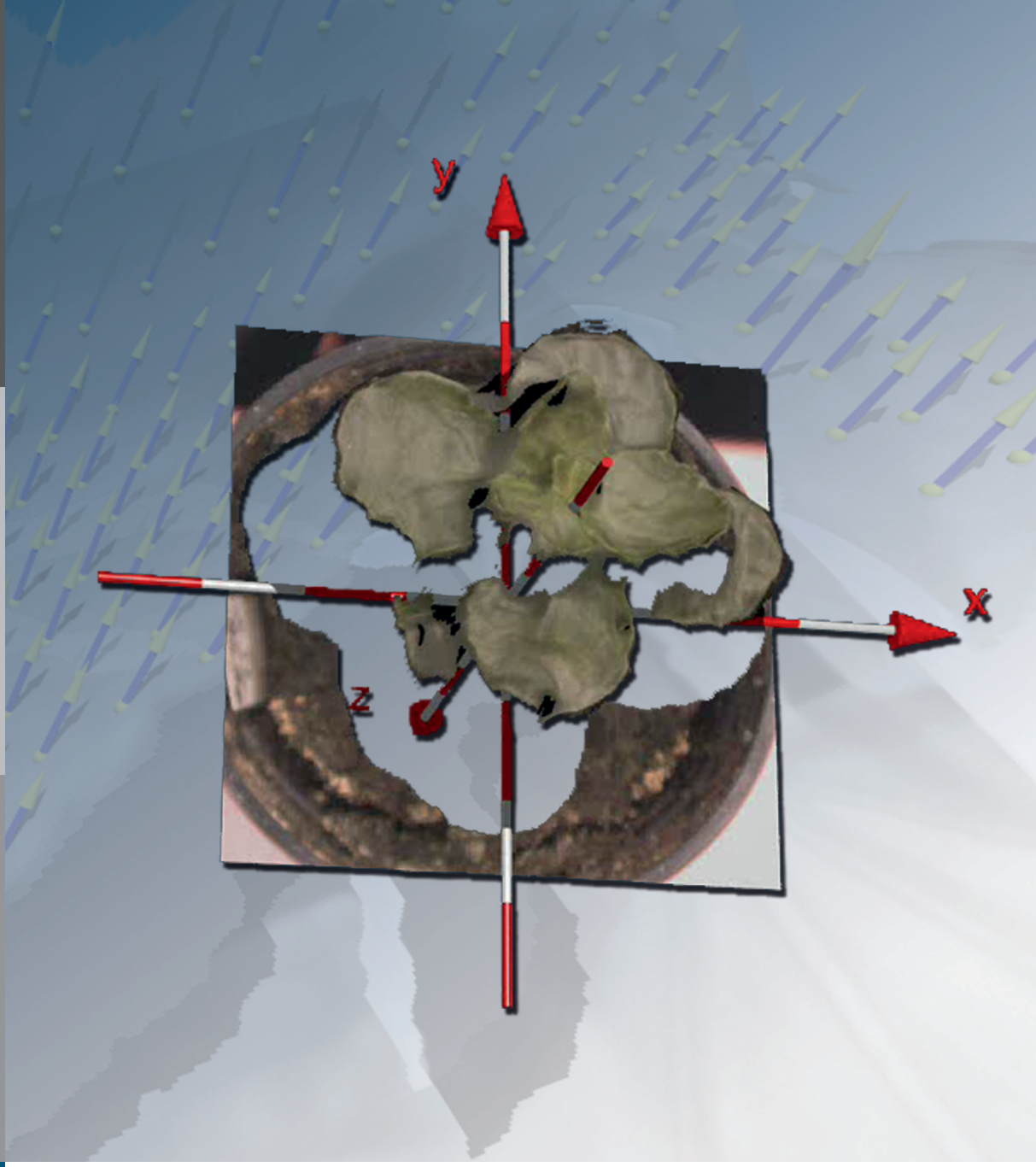


Plant Leaf Motion Estimation Using A 5D Affine Optical Flow Model

Tobias Schuchert

Mitglied der Helmholtz-Gemeinschaft

JÜLICH
FORSCHUNGSZENTRUM

Schriften des Forschungszentrums Jülich
Reihe Energie & Umwelt / Energy & Environment Band / Volume 57

Forschungszentrum Jülich GmbH
Institute of Chemistry and Dynamics of the Geosphere (ICG)
Phytosphere (ICG-3)

Plant Leaf Motion Estimation Using A 5D Affine Optical Flow Model

Tobias Schuchert

Schriften des Forschungszentrums Jülich
Reihe Energie & Umwelt / Energy & Environment Band / Volume 57

ISSN 1866-1793 ISBN 978-3-89336-613-2

Bibliographic information published by the Deutsche Nationalbibliothek.
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Publisher and Distributor:	Forschungszentrum Jülich GmbH Zentralbibliothek, Verlag D-52425 Jülich phone:+49 2461 61-5368 · fax:+49 2461 61-6103 e-mail: zb-publikation@fz-juelich.de Internet: http://www.fz-juelich.de/zb
Cover Design:	Grafische Medien, Forschungszentrum Jülich GmbH
Printer:	Grafische Medien, Forschungszentrum Jülich GmbH
Copyright:	Forschungszentrum Jülich 2010

Schriften des Forschungszentrums Jülich
Reihe Energie & Umwelt / Energy & Environment Band / Volume 57

D 82 (Diss., RWTH Aachen, Univ., 2010)

ISSN 1866-1793
ISBN: 978-3-89336-613-2

The complete volume is freely available on the Internet on the Jülicher Open Access Server (JUWEL) at http://www.fz-juelich.de/zb/juwel

Abstract

High accuracy motion analysis of plant leafs is of great interest for plant physiology, e.g., estimation of plant leaf orientation, or temporal and spatial growth maps, which are determined by divergence of 3D leaf motion. In this work a new method for plant leaf motion estimation is presented. The model is based on 5D affine optical flow, which allows simultaneous estimation of 3D structure, normals and 3D motion of objects using multi camera data.

The method consists of several consecutive estimation procedures. In a first step the affine transformation in a 5D data set, i.e., 3D image sequences (x,y,t) of a 2D camera grid (s_x,s_y) is estimated within a differential framework. In this work the differential framework, based on an optical flow model, is extended by explicitly modeling of illumination changes.

A second estimation process yields 3D structure and 3D motion parameters from the affine optical flow parameters. Modeling the 3D scene with local surface patches allows to derive a matrix defining the projection of 3D structure and 3D motion onto each camera sensor. The inverse projection matrix is used to estimate 3D structure (depth and surface normals) and 3D motion, including translation, rotation and acceleration from up to 24 affine optical flow parameters.

In order to stabilize the estimation process optical flow parameters are estimated additionally separated for all cameras. A least squares estimator yields the solution minimizing the difference between optical flow parameters and the back projection of the 3D scene motion onto all cameras.

Experiments on synthetic data demonstrate improved accuracy and improved robustness against illumination changes compared to methods proposed in recent literature. Moreover the new method allows estimation of additional parameters like surface normals, rotation and acceleration. Finally, plant data acquired under typical laboratory conditions is analyzed, showing the applicability of the method for plant physiology.

Kurzfassung

Eine detaillierte Bewegungsanalyse von Pflanzenblättern ist für die Pflanzenphysiologie von großem Interesse, z.B. die Bestimmung der Blattwinkelstellung oder zeitlich und räumlich hochaufgelöster Blattwachstumskarten, welche sich aus der 3-D-Bewegung eines Blattes berechnen lassen. Im Rahmen dieser Arbeit wurde eine neue Methode zur Analyse von Pflanzenblattbewegungen entwickelt. Die Methode basiert auf dem 5-D-affinen optischen Fluss und ermöglicht die simultane Bestimmung von 3-D-Struktur, Oberflächennormalen und 3-D-Bewegung eines Objektes aus Multi-Kamerasequenzen.
Die Methode basiert auf mehreren, hintereinander ausgeführten Schätzungen. Zunächst wird die affine Transformation einer Umgebung innerhalb eines 5-D-Datensatzes, d.h. 3-D-Bildsequenzen (x,y,t) eines 2-D-Kamera-Arrays (s_x,s_y), mit einem differentiellen Ansatz nach dem Prinzip des optischen Flusses bestimmt. Das in dieser Arbeit vorgestellte erweiterte Modell des optischen Flusses modelliert auftretende Helligkeitsänderungen explizit und erhöht somit die Robustheit gegenüber Beleuchtungsänderungen.
Nach Bestimmung der 5-D-optischen Flussparameter werden die 3-D-Struktur und die 3-D-Bewegung, basierend auf einem sogenannten *Surface Patch* Model, geschätzt. Die Matrix, die die Projektion der 3-D-Struktur und der 3-D-Bewegung des Surface Patches auf den jeweiligen Kamerasensor beschreibt, kann mit Hilfe projektiver Geometrie bestimmt werden. Die Inverse dieser Projektionsmatrix ermöglicht dann die Ermittlung von 3-D-Struktur (Tiefe und Oberflächennormalen) und 3-D-Bewegung (Translation, Beschleunigung und Rotation) aus den bis zu 24 Parametern des affinen optischen Flusses.
Zur Stabilisierung der Schätzung werden Parameter des optischen Flusses zusätzlich separat in allen Kameras geschätzt. Ein Least-Squares Schätzer liefert dann die Lösung, welche die Differenz zwischen den einzelnen Parametern des optischen Flusses und der Rückprojektion der 3-D-Bewegung in die einzelnen Kameras minimiert.
Experimente mit synthetischen Daten belegen die höhere Genauigkeit und die höhere Robustheit gegenüber Beleuchtungsänderungen im Vergleich zu bekannten Verfahren aus der Literatur. Zudem ist die explizite Bestimmung zusätzlicher Parameter wie Oberflächennormalen, Beschleunigung und Rotation möglich. Die erfolgreiche Auswertung von unter normalen Laborbedingungen erhobenen Pflanzensequenzen belegt die Anwendbarkeit der neuen Methode in der Pflanzenphysiologie.

