

The inner product of the restriction operator is just the identity:

$$R^\top_{\mathcal{J}\to\mathcal{I}} R_{\mathcal{J}\to\mathcal{I}} = \left(\sum_{k=1}^{|\mathcal{I}|} \delta_{\mathcal{I}_k,\mathcal{J}_i} \delta_{\mathcal{I}_k,\mathcal{J}_j} \right)_{i,j} = \left(\delta_{\mathcal{J}_i,\mathcal{J}_j}\right)_{i,j} = I \in \{0,1\}^{|\mathcal{J}|\times|\mathcal{J}|}$$

Furthermore, for d disjoint sets, $\mathcal{J}^1, \ldots, \mathcal{J}^d \subset \mathcal{I}$ and $\mathcal{J}^i \cap \mathcal{J}^j = \emptyset \ \forall i,j \in \mathcal{L} := \{1,\ldots,d\}, i \neq j$, the following equality holds (denoting $\mathcal{J} := \bigcup_{l\in\mathcal{L}} \mathcal{J}^l$):

$$\begin{aligned} \sum_{l\in\mathcal{L}} R_{\mathcal{J}^l\to\mathcal{I}} R^\top_{\mathcal{J}^l\to\mathcal{I}} &= \sum_{l\in\mathcal{L}} \operatorname{diag}\left(\left(\chi_{\mathcal{J}^l}(\mathcal{I}_i)\right)_{i=1,\ldots,|\mathcal{I}|} \right) = \operatorname{diag}\left(\left(\sum_{l\in\mathcal{L}} \chi_{\mathcal{J}^l}(\mathcal{I}_i) \right)_{i=1,\ldots,|\mathcal{I}|} \right) \\ &= \operatorname{diag}\left(\left(\chi_{\mathcal{J}}(\mathcal{I}_i)\right)_{i=1,\ldots,|\mathcal{I}|} \right) = R_{\mathcal{J}\to\mathcal{I}} R^\top_{\mathcal{J}\to\mathcal{I}} \ . \end{aligned} \tag{5.5}$$

Assignment of Terms to Subfunctions

For the decomposition of the objective function we identify each row of $D \in \mathbb{R}^{m\times n}$ and $c \in \mathbb{R}^m$ with a single quadratic term:

$$f(u) = \frac{1}{2}\|Du + c\|_2^2 = \frac{1}{2} \left\| \begin{pmatrix} D_{1,*} \\ \vdots \\ D_{m,*} \end{pmatrix} u + \begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix} \right\|_2^2 = \sum_{i=1}^{m} \underbrace{\frac{1}{2}\left(D_{i,*}u + c_i\right)^2}_{\text{term } i} \tag{5.6}$$

where $D_{i,*}$ denotes the i-th row of the matrix D and c_i the i-th element of c. To simplify the presentation and without loss of generality we assume that D contains no all-zero columns nor rows. This can be achieved by eliminating all invariant terms i, i.e. $e_i^\top \tilde{D} = \tilde{D}_{i,*} = 0$, and void variables u_j, i.e. $\tilde{D}e_j = 0$, from a matrix $\tilde{D}$ in before. The vectors e_i are of appropriate size and have an one at entry i and zero elsewhere. Furthermore, we define the all-one vector e.

Next, we choose a matrix $W \in [0,1]^{m\times d}$ with $We = e$ which assigns term i to subfunction l with weight $W_{i,l}$. This generalises "hard" decomposition methods (i.e. $W \in \{0,1\}^{m\times d}$) by permitting assignment of each term to more than one subfunction and with different impact. The choice of the weights can be based on geometrical cues but also on more abstract decomposition methods, e.g. based on graphs. For continuity, we temporarily skip the details on the method used to choose W and refer to Sect. 5.1.3.

Let us denote the set of the indices of all terms by $\mathcal{I} := \{1,\ldots,m\}$. Furthermore, we define the index sets $\mathcal{I}^1, \ldots, \mathcal{I}^d \subset \mathcal{I}$ which represent the terms involved in subfunction l, i.e.

$$\mathcal{I}^l := \{i \in \mathcal{I} \,|\, W_{i,l} > 0\} \ . \tag{5.7}$$

With $W_l := \operatorname{diag}(W_{*,l})$ the objective function decomposes into

$$f(u) = \sum_{i\in\mathcal{I}} \frac{1}{2}(D_{i,*}u + c_i)^2 = \sum_{i\in\mathcal{I}} \sum_{l\in\mathcal{L}} \frac{1}{2} W_{i,l} (D_{i,*}u + c_i)^2 = \sum_{l\in\mathcal{L}} \frac{1}{2} (Du + c)^\top W_l (Du + c) \ .$$

Due to the definition (5.7) of $\mathcal{I}^l$,

$$W_l^{\frac{1}{2}} R_{\mathcal{I}^l\to\mathcal{I}} R_{\mathcal{I}^l\to\mathcal{I}}^\top W_l^{\frac{1}{2}} = W_l^{\frac{1}{2}} \operatorname{diag}\left(\left(\chi_{\mathcal{I}^l}(\mathcal{I}_i)\right)_{i=1,\dots,|\mathcal{I}|}\right) W_l^{\frac{1}{2}} = W_l$$

holds. Then we can define $P_l := R_{\mathcal{I}^l\to\mathcal{I}}^\top W_l^{\frac{1}{2}}$ and rewrite the objective as

$$\begin{aligned} f(u) &= \sum_{l\in\mathcal{L}} \frac{1}{2} (Du+c)^\top W_l (Du+c) \\ &= \sum_{l\in\mathcal{L}} \frac{1}{2} (Du+c)^\top W_l^{\frac{1}{2}} R_{\mathcal{I}^l\to\mathcal{I}} R_{\mathcal{I}^l\to\mathcal{I}}^\top W_l^{\frac{1}{2}} (Du+c) \\ &= \sum_{l\in\mathcal{L}} \frac{1}{2} \left\| R_{\mathcal{I}^l\to\mathcal{I}}^\top W_l^{\frac{1}{2}} (Du+c) \right\|_2^2 = \sum_{l\in\mathcal{L}} \frac{1}{2} \left\| P_l (Du+c) \right\|_2^2 . \end{aligned}$$

which is a sum of convex terms just as required by the dual decomposition approach. However, still each term involves *all* variables in u and does not distinguish between local and global ones.

Identification of Local and Complicating Variables

In the next step we make the reduced number of variables involved in $P_l(Du+c)$ and their classification explicit. Analog to the index set for the terms, $\mathcal{J} := \{1,\dots,n\}$ represents the indices of the variables u_j of the original problem vector u. Then we define the subsets $\mathcal{J}^l \subseteq \mathcal{J}$ of variables with non-zero contribution to $P_l Du$, i.e.

$$\mathcal{J}^l := \{ j \in \mathcal{J} \,|\, P_l D e_j \neq 0 \} .$$

This definition directly implies the equality

$$P_l D = P_l D \operatorname{diag}\left(\left(\chi_{\mathcal{J}^l}(\mathcal{J}_i)\right)_{i=1,\dots,|\mathcal{J}|}\right) = P_l D R_{\mathcal{J}^l\to\mathcal{J}} R_{\mathcal{J}^l\to\mathcal{J}}^\top . \tag{5.8}$$

The number of subfunctions in which a variable is involved in is given by the entries of the diagonal matrix

$$V := \sum_{l\in\mathcal{L}} R_{\mathcal{J}^l\to\mathcal{J}} R_{\mathcal{J}^l\to\mathcal{J}}^\top = \operatorname{diag}\left(\left(\sum_{l\in\mathcal{L}} \chi_{\mathcal{J}^l}(\mathcal{J}_j)\right)_{j=1,\dots,|\mathcal{J}|}\right) ,$$

or individually for variable u_j

$$V_j := V_{j,j} = \sum_{l\in\mathcal{L}} \chi_{\mathcal{J}^l}(\mathcal{J}_j) = \left| \{ l \in \mathcal{L} \,|\, j \in \mathcal{J}_l \} \right| .$$

Then for each subfunction l we identify the indices of the complicating variables by

$$\mathcal{J}_{\mathrm{C}}^l := \left\{ j \in \mathcal{J}^l \,\middle|\, V_j > 1 \right\} ,$$

and the internal ones by

$$\mathcal{J}_{\mathrm{I}}^{l} := \left\{ j \in \mathcal{J}^{l} \,\middle|\, V_j = 1 \right\} .$$

The set of complicating variables of the complete problem is denoted by

$$\mathcal{J}_{\mathrm{C}} := \bigcup_{l \in \mathcal{L}} \mathcal{J}_{\mathrm{C}}^{l} .$$

Note that the assumptions made at the beginning of this section and ruling out void variables ensure $V_j > 0$ for all $j \in \mathcal{J}$. This implies that the subsets $\left\{\mathcal{J}_{\mathrm{I}}^{l}\right\}_{\mathcal{L}}$ and $\mathcal{J}_{\mathrm{C}}$ define a partition of $\mathcal{J}$.

Decomposed Formulation of the Objective Function

Next we define an operator that selects the variables involved in subproblem l from u:

$$Q_l := \begin{pmatrix} R^{\top}_{\mathcal{J}_{\mathrm{I}}^{l} \to \mathcal{J}} \\ R^{\top}_{\mathcal{J}_{\mathrm{C}}^{l} \to \mathcal{J}} \end{pmatrix}$$

In addition we can introduce names for the combined vectors of local and complicating variables:

$$v_l := \begin{pmatrix} x_l \\ y_l \end{pmatrix} = \begin{pmatrix} R^{\top}_{\mathcal{J}_{\mathrm{I}}^{l} \to \mathcal{J}} \\ R^{\top}_{\mathcal{J}_{\mathrm{C}}^{l} \to \mathcal{J}} \end{pmatrix} u = Q_l u$$

The sets $\mathcal{J}_{\mathrm{C}}^{l}$ and $\mathcal{J}_{\mathrm{I}}^{l}$ are disjoint by construction and due to (5.5)

$$\begin{aligned} Q_l^{\top} Q_l &= \begin{pmatrix} R^{\top}_{\mathcal{J}_{\mathrm{I}}^{l} \to \mathcal{J}} \\ R^{\top}_{\mathcal{J}_{\mathrm{C}}^{l} \to \mathcal{J}} \end{pmatrix}^{\top} \begin{pmatrix} R^{\top}_{\mathcal{J}_{\mathrm{I}}^{l} \to \mathcal{J}} \\ R^{\top}_{\mathcal{J}_{\mathrm{C}}^{l} \to \mathcal{J}} \end{pmatrix} = \left(R_{\mathcal{J}_{\mathrm{I}}^{l} \to \mathcal{J}} R^{\top}_{\mathcal{J}_{\mathrm{I}}^{l} \to \mathcal{J}} + R_{\mathcal{J}_{\mathrm{C}}^{l} \to \mathcal{J}} R^{\top}_{\mathcal{J}_{\mathrm{C}}^{l} \to \mathcal{J}} \right) \\ &= R_{(\mathcal{J}_{\mathrm{I}}^{l} \cup \mathcal{J}_{\mathrm{C}}^{l}) \to \mathcal{J}} R^{\top}_{(\mathcal{J}_{\mathrm{I}}^{l} \cup \mathcal{J}_{\mathrm{C}}^{l}) \to \mathcal{J}} = R_{\mathcal{J}^{l} \to \mathcal{J}} R^{\top}_{\mathcal{J}^{l} \to \mathcal{J}} \end{aligned}$$

holds.

Now, using (5.8), we can further reformulate the objective function so that it fits into the form required for the Dual Decomposition.

$$\begin{aligned} f(u) &= \sum_{l \in \mathcal{L}} \frac{1}{2} \left\| P_l D u + P_l c \right\|_2^2 = \sum_{l \in \mathcal{L}} \frac{1}{2} \left\| P_l D R_{\mathcal{J}^{l} \to \mathcal{J}} R^{\top}_{\mathcal{J}^{l} \to \mathcal{J}} u + P_l c \right\|_2^2 \\ &= \sum_{l \in \mathcal{L}} \frac{1}{2} \left\| P_l D Q_l^{\top} Q_l u + P_l c \right\|_2^2 = \sum_{l \in \mathcal{L}} \frac{1}{2} \left\| D_l v_l + c_l \right\|_2^2 = \sum_{l \in \mathcal{L}} f_l(v_l) \end{aligned}$$

with

$$\begin{aligned} D_l &:= P_l D Q_l^{\top} \\ c_l &:= P_l c . \end{aligned}$$

Each subfunction is quadratic and convex and can be brought into standard form (4.1) for the objective function of a quadratic program.

$$f_l(v_l) = \frac{1}{2} v_l^\top A_l v_l - b_l^\top v_l + \text{const}_l \tag{5.9}$$

$$\text{with } A_l := D_l^\top D_l = Q_l D^\top P_l^\top P_l D Q_l^\top \tag{5.10}$$

$$\text{and } b_l := -D_l^\top c_l = -Q_l D^\top P_l^\top P_l c \tag{5.11}$$

$$\text{and constant } \text{const}_l := c^\top P_l^\top P_l c \tag{5.12}$$

The structure of Q_l and P_l directly indicate the parts (rows and columns) of D and c that are relevant for subproblem l. This opens up the possibility to avoid constructing the whole problem description (D, c) at once and thus to save memory resources.

Summing up, we presented a method which decomposes the objective function of problem (5.1) into a sum of convex quadratic subfunctions. The only prerequisite is that we have to specify the matrix W which has to comply with some simple rules. The overall method is summarised in Algorithm 5.1.

Note, that the design of the weighting matrix is critical for the performance of the overall method as the following – extreme – example demonstrates: If we set all entries of W to d^{-1}, the matrix is considered as valid and a correct decomposition is found, i.e. $f(v) = \sum_{l \in \mathcal{L}} f_l(v_l)$ with f_l convex. However, every term is assigned to all subfunction, and thus each variable is identified as a complicating one. Then in every iteration each subproblem has to solve the complete problem.

Furthermore, the construction method leads to positive semidefinite subproblem matrices A_l, but cannot guarantee their positive definiteness. Let us examine the following toy problem (with arbitrary c) which is already in the form used in (5.3):

$$f(u) = \frac{1}{2}\|Du + c\|_2^2 = \frac{1}{2}\left\| \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_{1,1} \\ x_{2,1} \\ x_{2,2} \\ y \end{pmatrix} + \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix} \right\|_2^2$$

The matrix D has full rank and thus $D^\top D$ is positive definite and a unique solution exists. The naming of the variables suggests a possible decomposition:

$$\begin{aligned} f(u) &= \frac{1}{2}\left\| \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_{1,1} \\ y \end{pmatrix} + \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \right\|_2^2 + \frac{1}{2}\left\| \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_{2,1} \\ x_{2,2} \\ y \end{pmatrix} + \begin{pmatrix} c_3 \\ c_4 \end{pmatrix} \right\|_2^2 \\ &= f_1(x_1, y) + f_2(x_2, y) \end{aligned}$$

The first subfunction is strictly convex, however the second one is not, because the matrix

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}^\top \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

does not have full rank (e.g. the first and last row sum up to the middle one). This case shows, that in fact strictly convexity cannot be guaranteed.

However, this property is required by the conjugate gradient method we use to solve the subproblems. In Sect. 5.2 we present an extension to the Dual Decomposition method which allows to modify the subproblems and thus to enforce strict convexity. The overall objective is not changed by this measure.

In practice, e.g. in the investigation of the case studies in Sect. 5.3, we did not encounter any subproblem with semidefinite matrices A_l. Thus, we are confident that it is possible to define a method to choose the term weights W, which guarantees the discussed attribute. Motivated by this experiences and the fact that our convergence analysis presented in Sect. 5.2.2 presumes this property, in the remaining work we assume that all A_l are positive definite.

Algorithm 5.1: Decompose a problem given in the form $\frac{1}{2}\left\|Du+c\right\|_2^2$ into d subproblems. Return the subfunction description $\{A_l, b_l\}_{\mathcal{L}}$, the consistency constraint operators $\{C_l\}_{\mathcal{L}}$ and the variable assignment operators $\{Q_l\}_{\mathcal{L}}$.

```
Algorithm: Decomposition of a quadratic problem
Input: original problem $D, c$, number of subproblems $d$
Output: decomposed problem $\{A_l, b_l, C_l, Q_l\}_{\mathcal{L}}$
/* assign terms to subproblems */
determine weights of terms in the subproblems: $W$
determine term assignment: $\mathcal{I}^l \leftarrow \{i \in \mathcal{I}\,|W_{i,l} > 0\}$
construct term assignment operator: $P_l \leftarrow R^\top_{\mathcal{I}^l\to\mathcal{I}}\,\mathrm{diag}\,(W_{*,l})^{\frac{1}{2}}$
/* assign variables to subproblems */
determine variable use multiplicity: $V \leftarrow \sum_{l\in\mathcal{L}} R_{\mathcal{J}^l\to\mathcal{J}} R^\top_{\mathcal{J}^l\to\mathcal{J}}$
determine complicating variables: $\mathcal{J}^l_{\mathrm{C}} \leftarrow \left\{j \in \mathcal{J}^l\,\middle|V_j > 1\right\}$
determine internal variables: $\mathcal{J}^l_{\mathrm{I}} \leftarrow \left\{j \in \mathcal{J}^l\,\middle|V_j = 1\right\}$
construct variable assignment operators: $Q_l \leftarrow \begin{pmatrix} R^\top_{\mathcal{J}^l_{\mathrm{I}}\to\mathcal{J}} \\ R^\top_{\mathcal{J}^l_{\mathrm{C}}\to\mathcal{J}} \end{pmatrix}$
/* construct decomposed problem description */
$A_l \leftarrow Q_l D^\top P_l^\top P_l D Q_l^\top$
$b_l \leftarrow -Q_l D^\top P_l^\top P_l c$
/* generate consistency constraint operators */
construct $\{C_l\}_{\mathcal{L}}$
```

5.1.3 Solving a Decomposed Problem

In the previous section we showed how to decompose the objective function of the quadratic problem as a prerequisite for the application of the Dual Decomposition method. With this results, the problem to solve can be written as

$$\inf_{\{v_l\}_{\mathcal{L}}} \sum_{l\in\mathcal{L}} \left(\frac{1}{2} v_l^\top A_l v_l - b_l^\top v_l + \text{const}_l\right) \tag{5.13}$$

$$\text{s.t.} \sum_{l\in\mathcal{L}} C_l v_l = 0 \; . \tag{5.14}$$

For the minimisation of the objective function, it is possible to omit the constant part. In dual form the problem reads

$$\sup_{\lambda} \sum_{l\in\mathcal{L}} \left(\inf_{v_l} \frac{1}{2} v_l^\top A_l v_l - (b_l - C_l^\top \lambda)^\top v_l\right)$$

and we can apply the iterative Algorithm 4.1 for solving this class of problems. Each subproblem is now a quadratic, convex optimisation problem and can efficiently be solved by accessing standard tools such as conjugate gradient methods to update $\{v_l\}_{\mathcal{L}}$.

The dual variables are updated by following the gradient of the Lagrangian function,

$$s^{(k)} = \nabla_\lambda L\left(\left\{v_l^{(k+1)}\right\}_{\mathcal{L}}, \lambda\right) = \sum_{l\in\mathcal{L}} C_l v_l^{(k+1)} \; .$$

Here we use

$$\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \alpha^{(k)} s^{(k)}$$

and choose the step scale parameter as

$$\alpha^{(k)} = \frac{1}{a + bk}$$

with scalar parameters a and b.

The update rules for the primal and dual variables can be summarised as

$$v_l^{(k+1)} \leftarrow A_l^{-1}\left(b_l - C_l^\top \lambda^{(k)}\right) \; , \qquad l \in \mathcal{L} \tag{5.15}$$

$$\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \alpha^{(k)} s^{(k)} \; . \tag{5.16}$$

The iterative solution method is summarised in Algorithm 5.2. It is wrapped in the complete approach (Algorithm 5.3) for solving convex quadratic problems via Dual Decomposition. Further technical details on the term assignment (yielding W), the constraint construction (yielding C_l) and the reconstruction of u from the subproblem solutions follow in the remaining section. Convergence properties of this approach are covered as a special case of an extended version in Sect. 5.2.2

Algorithm 5.2: Solving a decomposed convex quadratic problem via its dual representation.

```
Algorithm: Iterative Solution of Dual Decomposed Quadratic Problems
Input: decomposed problem: ${A_l, b_l, C_l}_{\mathcal{L}}$
Output: solution ${v_l}_{\mathcal{L}}$
/* initialise variables */
$\lambda^{(0)} \leftarrow 0$
$v_l^{(0)} \leftarrow 0 \quad \forall l \in \mathcal{L}$
$k \leftarrow 0$
/* iterate the update of primal and dual variables */
repeat
    /* update primal variables */
    solve subproblems $l \in \mathcal{L}$ in parallel:
        $v_l^{(k+1)} \leftarrow A_l^{-1} \left( b_l - C_l^\top \lambda^{(k)} \right)$
    /* update dual variables */
    $s^{(k)} \leftarrow \sum_{l \in \mathcal{L}} C_l v_l^{(k+1)}$
    $\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \alpha^{(k)} s^{(k)}$
    $k \leftarrow k + 1$
until convergence
```

Term Assignment Based on Geometrical Cues

In our work we use geometrical cues to assign the terms in (5.6) to the subproblems. We split up the problem domain Ω into d – possibly overlapping – regions $\Omega_1, \ldots, \Omega_d$, so $\cup_{l \in \mathcal{L}} \Omega_l = \Omega$. Roughly spoken, a membership for each term to each regions is determined, which is based on the usage of the variables and their grid position in Ω.