Contents

Chapter 1

Introduction

This thesis presents a new approach for estimation of 3D structure, surface slopes and 3D motion of objects from multi-camera image sequences. In the last years more and more applications using 3D reconstruction and/or motion estimation of image sequences are developed. Miniaturization and increasing performance of image processing setups including cameras and signal processing hardware as well as falling costs, allow fast and cheap measurements from large to small scale applications. Another important fact is that measurements are non-invasive. Image processing applications can be found almost everywhere in daily life, starting from motion estimation in consumer hardware such as cameras or entertainment products as *Sony's EyeToy* [66], to driver assistance systems, surveillance systems, satellite data products, so-called web mapping service applications, like *Google Street View* [22], and so on. Motion estimation in image sequences is a prerequisite of many image processing algorithms, such as depth reconstruction, 3D motion estimation, tracking, segmentation and coding, to name only few of them. 2D motion in image sequences is sometimes called optical flow [26]. Several estimation frameworks have been proposed for optical flow estimation. We may distinguish between differential, often called optical flow based techniques and point/feature correspondence or correlation based techniques. In this work we focus on differential frameworks, which yield high accuracy and are developed since the early 80's [28, 42]. There is rich literature on optical flow estimation techniques (see e.g., [4, 26, 10, 14, 49]) and many extensions have been developed. An overview of recent optical flow estimation algorithms is presented in [3]. Optical flow can be used for depth reconstruction and 3D motion estimation, depending on the images on which the optical flow is determined:

- in case of images of different cameras displaced parallel to their sensor planes the optical flow is inversely proportional to the depth of the 3D point projected onto the image sensor element,

- for consecutive images from one fixed camera the motion seen on the sensor is the projection of the 3D motion in the scene, the so called *Scene Flow*.

Considerable work has already been carried out on estimating 3D motion fields. *Range Flow* estimation [80, 21] uses solely data from range sensors, whereas Spies et al. [67, 69] incorporate information from both range and image sensors. Reconstruction of *Scene Flow* and 3D structure from optical flow observed in several cameras has been proposed by [81, 76, 40, 57, 31].

1.1 Motivation and Approach

In this work we focus on plant leaf motion estimation. The Institute of Chemistry and Dynamics of the Geosphere: Phytosphere (ICG-3) at the Forschungszentrum Jülich investigates plant leaf motion and growth in order to examine the influence of environmental change, biotic/abiotic stresses, and light quality and quantity on growth dynamics. For the estimation of plant leaf growth a system based on a single camera was developed by Schmundt and Schurr [59]. There the camera images a leaf from the top and the divergence of the calculated velocity field yields the sought growth rate. Typical growth rates of fast growing leaves are below 3% per hour or 0.1%/frame when a typical frame rate of 0.5 frames per minute is assumed [78]. Therefore a suitable estimation technique has to be highly accurate. However, any change in the distance between the camera and the plant leaf causes a diverging optical flow field. This motion cannot be distinguished from growth of the leaf. Therefore the leaf is fixed in order to restrict motion to the horizontal plane. The procedure how to fix the leaf in order not to influence overall leaf growth was thoroughly tested [63]. Nevertheless, fixing leafs influences plants and is not appropriate for screening, i.e., high throughput measurements. In order to estimate plant growth of freely moving plant leaves, 3D structure and motion has to be estimated in high accuracy.

In this work we investigate three 3D motion estimation techniques, namely *Scene Flow* [74], *Range Flow* [67] and a technique based on an affine motion model [57] (labeled as the *Affine Model*). All these techniques are based on motion models, which do not allow reliable estimation of plant motion. We extend these models and propose a novel estimation technique incorporating advantages of all three approaches. The novel affine optical flow-like differential model allows estimation of 3D structure and 3D motion using image sequences of a camera grid. The model is valid for instantaneously moving cameras observing moving surfaces. This is unlike previous work (e.g., [41, 37, 1, 79]) where either an observed surface moves, or a camera, but not both. The model introduced here combines motion estimation in the sequence in camera displacement direction, i.e., disparity, and in time direction, i.e., optical flow. It is an extension of a model first introduced by Scharr [55] and further refined and extended in [56, 60]. It allows simultaneous local estimation of 3D motion, 3D position and surface normals in a spatio-temporal affine optical flow-like model.

3D motion estimation is used for many different applications, i.e., driver assistance, machine vision or tracking, to name only few of them. Obviously applications have

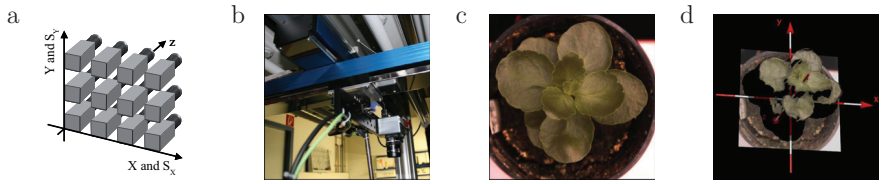


Figure 1.1: Setup for our target application plant growth analysis. A camera grid (a) is simulated using a single camera on a moving stage (b). From images (c) taken from vertical view a 3D reconstruction (d) is calculated.

different demands on the estimated 3D motion field, e.g., real-time processing and reliability are basic prerequisites for a driver assistance system, whereas accuracy in the range of *cm* may be sufficient.

Although we do not aim at presenting a best *method* yet, we do have the target application plant leaf growth analysis in mind when designing our experiments. Accuracy in the range of μm is a basic requisite. Plant leaf motion is high compared to plant leaf growth. In order to estimate growth from the divergence of plant leaf motion these estimates have to be highly accurate and reliable. Moreover many plant species have reflective leaf surfaces, i.e., directed illumination may cause specular highlights. Further plants react on visible light. Nevertheless real-time processing is not necessary, i.e., motion and growth parameter estimation can be done off-line. In order to develop an appropriate *method* all these requirements have to be taken into account. For imaging we use a single camera on an x-y-moving stage instead of a real camera grid. This allows us to use very small baselines, much smaller than camera dimensions. Obviously images are not taken at exactly the same point in time, which could be accounted for in the discretization of derivative kernels applied to the images. We neglect this effect here, because plants move slowly enough such that time intervals between two images at the same camera position are much larger than the overall acquisition time for all camera images representing 'one' point in time. Figure 1.1 shows the setup for our target application, and a rendered image of a plant obtained with the presented model.

1.2 Thesis Outline

In this work we derive a novel model for 3D depth, slopes and 3D motion estimation. In Chap. 2 we demonstrate the basic elements of motion estimation techniques, i.e., model derivation, discretization and parameter estimation. We discuss motion models suitable for optical flow estimation and optimal filter sets for data discretization. Several estimators are proposed and evaluated on synthetic sequences.

Chapter 3 reviews the 3D motion estimation techniques *Scene Flow* and *Range Flow*. Furthermore the *Affine Model* is introduced, which is an extension of the

multi-camera model presented by [55]. *Range Flow* and the *Affine Model* have already been proposed for plant leaf emotion estimation, whereas *Scene Flow* showed high performance in human motion estimation. We evaluate the performance of the three models on synthetic data and on a plant leaf sequence.

None of the models yields the accuracy needed for reliable plant leaf motion estimation. The main reason are illumination changes in the image sequences due to motion of the leaf or the light source. Chapter 4 therefore shows an investigation of different approaches to handle varying illumination. Combinations of prefilters and brightness change constraints are thoroughly tested and a novel brightness change constraint, which explicitly models brightness changes is derived.