First of all we define a measurement $U_{i,j} \geq 0$ that describes the "degree of usage" of variable u_j in term i.

$$U_{i,j} = \begin{cases} 1 & \text{if } D_{i,j} \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

As we presumed in Sect. 5.1.2, all-zero rows in D were eliminated in before, and thus for each variable u_i, there is at least one j, so $U_{i,j} = 1$.

Next, each single variable u_j in u is identified with a geometrical position $p_j \in \Omega$, which can easily be derived from the problem discretisation. Based on this, each variable is assigned a weight $M_{j,l} \geq 0$ that describes the membership of variable u_j to Ω_l. Here we simply use the set characteristic function:

$$M_{j,l} := \chi_{\Omega_l}(p_j)$$

Algorithm 5.3: General approach for decomposition and solving of a convex quadratic optimisation problem in the form $\frac{1}{2}\|Du + c\|_2^2$ by Dual Decomposition.

```
Algorithm: solve a convex quadratic problem via Dual Decomposition
Input: problem description $D, c$
Output: solution $u$
create decomposition $\{A_l, b_l, C_l, Q_l\}_{\mathcal{L}}$
solve problem via Dual Decomposition, obtaining $\{v_l\}_{\mathcal{L}}$
construct solution $u$ of the original problem from $\{v_l\}_{\mathcal{L}}$
```

The "usage" of term i in region l is then measured by

$$\tilde{W}_{i,l} := \sum_{j=1}^{n} U_{i,j} M_{j,l} \ , \tag{5.17}$$

or in compact form $\tilde{W} = UM$.

Note that because each term i involves at least one variable, and each variable is assigned to at least one region, there is always a $d \in \mathcal{L}$, so $\tilde{W}_{i,d} > 0$.

Finally we normalise the rows, so they sum up to 1. A simple way to achieve this aim is to set

$$W = \operatorname{diag}(\tilde{W}e)^{-1}\tilde{W} \ .$$

However, the nonlinear version used here leads to more sparse results, i.e. terms tend to appear in less subdivision.

$$W_{i,l} = \frac{\chi_{\omega_i}(l)}{|\omega_i|}$$
$$\text{where } \omega_i := \left\{ l \in \mathcal{L} \,\middle|\, \tilde{W}_{i,l} = \sup_{l \in \mathcal{L}} \tilde{W}_{i,l} \right\} \ .$$

Basically, each term is assigned to the subfunction where it has the highest impact, as measured by $\tilde{W}$. In case of a tie (i.e. $|\omega_i| > 1$) it is equally distributed to all of them.

In this vein the requirements $W \in [0,1]^{m \times d}$ and $We = e$, posed in Sect. 5.1.2, hold. Thus, this method – although somewhat heuristic – chooses W adequately, so the proposed decomposition method defined in Sect. 5.1.2 can be applied and decomposes the initial objective function correctly. In our experiments, this method worked well, but more sophisticated approaches might further improve the performance of the decomposition.

For a parallel implementation of the solver it is desirable that the decomposition does not presume that the problem description D and c must be available as a whole. Here, U involves not the actual values but only the sparseness information of D, which can be stored more efficiently. In addition the definition (5.17) possesses the possibility to reduce the evaluation of $U_{i,j}$ to those j where $M_{j,l}$ does not vanish.

Construction of the Consistency Constraints

In the considered Dual Decomposition approach, the consistency constraints enforce the equivalence of all copies of the complicating variables, i.e.

$$y_1 = y_2 = \cdots = y_d \,. \tag{5.18}$$

The investigated approach requires them to be formulated as an homogeneous linear system in the variables of all subproblems,

$$\sum_{l \in \mathcal{L}} C_l v_l = 0 \,.$$

To obtain this form we decompose the chain-equation (5.18) into pairwise relations, which can then easily be casted as linear system as required in (4.10). In the case $d = 4$ the two steps are

$$\begin{aligned}
& y_1 = y_2 = y_3 = y_4 \\
\text{(split into pairwise equations)} \Leftrightarrow\ & y_1 - y_2 = 0 \wedge y_2 - y_3 = 0 \wedge y_3 - y_4 = 0 \\
\text{(setup linear system)} \Leftrightarrow\ & \underbrace{\begin{pmatrix} +I \\ 0 \\ 0 \end{pmatrix}}_{C_1} y_1 + \underbrace{\begin{pmatrix} -I \\ +I \\ 0 \end{pmatrix}}_{C_2} y_2 + \underbrace{\begin{pmatrix} 0 \\ -I \\ +I \end{pmatrix}}_{C_3} y_3 + \underbrace{\begin{pmatrix} 0 \\ 0 \\ -I \end{pmatrix}}_{C_4} y_4 = 0 \,.
\end{aligned}$$

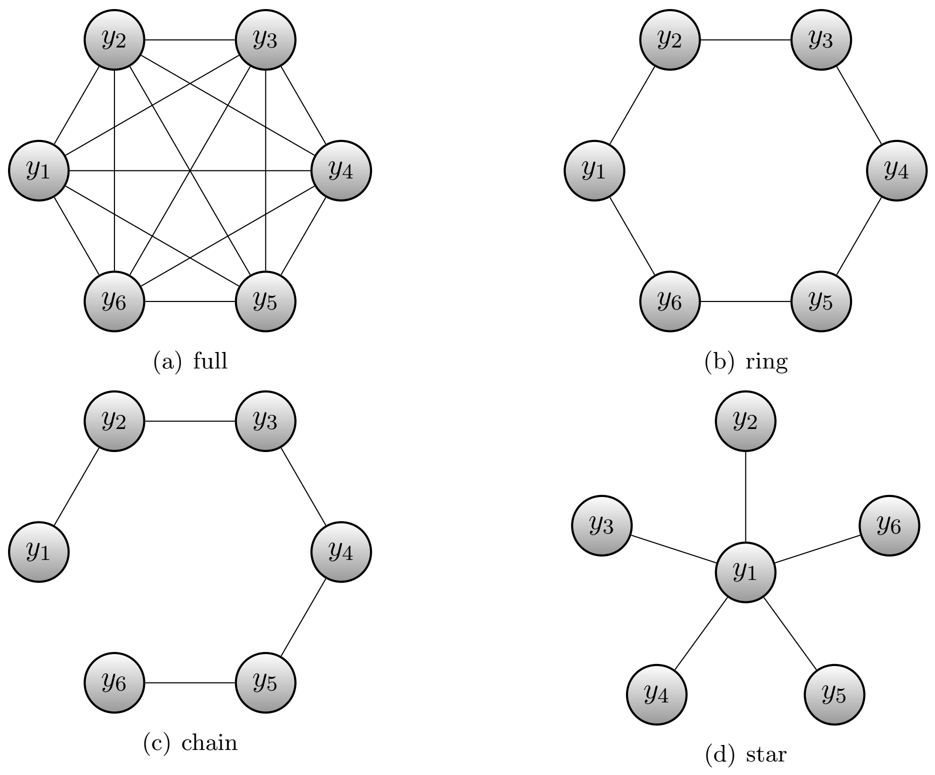
The decomposition of (5.18) into pairwise equations can be implemented in several – mathematically equivalent – ways, which however may influence numerical properties of the resulting optimisation problem.

For a visual description of the variants of the decomposition of a single, *scalar-valued* equation

$$y_{1,j} = y_{2,j} = \cdots = y_{d,j} \,, \qquad j \in \{1, \ldots, |\mathcal{J}_\mathrm{C}|\} \,,$$

we represent a set of pairwise constraints as an *undirected graph* $G_j = (N_j, E_j)$ with nodes $N_j = \{y_{1,j}, \cdots, y_{d,j}\}$ and edges E_j, where each $(y_{l_1,j}, y_{l_2,j}) \in E_j$ represents an equation $y_{l_1,j} = y_{l_2,j}$. For definitions and terminology in conjunction with graphs we refer to [66, Chapter 1].

Figure 5.1(a) represents the required pairwise identity between the variables, resulting in a fully connected graph with $\frac{d(d-1)}{2}$ edges. However, due to the transitivity of the equation relation, it is possible to omit some of the edges, as long as the graph is connected, i.e. there is a path between any pair of vertices in G_j. A graph with the minimum required number of edges that connects all nodes in N_j is called a *spanning tree*, e.g. Fig. 5.1(c) and Fig. 5.1(d). Adding redundant edges, i.e. more than the minimum required number $|N_j| - 1$, does not further reduce the feasible set $\{v \,|\, Cv = 0\}$ and thus leads to a rank-deficient matrix C. For example, a further edge added between the ends of a chain leads to a ring-shaped graph Fig. 5.1(d). Once we have set up



(a) full (b) ring (c) chain (d) star

Figure 5.1: Visualisation of some alternative implementations of a scalar consistency constraint $y_1 = \ldots = y_d$ (here: $d = 6$) as pairwise equations: In the graph $G = (N, E)$, each node represents a *scalar* variable and each edge $(y_i, y_j) \in E$ a pairwise constraint $y_i = y_j$. (a) Fully-connected graph (with $\frac{1}{2}|E|(|E| - 1)$ edges), referred to as *full*; (b) *ring* ($|E|$ edges); spanning trees (with minimum necessary number of edges, $|E| - 1$): (c) *chain* and (d) *star*

the graph representations for all constraints, the creation of the constraint matrices is straightforward. Every edge $(y_{l_1,j}, y_{l_2,j}) \in E_j$ casts an entry $+1$ in C_{l_1} and a -1 in C_{l_2}.

Algorithm 5.4 summarises the method used in this work for constructing the constraint matrices. In addition to the description given above, it also respects that complicating variables may not have copies in all subproblems and does not generate related constraints. The pattern used for creating the graph is a parameter and can be chosen from those visualised in Fig. 5.1.

Reconstruction of the the Original Solution Vector

Given the solution in decomposed form, $\{v_l\}_{\mathcal{L}}$, we need to reconstruct the solution of the original problem. The local variables x_l can directly be mapped to u as they only appear once by definition:

$$\tilde{u}_{\mathrm{I}} := \sum_{l\in\mathcal{L}} R_{\mathcal{J}_{\mathrm{I}}^l\to\mathcal{J}} x_l = V^{-1} \sum_{l\in\mathcal{L}} R_{\mathcal{J}_{\mathrm{I}}^l\to\mathcal{J}} x_l$$

where in the last transformation we used the fact that $V_j = 1$ for all internal variables. In contrast, for each complicating variable at least two copies exist in separate subproblems. In theory, the complicating constraints enforce their exact identity. However, in practice, the relaxation of the equalities and numerical inaccuracies lead to small differences. Hence we reconstruct the non-local entries of u from an average of all corresponding instances of complicating variables. By definition, V_j gives the multiplicity of u_j and we can write

$$\tilde{u}_{\mathrm{C}} := V^{-1} \sum_{l\in\mathcal{L}} R_{\mathcal{J}_{\mathrm{C}}^l\to\mathcal{J}} y_l$$

The reconstruction of the original problem can be formulated in a compact way as

$$\tilde{u} = \tilde{u}_{\mathrm{I}} + \tilde{u}_{\mathrm{C}} = V^{-1} \sum_{l\in\mathcal{L}} \left(R_{\mathcal{J}_{\mathrm{I}}^l\to\mathcal{J}} x_l + R_{\mathcal{J}_{\mathrm{C}}^l\to\mathcal{J}} y_l \right) = V^{-1} \sum_{l\in\mathcal{L}} Q_l^\top v_l \ .$$

Substituting $v_l = Q_l u$ shows that for the correct solution represented in decomposed form, the reconstruction is exact:

$$\tilde{u} = V^{-1} \sum_{l\in\mathcal{L}} Q_l^\top v_l = V^{-1} \sum_{l\in\mathcal{L}} Q_l^\top Q_l u = V^{-1} \left(\sum_{l\in\mathcal{L}} Q_l^\top Q_l \right) u = V^{-1} V u = u \ .$$

5.2 Improving Ill-Conditioned Subproblems

The matrices A_l in subproblem l which arise from the decomposition (5.9) are positive semidefinite by construction. However, they might be rank-deficient, which rules out the use of conjugate gradient methods, or ill-conditioned, i.e. the condition number for a symmetric matrix A_l (with respect to the spectral matrix norm $\|\cdot\|_2$),

$$\kappa(A_l) := \|A_l\|_2 \left\|A_l^{-1}\right\|_2 = \sqrt{\frac{\lambda_{\max}(A_l^\top A_l)}{\lambda_{\min}(A_l^\top A_l)}} \ ,$$

is large. Here, $\lambda_{\min}(A_l)$ and $\lambda_{\max}(A_l)$ denote the smallest and largest eigenvalues of A_l, respectively. [1, Theorem 10.2.6] gives an estimation on the convergence rate of a conjugate gradient method ([1, Algorithm 10.2.1]) for solving problem $Ax = b$ with $A \succ 0$:

$$\left\| x^{(i)} - x \right\|_A \le 2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^i \left\| x^{(0)} - x \right\|_A \ ,$$

with $\|x\|_A = \sqrt{x^\top A x}$ and $x^{(i)}$ is the sequence created by the conjugate gradient algorithm. Thus, the condition numbers of A_l directly influence the performance of the overall method, if a conjugate gradient method is used to solve each subproblem in every iteration k. In practice, the embedded linear solvers may even fail due to the limited accuracy of the internal number representation. For this reason we propose a framework, which allows to improve the properties of the subproblems by adding arbitrary symmetric, positive semidefinite matrices B_l to A_l. The condition number then changes from $\kappa(A_l)$ to $\kappa(A_l + B_l)$. We ensure that the overall objective is not altered.

5.2.1 Definition

The objective function of the underlying optimisation problems,

$$\inf_{v_l} f_l(v_l) + C_l^\top \lambda ,$$

is extended by a regularisation term that involves the value from the previous iteration, $v_l^{(k)}$:

$$\begin{aligned}\tilde{f}_l(v_l) :=& f_l(v_l) + \frac{1}{2}\left(v_l - v_l^{(k)}\right)^\top B_l \left(v_l - v_l^{(k)}\right) \\ =& \frac{1}{2} v_l^\top \left(A_l + B_l\right) v_l - v_l^\top \left(b_l + B_l v_l^{(k)}\right) + \frac{1}{2}\left(v_l^{(k)}\right)^\top B_l v_l^{(k)}\end{aligned}$$

with some arbitrary, symmetric and positive semidefinite matrix B_l. Then the linear problem (in v_l) to solve in each subproblem is

$$\left(A_l + B_l\right) v_l = b_l + B_l v_l^{(k)} - C_l^\top \lambda^{(k)} , \tag{5.19}$$

or expressed as iteration step (replacing (5.15)) for the primal variables:

$$v_l^{(k+1)} \leftarrow \left(A_l + B_l\right)^{-1} \left(b_l + B_l v_l^{(k)} - C_l^\top \lambda^{(k)}\right) , \qquad l \in \mathcal{L} . \tag{5.20}$$

In this representation it becomes apparent that B_l allows to directly modify the matrix involved in the linear system that has to be solved for each subproblem. By setting $B_l = 0$ for all $l \in \mathcal{L}$, we obtain the original update step (5.15). In Sect. 5.3 we demonstrate how knowledge about the problem structure can be used to choose the modifying matrices and to improve the numerical properties significantly. The modified iterative update of the variables is summarised in Algorithm 5.5 and the final method can be found in Algorithm 5.6.

There is a connection to proximal point methods [67] used to obtain strictly convex subproblems for sequentially solving a convex optimisation problem. The additional term in the modified objective function can be regarded as a unit-weighted distance measurement $d_B(x_1, x_2) := \frac{1}{2} x_1^\top B x_2$, so

$$\tilde{f}_l(v_l) = f_l(v_l) + d_B\left(v_l, v_l^{(k)}\right)$$

However, the author of [67] only considers the case $B = I$ and we are not aware of any result for general positive semidefinite B. Furthermore, the convergence properties of the overall iteration step cannot be analysed by investigating the subproblems separately due to the interconnection caused by the consistency constraints, which occur as linear terms in λ.

5.2.2 Convergence

In order to show that the modified approach still solves the original problem, and to derive some convergence properties, we rewrite the algorithm in a compact form. We combine the new update steps (5.20) for the primal variables and the unmodified ones for the dual variables (5.16) (multiplied by $-\alpha^{-1}$) in a single linear system (implicitly defining M, N, $z^{(k)}$ and f),

$$\underbrace{\begin{pmatrix} A+B & 0 \\ C & -\alpha^{-1}I \end{pmatrix}}_{M} \underbrace{\begin{pmatrix} v^{(k+1)} \\ \lambda^{(k+1)} \end{pmatrix}}_{z^{(k+1)}} = \underbrace{\begin{pmatrix} B & -C^\top \\ 0 & -\alpha^{-1}I \end{pmatrix}}_{N} \underbrace{\begin{pmatrix} v^{(k)} \\ \lambda^{(k)} \end{pmatrix}}_{z^{(k)}} + \underbrace{\begin{pmatrix} b \\ 0 \end{pmatrix}}_{f} ,$$

where we joined the subproblem descriptions into

$$\begin{aligned} A &:= \operatorname{diag}(A_1, \dots, A_d) , \\ B &:= \operatorname{diag}(B_1, \dots, B_d) , \\ C &:= \begin{pmatrix} C_1 & \cdots & C_d \end{pmatrix} , \\ \text{and} \qquad b &:= \begin{pmatrix} b_1^\top & \cdots & b_d^\top \end{pmatrix}^\top . \end{aligned}$$

The update step scaling α is restricted to be constant here. In this form the approach can be interpreted as a splitting method, that iteratively finds a solution to the linear system $(M-N)z = f$. Although there exist general results on the convergence of this method, e.g. [1, Theorem 10.1.1], we were not able to derive comprehensive conditions for the considered method. However, using the introduced compact notation, we can slightly reformulate the update steps as

$$v^{(k+1)} \leftarrow v^{(k)} + Q_A^{-1}\left(b - \left(Av^{(k)} + C^\top \lambda^{(k)}\right)\right) , \tag{5.21}$$

$$\text{and} \qquad \lambda^{(k+1)} \leftarrow \lambda^{(k)} + Q_C^{-1}\left(Cv^{(k+1)} - 0\right) , \tag{5.22}$$

with matrices $Q_A := A + B$ and $Q_C := \alpha^{-1}I$. This is known as the *inexact Uzawa algorithm*, see [68] or [69, Algorithm 2.3], a modification of the original Uzawa update method [70]. Then Q_A and Q_C can be interpreted as preconditioning matrices for the linear problems which are solved to update the primal and dual variables, respectively. For any starting point $(v^{(0)}, \lambda^{(0)})$, the inexact Uzawa algorithm finds a solution of the saddle point problem

$$\begin{pmatrix} A & C^\top \\ C & 0 \end{pmatrix} \begin{pmatrix} v \\ \lambda \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix} , \tag{5.23}$$

if it converges. In [69] the authors examine the inexact Uzawa algorithm and give sound convergence conditions and an estimation on the convergence rate for general symmetric Q_A and Q_C. Let $\|x\|_S := \sqrt{x^\top S x}$ be the norm defined with respect to a symmetric, positive definite matrix S. Furthermore, we abbreviate $Q := \operatorname{diag}(Q_A - A, Q_C)$, so

$$\left\| \begin{pmatrix} v \\ \lambda \end{pmatrix} \right\|_Q = \left(v^\top (Q_A - A)v + \lambda^\top Q_C \lambda\right)^{\frac{1}{2}} .$$

Theorem 5.2.1. *Assume that A is a symmetric and positive definite linear matrix, and C is a matrix of full rank. Let $\delta, \gamma \in [0,1)$ be the smallest values satisfying*

$$(1-\delta)Q_A \preceq A \,, \tag{5.24}$$

$$\text{and} \qquad (1-\gamma)Q_C \preceq CA^{-1}C^\top \,. \tag{5.25}$$

Furthermore, presume that

$$A \preceq Q_A \,, \tag{5.26}$$

$$\text{and} \qquad CA^{-1}C^\top \preceq Q_C \tag{5.27}$$

hold. Then the sequence $z^{(k)} = (v^{(k)}, \lambda^{(k)})$ generated by (5.21)-(5.22) *converges to the solution (v^*, λ^*) of* (5.23) *with*

$$\left\| v^{(k)} - v^* \right\|_A \leq \rho^{k-1} e_0 \,,$$

$$\text{and} \qquad \left\| \lambda^{(k)} - \lambda^* \right\|_{Q_C} \leq \rho^k e_0 \,,$$

with the initial error $e_0 := \left\| z^{(0)} - z^* \right\|_Q$. The convergence rate is given by

$$\rho := \frac{\gamma(1-\delta) + \sqrt{\gamma^2(1-\delta)^2 + 4\delta}}{2} \,. \tag{5.28}$$

Proof. see Corollary 3.2 in [69]. □

Let's apply this insight to our approach for solving a decomposed problem via its dual representation. It covers the modified variant and the original one as a special case.

Corollary 5.2.2. *The iteration* (5.21)-(5.22) *solves the decomposed problem* (5.13) *for any initial $(v^{(0)}, \lambda^{(0)})$, if $A \succ 0$, $B \succeq 0$, C has full row rank and the step size parameter is chosen as*

$$\alpha = \beta \left(\lambda_{\max}(CA^{-1}C^\top) \right)^{-1}$$

with $\beta \in (0,1]$. The convergence rate is (5.28), *with*

$$\delta = \lambda_{\max}\left((A+B)^{-1}B \right) \tag{5.29}$$

$$\text{and} \qquad \gamma = 1 - \frac{\beta}{\kappa\left(CA^{-1}C^\top\right)} \,. \tag{5.30}$$

Proof. For our approach we have $Q_A = A + B$ and $Q_C = \alpha^{-1} I$ and the constraints (5.26) and (5.27) turn into

$$0 \preceq B$$

$$\text{and} \qquad CA^{-1}C^\top \preceq \lambda_{\max}(CA^{-1}C^\top)\, I \preceq \beta^{-1}\lambda_{\max}(CA^{-1}C^\top)\, I = \alpha^{-1} I = Q_C \,,$$

respectively, which are fulfilled by presumption. Furthermore (5.25) reads

$$1 - \gamma \leq \alpha \lambda_{\min} \left(CA^{-1}C^\top \right) \ ,$$

and substituting the definition of α, we get a bound for γ:

$$\gamma \geq 1 - \alpha \lambda_{\min} \left(CA^{-1}C^\top \right) = 1 - \beta \frac{\lambda_{\min}\left(CA^{-1}C^\top\right)}{\lambda_{\max}(CA^{-1}C^\top)} = 1 - \frac{\beta}{\kappa(CA^{-1}C^\top)} \tag{5.31}$$

The condition number of $CA^{-1}C^\top$ is bounded from above, which follows from the positive definiteness of A and the full rank of C (see [1, Theorem 4.2.1]). As $0 < \beta \leq 1$ and $\kappa(\cdot) \geq 1$, there is always a γ in $[0, 1)$, which fulfils relation (5.31), in particular with equality. Substituting the definition of Q_A into (5.24) gives

$$\delta \geq \lambda_{\max} \left(I - (A+B)^{-1}A \right) = \lambda_{\max} \left((A+B)^{-1}B \right) \ .$$

From the preconditions $A \succ 0$ and $B \succeq 0$ follows that there is always a $\delta \in [0, 1)$ that satisfies this relation. Thus, Theorem 5.2.1 applies and the proposed variable update sequence converges for any initial value, and with convergence rate (5.28). Furthermore, it finds a solution of the linear system

$$\begin{pmatrix} A & C^\top \\ C & 0 \end{pmatrix} z = \begin{pmatrix} A_1 & 0 & \cdots & 0 & C_1^\top \\ 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & A_d & C_d^\top \\ C_1 & \cdots & \cdots & C_d & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ \vdots \\ v_d \\ \lambda \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ b_d \\ 0 \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix} \ .$$

However, these equations exactly represent the first-order Karush-Kuhn-Tucker conditions for (5.13),

$$\begin{aligned} & \nabla_{v_l} L(v, \lambda) = A_l v_l + C_l^\top \lambda - b_l = 0 \ , \qquad \forall l \in \mathcal{L} \\ \text{and} \qquad & \nabla_\lambda L(v, \lambda) = \sum_{l \in \mathcal{L}} C_l v_l = 0 \ . \end{aligned}$$

Thus, the proposed method solves the considered problem in its decomposed form under the stated conditions. □

The requirement $A \succ 0$ holds by the assumption made in Sect. 5.1.2 that all A_l are positive definite. The full rank of C can easily be enforced as discussed in Sect. 5.1.3.

On the basis of (5.28) and (5.30) the dual update step scaling can always be chosen with $\beta = 1$ to maximise convergence speed. However, other aspects, such as numerical inaccuracies, might require to reduce this value, and thus we will not eliminate β in the further analysis.

For the unmodified primal update steps, i.e. $B_l = 0$ for all $l \in \mathcal{L}$, the convergence rate simplifies to

$$\rho = \gamma = 1 - \frac{\beta}{\kappa\left(CA^{-1}C^\top\right)} = 1 - \frac{\beta}{\kappa\left(\sum_{l \in \mathcal{L}} C_l A_l^{-1} C_l^\top\right)} \ ,$$

so it is necessary to find an estimate on the condition number of $C_l A_l^{-1} C_l^\top$. The special structure of the constraint matrices C_l, which contain not more than one non-zero entry ± 1 per row in our definition (see Sect. 5.1.3), should be considered.

Adding regularising terms influences the approach as following: On the one hand, the condition number $\kappa(A_l + B_l)$ of the subproblems can be reduced by choosing appropriate matrices B_l. This improves the performance of the embedded solvers as discussed at the beginning of this section. On the other hand, the convergence of the outer iteration also depends on the regularisation as (5.29) indicates. More precisely, the convergence rate (5.28) monotonously approaches one with larger δ as following relation shows (for $\delta, \gamma \in [0,1)$):

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\delta}\rho(\gamma,\delta) &= \frac{2-\gamma^2(1-\delta)-\gamma\sqrt{\gamma^2(1-\delta)^2+4\delta}}{2\sqrt{\gamma^2(1-\delta)^2+4\delta}} \\
&\geq \frac{2-\gamma^2(1-\delta)-\gamma\sqrt{\gamma^2(1-\delta)^2+4\delta+4(1-\gamma)(1-\delta)}}{2\sqrt{\gamma^2(1-\delta)^2+4\delta}} \\
&= \frac{2-\gamma^2(1-\delta)-\gamma\sqrt{(2-\gamma(1-\delta))^2}}{2\sqrt{\gamma^2(1-\delta)^2+4\delta}} \\
&= \frac{(2-2\gamma)+\gamma(2-\gamma(1-\delta))-\gamma(2-\gamma(1-\delta))}{2\sqrt{\gamma^2(1-\delta)^2+4\delta}} \\
&= \frac{1-\gamma}{\sqrt{\gamma^2(1-\delta)^2+4\delta}} > 0
\end{aligned}$$

As ρ should be low, a small δ is always preferable. Due to (5.29) the optimal choice is $B_l = 0$, which, however, might contradict the aim to have well-posed subproblems.

Summing up, the proposed method of the algorithm allows to improve the numerical properties of the subproblems. Despite the modification, still a solution to the initial problem is found. Convergence conditions and an estimation of the rate were derived. The regularising matrices are only constrained to be symmetric and positive semidefinite and must be carefully chosen to balance the convergence rate of the inner (here: conjugate gradient) and outer (update of the dual variables) iteration loop.

Algorithm 5.4: Construction of the consistency constraints.

```
Algorithm: construct consistency constraints
Input: $\{\mathcal{J}^l_{\mathrm{C}}, \mathcal{J}^l\}_{\mathcal{L}}$
Output: $\{C_l\}_{\mathcal{L}}$
/* initialise consistency constraint matrices */
$C_l \leftarrow$ (empty) matrix of size $0 \times \left|\mathcal{J}^l\right|$ ,    $l \in \mathcal{L}$
/* for each complicating variable construct the necessary
   constraints */
foreach $j \in \bigcup_{l \in \mathcal{L}} \mathcal{J}^l_{\mathrm{C}}$ do
   /* identify the subproblems the variable is involved in */
   $L \leftarrow \left\{l \in \mathcal{L} \,\middle|\, j \in \mathcal{J}^l_{\mathrm{C}}\right\}$
   /* create a graph that represents the pairwise constraints */
   setup graph $G := (N, E)$ with
       $N = L$
       $E =$ edges according to some pattern (see Fig. 5.1), so $G$ is connected
   /* for each edge in the graph create a pairwise constraint */
   foreach $(l_1, l_2) \in E$ do
      foreach $l \in \mathcal{L}$ do
         /* determine sign of variable in this constraint */
         $\alpha \leftarrow \begin{cases} +1 & \text{if } l = l_1 \\ -1 & \text{if } l = l_2 \\ 0 & \text{otherwise} \end{cases}$
         /* append a row to the constraint matrix */
         $C_l \leftarrow \begin{pmatrix} C_l \\ \alpha R^{\top}_{\{j\} \rightarrow \mathcal{J}^l} \end{pmatrix}$
      end
   end
end
```

Algorithm 5.5: Iterative solving of a decomposed convex quadratic problem via its dual representation.

```
Algorithm: Iterative Solution of Dual Decomposed Quadratic Problems (Modified)
Input: decomposed problem: ${A_l, b_l, C_l}_\mathcal{L}$
Input: regularisation terms: ${B_l}_\mathcal{L}$
Output: solution ${v_l}_\mathcal{L}$

/* initialise variables */
$\lambda^{(0)} \leftarrow 0$
$v_l^{(0)} \leftarrow 0 \quad \forall l \in \mathcal{L}$
$k \leftarrow 0$

/* iterate the update of primal and dual variables */
repeat
    /* update primal variables */
    solve subproblems $l \in \mathcal{L}$ in parallel:
        find $v_l^{(k+1)}$ so $(A_l + B_l)\, v_l^{(k+1)} = \left(b_l + B_l v_l^{(k)} - C_l^\top \lambda^{(k)}\right)$

    /* update dual variables */
    $s \leftarrow \sum_{l \in \mathcal{L}} C_l v_l^{(k+1)}$
    $\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \alpha^{(k)} s$

    $k \leftarrow k + 1$
until convergence
```

Algorithm 5.6: General approach for decomposition and solving of a convex quadratic optimisation problem in the form $\frac{1}{2}\left\|Du + c\right\|_2^2$ by Dual Decomposition, extended by the regularisation of the underlying subproblems.

```
Algorithm: solve a convex quadratic problem via Dual Decomposition
Input: problem description $D, c$
Output: solution $u$

create decomposition ${A_l, b_l, C_l, Q_l}_\mathcal{L}$

create regularisation terms ${B_l}_\mathcal{L}$

solve problem via Dual Decomposition, obtaining ${v_l}_\mathcal{L}$

construct solution $u$ of the original problem from ${v_l}_\mathcal{L}$
```

5.3 Case Studies

In this section we present three variational problem which we exemplary consider for decomposition with the proposed method. We give their definition and detail the discretisation we used to obtain a convex quadratic optimisation problem of the form

$$\inf_u \frac{1}{2} \|Du + c\|_2^2 \ .$$

In Sect. 5.3.1 we start with a simple variational image denoising approach with first- and second-order terms. We will use the abbreviation ID to identify this problem uniquely.

Furthermore we examine two variational approaches to motion estimation based on the optical flow constraint: The well-known approach by Horn and Schunck (denoted as HS) is considered in Sect. 5.3.2. In contrast the formulation discussed in Sect. 5.3.3 (DC) uses higher-order regularisation terms. In this context we employ the presented algorithm extension to obtain numerical well posed subproblems.

5.3.1 Image Denoising

The first problem we considered is a global approach to image denoising [71]. Given a grey-valued image g, defined on Ω, we want to find the image u, that solves the following optimisation problem:

$$\inf_u \frac{1}{2} \int_\Omega (g-u)^2 + \beta_1^2 \|\nabla u\|_2^2 + \beta_2^2 \left\|\nabla \nabla^\top u\right\|_F^2 \, \mathrm{d}x \tag{5.32}$$

The first term ("data term") ensures similarity between the resulting and original image, while the second and third ones ("regularisation terms") enforce smoothness of u and ∇u, respectively. The scalar parameters β_1 and β_2 control the balance between the three goals.

Discretisation

For discretisation we use a regular grid with square elements, see Fig. 5.2(a), and approximate derivatives using finite differences. Denoting the element of variable x located at position (i, j) as $x_{i,j}$, we define a vector representation,

$$\begin{aligned} \mathrm{vec} &: \mathbb{R}^{n \times m} \mapsto \mathbb{R}^{nm} \\ \mathrm{vec}(x) &:= \begin{pmatrix} x_{1,1} & \cdots & x_{n,1} & x_{1,2} & \cdots & x_{n,2} & \cdots & x_{1,m} & \cdots & x_{n,m} \end{pmatrix}^\top \ . \end{aligned}$$

In addition we introduce the following spatial differential operators, based on finite differences:

$$
\begin{aligned}
D_{\partial_x} &:= \begin{pmatrix} I_m \otimes D_n & 0 \\ 0 & I_m \otimes D_n \end{pmatrix} \in \mathbb{R}^{2(m-1)n \times 2nm} && \text{(derivative in x-direction)}\ , \\
D_{\partial_y} &:= \begin{pmatrix} D_n \otimes I_m & 0 \\ 0 & D_n \otimes I_m \end{pmatrix} \in \mathbb{R}^{2(n-1)m \times 2nm} && \text{(derivative in y-direction)}\ , \\
\text{with } D_n &:= \begin{pmatrix} -1 & +1 & 0 & \cdots & 0 \\ 0 & -1 & +1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 & +1 \end{pmatrix} \in \mathbb{R}^{(n-1)\times n}\ ,
\end{aligned}
$$

and the identity matrix $I_n \in \mathbb{R}^{n\times n}$. For the sake of readability we will omit the specification of the operator dimensions in the further description.

Based on the derivative operators we define some more complex matrices,

$$
\begin{aligned}
D_\nabla &:= \begin{pmatrix} D_{\partial_x} \\ D_{\partial_y} \end{pmatrix} && \text{first spatial derivatives (gradient)} \\
D_{\mathrm{H}} &:= \begin{pmatrix} D_{\partial_x} D_{\partial_x} \\ D_{\partial_y} D_{\partial_x} \\ D_{\partial_x} D_{\partial_y} \\ D_{\partial_y} D_{\partial_y} \end{pmatrix} && \text{second spatial derivatives (Hessian)}
\end{aligned}
$$

In the discrete representation of the problem, we implicitly identify the image data g and the variables u with their vector representation, i.e. $g \to \operatorname{vec}(g)$ and $u \to \operatorname{vec}(u)$. Then the discretised objective function of problem (5.32) is given by

$$
f_{\mathrm{ID}}(u) := \frac{1}{2}\,\|u-g\|_2^2 + \frac{1}{2}\beta_1^2\,\|D_\nabla u\|_2^2 + \frac{1}{2}\beta_2^2\,\|D_{\mathrm{H}} u\|_2^2 \tag{5.33}
$$

An important observation is, that any sum of $\|Du + c\|_2^2$-formed terms can be joined into a single one,

$$
\sum_{k=1}^{n} \frac{1}{2}\,\|D_k u + c_k\|_2^2 = \sum_{k=1}^{n} \frac{1}{2}\,(D_k u + c_k)^\top (D_k u + c_k) = \frac{1}{2} \left\| \begin{pmatrix} D_1 \\ \vdots \\ D_n \end{pmatrix} u + \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix} \right\|_2^2 ,
$$

which is exactly the form expected by our decomposition approach. When applied to (5.33), the objective function reads

$$
f_{\mathrm{ID}}(u) = \frac{1}{2}\,\|u-g\|_2^2 + \frac{1}{2}\,\|\beta_1 D_\nabla u\|_2^2 + \frac{1}{2}\,\|\beta_2 D_{\mathrm{H}} u\|_2^2 = \frac{1}{2} \left\| \begin{pmatrix} I \\ \beta_1 D_\nabla \\ \beta_2 D_{\mathrm{H}} \end{pmatrix} u + \begin{pmatrix} -g \\ 0 \\ 0 \end{pmatrix} \right\|_2^2 .
$$

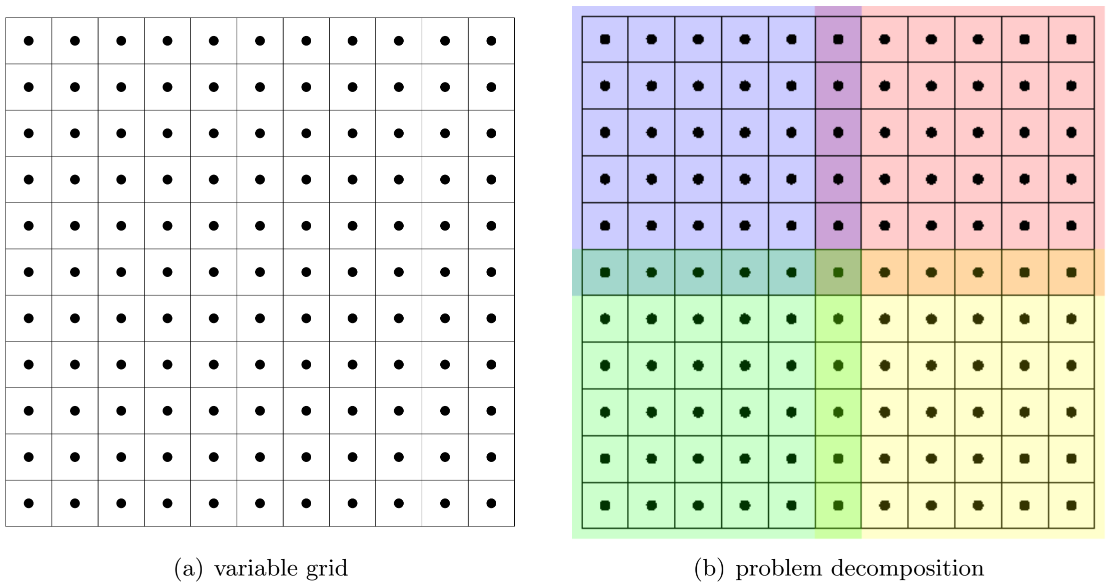


Figure 5.2: Variable discretisation grid for finite differences: Each dot represents the position of a variable. The box grid represents the pixels of the image data. (a) Original grid. (b) Problem split up into four overlapping subdivisions $\Omega_1, \ldots, \Omega_d$, marked by colours.

Decomposition

For the decomposition we defined d equally-sized subdivisions $\Omega_1, \ldots \Omega_d$ of Ω, which overlap by one grid unit. Each variable is assigned to the domains according to its position which is identical to the centre of the image pixels. Figure 5.2(b) illustrates an exemplary decomposition for $d = 4$.

5.3.2 Optical Flow Estimation with First-Order Regularisation

In [14] a variational approach to optical flow estimation is proposed. The aim is to find a vector field $u(x) := \big(u_1(x), u_2(x)\big)^\top$ for a given image sequence $g(x,t)$:

$$\inf_u \int_\Omega \frac{1}{2}\left(\nabla g^\top u + g_t\right)^2 + \frac{1}{2}\beta^2\left(\|\nabla u_1\|_2^2 + \|\nabla u_2\|_2^2\right) \mathrm{d}x \,,$$

where ∇g and g_t represent the spatial and temporal derivatives of g. The first term measures the coherence with the optical flow constraint, i.e. whether the displacement vectors u describe the intensity changes in the image sequence well. The second term enforces smoothness of the vector field, rendering the overall optimisation problem well-posed.