In Chap. 5 we introduce a new motion model for the *Affine Model* in order to estimate rotational motion. Furthermore we extend the model by estimating additional affine optical flow parameters. These parameters are used to make subsequent parameter estimation of 3D structure and 3D motion more robust. This extension boils down to be a combination of the *Affine Model* and *Range Flow*.

Chapter 6 demonstrates how to integrate *Scene Flow* in the *Affine Model*. The novel *Affine Scene Flow Model* combines advantages of all three 3D motion estimation models and yields highest accuracies on synthetic sequences as well as on plant leaf sequences.

Finally a concluding summary and an outlook on possible future work is given in Chap. 7.

Chapter 2

Parameter Estimation of Dynamic Processes

Parameter estimation from noisy data is a basic requisite in many image and signal processing applications, like line fitting, camera calibration, image matching, signal reconstruction or position detection. In this work we focus on motion estimation on image sensors. The motion seen on a 2D image sensor is called optical flow and describes the change of a value, e.g., brightness or color, in consecutive images. This optical flow may be used as input for subsequent processing steps including motion detection, motion compensation, motion-based data compression, 3D scene reconstruction, 3D motion estimation or segmentation,to name only a few of them. In some publications [74, 40] optical flow is referred to as the projection of the 3D motion. In fact it is only an approximation of the projected motion. Motion on the image plane is identified by changes of a certain value, e.g., the intensity for gray value images. Therefore optical flow in an intensity image sequence can only be estimated in textured areas or by varying illumination. This leads to problems for interpretation of the projected 3D motion. Horn [27] denotes several problems when estimating optical flow, e.g., the motion of a rigid sphere. Consider a rigid sphere without texture and homogeneous surface reflectance rotating around an axis through its origin. With a constant illumination source, this 3D motion does not generate optical flow, because no brightness changes are visible. If the light source itself is moving optical flow occurs also when the illuminated object does not move. Other problems for motion estimation are e.g., occlusion, transparent and multiple motions, aperture problems and correspondence problems.

Figure 2.1 shows two frames of the well known synthetic *Yosemite* sequence [4], which is very popular as a benchmark for motion estimation techniques. The sequence was originally generated by Lynn Quam at SRI and shows a flight through the Yosemite valley. The optical flow for frame 7 is depicted in Fig. 2.1c. Several techniques have been proposed for motion estimation between consecutive frames. Haussecker and Spies [26] distinguished two groups, namely *optical flow*-based and *correspondence*-based techniques. The definition of *optical flow*-based is somewhat

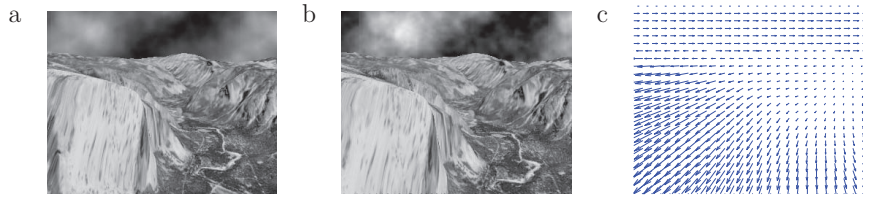


Figure 2.1: a and b: Frames 5 and 9 of the *Yosemite* sequence. c: Ground truth flow vectors for frame 7.

misleading, as *optical flow* is in general defined as the motion in the image, which is estimated by both techniques. Haussecker and Spies define techniques as *optical flow*-based techniques, when the relationship between the temporal variations of image intensities or the frequency distribution in the Fourier domain is analyzed, e.g., differential, energy-based, tensor-based and phase-based techniques. These techniques imply compliance with the temporal sampling theorem.
In contrast to this, correspondence techniques estimate best matches of features between two frames. These techniques can be classified into correlation methods and feature tracking techniques. Whereas correlation techniques find the best match based on a so called correlation window, the feature tracking techniques extract features like borders or edges and track them over time.
Haussecker and Spies [26] present an extensive study of these two classes of 2D motion estimation techniques. They propose that the tensor based techniques perform best with respect to systematic error and noise sensitivity and should be used for subpixel displacement estimation. Correspondence based techniques are less sensitive to illumination changes and are capable of estimating long range displacements, where standard differential techniques fail. Illumination models for differential techniques [25, 60] and warping techniques (e.g., [13]), which address these disadvantages, have been presented recently.
Overviews of optical flow-based approaches are given by Barron et al. [4] and Haussecker and Spies [26]. Baker et al. [3] established a set of benchmarks and evaluation methods for optical flow algorithms. This database is freely available on the web together with results of the most recent optical flow-based techniques at `http://vision.middlebury.edu/flow/`. A taxonomy of dense two frame stereo correspondence based techniques was presented by Scharstein and Szeliski [58]. Benchmarks and results of these and more recent techniques are also available on the web at `http://vision.middlebury.edu/stereo/`.
A complete motion estimation technique is based on an appropriate combination of a data model, a discretization and an estimator. The result of a motion estimation technique depends on all three parts, e.g., combining a highly accurate data model and an estimator based on statistics not appropriate for the data, leads to reduced accuracy of estimation results, limited by the estimator.
Optical flow estimation techniques are commonly classified into *local* and *global*

methods. *Local* estimators use data exclusively from a local neighborhood and minimize an energy function based on a local data-model, i.e., the so-called *data term*[1]. Methods based on these models may be implemented very efficiently and are often more robust under noise compared to *global* methods [19, 14]. *Global* estimators incorporate a prior term on the solution that couples parameters not only in a local neighborhood but in the whole data set [39]. In most cases this prior term is a smoothness assumption of the solution, therefore often called *smoothness term*. The prior term allows to estimate parameters also in regions where the estimation problem is ill-posed [5], i.e., when the solution a) is not unique, b) does not exist or c) does not depend continuously on the data. However, the prior term may include any prior knowledge on the solution parameters.
Global and combined local/global methods are widely used in recent work and showed high performance and accuracy [3]. A drawback of these methods is the prior term influencing all parameters, making detailed error analysis difficult. In contrast to this several error measurements for local methods, e.g., the structure tensor method, have been studied intensively [35].
We briefly review differential-based estimation of optical flow. We start by deriving flow models and the discretization of the image space. The second part of this chapter addresses different types of *local* estimation techniques.

2.1 Model Derivation

Parameter estimation is a part of statistics and includes appropriate mathematical modeling of data structures and estimation of constant parameters of this model. A common assumption on optical flow is that the image brightness $I(\mathbf{x}, t)$ at a point $\mathbf{x} = [x, y]^{\mathrm{T}}$ at time t should only change because of motion $\mathbf{u} = [u_x, u_y]^{\mathrm{T}}$. We get

$$I(\mathbf{x}, t) = I(\mathbf{x} + \mathbf{u}\delta t, t + \delta t) \tag{2.1}$$

with optical flow $\mathbf{u}$. Assuming that δt is small we approximate $I(\mathbf{x} + \mathbf{u}\delta t, t + \delta t)$ by its Taylor expansion. Ignoring higher order terms we obtain a gradient based formulation for the total change of intensity $I(\mathbf{x}, t)$ in time

$$\begin{aligned} & \mathrm{d}I = \left(I(\mathbf{x}, t) + \tfrac{\partial I}{\partial x}\mathrm{d}x + \tfrac{\partial I}{\partial y}\mathrm{d}y + \tfrac{\partial I}{\partial t}\mathrm{d}t \right) - I(\mathbf{x}, t) \\ \Leftrightarrow \quad & \mathrm{d}I = \tfrac{\partial I}{\partial x}\mathrm{d}x + \tfrac{\partial I}{\partial y}\mathrm{d}y + \tfrac{\partial I}{\partial t}\mathrm{d}t \\ \Leftrightarrow \quad & \tfrac{\mathrm{d}I}{\mathrm{d}t} = \tfrac{\partial I}{\partial x}u_x + \tfrac{\partial I}{\partial y}u_y + \tfrac{\partial I}{\partial t} \end{aligned} \tag{2.2}$$

with the optical flow components denoted by $u_x = \mathrm{d}x\,/\mathrm{d}t$ and $u_y = \mathrm{d}y\,/\mathrm{d}t$. Equation (2.2) describes brightness changes in the data due to the optical flow and is therefore called the brightness change constraint equation (BCCE). A special case of the BCCE

[1] In literature also named likelihood term [51] or bones [36].

is the brightness constancy constraint equation (in literature sometimes also called BCCE), where intensities are assumed constant with time, i.e., $\mathrm{d}I\,/\mathrm{d}t = 0$. This results in

$$I_x u_x + I_y u_y + I_t = 0 \tag{2.3}$$

with partial image derivatives $I_q = \frac{\partial I}{\partial q}$ with $q \in x, y, t$. To solve for optical flow $\mathbf{u}$ the flow field is modeled for neighboring image points in Sec. 2.1.1. Additional equations then allow to solve for $\mathbf{u}$.