Discretisation

For our experiments we used the same discretisation as for the image denoising problem, see Sect. 5.3.1 and Fig. 5.2. The input data is given by two image frames $g_1(x) := g(x, t_1)$ and $g_2(x) := g(x, t_2)$ with $t_2 - t_1 = 1$. The spatial image derivatives $g_x(i, j)$ and $g_y(i, j)$ were estimated from $\frac{1}{2}(g_1 + g_2)$ using binomial finite impulse response (FIR) filters (see [47] or [48]). These are, in stencil notation,

$$\frac{1}{8}\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{pmatrix} \quad \text{and} \quad \frac{1}{8}\begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix} ,$$

respectively. A simple difference $g_t = g_2 - g_1$ was used for the temporal derivatives. For the optical flow constraint (OFC) these values were arranged according to

$$\begin{aligned} D_{\text{OFC}} &:= \begin{pmatrix} \operatorname{diag}(\operatorname{vec}(g_x)) & \operatorname{diag}(\operatorname{vec}(g_y)) \end{pmatrix} \\ \text{and } c_{\text{OFC}} &:= \operatorname{vec}(g_t) \, . \end{aligned}$$

The variables u are stacked together coordinate- and then component-wise, i.e.

$$\begin{pmatrix} \operatorname{vec}(u_1) \\ \operatorname{vec}(u_2) \end{pmatrix} .$$

Then the objective function for the approach by Horn and Schunck in discrete form reads

$$\begin{aligned} f_{\text{HS}}(u) &:= \frac{1}{2}\left\| D_{\text{OFC}} u + c_{\text{OFC}} \right\|_2^2 + \frac{1}{2}\beta^2 \left(\| D_\nabla u_1 \|_2^2 + \| D_\nabla u_2 \|_2^2 \right) \\ &= \frac{1}{2}\left\| D_{\text{OFC}} u + c_{\text{OFC}} \right\|_2^2 + \frac{1}{2}\left\| \beta D_{\text{J}} u \right\|_2^2 \\ \text{with } D_{\text{J}} &:= \begin{pmatrix} D_\nabla & 0 \\ 0 & D_\nabla \end{pmatrix} . \end{aligned}$$

This can easily be reformulated as

$$f_{\text{HS}}(u) = \frac{1}{2}\left\| \begin{pmatrix} D_{\text{OFC}} \\ \beta D_{\text{J}} \end{pmatrix} u + \begin{pmatrix} c_{\text{OFC}} \\ 0 \end{pmatrix} \right\|_2^2 ,$$

making the proposed decomposition method applicable.

Decomposition

For this case we basically used the same method to decompose the objective function as for the image denoising approach. Both the x- and the y-components of the displacement vector are assigned to the position of the according pixel. For four subdivisions, the decomposition in illustrated in Fig. 5.2(b).

5.3.3 Optical Flow Estimation with Higher-Order Regularisation

The optical flow estimation approach proposed in [17] uses higher-order regularisation on the vector field:

$$\inf_u \frac{1}{2}\int_\Omega \left(\nabla g^\top u + g_t\right)^2 + \beta_1^2 \left\|\nabla \mathrm{div}\; u\right\|_2^2 + \beta_2^2 \left\|\nabla \mathrm{curl}\; u\right\|_2^2 \,\mathrm{d}x + \frac{1}{2}\beta_3^2 \int_{\partial\Omega} \left\|\partial_n u\right\|_2^2 \,\mathrm{d}x \;,$$

where div $:= \partial_x + \partial_y$, curl $:= \partial_x - \partial_y$ and ∂_n is the normal derivative on the image domain boundary $\partial\Omega$. The parameters β_1, β_2 and β_3 weight the regularisation terms, which control smoothness of the divergence and of the flow rotation in Ω, while the third one adds boundary terms to improve the numerical stability of the problem.

Discretisation

For the discretisation we used the framework based on mimetic differences described in [72] and [73]. Details on the operator discretisation can be found in [17]. In Fig. 5.3(a) the underlying variable grid is depicted. Shading indicates the location of boundary terms $\|\partial_n u\|_2^2$. Using the notation introduced in [17], the discrete version of the objective function is

$$f_{\mathrm{DC}}(u) := \frac{1}{2}\left\|Gu + \partial_t g\right\|_{H_V}^2 + \frac{1}{2}\beta_1^2 \left\|\overline{\mathbb{G}}\mathbb{D}iv\; u\right\|_{H_S}^2 + \frac{1}{2}\beta_2^2 \left\|\mathbb{G}\overline{\mathbb{C}url}\; u\right\|_{H_E}^2 + \frac{1}{2}\beta_3^2 \left\|\mathbb{P}u\right\|_{bc}^2 \;,$$

where $G = \overline{\mathbb{G}}\,\mathrm{diag}(g)$. In addition $\overline{\mathbb{G}}$, $\mathbb{D}iv$, $\overline{\mathbb{C}url}$ and $\mathbb{P}$ are linear operators for the gradient of a scalar field, and the divergence, curl and the normal derivative of a vector field. Here $g = \frac{1}{2}(g_1 + g_2)$ and $g_t = g_2 - g_1$ are both represented in vector representation. Due to their definition, the H_V-, H_S-, H_E- and bc-norms are equivalent to the Euclidean norm. Hence, we can rewrite the objective function as

$$f_{\mathrm{DC}}(u) = \frac{1}{2}\left\| \begin{pmatrix} G \\ \beta_1\overline{\mathbb{G}}\mathbb{D}iv \\ \beta_2\mathbb{G}\overline{\mathbb{C}url} \\ \beta_3\mathbb{P} \end{pmatrix} u + \begin{pmatrix} \partial_t g \\ 0 \\ 0 \\ 0 \end{pmatrix} \right\|_2^2 \;,$$

obtaining a representation as presumed by the proposed decomposition algorithm.

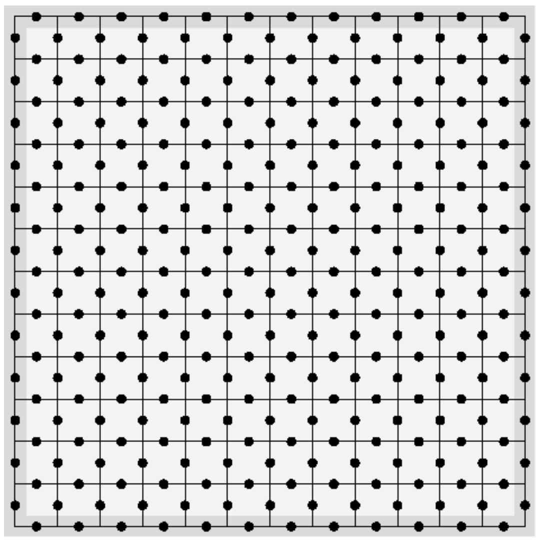
Decomposition and Additional Subproblem Regularisation

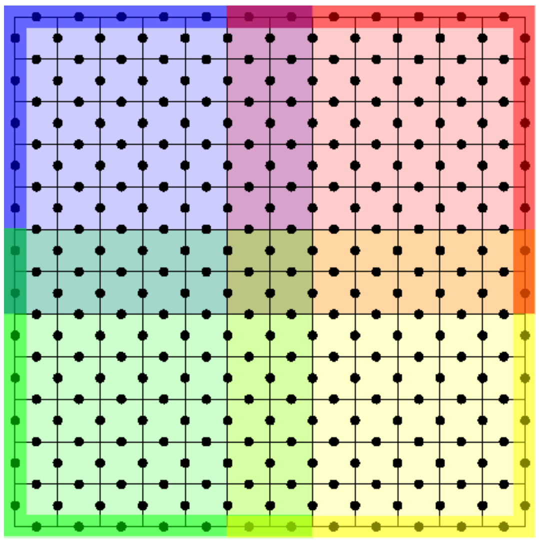
Figure 5.3(b) illustrates the decomposition of the original problem into four regions which overlap by two grid cells. The location of boundary terms $\frac{1}{2}\left\|\partial_n u\right\|_2^2$ of the original problem are shaded.

However, first experiments showed that a decomposition by the proposed method leads to convex subfunctions but with ill-conditioned matrices A_l. This causes a divergent behaviour of the algorithm. In order to improve the numerical properties, we introduce additional terms for the subproblems by means of the framework proposed in Sect. 5.2.

The objective functions f_l were supplemented by regularisation terms $\frac{1}{2}\left\|\partial_n u\right\|_2^2$ on the inner, artificial boundaries introduced by the decomposition. Thereby each subproblem is regularised on *all* its boundaries just as if it is a smaller instance of the original problem whose convergence properties are well analysed. Figure 5.3(c) shows the layout of the decomposed region and the location of the additional boundary terms.

The discrete forms of the new regularisation terms are concentrated into the matrices B_l. As a result of this modification the condition number of the subproblem matrices dropped from $\kappa(A_l) \approx 10^8$ to $\kappa(A_l + B_l) \approx 10^5$ and the algorithm converged quickly.


(a) variable grid


(b) problem decomposition

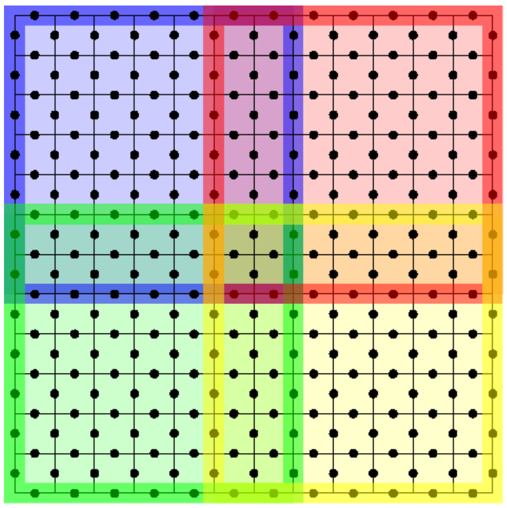

(c) problem decomposition, with additional boundary terms

Figure 5.3: Variable discretisation grid for mimetic differences: Each dot represents the position of a variable. The box grid represents the pixels of the image data. Locations of boundary terms are shaded. (a) Original grid. (b) Problem split into four subproblems with overlapping subdivisions $\Omega_1, \dots, \Omega_4$, marked by colours. (c) Decomposed problem with additional boundary terms (shaded) on the artificial boundaries.

5.4 Experiments

In this section we apply the proposed decomposition approach to the previously introduced case studies. The main objective of this work is to find a solution of the initial problem in a decomposed manner. Thus, we compared the solutions obtained from the proposed decomposition method to one calculated by solving the problem as a whole. Then we verified the quality of the decomposition by means of three error measurements. However, this implies that we have to restrict ourselves to problem sizes that can be handled on a single computational node so that we can generate a reference solution.

The computation was entirely performed in a *MATLAB* environment. The quadratic subproblems as well as the non-decomposed problem were solved using the built-in conjugate gradients squared method (CGS). No efforts were made to actually distribute the subproblems to separate hardware nodes. Instead they were solved sequentially on a single CPU. In Sect. 5.4.1 we further detail on the experimental setups including the problem data, parameters and error measurements.

In the remaining section we investigate the influence of parameter variations on the accuracy of our approach. The update of the dual variables is considered in Sect. 5.4.2. Next, we measure the scalability of the algorithm with respect to the number of subproblems in Sect. 5.4.3. In Sect. 5.4.4 we compare the performance of our approach for different implementations of the consistency constraints. Finally, in Sect. 5.4.5 we investigate to what grade the accuracy of the subproblem solutions influences the overall result. We summarise and conclude in Sect. 5.4.6.

5.4.1 Experimental Setup and Evaluation Procedure

The parameters of the algorithm, including the update rule for the dual variables, were chosen experimentally. However, it became apparent that they scale well with the size of the problem and we could determine a good set of parameters by means of a very small instance.

The maximum number of iterations was fixed in before, usually to ten. No other stopping criteria were used. Instead, we analysed the error measurements over the number of iterations afterwards.

To avoid repetition we will only mention if we deviate from the following default parameter values: The consistency constraint layout (see Sect. 5.1.3) was chosen as *star*, which guarantees the minimum number of constraints (as *chain* would do). The problem domain is decomposed into equally-sized (rounded to full units of the variable grid) regions which overlap by one or two variable grid cells as described in Sect. 5.3.

The CGS solver tolerance is parametrised by an upper bound on $\frac{\|b-Ax\|_2}{\|b\|_2}$ for the approximate solution of an arbitrary problem $Ax = b$. For both the reference as well as the subproblems we set it to a very low value to minimise influence from this error source. For the image denoising problem and the optical flow approach by Horn&Schunck it was chosen as 10^{-12}. However, for the higher-order regularised motion estimation problem (DC) we were forced to reduce it to 10^{-10} because CGS stagnated without reaching an accurate solution, even for the non-decomposed problem. The maximum number of CGS iterations was set to 10000 for both the reference solution as well as the quadratic subproblems.

The step scale $a^{(k)}$ for the update of the dual variables is chosen by

$$\alpha^{(k)} = \frac{1}{a + bk} ,$$

see Sect. 4.3.4. The parameters a and b were adapted for each problem and their influence is investigated in Sect. 5.4.2.

Image Denoising

For the image denoising approach defined in Sect. 5.3.1 we chose the well-known *Lena* image, see Fig. 5.4(a) and normalised its grey-values to the range $[0, 1]$. Then we added normal distributed noise with the same standard deviation as the image itself and obtained Fig. 5.4(b) which acts as input data g. The image data is 512px × 512px in size, thus the problem has about 262000 variables.

The earlier defined variational problem was solved for this image with the denoising parameters set to $\beta_1 = \beta_2 = 5$. The result, depicted in Fig. 5.5, acts as reference u^* for the following experiments. The high degree of smoothness attests the strong interdependency of the variables.

The quality of a solution u obtained from the decomposed approach is evaluated by means of three error measurements which are defined on the pixel-wise absolute difference $\varepsilon(i, j) := \left| u_{i,j} - u^*_{i,j} \right|$ to the reference image.

$$\text{mean error } \mu(\varepsilon) := \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \varepsilon(i, j) , \tag{5.34}$$

$$\text{standard deviation } s(\varepsilon) := \sqrt{\frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (\varepsilon(i, j) - \mu(\varepsilon))^2} , \tag{5.35}$$

$$\text{maximum error } \max(\varepsilon) := \max_{i,j} \varepsilon(i, j) . \tag{5.36}$$

Here, n and m denote the variable grid size.

Optical Flow Estimation

For the optical flow approaches with first- and higher-order regularisation, the same image pair g_1 and g_2 was used. For this purpose we deformed a synthetic particle image by an affine vector field. The images are each 500px × 500px in size. The first of the two frames (g_1) is depicted in Fig. 5.7(a). The grey values were constrained to the range $[0, 1]$. The displacements were scaled to meet $\|u_{i,j}\| \leq 1$ everywhere to make the optical flow constraint applicable. The ground truth motion field is illustrated in Fig. 5.7(b), using the colour encoding for displacements, which was introduced in Chap. 3. For convenience we repeat it in Fig. 5.6.

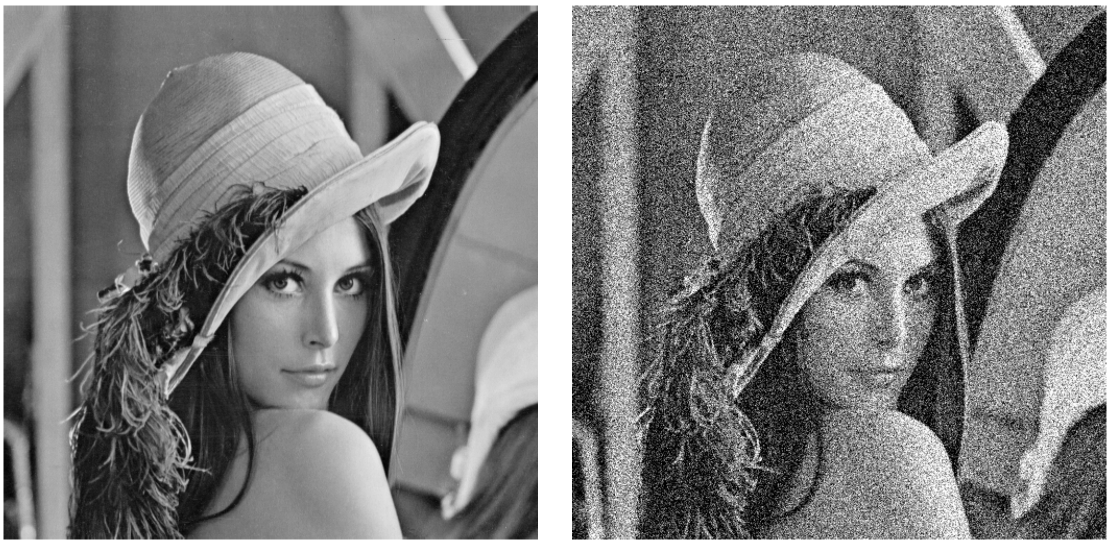


Figure 5.4: Data used for the image denoising experiments: The Lena image, 512px × 512px in size. The grey-values were scaled to the range $[0, 1]$. (a) Original and (b) with additive normal distributed noise.

For all experiments with the optical flow approach by Horn and Schunck (HS) the regularisation parameter was set to $\beta = 1$. The vector field in Fig. 5.8 was calculated without decomposition and acts as reference for the following experiments. The variable vector consists of half a million scalars.

The parameters of the optical flow approach with higher-order regularisation (DC) were chosen as $\beta_1 = \beta_2 = 1$ and $\beta_3 = 0.5$, resulting in the reference vector field depicted in Fig. 5.9. It was necessary to reduce the accuracy of the incorporated CGS-solver to 10^{-10} because otherwise no solution could be determined. Due to the discretisation based on mimetic differences, the number of variables increased to slightly more than half a million.

The error measurements were defined in the same way as for the image denoising experiments in (5.34)-(5.36), with the exception that the point-wise measurement is adapted to vector-valued data: $\varepsilon(i, j) := \|u_{i,j}\|_2$.

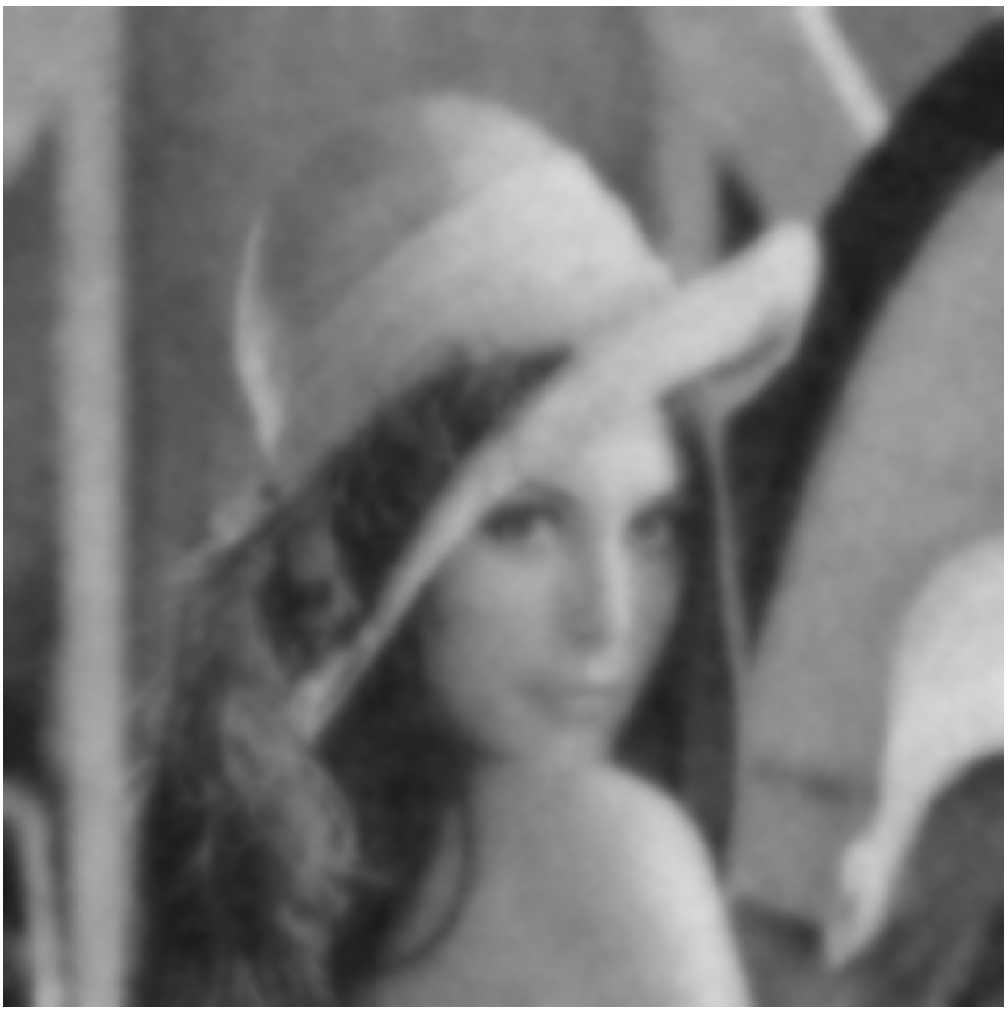


Figure 5.5: Reference for the image denoising experiments: Result of applying the approach to the noisy data, when solved without decomposition.

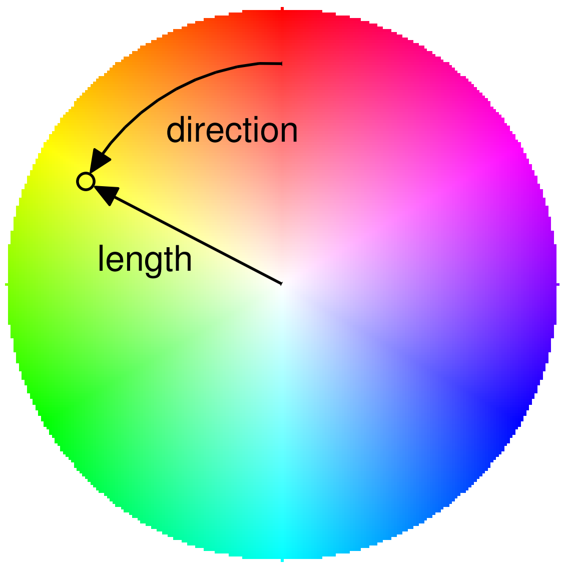


Figure 5.6: Colour encoding for vector fields: Each vector u is represented by a value within the depicted colour range: Hue for the vector direction and saturation for its length from white ($u = 0$) to the pure colour when $\|u\|_2 \geq v_{\max}$, where $v_{\max}$ is the maximum vector length chosen for the representation.

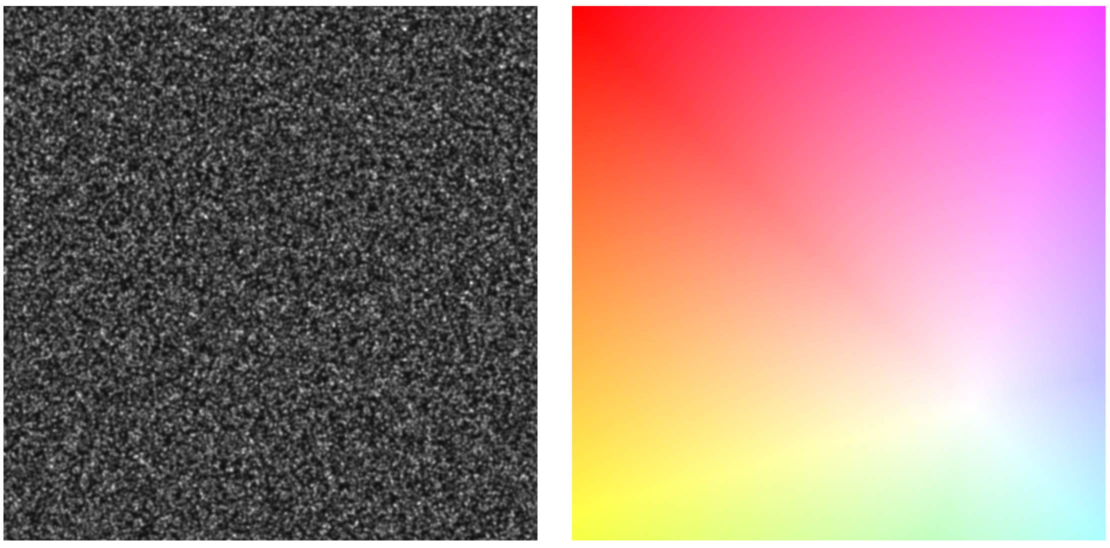


Figure 5.7: Data used for the optical flow estimation experiments: (a) The first frame of the image pair: A grey-valued synthetic particle image, 500px × 500px in size. (b) Ground truth: Affine vector field with a maximum displacement length of 1.

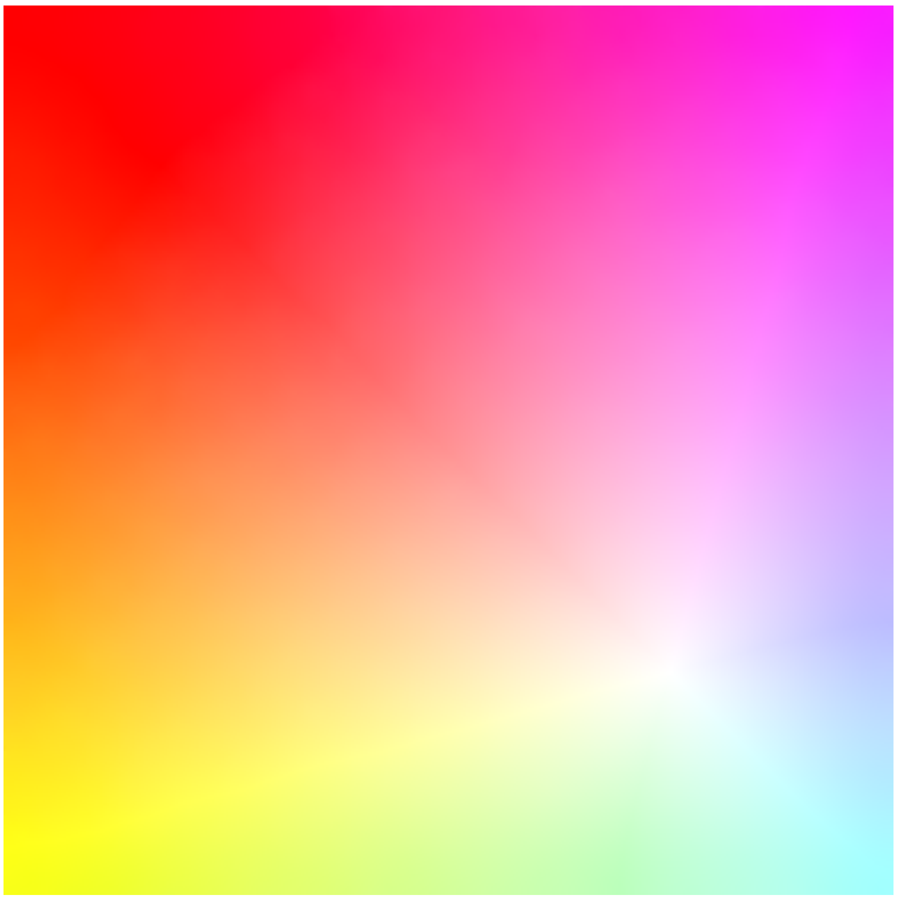


Figure 5.8: Reference for the motion estimation experiments with first-order regularisation: Result of the approach with first-order regularisation (Horn&Schunck), applied to the experimental data defined in Fig. 5.7, when solved without decomposition.

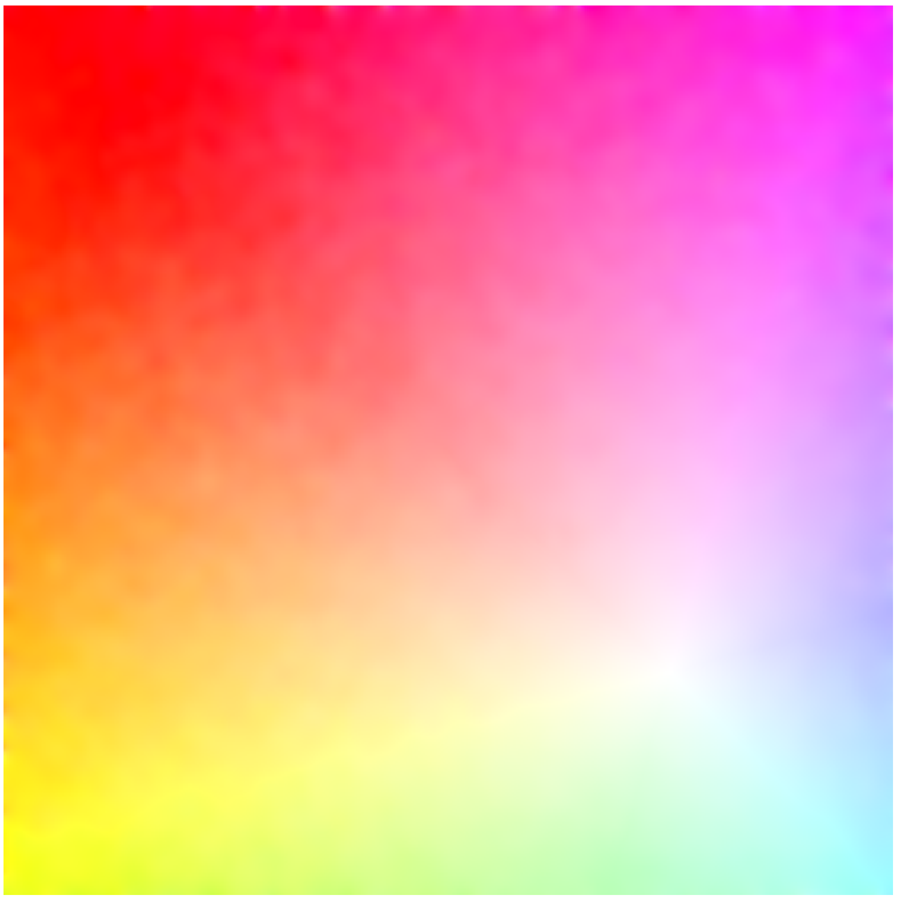


Figure 5.9: Reference for the motion estimation experiments with higher-order regularisation: Result of the optical flow estimation approach with higher-order regularisation, applied to the experimental data defined in Fig. 5.7, when solved without decomposition.

Table 5.1: Error measurements for the decomposed image denoising (ID) approach: Comparison of different parameters for the update of the dual variables.

a	b	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$
0.25	0	—		diverges	
0.5	0	5.11		diverges	
1	0	—	$85.6 \cdot 10^{-6}$	$1339 \cdot 10^{-6}$	$143.0 \cdot 10^{-3}$
2	0	5.10	$41.4 \cdot 10^{-6}$	$269 \cdot 10^{-6}$	$11.4 \cdot 10^{-3}$
4	0	—	$71.8 \cdot 10^{-6}$	$408 \cdot 10^{-6}$	$13.8 \cdot 10^{-3}$
8	0	5.12	$181.7 \cdot 10^{-6}$	$1035 \cdot 10^{-6}$	$36.2 \cdot 10^{-3}$

5.4.2 Update of the Dual Variables

One important aspect of the proposed decomposition method is the update of the dual variables. The theoretical results on the convergence rate presented in our work could not be applied as we lack the necessary estimates on the eigenvalues of the involved matrices. Thus, we experimentally investigated the influence of the parameters a and b for the three case studies.

In Fig. 5.10 we present the error measurement for the image denoising problem when solved in decomposed manner. The parameters of the step size were set to $a = 2$ and $b = 0$ which corresponds to a constant scaling $\alpha^{(k)} = 0.5$ for all k.

The error history plot in Fig. 5.10(a) documents the convergence of the approach to the reference solution, using a logarithmic vertical axis. In iteration $k = 0$ the error to the initial value $v^{(0)}$ – an all-zero vector – is measured and gives an impression of the scale of the error for this problem. The improvement between $k = 0$ and $k = 1$ corresponds to the solution of the subproblems without any consistency constraints due to $\lambda^{(0)} = 0$ at initialisation. In the following steps, the algorithm continues to improve the solution by enforcing the consistency between the subproblems. However, after about four iterations, the process stalls after reaching a high accuracy level.

We omit the illustration of the solution as it is impossible to visually distinguish it from the reference in Fig. 5.5 for typical error values. Instead, the error map in Fig. 5.10(b) employs a logarithmic colour encoding to give an impression of the spatial distribution of the errors $\varepsilon(i, j)$ after the last iteration. Not surprisingly, the significant errors are located at and close to the boundaries between the subproblems while in the remaining regions the deviations are very small. The thin lines of lower errors are exactly in the regions where the subdivisions overlap. They are caused by the averaging of variable values while reconstructing the overall solution from the subproblem variables.

In Table 5.1 we compare the results for different step lengths and give references to plots for two further experiments. For a larger step scaling ($a \to 0$) the mean error tends to diverge. On the other hand, for small step scales ($a > 2$) the algorithm converges only slowly and does not reach a similar accuracy within ten iterations.

In Table 5.2 the results for the decomposition of the optical flow approach by Horn and Schunck are listed for different parameters for the dual variable update. Again, we

Table 5.2: Error measurements for the decomposed optical flow approach with first-order regularisation (HS): Comparison of different parameters for the update of the dual variables.

a	b	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$
20	0	5.14	diverges		
50	0	5.13	$5.1 \cdot 10^{-6}$	$27.1 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$
100	0	—	$91.8 \cdot 10^{-6}$	$173.4 \cdot 10^{-6}$	$7.4 \cdot 10^{-3}$

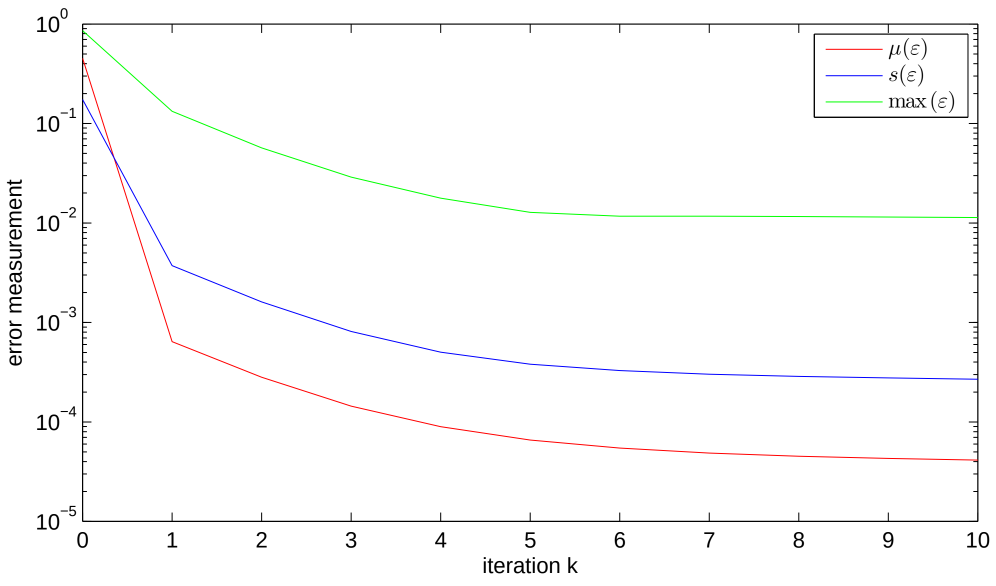
Table 5.3: Error measurements for the decomposed optical flow approach with higher-order regularisation (DC): Comparison of different parameters for the update of the dual variables.

a	b	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$
600	0	5.16	diverges		
500	100	—	$172 \cdot 10^{-6}$	$795 \cdot 10^{-6}$	$80 \cdot 10^{-3}$
600	100	5.15	$176 \cdot 10^{-6}$	$822 \cdot 10^{-6}$	$85 \cdot 10^{-3}$
600	200	—	$195 \cdot 10^{-6}$	$970 \cdot 10^{-6}$	$108 \cdot 10^{-3}$

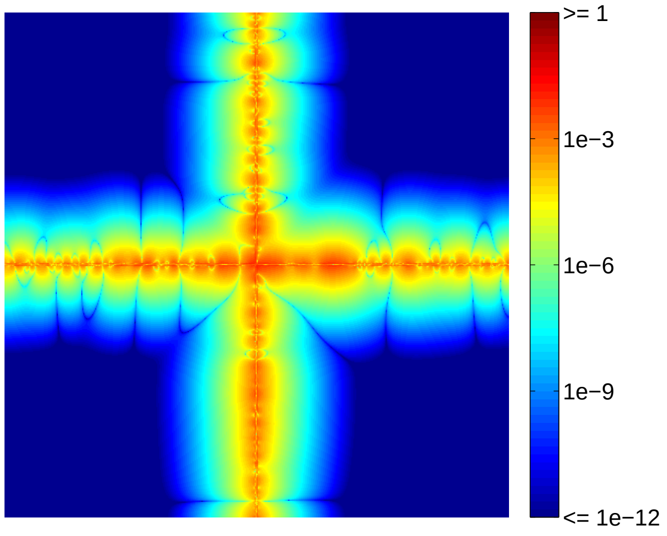
can find a parameter, so the approach converges quickly to a good solution, see Fig. 5.13, while for too large steps, the error diverges (Fig. 5.14).

From our experiments it became clear that the decomposition of the optical flow approach with *higher-order* regularisation is more involved than the previous ones. This is caused by the fact that the higher-order derivatives introduce more complex spatial dependencies between the variables. Without the additional regularisation terms as described in Sect. 5.3.3, the method did not converge at all. Only after defining B_l it was possible to calculate a solution with good accuracy, see Fig. 5.15. The results are more sensible to the choice of the step size parameters, but it was possible to find an appropriate value by examining a smaller instance of the same problem. Furthermore, it was necessary to choose $b > 0$ which reduces the step scaling α with the number of iterations.

The error measurements for four parameter choices are compared in Table 5.3. The best achieved mean error is about 35 times higher than for the approach with first-order constraints, but is still at a good level. A result of marginal better quality than Fig. 5.15 could be achieved by reducing a to 500. In both cases, the rate of improvement after the first two iterations is quite slow due to the small step scaling. However, setting b to zero leads to a divergent behaviour, see Fig. 5.16.

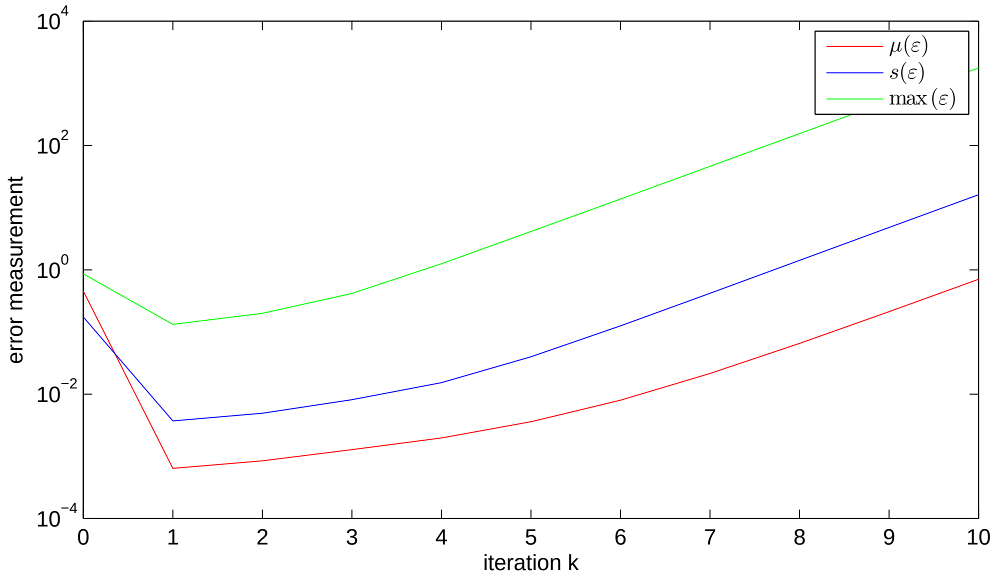


(a) error history

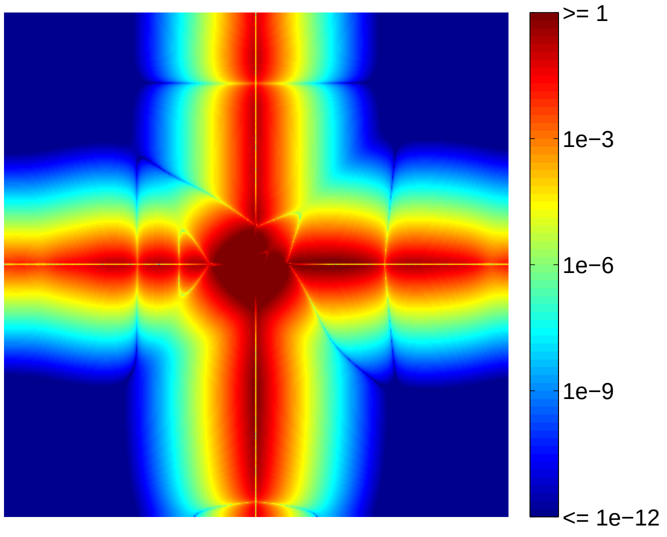


(b) error map $\varepsilon(i, j)$ after the last iteration

Figure 5.10: Results for the image denoising problem (ID), decomposed into $d = 4$ subproblems, with step size parameters $a = 2$ and $b = 0$, *star*-shaped consistency constraints and subproblem accuracy of 10^{-12}: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations.
The approach finds a very accurate solution within few iterations. The remaining errors are located around the boundaries between the subproblems.