2.1.1 Modeling of Flow Fields

In order to solve under-determined constraints like the BCCE (2.2), constraints may be grouped together over a local neighborhood Λ. The resulting equation system allows estimation of a constant solution vector (here: optical flow vector $\mathbf{u}$). The neighborhood Λ should be chosen as small as possible to ensure a local estimate of optical flow $\mathbf{u}$. The larger the neighborhood is, the more likely it is that the optical flow varies in Λ, or that multiple flows occur. On the other hand, Λ should be large enough to contain enough information to constrain the solution and be robust to noise. In this section we parameterize the flow vector $\mathbf{u}$ assuming a single motion. The most common models are

- Local constancy of $\mathbf{u}(\mathbf{x})$,
- Local smoothness of $\mathbf{u}(\mathbf{x})$, i.e., the so-called *affine* optical flow and
- Higher order models of $\mathbf{u}(\mathbf{x})$.

Local constancy of $\mathbf{u}(\mathbf{x})$ assumes that the flow in a certain neighborhood Λ of $\mathbf{x}$ is constant. Assuming brightness constancy (cmp. (2.3)), optical flow $\mathbf{u}$ is then determined by solving the overdetermined equation system

$$\begin{pmatrix} I_{x,1} & I_{y,1} & I_{t,1} \\ I_{x,2} & I_{y,2} & I_{t,2} \\ \vdots & \vdots & \vdots \\ I_{x,N} & I_{y,N} & I_{t,N} \end{pmatrix} \begin{pmatrix} u_x \\ u_y \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2.4}$$

for neighborhood Λ with size N.
The assumption of locally constant motion is not fulfilled for most motion vector fields or holds only for small sizes of neighborhood Λ. A more sophisticated motion vector field model is the assumption of local smoothness. Local smoothness and local constancy are sometimes used synonymously. But in a stricter sense local smoothness means that within a local neighborhood Λ the optical flow field is allowed to vary smoothly. This may be achieved e.g., by means of a Taylor series of $\tilde{\mathbf{u}}(\mathbf{x})$ in local coordinates $\Delta\mathbf{x} = (x - x_0, y - y_0)^{\mathrm{T}} =: (\Delta x, \Delta y)^{\mathrm{T}}$ around the center (x_0, y_0) of Λ. This is typically called an *affine* model and estimation boils down to estimating a

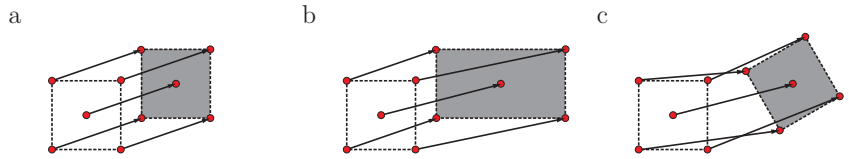


Figure 2.2: Motion vector field transformations. a: Constant, b: stretching and c: rotation.

larger parameter vector also containing smoothness information, which is constant in a neighborhood. The displacement vector field $\mathbf{v}$ is modeled locally by

$$\mathbf{v} = \mathbf{u} + \mathbf{A}\Delta\mathbf{x} \tag{2.5}$$

with a 2×2-matrix $\mathbf{A}$. (x_0, y_0) is the point for which a displacement vector is to be calculated, usually the center of the neighborhood Λ. Reformulating the brightness constancy constraint (2.3) to $\nabla_{\mathbf{x}}^{\mathrm{T}} I\mathbf{u} + I_t = 0$ with spatial gradient $\nabla_{\mathbf{x}} = (\partial_x, \partial_y)^{\mathrm{T}}$ and replacing $\mathbf{u}$ by $\mathbf{v}$ yields

$$\begin{aligned} & 0 &=& \nabla_{\mathbf{x}}^{\mathrm{T}} I(\mathbf{u} + \mathbf{A}\Delta\mathbf{x}) + I_t \\ \Leftrightarrow\quad & 0 &=& \nabla_{\mathbf{x}}^{\mathrm{T}} I\mathbf{u} + \nabla_{\mathbf{x}}^{\mathrm{T}} I\mathbf{A}\Delta\mathbf{x} + I_t \\ \Leftrightarrow\quad & 0 &=& I_x u_x + I_y u_y + I_x \Delta x a_{11} + I_x \Delta y a_{12} \\ & & & \quad + I_y \Delta x a_{21} + I_y \Delta y a_{22} + I_t \,. \end{aligned} \tag{2.6}$$

Flow vector $\mathbf{u}$ represents the translational motion of the center (x_0, y_0) of Λ. The matrix $\mathbf{A}$ addresses smooth changes of the velocity field in the neighborhood, based on elementary geometric transformations, namely rotation, dilation, shear and stretching. Two examples of vector field transformations are compared to a constant vector field in Fig. 2.2.

An affine optical flow model describes variations of $\mathbf{u}$ by matrix $\mathbf{A}$ and linear local coordinates $\Delta\mathbf{x}$. Obviously motion models based on quadratic local coordinates $\Delta\mathbf{x}$ may be used as well. Recently, Li and Sclaroff [40] fit flow vectors to an eight-parameter model corresponding to quadratic motion (cmp. [11]). The optical flow model is given by

$$\begin{aligned} & 0 &=& \nabla_{\mathbf{x}}^{\mathrm{T}} I(\mathbf{u} + \mathbf{A}_{quad}\Delta\mathbf{x}_{quad}) + I_t \\ \Leftrightarrow\quad & 0 &=& I_x u_x + I_y u_y + I_x \Delta x a_{11} + I_x \Delta y a_{12} \\ & & & \quad + I_y \Delta x a_{21} + I_y \Delta y a_{22} \\ & & & \quad + I_x \Delta x \Delta y a_{13} + I_x \Delta x^2 a_{14} \\ & & & \quad + I_y \Delta x \Delta y a_{23} + I_y \Delta y^2 a_{24} + I_t \end{aligned} \tag{2.7}$$

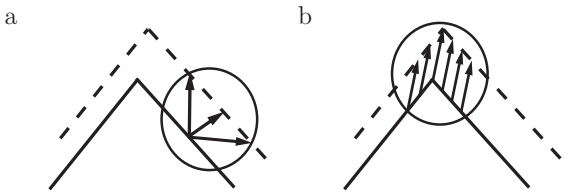


Figure 2.3: a: Aperture problem, flow not uniquely defined and b: full flow at a feature point.

with

$$\mathbf{A}_{quad} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & 0 \\ a_{21} & a_{22} & a_{23} & 0 & a_{24} \end{bmatrix} \tag{2.8}$$

and $\mathbf{x}_{quad} = [\Delta x, \Delta y, \Delta x \Delta y, \Delta x^2, \Delta y^2]^{\mathrm{T}}$. Quadratic models have also been used to improve change detection algorithms [29]. There the gray-value distribution within a region was modeled by a so-called quadratic picture function.

2.1.2 Model Limitations

Using a neighborhood assumption to solve equation (2.2) does not always yield the correct flow as shown in Fig. 2.3. The so called aperture problem occurs if there is no change of the spatial gradient $\nabla_{\mathbf{x}} I$ in the neighborhood. In this case the minimum norm solution is the *normal flow*, pointing parallel to the direction of the spatial gradient. To avoid the aperture problem, either global smoothness assumptions may be applied or, for local methods, the size of the neighborhood has to be enlarged. However, this may extend the region over motion boundaries, thus that constraints are no longer consistent. In addition enlarging kernels blurs results. In order to overcome this problem robust statistics [8] may be used. A sophisticated estimator is presented in Sec. 2.3.
Another limitation of differential models is that differentiation in the BCCE (2.2) assumes small motions. Therefore compliance with the temporal sampling theorem is a basic prerequisite. So called multi-grid methods ([72, 8, 49, 65]) help to achieve also larger motions reliably. These methods calculate motion estimates on image pyramids, but so far are only available for 2 frame algorithms. Motion between the coarsest images is significantly smaller than for the original images and the sampling theorem is fulfilled. The motion estimates from the coarse images are used to warp the image on the next resolution. The remaining motion vector field typically also fulfills the sampling theorem. In subsequent steps only an incremental motion needs to be estimated. These steps are repeated till the original image resolution is reached.