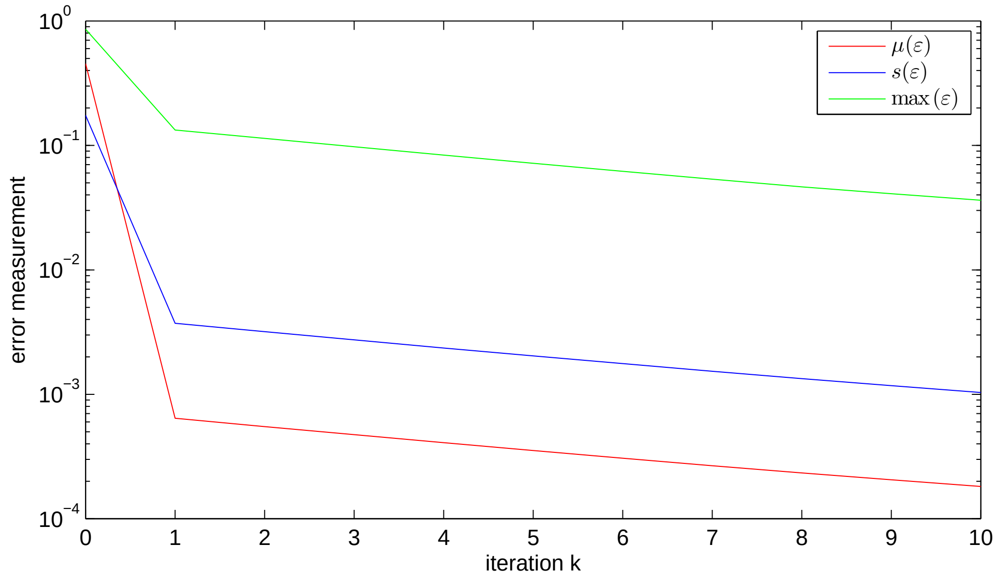


(a) error history

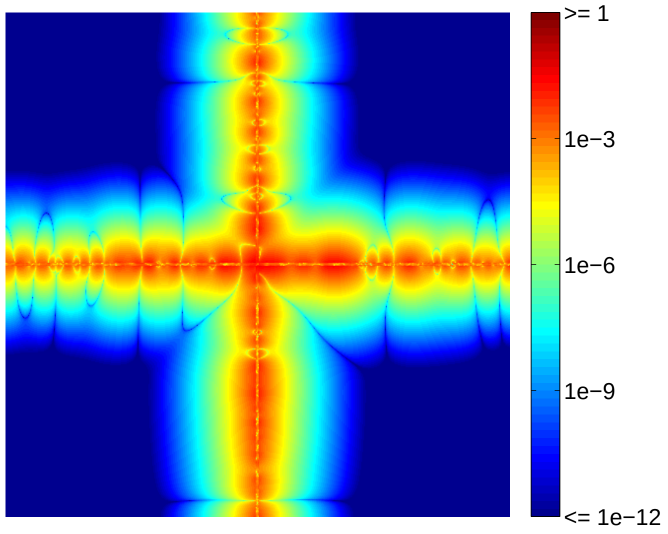


(b) error map $\varepsilon(i, j)$ after the last iteration

Figure 5.11: Results for the decomposed image denoising problem (ID), with the same parameters as in Fig. 5.10, but with $a = 0.5$: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations.
The error increases with the number of iterations, because the step size was chosen to high.

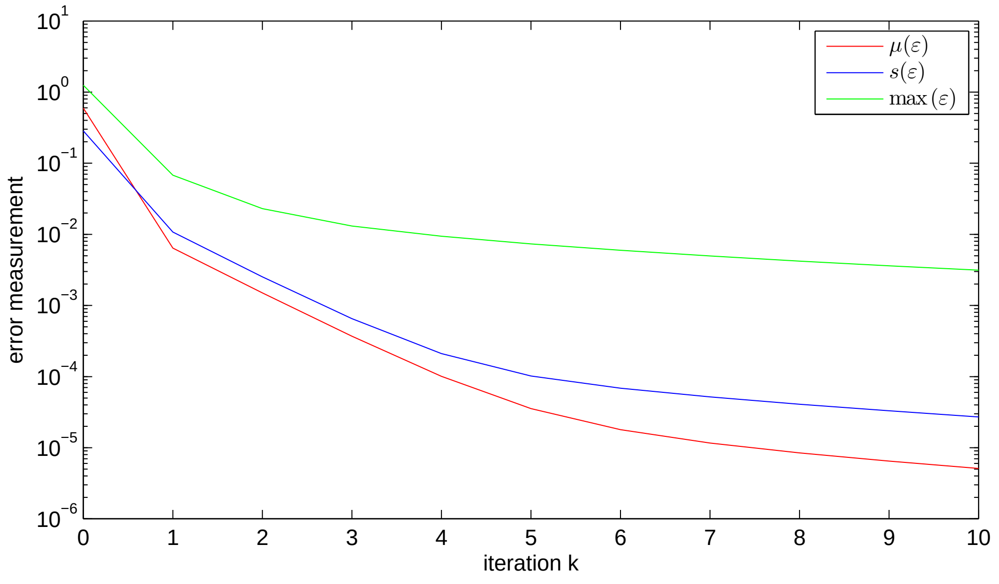


(a) error history

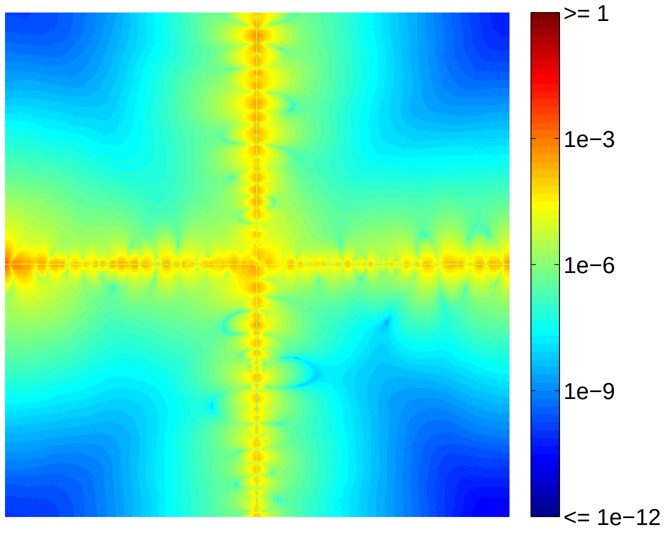


(b) error map $\varepsilon(i,j)$ after the last iteration

Figure 5.12: Results for the decomposed image denoising problem (ID), with the same parameters as in Fig. 5.10, but with $a = 8$: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations.
The approach converges, but only slowly reduces the error, because the step size was chosen too small.

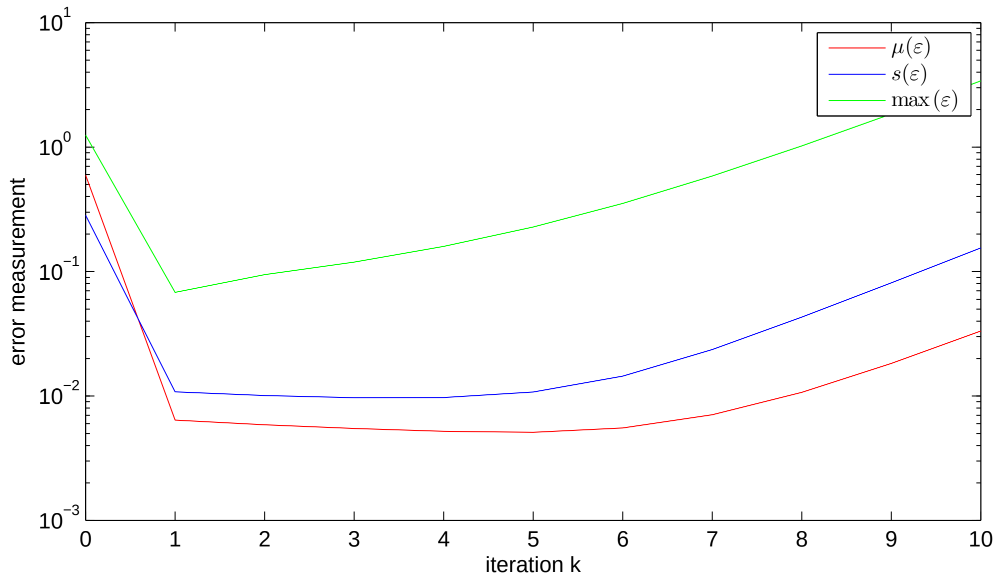


(a) error history

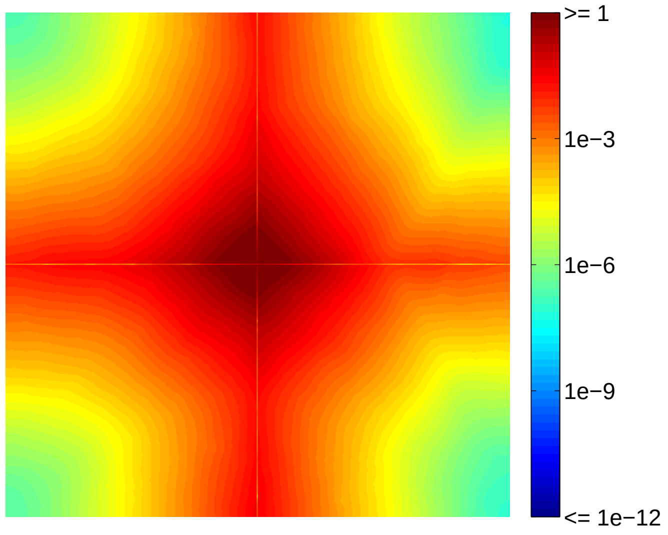


(b) error map $\varepsilon(i,j)$ after the last iteration

Figure 5.13: Results for the optical flow estimation problem with first-order regularisation terms (HS), decomposed into $d = 4$ subproblems, with step size parameters $a = 50$ and $b = 0$, *star*-shaped consistency constraints and subproblem accuracy of 10^{-12}: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations. The approach finds a very accurate solution within few iterations. The remaining errors are spread wider around the artificial boundaries than in the image denoising experiments.

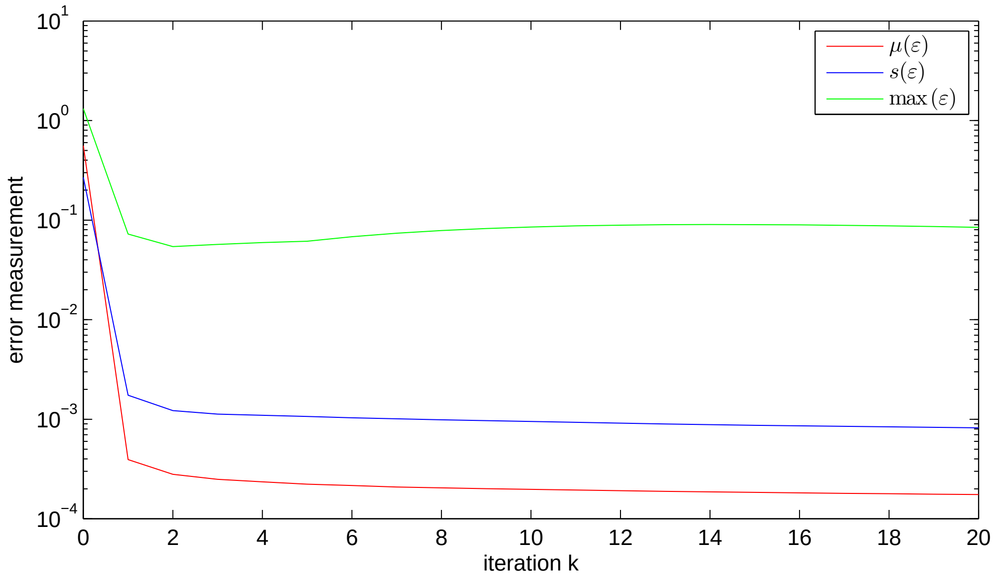


(a) error history

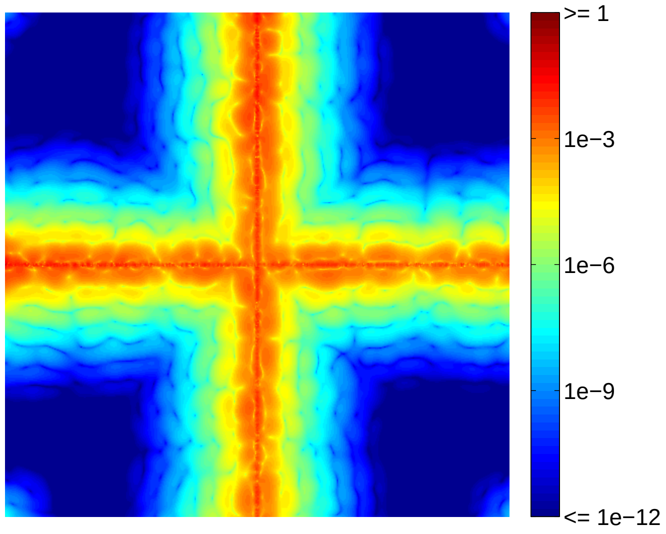


(b) error map $\varepsilon(i,j)$ after the last iteration

Figure 5.14: Results for the decomposed optical flow estimation problem with first-order regularisation terms (HS), with the same parameters as in Fig. 5.13, but with $a = 20$: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations.
After few iterations with marginal improvement, the error diverges, because the step size was chosen too high.

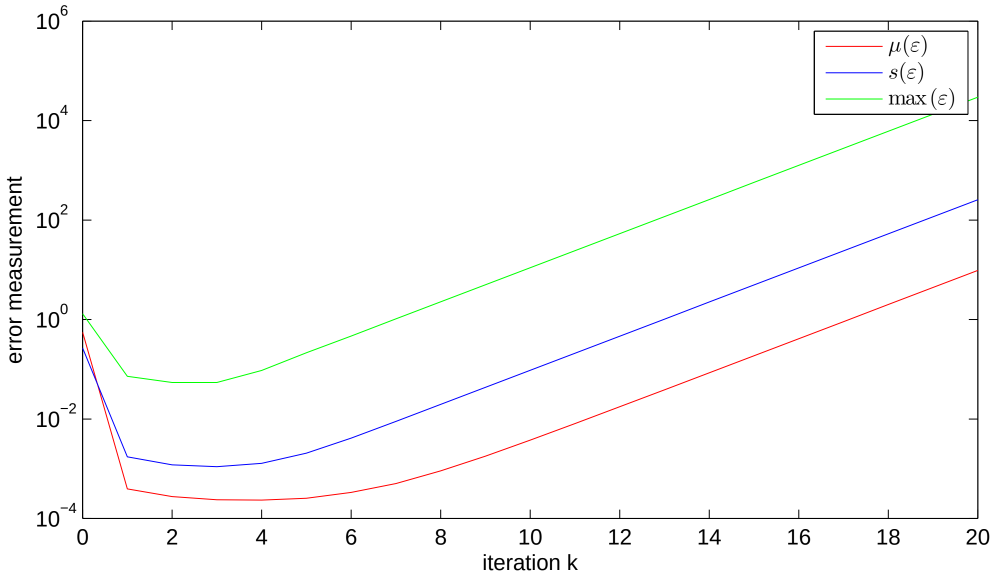


(a) error history

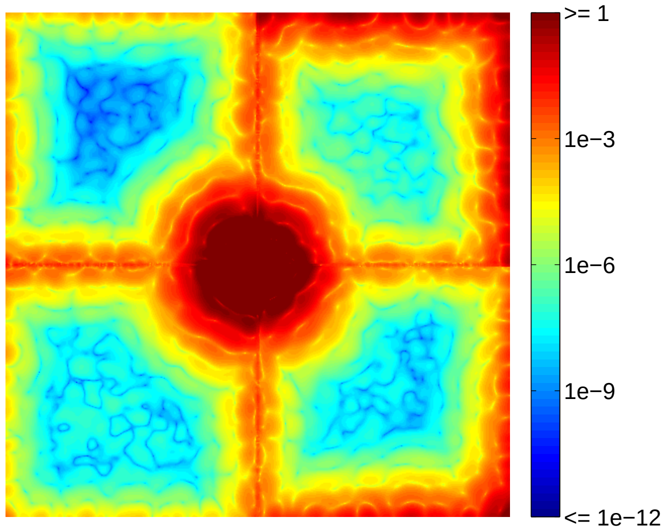


(b) error map $\varepsilon(i,j)$ after the last iteration

Figure 5.15: Results for the optical flow estimation problem with higher-order regularisation terms (DC), decomposed into $d = 4$ subproblems, with step size parameters $a = 600$ and $b = 100$, *star*-shaped consistency constraints and subproblem accuracy of 10^{-10}: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 20 iterations. The approach finds an accurate solution after few iterations, but only slowly reduces the mean error in the following iterations. The errors are located around the boundaries between the subproblems.



(a) error history



(b) error map $\varepsilon(i, j)$ after the last iteration

Figure 5.16: Results for the decomposed optical flow estimation problem with first-order regularisation terms (DC), with the same parameters as in Fig. 5.15, but with $b = 0$: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 20 iterations.
After few iterations, the approach diverges with exponential rate, because the step size was chosen too large.

Table 5.4: Error measurements and extrapolated timings for a parallel implementation for the decomposed image denoising approach (ID): Comparison of different number of subproblems d.

d	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$	t/s
1	5.5	0	0	0	88.4
4	5.10	$41 \cdot 10^{-6}$	$269 \cdot 10^{-6}$	$11.4 \cdot 10^{-3}$	139.3
9	—	$109 \cdot 10^{-6}$	$448 \cdot 10^{-6}$	$11.7 \cdot 10^{-3}$	65.5
16	—	$137 \cdot 10^{-6}$	$520 \cdot 10^{-6}$	$15.1 \cdot 10^{-3}$	33.8
25	—	$177 \cdot 10^{-6}$	$545 \cdot 10^{-6}$	$16.1 \cdot 10^{-3}$	28.1
36	5.17	$225 \cdot 10^{-6}$	$605 \cdot 10^{-6}$	$11.7 \cdot 10^{-3}$	18.1

Table 5.5: Error measurements and extrapolated timings for a parallel implementation for the decomposed optical flow approach with first-order regularisation (HS): Comparison of different number of subproblems d.

d	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$	t/s
1	5.8	0	0	0	151.9
4	5.13	$5.1 \cdot 10^{-6}$	$27.1 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$	399.2
9	—	$10.9 \cdot 10^{-6}$	$31.9 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$	111.5
16	—	$16.5 \cdot 10^{-6}$	$41.6 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$	60.5
25	—	$22.6 \cdot 10^{-6}$	$44.7 \cdot 10^{-6}$	$1.0 \cdot 10^{-3}$	35.3
36	5.18	$29.6 \cdot 10^{-6}$	$58.5 \cdot 10^{-6}$	$4.1 \cdot 10^{-3}$	24.9

5.4.3 Number of Subproblems

We investigated the scalability of the approach with respect to the number of subproblems d. The problem domain was split into a regular grid of 2×2 up to 6×6 subdivisions. The size of the tiles was chosen the same up to one pixel difference caused by rounding to integer coordinates.

The error measurements for the image denoising problem are listed in Table 5.4. Exemplary results for 36 subproblems are illustrated in Fig. 5.17. Table 5.5 presents the error measurements for the optical flow estimation problem with first-order regularisation. Figure 5.18 illustrates the error location map and the evolution over time for 36 subproblems. For the same problem but with higher-order regularisation terms, the results can be found in Table 5.6 and Fig. 5.19, respectively.

In all three cases, the mean error increases sub-linearly with the number of subproblems. Thus, from the viewpoint of accuracy, the method scales well with d, which, in combination with parallel hardware, allows to solve very large problems.