2.2 Discretization

Solving the BCCE (2.2) requires multi-dimensional, i.e., spatial and temporal, image gradients. In a first step the image is discretized in order to be processed on a digital computer. In a next step discrete filters are applied to calculate the sought image gradients ∇I with $\nabla = (\partial_x, \partial_x, \partial_t)^{\mathrm{T}}$. It is desired to minimize errors in both steps. In general, discretization of an image sequence is already performed by the camera. Errors coming from spatial and temporal discretization depend on camera properties, like resolution of the image sensor, lens and frame-rate. Discretization error should not be confused with quantization error, which also appears by digitizing an image in the camera or a frame-grabber and is a rounding in the value range.
In order to minimize the discretization error caused by filtering, optimized filters have been proposed in [64, 53, 54]. Multi-dimensional image derivatives are usually computed via differences between neighboring pixel with backward $[-1, 0]$ or central $[-0.5, 0, 0.5]$ differences. These traditional derivatives are not well suited for gradient direction estimation [64]. In order to minimize the systematic error and to increase the accuracy the authors in [64] and [53] propose the use of matched pairs of derivative filters and lowpass filters. Simoncelli [64] presented optimized first order derivatives and the corresponding smoothness filters. Scharr [53, 54] proposed a general optimization framework for nonlinear composition of arbitrary filters. The filter sets presented in [54] are optimized for optical flow estimation. It was shown that these optimized filters get more accurate, the larger they get. However, larger filters increase smoothing. Symmetric filters are denoted as $h(r) = [h_R, ..., h_1, h_0, h_1, ..., h_R]$, i.e., with transfer function

$$\hat{h}(\tilde{k}) = h_0 + 2\sum_{r=1}^{R} h_r \cos(\pi r \tilde{k}) \tag{2.9}$$

and antisymmetric filters as $h(r) = [h_R, ..., h_1, h_0, -h_1, ..., -h_R]$ with $h_0 = 0$, i.e., with transfer function

$$\hat{h}(\tilde{k}) = 2i\sum_{r=1}^{R} h_r \sin(\pi r \tilde{k}) \; . \tag{2.10}$$

In order to demonstrate the influence of optimized filter sets on optical flow estimation we compare the average angular error (see App. A) and its standard deviation obtained for the *Yosemite* sequence (cmp. Fig. 2.1) using the structure tensor estimator (cmp. Sec. 2.3.2) and different filter sets. A selection of first order filter sets proposed in [54], central differences ($3 \times 1 \times 1$) and the corresponding average angular errors are shown in Tab. 2.1. The $5 \times 5 \times 5$ filter sets, which reduce the angular error up to a factor of 4 compared to central differences, seem to yield a good trade-off between accuracy and length. Therefore in this work the $5 \times 5 \times 5$ filter sets proposed in [54] are used in all experiments, if not stated otherwise.

Table 2.1: Filter coefficients for different filter sets from [54] and the average angular error (AAE) and its standard deviation of optical flow estimates for the *Yosemite* sequence obtained when using the structure tensor TLS-estimator.

derivative, h_0 to h_R	smoothing, h_0 to h_R	AAE	size
$[0, 0.5]$	$[1]$	16.8 ± 25.2	$3 \times 1 \times 1$
$[0, 0.5]$	$[0.6326, 0.1837]$	10.8 ± 24.7	$3 \times 3 \times 3$
$[0, 0.3327, 0.0836]$	$[0.4704, 0.2415, 0.0233]$	4.0 ± 11.1	$5 \times 5 \times 5$
$[0, 0.2232, 0.1190, 0.0130]$	$[0.3845, 0.2461, 0.0583, 0.0031]$	3.1 ± 7.6	$7 \times 7 \times 7$

2.3 Parameter Estimation

A general linear parameter estimation problem is of the form $\mathbf{d}^{\mathrm{T}}\mathbf{p} = 0$, where the vector $\mathbf{d}$ contains data depending terms and the parameter vector $\mathbf{p}$ needs to be estimated. Rearranging the brightness constancy constraint (2.3) yields

$$I_x u_x + I_y u_y + I_t = \nabla_{\mathbf{x}}^{\mathrm{T}} I \mathbf{u} + I_t = \mathbf{d}^{\mathrm{T}}\mathbf{p} = 0 \tag{2.11}$$

with spatial gradient $\nabla_{\mathbf{x}} I = [I_x, I_y]^{\mathrm{T}}$, data vector $\mathbf{d} = [I_x, I_y, I_t]^{\mathrm{T}}$ and parameter vector $\mathbf{p} = [\mathbf{u}^{\mathrm{T}}, 1]^{\mathrm{T}} = [u_x, u_y, 1]^{\mathrm{T}}$. The parameter vector $\mathbf{p}$ has more than one degree of freedom, thus $\mathbf{d}^{\mathrm{T}}\mathbf{p} = 0$ is an under-determined system of equations. Consequently more information has to be provided if a unique solution $\tilde{\mathbf{p}}$ shall be given. Modeling the parameter field as shown in Sec. 2.1.1 yields additional constraints for the parameter vector $\mathbf{p}$.
In general one then solves a system of equations

$$\mathbf{d}_i^{\mathrm{T}}\mathbf{p} = 0 \quad \forall i \in 1, \ldots, N \tag{2.12}$$

for a *single* vector $\mathbf{p}$. Usually the number of equations N is greater than the degrees of freedom in $\mathbf{p}$. Then the system of equations is not solved exactly, but in a least squares sense or minimizing a robust error norm.

2.3.1 Least Squares

In many models $\mathbf{p}$ is a homogeneous vector, with one component of it being a constant, which is usually set to 1. Let this component be the last one. Equation (2.12) can then be rewritten

$$\sum_{j=1}^{J-1} (d_i)_j p_j = -(d_i)_J \quad \forall i \in \{1, \ldots, N\} \tag{2.13}$$

where J is the number of components of $\mathbf{p}$. We rewrite equation (2.13) as

$$\mathbf{A}\,\mathbf{u} = \mathbf{b} \tag{2.14}$$

where we renamed $A_{ij} = (d_i)_j$, $u_j = p_j$ with $\mathbf{p} = (\mathbf{u}^\mathrm{T}, 1)^\mathrm{T}$, and $b_i = -(d_i)_J$ for $j \in \{1, \ldots, J-1\}$ and $i \in \{1, \ldots, N\}$. One can then solve for $\mathbf{u}$ in least squares sense using the Moore-Penrose-pseudo-inverse $A^+ = (\mathbf{A}^\mathrm{T}\mathbf{A})^{-1}\mathbf{A}^\mathrm{T}$

$$\tilde{\mathbf{u}} = A^+\mathbf{b}\,. \tag{2.15}$$

In general $\mathbf{A}^\mathrm{T}\mathbf{A}$ is a symmetric positive semi-definite matrix and therefore diagonalizable. But it is not necessarily invertible as one or more eigenvalues may be (close to) zero, i.e., $\mathbf{A}^\mathrm{T}\mathbf{A}$ may be rank deficient. If this is the case, one may calculate A^+ by diagonalization of $\mathbf{A}^\mathrm{T}\mathbf{A}$ and inversion of all non-zero (or large enough) eigenvalues only, instead of $(\mathbf{A}^\mathrm{T}\mathbf{A})^{-1}$ where all eigenvalues are inverted.

If $\mathbf{A}^\mathrm{T}\mathbf{A}$ is invertible then solution $\tilde{\mathbf{u}}$ is fully determined. If it is not invertible then the eigenvectors corresponding to the (very small or) zero eigenvalues span the null space $\mathrm{Null}(\mathbf{A}^\mathrm{T}\mathbf{A}) = \{\mathbf{u} | \mathbf{A}^\mathrm{T}\mathbf{A}\mathbf{u} = 0\}$. This means that solution $\tilde{\mathbf{u}}$ is only determined up to an additive arbitrary vector $\mathbf{u}_0 \in \mathrm{Null}(\mathbf{A}^\mathrm{T}\mathbf{A})$. Solution $\tilde{\mathbf{u}}$ given in (2.15) is the shortest vector of all possible solutions and therefore is normal to $\mathrm{Null}(\mathbf{A}^\mathrm{T}\mathbf{A})$. The solution is then called *normal solution*. $\mathbf{A}^\mathrm{T}\mathbf{A}$ is rank deficient, if $\mathbf{d}(\mathbf{x}_i)$ (and therefore the intensity signal $I(\mathbf{x})$) does not vary enough in the local neighborhood Λ. As Λ acts as an aperture on $I(\mathbf{x})$ and is, in the rank deficient case, selected too small with respect to changes in $I(\mathbf{x})$, this phenomenon is sometimes called *aperture problem* (cmp. Sec. 2.1.2).

2.3.2 Total Least Squares

As for standard least squares estimation (see Sec. 2.3.1) total least squares parameter estimation solves (2.12) for $\mathbf{p}$. The standard least squares approach assumes that matrix $\mathbf{A}$ contains no errors, i.e., all errors of (2.14) are confined to $\mathbf{b}$. In case of solving (2.11), this assumption is not fulfilled, because both $\mathbf{A}$ and $\mathbf{b}$ are composed of image gradients. In contrast to standard least squares, total least squares addresses erroneous data in $\mathbf{A}$. It is assumed that solution vector $\tilde{\mathbf{p}}$ approximately solves all equations in the local neighborhood Λ and therefore $\mathbf{d}_i^\mathrm{T}\mathbf{p}$ only approximately equals 0. We get

$$\mathbf{d}_i^\mathrm{T}\mathbf{p} = e_i \quad \forall i \in 1, \ldots, N \tag{2.16}$$

with errors e_i which have to be minimized by the sought solution $\tilde{\mathbf{p}}$. We define matrix $\mathbf{D} := [\mathbf{A}, \mathbf{b}]$ (cmp. (2.14)) composed of the vectors $\mathbf{d}_i$ via $\mathbf{D}_{ij} = (\mathbf{d}_i)_j$ for $j \in \{1, \ldots, J\}$ (in contrast to $\mathbf{A}$ in the least squares case in Sec. 2.3.1) and $i \in \{1, \ldots, N\}$. Equation (2.16) becomes $\mathbf{D}\mathbf{p} = \mathbf{e}$.

We minimize $\mathbf{e}$ in a weighted 2-norm

$$\begin{aligned} &||\mathbf{e}||_2 = ||\mathbf{D}\mathbf{p}||_2 = \mathbf{p}^\mathrm{T}\mathbf{D}^\mathrm{T}\mathbf{W}\mathbf{D}\mathbf{p} =: \mathbf{p}^\mathrm{T}\mathbf{J}\mathbf{p} \stackrel{!}{=} \min \\ \Rightarrow \quad &\mathbf{p} = \arg\min_{\mathbf{p}} \mathbf{p}^\mathrm{T}\mathbf{J}\mathbf{p} \end{aligned} \tag{2.17}$$

where $\mathbf{W}$ usually is a diagonal matrix containing positive or zero weights; typically Gaussian weights are used. The matrix $\mathbf{J} = \mathbf{D}^\mathrm{T}\mathbf{W}\mathbf{D}$ is called *structure tensor*. Obviously $||\mathbf{e}||_2$ depends quadratically on the length of $\mathbf{p}$ and $||\mathbf{e}||_2 = 0$ if $\mathbf{p} = \mathbf{0}$. The additional constraint $|\mathbf{p}| = 1$ is introduced to avoid this trivial solution.

The space of solutions $\tilde{\mathbf{p}}$ is spanned by the eigenvector(s) to the smallest eigenvalue(s) of $\mathbf{J}$. We call this space the null-space of $\mathbf{J}$ as the smallest eigenvalue(s) are 0 if the model is fulfilled perfectly, i.e., $\mathbf{e} = \mathbf{0}$, this space is Null($\mathbf{J}$). For most models, as e.g., standard optical flow with or without brightness changes the null-space is 1D [26, 25]. Typically the last entry of $\mathbf{p}$ in these models is 1 and therefore $\tilde{\mathbf{p}}$ has to be divided by its last entry to get the sought model parameters.

For simultaneous estimation of depth and motion, and similar models presented in Sec. 3.3 Null($\mathbf{J}$) is 2D or 3D, depending on the model used. There the sought for solution is the set of orientation vectors, i.e., the eigenvectors spanning Null($\mathbf{J}$). Parameters are disentangled using linear combinations of the found eigenvectors, such that one of the last 2 or 3 entries (of the 2D or 3D case, respectively) is 1 and the others are 0. See Sec. 3.3.2 for a detailed description of how the parameters are subsequently derived.

If there is not enough variation in the data, the aperture problem occurs as it does in standard least squares estimation (cmp. Sec. 2.1.2 and Sec. 2.3.1). This means the null-space has a higher dimension as indicated by the model. The normal solution is then the shortest vector in Null($\mathbf{J}$) with the last entry (1D null-space) or one of the last 2 or 3 entries (2D or 3D null-space) equal to 1. See e.g., [68] for an analysis of the aperture problem and possible solutions.

If there is too much variation in the data, the smallest eigenvalues are considerably larger than 0. Therefore the model error $||\mathbf{e}||_2$ from (2.17), also called residual of the local fit, is large because it equals the smallest eigenvalue. If this value is much larger than expected from the noise level in the data the model assumption does not fit the data. Estimated parameters are then unreliable. We can use the smallest eigenvalue $\mu_{\min}$ of $\mathbf{J}$ to define a confidence measure as follows:

$$\omega = \begin{cases} 0 & \text{if } \mu_{\min} \geq \sigma^2, \\ \left(\frac{\sigma^2 - \mu_{\min}}{\sigma^2}\right)^2 & \text{else.} \end{cases} \tag{2.18}$$

where σ^2 is the variance in the structure tensor components due to noise in the data. Variance σ^2 of the structure tensor components can be calculated from the noise variance in the image data by error propagation. Estimation of noise in the image data itself is difficult as noise has to be separated from structure. One approach is

using a Laplacian operator and adaptive edge detection following [71].

In most cases it does not make a lot of sense to distinguish between full and normal solutions and cases where model assumptions are not met. Instead reliability of an estimated parameter should be calculated by a suitable error analysis (see e.g., [46]). If an aperture problem occurs variance of an estimated parameter in the undetermined direction will be close to infinity. In the following we therefore assume that enough data variation is present and an error analysis is performed, if needed.

Structure Tensor Method

Optical flow estimation via the structure tensor is discussed in [6, 33, 26]. There an image sequence $I(\mathbf{x})$ is treated as a *volume* of scalar, unit-free gray values with $\mathbf{x} = (x_1, x_2, x_3)^{\mathrm{T}}$ where x_3 is time. The single motion model given in (2.3) can thus be written

$$\nabla^{\mathrm{T}} I \cdot (\mathbf{u}^{\mathrm{T}}, 1)^{\mathrm{T}} = 0 \tag{2.19}$$

with data vector $\mathbf{d} = \nabla I$ and parameter vector $\mathbf{p} = (\mathbf{u}^{\mathrm{T}}, 1)^{\mathrm{T}}$. The data vector can be calculated by convolving intensity signal I with derivative filters optimized for optical flow computation (cmp. Sec. 2.2). The structure tensor $\mathbf{J} = \mathbf{D}^{\mathrm{T}}\mathbf{W}\mathbf{D}$ from (2.17) may then be calculated as

$$\mathbf{J}_\rho = \mathcal{G}_\rho * (\nabla I \nabla^{\mathrm{T}} I) = \begin{pmatrix} J_{11} & J_{12} & J_{13} \\ J_{12} & J_{22} & J_{23} \\ J_{13} & J_{23} & J_{33} \end{pmatrix} \quad , \tag{2.20}$$

where weights $\mathbf{W}$ are given by a Gaussian filter kernel $\mathcal{G}_\rho$ with variance ρ^2. The support of this kernel defines the local neighborhood Λ. In Sec. 2.3.1 matrix $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ was used to generate the Moore-Penrose-inverse. Gaussian filtering of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ yields the upper left 2×2 submatrix of $\mathbf{J}_\rho$ and has been used for local orientation estimation in 2D data [7, 44]. Eigenvectors $\mathbf{e}_i$ $(i \in \{1, 2, 3\})$ of $\mathbf{J}_\rho$ give preferred local orientations, and corresponding eigenvalues μ_i denote local contrast or gray-value variation along these directions. Typically we deal with noisy data, thus part of this contrast comes from the noise. For independently distributed, additive Gaussian noise of variance σ_n^2 each diagonal entry of $\mathbf{J}_\rho$ on average is increased by the noise variance σ_d^2 in the derivative estimates[2]. As this does not change the eigensystem of $\mathbf{J}_\rho$ the structure tensor method is robust under this kind of noise [34]. In addition it can be implemented efficiently [26].

Eigenanalysis

An eigenvalue analysis of the structure tensor corresponds to a total least squares fit of a locally (on the scale of $\mathcal{G}_\rho$) constant displacement vector field to the intensity

[2]This variance σ_d^2 depends on the noise variance σ_n^2 and the actual implementation of the derivatives.

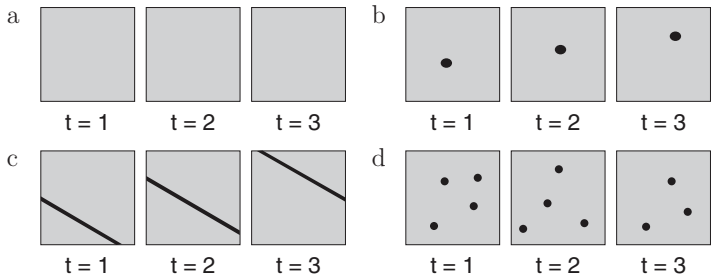


Figure 2.4: The four considered spatio-temporal neighbourhoods. a: no structure, b: point-like structure, c: linear structure and d: no coherent structure.

data (see Sec. 2.1.1 and e.g., [26]). In the following we assume the eigenvalues $\mu_i\,(i \in \{1,2,3\})$ of $\mathbf{J}_\rho$ to be sorted in descending order and the corresponding eigenvectors $\mathbf{e}_i$ to be normalized. $\mathbf{J}_\rho$ thus can be written

$$\mathbf{J}_\rho = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)\mathbf{M}(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)^{\mathrm{T}} \tag{2.21}$$

where $\mathbf{M}$ is a 3×3 diagonal matrix with $M_{ii} = \mu_i$.