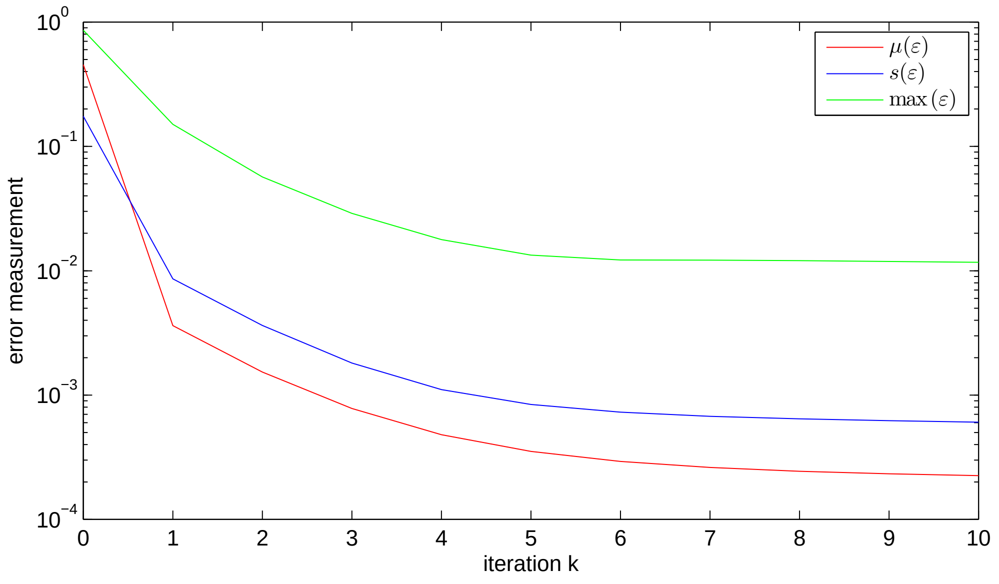
The computational effort for decomposing the problem and solving it sequentially on a single node was higher than for the original problem with a single domain. However, we give estimates of the expected time complexity for solving the problem on parallel hardware: We assume that the decomposition of the objective function can be split up as

Table 5.6: Error measurements and extrapolated timings for a parallel implementation for the decomposed optical flow approach with higher-order regularisation (DC): Comparison of different number of subproblems d.

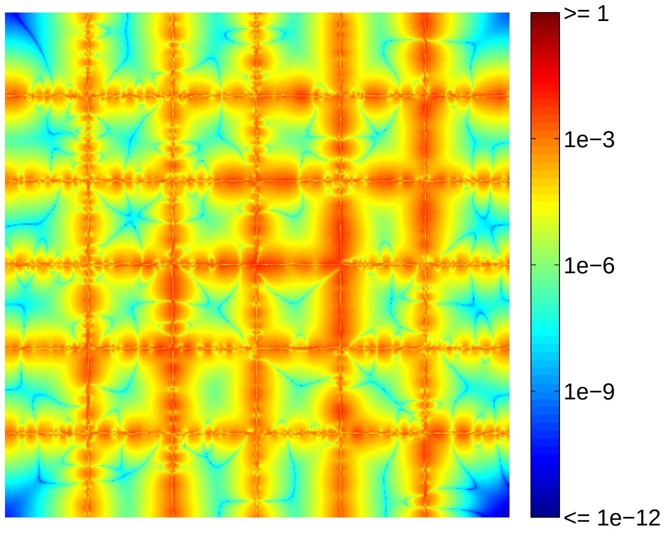
d	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$	t/s
1	5.9	0	0	0	932.9
4	5.15	$176 \cdot 10^{-6}$	$822 \cdot 10^{-6}$	$85 \cdot 10^{-3}$	4456.2
9	—	$405 \cdot 10^{-6}$	$1249 \cdot 10^{-6}$	$29 \cdot 10^{-3}$	1807.0
16	—	$577 \cdot 10^{-6}$	$1472 \cdot 10^{-6}$	$85 \cdot 10^{-3}$	956.7
25	—	$794 \cdot 10^{-6}$	$1756 \cdot 10^{-6}$	$61 \cdot 10^{-3}$	795.7
36	5.19	$947 \cdot 10^{-6}$	$1901 \cdot 10^{-6}$	$115 \cdot 10^{-3}$	470.8

indicated in Sect. 5.1.3. Then, for each iteration we picked the subproblem which had the longest runtime for updating the primal variables. We added the time for updating the dual variables, and presumed this to be the time consumption for each iteration. Finally, we summed up the timings of the decomposition, initialisation and all iterations, and list this value along with the error measures in the tables. For the reference solution, we just used their overall time consumption. Note that in this estimation, we disregarded the time for data transfer and synchronisation between the parallel computational nodes. All measurements were performed on an *Intel Core2* CPU running at 2.40GHz. Only one of the CPU cores was used.

Although the timing information is only an estimate and disregards communication overhead, the results promise noticeable increase in performance with the number of subproblems. However, compared to the non-decomposed problem, the speedup is disappointing. This situation could be improved by finding a trade-off between reducing the number of iterations and speeding up the single subproblems on the one hand, and the inevitable, but possibly small, decrease in accuracy on the other.

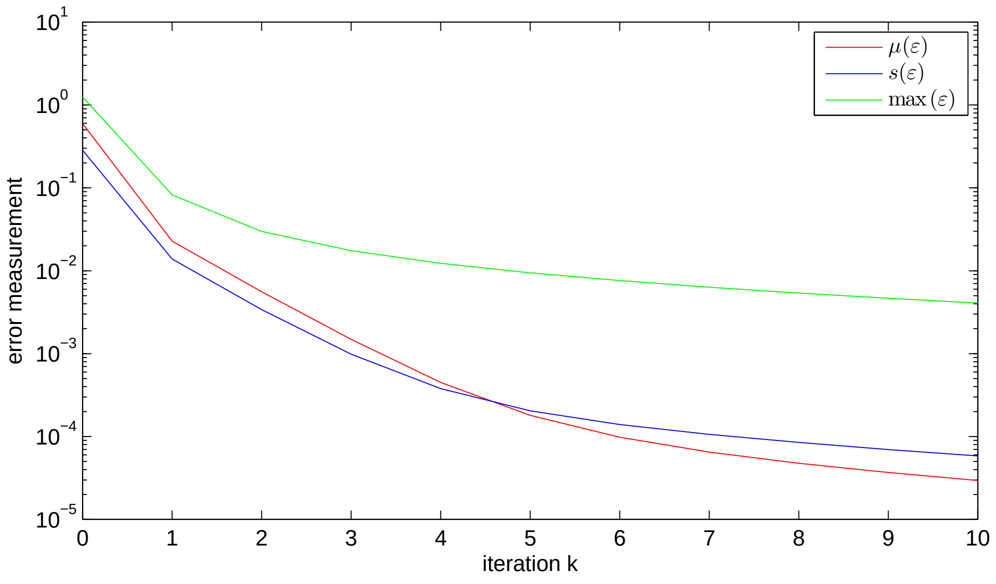


(a) error history

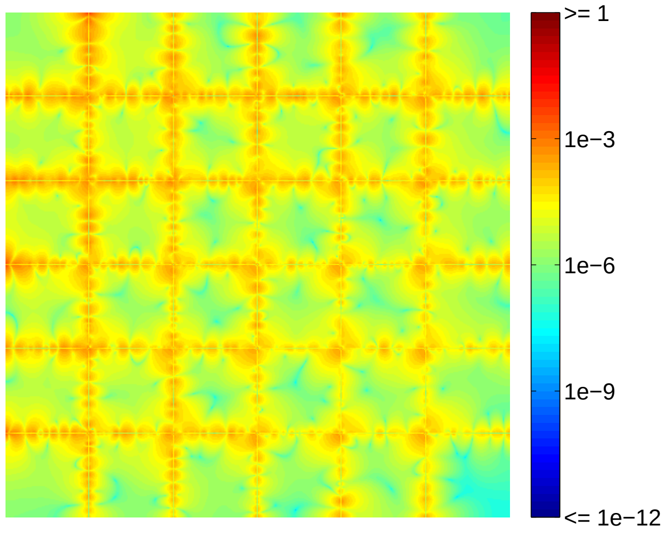


(b) error map $\varepsilon(i,j)$ after the last iteration

Figure 5.17: Results for the decomposed image denoising problem (ID), with the same parameters as in Fig. 5.10, but decomposed into 36 subproblems: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations.
The approach converges quickly to a solution with high accuracy. The remaining errors are located at the artificial boundaries between the subproblems.

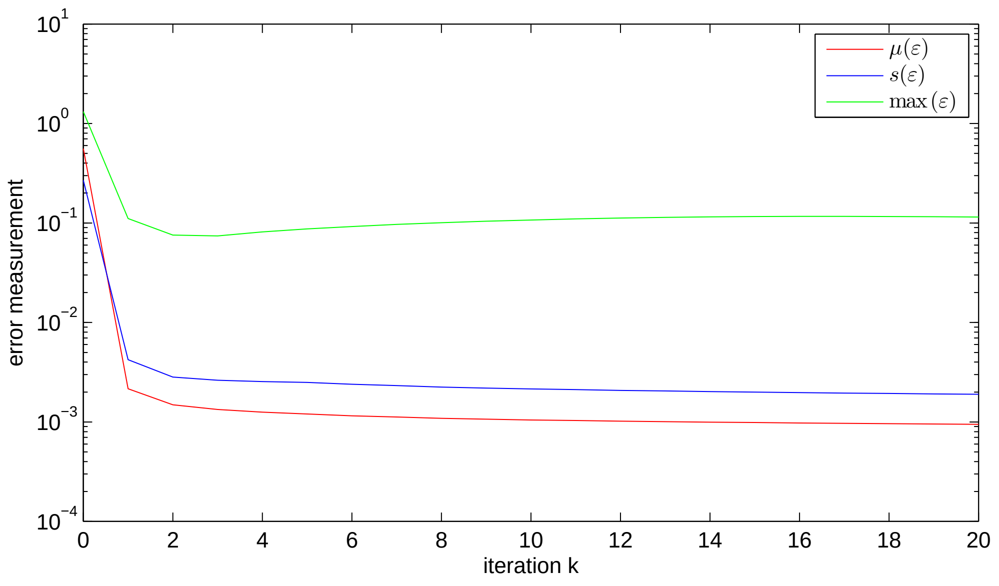


(a) error history

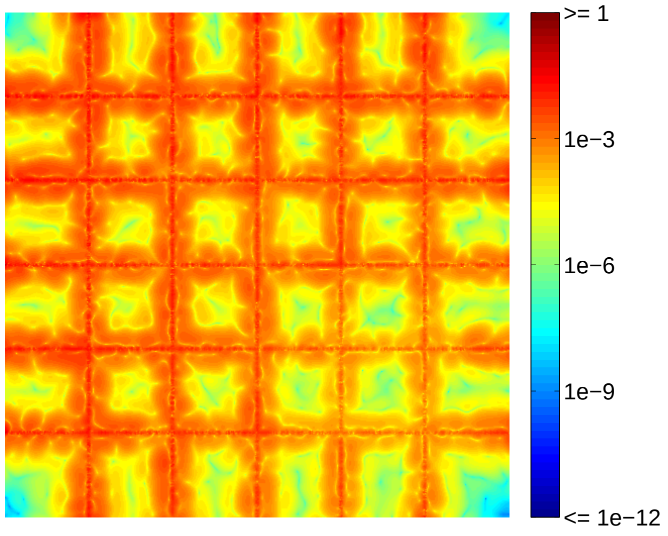


(b) error map $\varepsilon(i,j)$ after the last iteration

Figure 5.18: Results for the optical flow estimation problem with first-order regularisation terms (HS), with the same parameters as in Fig. 5.13, but decomposed into 36 subproblems: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations.
The approach finds a very accurate solution within few iterations. The remaining errors are located around the artificial boundaries between the subproblems.



(a) error history



(b) error map $\varepsilon(i, j)$ after the last iteration

Figure 5.19: Results for the optical flow estimation problem with higher-order regularisation terms (DC), with the same parameters as in Fig. 5.15, but decomposed into 36 subproblems: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 20 iterations.
The approach finds a good solution after few iterations, but only slowly reduces the mean error afterwards, while the maximum error increases. The errors are distributed around the boundaries between the subproblems.

Table 5.7: Error measurements for the decomposed image denoising approach (ID): Comparison of consistency constraint layouts.

constraint layout	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$
star	5.10	$41.4 \cdot 10^{-6}$	$269.0 \cdot 10^{-6}$	$11.4 \cdot 10^{-3}$
chain	—	$41.3 \cdot 10^{-6}$	$267.3 \cdot 10^{-6}$	$11.0 \cdot 10^{-3}$
ring	—	$41.9 \cdot 10^{-6}$	$277.1 \cdot 10^{-6}$	$13.8 \cdot 10^{-3}$
full	—	$41.1 \cdot 10^{-6}$	$264.2 \cdot 10^{-6}$	$10.9 \cdot 10^{-3}$

Table 5.8: Error measurements for the decomposed optical flow approach with first-order regularisation (HS): Comparison of consistency constraint layouts.

constraint scheme	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$
star	5.13	$5.1 \cdot 10^{-6}$	$27.1 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$
chain	—	$5.0 \cdot 10^{-6}$	$26.8 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$
ring	—	$5.0 \cdot 10^{-6}$	$27.2 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$
full	—	$5.6 \cdot 10^{-6}$	$28.2 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$

5.4.4 Consistency Constraint Layout

In Sect. 5.1.3 we defined four exemplary rules for setting up the consistency constraints and conjectured that they might have an influence on the accuracy of the solution.

Table 5.7 and Table 5.8 list the errors for the image denoising problem and the approach by Horn&Schunck, respectively. Due to the high computational costs we omit the experiments for the third case study. However, there are no significant differences between the results for the individual constraint layouts. As *star* and *chain* generate the minimum number of constraints, they seem to be the method of choice.

Table 5.9: Error measurements and extrapolated timings for a parallel implementation for the decomposed image denoising (ID) approach: Comparison of different subproblem accuracies.

accuracy *CGS*	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$	t/s
10^{-12}	5.10	$41.4 \cdot 10^{-6}$	$269 \cdot 10^{-6}$	$11.4 \cdot 10^{-3}$	139.0
10^{-9}	—	$41.4 \cdot 10^{-6}$	$269 \cdot 10^{-6}$	$11.4 \cdot 10^{-3}$	187.8
10^{-6}	—	$41.4 \cdot 10^{-6}$	$269 \cdot 10^{-6}$	$11.4 \cdot 10^{-3}$	130.3
10^{-3}	5.20	$42.4 \cdot 10^{-6}$	$272 \cdot 10^{-6}$	$99.8 \cdot 10^{-3}$	48.7

Table 5.10: Error measurements and extrapolated timings for a parallel implementation for the decomposed optical flow approach with first-order regularisation (HS): Comparison of different subproblem accuracies.

accuracy *CGS*	Figure	$\mu(\varepsilon)$	$\sigma(\varepsilon)$	$\max(\varepsilon)$	t/s
10^{-12}	5.13	$5.1 \cdot 10^{-6}$	$27.1 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$	399.2
10^{-9}	—	$5.1 \cdot 10^{-6}$	$27.1 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$	278.0
10^{-6}	—	$5.1 \cdot 10^{-6}$	$27.1 \cdot 10^{-6}$	$3.1 \cdot 10^{-3}$	167.6
10^{-3}	5.21	$31.0 \cdot 10^{-6}$	$60.6 \cdot 10^{-6}$	$2.9 \cdot 10^{-3}$	74.4

5.4.5 Accuracy of Subproblems

Next we investigated the influence of the accuracy of the subproblems on the performance of the overall method.

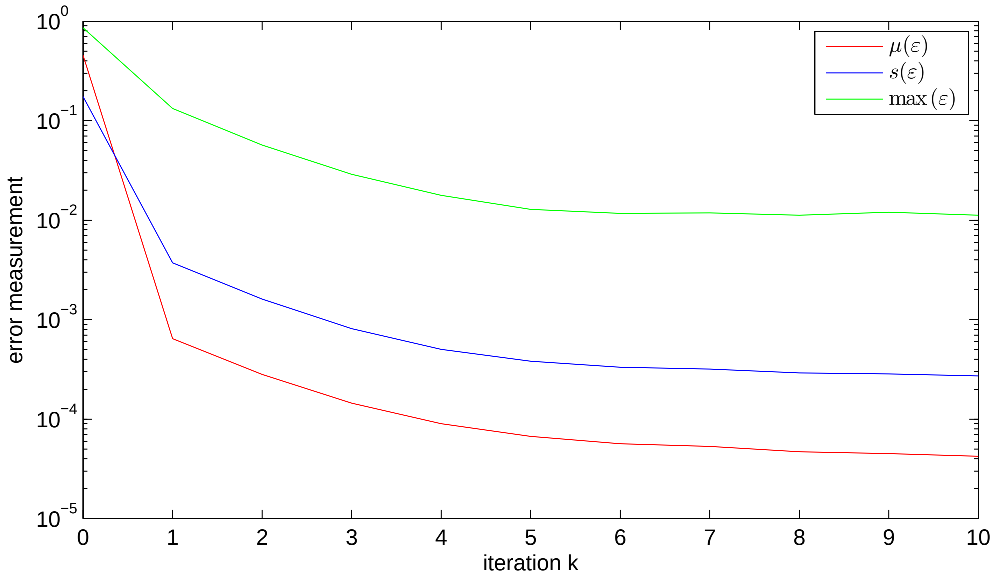
The results for the image denoising and optical flow approach with first-order regularisation are compared in Table 5.9 and Table 5.10, respectively. For the first approach, there is no considerable increase in the overall accuracy. The situation hardly changes for the motion estimation problem. Here, the mean error increases slightly for the lowest subproblem accuracy.

This experiments suggest, that the subproblems can be solved with low accuracy without significantly deteriorating the overall solution. In addition, the computational complexity reduces by a large factor, if we compare the timing measurements (as defined in Sect. 5.4.3) listed with the error measurements in the tables referenced above.

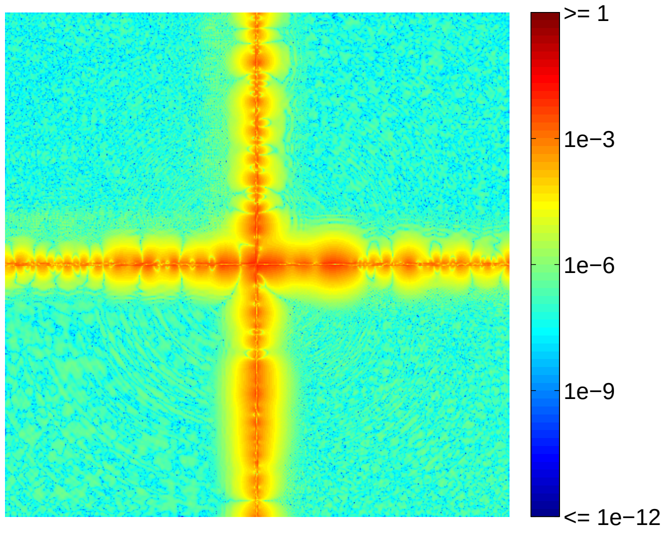
5.4.6 Summary and Conclusion

In this section we investigated the performance of the proposed decomposition method by means of the three case studies defined in Sect. 5.3. The accuracy was measured with respect to solutions of the non-decomposed problems.

The choice of the parameters for the dual variable update seems to be critical for the convergence of the algorithm and the achievable accuracy. However, in all cases, we could present solutions which were very close to the reference. The number of subproblems used to decompose the considered problem reduced the accuracy only slightly. Furthermore, an estimate of the computation time of a (future) parallel implementation was given and proved the potential to speed up the calculation. In further comparisons, we found that the implementation of the consistency constraints, which offers some freedom of choice, does not influence the results for the considered variants. Finally, the experiments showed, that it is possible to solve the linear subproblems only approximately, without deteriorating the overall solution significantly, but reducing computational complexity at the same time.