The noise variance σ_d^2 is a natural threshold below which an eigenvalue μ can be considered to be 0. As illustrated in Fig. 2.4 there are four cases that have to be considered:

1. **No structure** ($\mu_{1,2,3} \lesssim \sigma_d^2$): All eigenvalues vanish and no motion can be estimated.

2. **Corner- or point-like structure** ($\mu_3 \lesssim \sigma_d^2;\ \mu_1, \mu_2 > \sigma_d^2$): Moving corner- or point-like structures result in one-dimensional trajectories and are characterized by only the third eigenvalue of $\mathbf{J}_\rho$ vanishing. In this case the displacement vector is easily obtained from $\mathbf{e}_3$:

$$\mathbf{u} = \frac{1}{\mathbf{e}_{3,3}}(\mathbf{e}_{3,1}\ \mathbf{e}_{3,2})^{\mathrm{T}}\ . \tag{2.22}$$

3. **Linear structures** ($\mu_2, \mu_3 \lesssim \sigma_d^2;\ \mu_1 > \sigma_d^2$): This is the aperture problem and only the flow $\mathbf{n}$ normal to the intensity structure may be estimated. For the special case of optical flow (3D signal I, 1D null-space of $\mathbf{J}$) it can be estimated from $\mathbf{e}_1$ (cmp. [68]):

$$\mathbf{n} = \frac{-\mathbf{e}_{1,3}}{\mathbf{e}_{1,1}^2 + \mathbf{e}_{1,2}^2}(\mathbf{e}_{1,1}\ \mathbf{e}_{1,2})^{\mathrm{T}}\ . \tag{2.23}$$

4. **No coherent motion** ($\mu_1, \mu_2, \mu_3 > \sigma_d^2$): In this case the local fit of the flow model (here constant) failed and no sensible motion may be estimated. This happens for example in the presence of motion discontinuities.

2.3.3 Respecting Known Measurement Errors in Estimation Schemes

In Sec. 2.3.2 standard total least squares estimation is presented without regarding known measurement errors, i.e., covariance matrices. In general motion model components may have different variances, e.g., in the affine motion model (2.6) presented in Sec. 2.1.1. There, gradients are weighted by local coordinates $\Delta\mathbf{x}$. Assuming that the gradients have all the same variance σ, the covariance matrix of an affine model with

$$\mathbf{d} = [I_x, I_y, I_x\Delta x, I_x\Delta y, I_y\Delta x, I_y\Delta y, I_t]^{\mathrm{T}}$$

and

$$\mathbf{p} = [u_x, u_y, a_{11}, a_{12}, a_{21}, a_{22}, 1]^{\mathrm{T}}$$

is given by

$$\mathbf{C}(\Delta x, \Delta y) = \sigma^2 \begin{pmatrix} 1 & 0 & \Delta x & \Delta y & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta x & \Delta y & 0 \\ \Delta x & 0 & \Delta x^2 & \Delta x\Delta y & 0 & 0 & 0 \\ \Delta y & 0 & \Delta x\Delta y & \Delta y^2 & 0 & 0 & 0 \\ 0 & \Delta x & 0 & 0 & \Delta x^2 & \Delta x\Delta y & 0 \\ 0 & \Delta y & 0 & 0 & \Delta x\Delta y & \Delta y^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.24}$$

depending on local coordinates $\Delta\mathbf{x}$. In the affine model the local coordinates may influence estimation depending on their scale, because changing the scale directly changes the covariance matrix. In some computer vision tasks like computation of the fundamental matrix [23] they are scaled in the range $[-1; 1]$. However, the scale of the local coordinates and therefore the scale of parts of the data vector $\mathbf{d}$, may be set arbitrarily. Then it is desirable to weight parameters according to their covariances to minimize bias coming from different measurement errors. Re-scaling of variances may avoid these biased estimates [16, 38]. An overview of estimation approaches incorporating covariances can be found in [16]. In this section we briefly review three of the most common techniques, namely

- Sampson's scheme (*SMP*, [52]),
- the Fundamental Numerical Scheme (*FNS*, [16]) and
- First Order Renormalization Version III (*FOR III*, [38]).

Standard total least squares of Sec. 2.3.2 can be understood as minimizer of the energy

$$E = \sum_i^N \frac{\mathbf{p}^{\mathrm{T}}\mathbf{A}_i\mathbf{p}}{\mathbf{p}^{\mathrm{T}}\mathbf{p}} \tag{2.25}$$

for all constraints i in a neighborhood of size N and non-smoothed structure tensor $\mathbf{A} = \mathbf{J}_0$. The eigenvector to the smallest eigenvalue of $\sum_i^N \mathbf{A}_i$ yields $\mathbf{p}_{TLS}$, the solution of (2.25) as shown in Sec. 2.3.2. The standard total least squares estimator treats all data as being equally important. However, if information about measurement errors is available, e.g., the covariance matrix of the affine model (2.24), it is desirable to incorporate this information into the estimation process. Based on the principle of maximum likelihood and Kanatani's work on geometric fitting [38], Chojnacki et al. [16] derived a more sophisticated energy function. The measurement errors for each constraint i may be given by a covariance matrix $\mathbf{C}_i$. Then, an approximation of the energy of a maximum likelihood (*AML*) estimate can be derived by

$$E_{AML} = \sum_i^N \frac{\mathbf{p}^\mathrm{T}\mathbf{A}_i\mathbf{p}}{\mathbf{p}^\mathrm{T}\mathbf{C}_i\mathbf{p}} \,. \tag{2.26}$$

The minimum of (2.26) can be determined by finding $\mathbf{p}_{AML}$ with the variational approach $\partial_\mathbf{p}\mathbf{E}_{AML} = \mathbf{0}$, where

$$\frac{\partial \mathbf{E}_{AML}}{\partial \mathbf{p}} = 2\mathbf{X}_\mathbf{p}\mathbf{p} \stackrel{!}{=} \mathbf{0} \tag{2.27}$$

and

$$\mathbf{X}_\mathbf{p} = \sum_i^N \frac{\mathbf{A}_i}{\mathbf{p}^\mathrm{T}\mathbf{C}_i\mathbf{p}} - \sum_i^N \frac{\mathbf{p}^\mathrm{T}\mathbf{A}_i\mathbf{p}}{(\mathbf{p}^\mathrm{T}\mathbf{C}_i\mathbf{p})^2}\mathbf{C}_i \,. \tag{2.28}$$

This equation is non-linear and not every solution of the variational equation is a global minimum of $\mathbf{E}_{AML}$.
In practice $\mathbf{p}$ is calculated numerically. It is assumed that $\mathbf{p}_{TLS}$ already is close to $\mathbf{p}_{AML}$. Numerical methods with seed $\mathbf{p}_{TLS}$ are then likely to converge to $\mathbf{p}_{AML}$.

Fundamental Numerical Scheme

A vector $\mathbf{p}$ solves (2.28) if and only if it falls into the null space of matrix $\mathbf{X}_\mathbf{p}$. Starting with $\mathbf{p}_{TLS} = \mathbf{p}_{k-1}$ an improved estimate can be obtained by taking the eigenvector to the smallest eigenvalue of $\mathbf{X}_{\mathbf{p}_{k-1}}$. This eigenvector most closely approximates the null space of $\mathbf{X}_{\mathbf{p}_k}$. When the new solution vector $\mathbf{p}_k$ is sufficiently close to the previous one $\mathbf{p}_{k-1}$ this iterative procedure is terminated, otherwise k is incremented. This minimization scheme is presented in [16] and is the so-called *Fundamental Numerical Scheme*.

Sampson's Scheme

Another numerically estimation technique for (2.27) is the so-called *Sampson's Scheme*. There, solution vector $\mathbf{p}$ in the denominator of $\mathbf{E}_{AML}$ is frozen at a value ξ. The energy to be minimized becomes

$$\mathbf{E}_{SMP} = \mathbf{p}^{\mathrm{T}}\mathbf{M}_{\xi}\mathbf{p} \tag{2.29}$$

with

$$\mathbf{M}_{\mathbf{p}} = \sum_{i}^{N} \frac{\mathbf{A}_i}{\mathbf{p}^{\mathrm{T}}\mathbf{C}_i\mathbf{p}} \; . \tag{2.30}$$

Starting with an initial estimate $\mathbf{p}_0$ (typically $\mathbf{p}_0 = \mathbf{p}_{TLS}$) and a stopping criterion, (2.29) may be solved iteratively.
It can be shown that *Sampson's Scheme* has an inherent systematic bias. This bias may be avoided by the renormalization technique of Kanatani [38], see below.