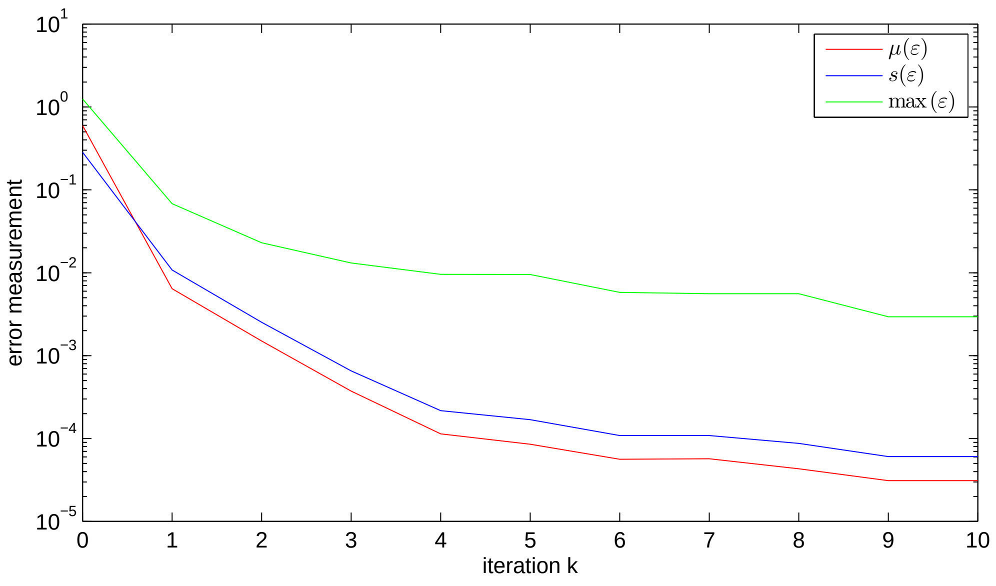


(a) error history

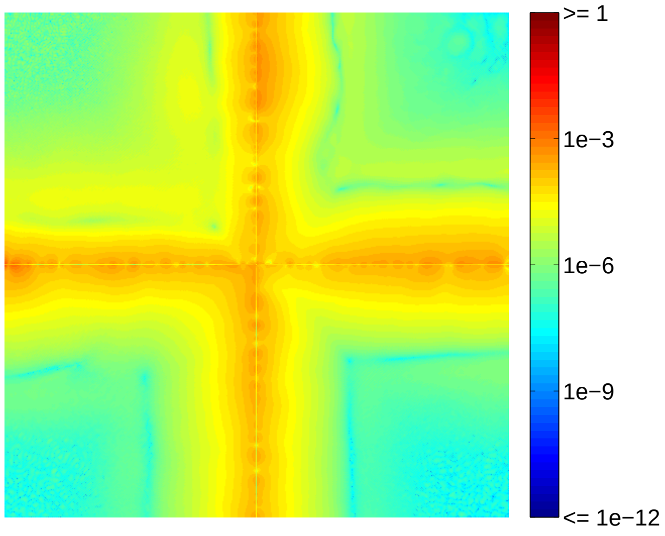


(b) error map $\varepsilon(i,j)$ after the last iteration

Figure 5.20: Results for the decomposed image denoising problem (ID), with the same parameters as in Fig. 5.10, but with subproblem accuracy of 10^{-3}: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations.
The approach finds a very accurate solution within few iterations. The lower accuracy of the subproblems does not significantly influence the mean error, however the maximum error increases by factor nine. The reduced accuracy in the inner regions of the subproblems becomes visible in the error map.



(a) error history



(b) error map $\varepsilon(i, j)$ after the last iteration

Figure 5.21: Results for the decomposed optical flow estimation problem with first-order regularisation terms (HS), with the same parameters as in Fig. 5.13, but with subproblem accuracy 10^{-3}: (a) Error measurements with respect to the reference solution over iterations. (b) Error distribution after 10 iterations. The general accuracy of the subproblems is reduced slightly by factor six, and the errors around the artificial boundaries are distributed wider. The maximum error, however, does not increase.

5.5 Conclusion and Further Work

5.5.1 Summary and Conclusion

In this chapter we applied the Dual Decomposition method to the class of convex unconstrained, quadratic optimisation problems. The aim was to render variational problems of extreme size (e.g. 3D motion estimation problems) manageable by distributing subproblems to parallel hardware. We detailed a procedure to split the problem, which guarantees convex subproblems, and thus allows to use standard methods to solve them. Furthermore, we extended the Dual Decomposition by a framework which allows to improve the numerical properties of the subproblems. Theoretical results on the convergence rate of the modified approach, which includes the standard Dual Decomposition as a special case, were presented.

The method was examined by means of three variational approaches from image processing and it showed its ability to solve the initial problems accurately. The experiments demonstrated that increasing the number of subproblems and reducing the exactness of the partial solutions – and thus their runtime – are adequate instruments to significantly improve computational performance with a marginal decrease in accuracy.

5.5.2 Further Work

From a technical point of view, a framework to distribute the problems to parallel hardware would be favourable in order to actually profit from the presented theoretical basis. Implementations for the efficient exchange of information between hardware nodes already exist, e.g. the *Message Passing Interface* [74].

The terms of the initial problem were assigned to subproblems by using simple geometrical cues, which have to be specified by the user. Although we expect that this works well for problems that are defined by means of local coherence, more general methods can be investigated which decompose the posed problem optimally with respect to the expected convergence rate and accuracy of the solution as well as strict convexity of the subproblems.

Our results on the convergence properties can be used as a starting point for deriving further, possibly problem-specific, conditions for the choice of the step parameter, the stopping criterion and the subproblem regularisation. Also further aspects, such as the influence of the accuracy of subproblem solutions, can be considered. All these approaches could potentially improve the performance – i.e. speed and accuracy – of the overall methods.

The Dual Decomposition method is capable of handling any convex, decomposable optimisation problem. Thus, further steps should consider more general classes of convex optimisation problems, including convex quadratic programs *with* constraints, second-order cone programs [75, 76], semidefinite programs [77, 44] and general non-smooth problems [78].

6 Summary and Conclusion

In the first part our work, Chap. 2–3, we presented a variational approach to motion estimation for image data gained from PIV experiments. We based the displacement estimation on cross-correlation measurement which is known to be robust against distortions typical for this kind of data. However, we used continuous optimisation methods instead of performing an exhaustive search for the optimal displacements vectors as it is done in state-of-the-art methods. Furthermore, we employed a continuously parameterisable Gaussian correlation window and defined a sound error model which estimates its influence on the correlation process. The minimisation of this function directly formulates the aim to increase the accuracy of the displacement measurement. The velocity estimation and window adaption were combined in a single optimisation problem, which was solved using methods for non-linear and non-convex optimisation. We finally demonstrated the ability of our approach to accurately measure motion with real and synthetic PIV data. In a direct comparison to a number of state-of-the-art methods, our approach outperformed most of its competitors.

The second part, covered by Chapter 4–5, is dedicated to the decomposition of convex and unconstrained quadratic optimisation problems and is based on Dual Decomposition. The aim was to make variational problems of extreme size (e.g. for 3D motion estimation problems) manageable by distributing subproblems to parallel hardware. We described in detail a procedure to split up the considered problem. It guarantees convex subproblems, and thus allows to employ standard methods for solving them. Furthermore, we extended the Dual Decomposition by a framework which allows to improve their numerical properties. Theoretical results on the convergence rate of the modified approach, which includes the standard Dual Decomposition as special case, were presented. The method was tested by means of three variational approaches from image processing and showed its ability to solve the initial problems to high accuracy.

Although both presented topics are still loosely connected, a beneficial link may be established in future: An adaptive variational correlation approach for PIV-data – possibly in 3D – which finds partial solutions efficiently by solving distributed, convex problems.

Bibliography

[1] G. H. Golub and C. F. van Loan. *Matrix Computations.* The John Hopkins University Press, third edition edition, 1996.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, March 2004.

[3] R. Bracewell and P. B. Kahn. *The Fourier Transform and its Applications.* McGraw-Hill, 2000.

[4] R. J. Adrian. Particle-Imaging Techniques for Experimental Fluid Mechanics. *Annu. Rev. Fluid Mech*, 23:261–304, 1991.

[5] M. Raffel, C. Willert, and J. Kompenhans. *Particle Image Velocimetry.* Springer-Verlag, Berlin, 2001.

[6] M. Gharib, M. Raffel, O. Ronneberger, and J. Kompenhans. Feasibility Study of Three-Dimensional PIV by Correlating Images of Particles within Parallel Light Sheet Planes. *Experiments in Fluids*, 19:69–77, 1995.

[7] O. Ronneberger, M. Raffel, and J. Kompenhans. Advanced Evaluation Algorithms for Standard and Dual Plane Particle Image Velocimetry. In *Proceedings of the 9th International Symposium on Applications of Laser Techniques to Fluid Mechanics*, 1998.

[8] A.K. Prasad. Stereoscopic Particle Image Velocimetry. *Experiments in Fluids*, 29:103–116, 2000.

[9] G. E. Elsinga, F. Scarano, B. Wieneke, and B. W. van Oudheusden. Tomographic Particle Image Velocimetry. *Experiments in Fluids*, 41:933–947, 2006.

[10] L. Alvarez, C. A. Castaño, M. Garcá, K. Krissian, L. Mazorra, A. Salgado, and J. Sánchez. 3D Motion Estimation Using a Combination of Correlation and Variational Methods for PIV. In *Computer Aided Systems Theory EUROCAST 2007*, volume 4739/2007, pages 612–620, 2007.

[11] H. G. Maas, A. Gruen, and D. Papantoniou. Particle Tracking Velocimetry in Three-Dimensional Flows, Part I. *Experiments in Fluids*, 15(2):133–146, 1993.

[12] N. A. Malik, Th. Draco, and D. A. Papantoniou. Particle Tracking Velocimetry in Three-Dimensional Flows, Part II. *Experiments in Fluids*, 15(4-5):279–294, 1993.

[13] P. Ruhnau, C. Guetter, T. Putze, and C. Schnörr. A Variational Approach for Particle Tracking Velocimetry. In *Measurement, Science and Technology*, volume 16, pages 1449–1458, 2005.

[14] B. Horn and B. Schunck. Determining Optical Flow. *Artificial Intelligence*, 17:185–203, 1981.

[15] P. Ruhnau, T. Kohlberger, H. Nobach, and C. Schnörr. Variational Optical Flow Estimation for Particle Image Velocimetry. *Experiments in Fluids*, 38:21–32, 2005.

[16] P. Ruhnau, A. Stahl, and C. Schnörr. Variational Estimation of Experimental Fluid Flows with Physics-Based Spatio-Temporal Regularization. *Measurement Science and Technology*, 18:755–763, 2007.

[17] J. Yuan, C. Schnörr, and E. Mémin. Discrete Orthogonal Decomposition and Variational Fluid Flow Estimation. *J. Math. Imaging Vis.*, 28(1):67–80, 2007.

[18] 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, 2006.

[19] G. Hermosillo, C. Chefd'Hotel, and O. Faugeras. Variational Methods for Multimodal Image Matching. *International Journal of Computer Vision*, 50(3):329–343, 2002.

[20] D. Heitz, P. Heas, V. Navaza, J. Carlier, and E. Mémin. Spatio-Temporal Correlation-Variational Approach for Robust Optical Flow Estimation. In *Symposium on Particle Image Velocimetry (PIV'07)*, pages 1–9, Roma, Italy, June 2007.

[21] D. Heitz, P. Héas, E. Mémin, V. Carlier, and V. Navaza. Dynamic Consistent Correlation-Variational Approach for Robust Optical Flow Estimation. *Experiments in Fluids*, 4(4):595–608, October 2008.

[22] A. Vlasenko and C. Schnörr. Variational Denoising of Incompressible Fluid Flow Estimates. In *Pattern Recognition – 30th DAGM Symposium*, volume 5096 of *Lecture Notes in Computer Science*, pages 406–415, 2008. unpublished.

[23] J. Nogueira, A. Lecuona, and P.A. Rodriguez. Local Field Correction PIV: On the Increase of Accuracy of Digital PIV Systems. *Experiments in Fluids*, 27:107–116, 1999.

[24] F. Scarano. Iterative Image Deformation Methods in PIV. *Meas. Sci. Technol.*, 13:R1–R19, 2002.

[25] J. Nogueira, A. Lecuona, P.A. Rodriguez, J.A. Alfaro, and A. Acosta. Limits on the Resolution of Correlation PIV Iterative Methods. Practical Implementation and Design of Weighting Functions. *Experiments in Fluids*, 39:314–321, 2005.

[26] R. Theunissen, F. Scarano, and R. L. Riethmuller. An Adaptive Sampling and Windowing Interrogation Method in Piv. *Meas. Sci. Technol.*, 18:265–287, 2007.

[27] R. Theunissen, F. Scarano, and M. L. Riethmuller. On Improvement of PIV image Interrogation near Stationary Interfaces. *Experiments in Fluids*, 45:557–572, 2008.

[28] D. Di Florio, F. Di Felice, and G. P. Romano. Windowing, Re-Shaping and Re-Orientation Interrogation Windows in Particle Image Velocimetry for the Investigation of Shear Flows. *Measurement Science and Technology*, 13:953–962, 2002.

[29] B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of DARPA Imaging Understanding Workshop*, pages 121–130, 1981.

[30] 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.

[31] F. Becker, B. Wieneke, J. Yuan, and C. Schnörr. A Variational Approach to Adaptive Correlation for Motion Estimation in Particle Image Velocimetry. In Gerhard Rigoll, editor, *Pattern Recognition – 30th DAGM Symposium*, volume 5096 of *Lecture Notes in Computer Science*, pages 335–344. Springer Verlag, 2008.

[32] F. Becker, B. Wieneke, J. Yuan, and C. Schnörr. Variational Correlation Approach to Flow Measurement with Window Adaption. In *14th International Symposium on Applications of Laser Techniques to Fluid Mechanics*, page 1.1.3, 2008.

[33] R. D. Keane and R. J. Adrian. Theory of Cross-Correlation Analysis of PIV Images. *Applied Scientific Research*, 49:191–215, 1992.

[34] J. W. Cooley, P. A. Lewis, and P. D. Welch. The Fast Fourier Transform and Its Applications. *IEEE Transactions on Education*, 12(1):27–34, 1969.

[35] C. E. Willert and M. Gharib. Digital Particle Image Velocimetry. *Experiments in Fluids*, 10:181–193, 1991.

[36] S. T. Wereley and C.D. Meinhart. Accuracy Improvements in Particle Image Velocimetry. In *10th International Symposium on Applications of Laser*, 2000.

[37] S. T. Wereley and C. D. Meinhart. Second-Order Accurate Particle Image Velocimetry. *Experiments in Fluids*, 31:256–268, 2001.

[38] J. Westerweel. On Velocity Gradients in PIV Interrogation. *Experiments in Fluids*, 44:831–842, 2008.

[39] K. B. Petersen and M. S. Pedersen. The Matrix Cookbook. `http://matrixcookbook.com`, November 2008. Version: November 14, 2008.

[40] H. J. Reinhardt. Die Methoden der Finiten Elemente. `http://www.uni-siegen.de/fb6/numerik/skripts/skripte/femscript.zip?lang=de`, 1995.

[41] R. Theunissen, F. Scarano, and M. L. Riethmuller. Statistical Adaptivity in PIV Interrogation for Mean Flow Estimation. In *14th Int Symp on Applications of Laser Techniques to Fluid Mechanics*, 2008.

[42] 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.

[43] 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.

[44] H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors. *Handbook of Semidefinite Programming: Theory, Algorithms, and Applications.* Kluwer Academic Publishers, Boston, MA, 2000.

[45] E. P. Simoncelli. *Distributed Representation and Analysis of Visual Motion.* PhD thesis, Massachusetts Institute of Technology, January 1993.

[46] R. Krishnamurthy, P. Moulin, and J. W. Woods. Multiscale Motion Estimation for Scalable Video Coding. In *Proceedings of International Conference on Image Processing*, 1996.

[47] R. A. Haddad and A. N. Akansu. A Class of Fast Gaussian Binomial Filters for Speech and Imageprocessing. *IEEE Transactions on Signal Processing*, 39(3):723–727, March 1991.

[48] M. Hashimoto and J. Sklansky. Multiple-Order Derivatives for Detecting Local Image Characteristics. *Comp. Vision, Graphics, and Image Proc.*, 39:28–55, 1987.

[49] M. Stanislas, K. Okamoto, C. J. Kähler, J. Westerweel, and F. Scarano. Main Results of the Third International PIV Challenge. *Experiments in Fluids*, 45:27–71, July 2008.

[50] PIV-Challenge: homepage. `http://www.pivchallenge.org/`.

[51] FLUID project: homepage. `http://fluid.irisa.fr/`.

[52] J. Carlier and B. Wieneke. Deliverable 1.2: Report On Production and Diffusion of Fluid Mechanic Images and Data. Activity Report, European Project FLUID Deliverable 1.2, Cemagref, LaVision, Nov. 30 2005.

[53] P. Anadan. A Computational Framework and an Algorithm For The Measurement of Visual Motion. *International Journal of Computer Vision*, 2(3):283–310, January 1989.

[54] A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations.* Oxford Univ. Press, 1999.

[55] T. Kohlberger, C. Schnörr, A. Bruhn, and J. Weickert. Domain Decomposition for Variational Optical Flow Computation. *IEEE Transactions on Image Processing*, 14(8):1125–1137, 2005.

[56] A. J. Conejo, E. Castillo, R. Minguez, and R. García-Bertrand. *Decomposition Techniques in Mathematical Programming.* Springer, 2006.

[57] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, 1995.

[58] A. J. Conejo, F. J. Nogales, and F. J. Prieto. A Decomposition Procedure Based on Approximate Newton Directions. *Mathematical Programming*, 93(3):495–515, 2002.

[59] M. Minoux. *Mathematical Programming - Theory and Algorithms.* Wiley & Sons, 1986.

[60] S. Boyd, L. Xiao, and A. Mutapcic. Notes on Decomposition Methods. Technical report, Stanford University, 10 2003.

[61] Y. Censor and S. A. Zenios. *Parallel Optimization – Theory, Algorithms, and Applications.* Oxford Univ. Press, 1997.

[62] J. Frédéric Bonnans, Jean Charles Gilbert, Claude Lemaréchal, and Claudia A. Sagastizábal. *Numerical Optimization: Theoretical and Practical Aspects.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[63] F. Becker and C. Schnörr. Decomposition of Quadratric Variational Problems. In Gerhard Rigoll, editor, *Pattern Recognition – 30th DAGM Symposium*, volume 5096 of *Lecture Notes in Computer Science*, pages 325–334. Springer Verlag, 2008.

[64] S. Boyd, L. Xiao, A. Mutapcic, and J. Mattingley. Notes on Decomposition Methods. Technical report, Stanford University, 2007.

[65] O. E. Flippo and A. H. G. Rinnooy Kan. Decomposition in General Mathematical Programming. *Mathematical Programming*, 60:361–382, 1993.

[66] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* The MIT Press, 2000.

[67] R. T. Rockafellar. Augmented Lagrangians and Applications of the Proximal Point Algorithm in Convex Programming. *Mathematics of Operations Research*, 1(2):97–116, 1976.

[68] H. C. Elman and G. H. Golub. Inexact and Preconditioned Uzawa Algorithms for Saddle Point Problems. *Siam Journal on Numerical Analysis*, 31(6):1645–1661, December 1994.

[69] J. H. Bramble, J. E. Pasciak, and A. T. Vassilev. Analysis of the Inexact Uzawa Algorithm for Saddle Point Problems. *SIAM J. Numer. Anal.*, 34(3):1072–1092, 1997.

[70] K.J. Arrow, L. Hurwicz, and H. Uzawa. *Studies in Linear and Non-Linear Programming*. Stanford mathematical studies in the social sciences. Stanford University Press, Stanford, Calif., 1958.

[71] G. Demoment. Image Reconstruction and Restoration: Overview of Common Estimation Structures and Problems. *IEEE Transactions on Acoutstivs, Speech, and Signal Processing*, 37(12):2024–2036, December 1989.

[72] J. M. Hyman and M. Shashkov. Adjoint Operators for the Natural Discretizations of the Divergence, Gradient and Curl on Logically Rectangular Grids. *Applied Numerical Mathematics: Transactions of IMACS*, 25(4):413–442, 1997.

[73] J. M. Hyman and M. Shashkov. Natural Discretizations for the Divergence, Gradient, and Curl on Logically Rectangular Grids. *Comput. Math. Appl.*, 33(4):81–104, 1997.

[74] W Gropp, E. Lusk, and A. Skjellum. *Using MPI: Portable Parallel Programming with the Message Passing Interface*. Massachusetts Institute of Technology, 1999.

[75] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of Second-Order Cone Programming. In *Linear Algebra and its Applications*, volume 284, pages 193–228, 1998.

[76] D. Goldfarb and F. Alizadeh. Second-Order Cone Programming. *Mathematical Programming*, 95(1):3–51, January 2003.

[77] L. Vandenberghe and S. Boyd. Semidefinite Programming. *Siam Review*, 38(1):49–95, March 1996.

[78] Y. Nesterov. Smooth Minimization of Non-Smooth Functions . *Mathematical Programming*, 103(1):127 –152, 05 2005.