First Order Renormalization Scheme

Following [16] the bias of $\mathbf{M}$ can be removed with the help of a compensating factor J_{com}. This leads to

$$\mathbf{Y}_{\mathbf{p}} = \sum_{i}^{N} \frac{\mathbf{A}_i - J_{com}(\mathbf{p})\mathbf{C}_i}{\mathbf{p}^{\mathrm{T}}\mathbf{C}_i\mathbf{p}} = \mathbf{M}_{\mathbf{p}} - J_{com}\mathbf{N}_{\mathbf{p}} \tag{2.31}$$

where $\mathbf{M}_{\mathbf{p}}$ is given by (2.30), and $\mathbf{N}_{\mathbf{p}}$ is defined by

$$\mathbf{N}_{\mathbf{p}} = \sum_{i}^{N} \frac{\mathbf{C}_i}{\mathbf{p}^{\mathrm{T}}\mathbf{C}_i\mathbf{p}} \; , \tag{2.32}$$

and

$$J_{com} = \frac{1}{N} \sum_{i}^{N} \frac{\mathbf{p}^{\mathrm{T}}\mathbf{A}_i\mathbf{p}}{\mathbf{p}^{\mathrm{T}}\mathbf{C}_i\mathbf{p}} \; . \tag{2.33}$$

The renormalization equation is then

$$\mathbf{Y}_{\mathbf{p}}\mathbf{p} = 0 \tag{2.34}$$

analog to the variational equation (2.27). Chojnacki et al. [16] present several schemes for solving the renormalization equation. Tests showed that the *First Order Renormalization Scheme Version III* yields best results regarding accuracy and speed.

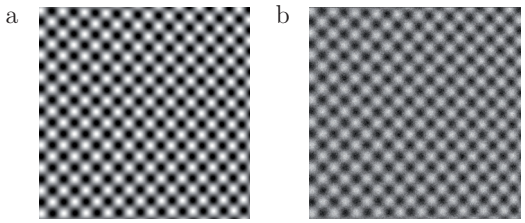


Figure 2.5: One frame of the sinusoidal input sequence. a: Without noise and b: with additive Gaussian noise with standard deviation $\sigma_n = 20$.

The procedure of the scheme is

1. Take $\mathbf{p}_{TLS}$ as initial estimate $\mathbf{p}_0$.
2. Assuming that $\mathbf{p}_{k-1}$ is known, compute matrices $\mathbf{M}_{\mathbf{p}_{k-1}}$ and $\mathbf{N}_{\mathbf{p}_{k-1}}$.
3. Compute the normalized eigenvector of the eigenvalue problem

$$\mathbf{M}_{\mathbf{p}_{k-1}}\xi = \mathbf{N}_{\mathbf{p}_{k-1}}\xi$$

   corresponding to the smallest eigenvalue and take this eigenvector as $\mathbf{p}_k$.
4. If $\mathbf{p}_k$ is sufficiently close to $\mathbf{p}_{k-1}$, then terminate the procedure, otherwise increment k and return to step 2.

For further details we refer to [16].

Experiments

For systematic error analysis we generate test sequences with the *Sequence Generator* (for details see App. B), showing a moving sinusoidal pattern with wavelength $\lambda = 32$ pixel in x- and y-direction. The movement of the surface patch generates a smooth optical flow field with translational and affine motion elements and displacements smaller than one pixel per frame to ensure compliance with the sampling theorem. The intensity values of the sequence are in the range [0; 255]. A frame of the sinusoidal sequence, without noise and with additive Gaussian noise with standard deviation $\sigma_n = 20$, is depicted in Fig. 2.5. The sequence is of size 301×301 pixel and contains 5 frames. Filters used for computing image gradients are the $5 \times 5 \times 5$ filter sets proposed in [54], see also Sec. 2.2. Size of the neighborhood is 165×165 pixel. We test robustness to noise of the proposed estimators. Therefore we add i.i.d. (independent and identically distributed) Gaussian noise with increasing standard deviations σ_n to the sequence and calculate the average of 25 parameter estimates around the center of the sequence. The experiment is repeated 20 times for each σ_n. The covariance matrix $\mathbf{\Lambda}$ is given in (2.24).

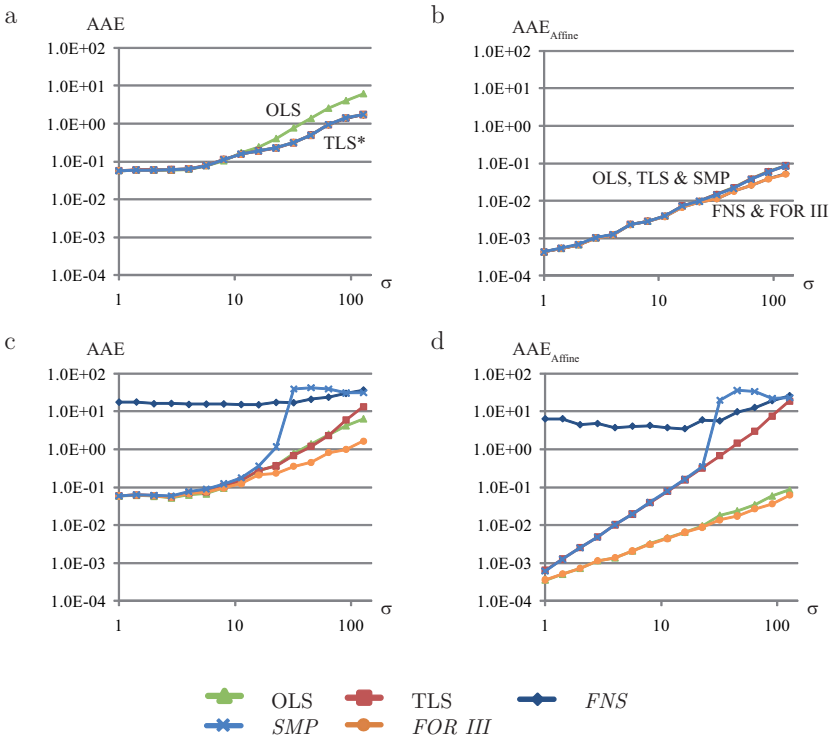


Figure 2.6: Parameter estimation with standard least squares (OLS), and total least squares (TLS*) techniques. The TLS* techniques are standard TLS, *FNS*, *SMP* and *FOR III*. Angular errors of translational (left) and affine (right) motion parameters for increasing additive Gaussian noise are shown. Scaling of local coordinates with $k = 1$ (top), i.e., normalized by pixelsize, and with $k = 1e - 4$ (bottom).

Figure 2.6 shows estimation results of standard least squares (OLS), total least squares (TLS) and the proposed numerical schemes for solving the variational equation, namely *FNS*, *SMP* and *FOR III*. The average angular error (App. A) of the translational parameters (left) and of the affine motion parameters (right) is shown for increasing standard deviations σ_n of additive Gaussian noise. Scaling of local coordinates may influence estimation (cmp. Sec. 2.3.3). In a standard affine motion model (2.6) weights of affine parameters are different to weights of the translational motion parameters. In order to examine the influence of different weights, we distinguish between angular errors of translational and affine motion parameters.
Figure 2.6a and b show angular errors using local coordinates scaled by $k = 1$, i.e., normalized by pixelsize, $\Delta\mathbf{x} \in [-82; 82]$. In this case, the standard least squares approach (Sec. 2.3.1) yields highest errors for translational motion estimates for high

noise variances ($\sigma_n > 20$). All other techniques perform similar for translational motion. Affine motion parameters are slightly better estimated by schemes incorporating covariance matrices (*FNS* and *FOR III*). Standard TLS, OLS and *SMP* yield similar results.
The effect on incorporating covariance matrices is only visible for high noise. Affine motion parameters for optical flow are typically up to a factor 100 smaller than translational motion parameters, i.e., the impact of re-scaling these parameters is small compared to the overall error. Experiments where incorporation of covariance information yield superior results are ellipse fitting [16] or fundamental matrix estimation [17]. There the influence of affine parameters is higher, because the scaling is different, e.g., typically in the range of $[-1; 1]$ for fundamental matrix estimation.
Figure 2.6c and d show results of a second experiment where affine parameters are scaled by a factor $k = 1e-4$ before estimation, i.e., $\Delta\mathbf{x} \in [-0.0082; 0.0082]$ and re-scaled afterwards. Using this scaling, errors in affine parameters influence errors in translational parameters much more than using a scaling factor $k = 1$.
In this case, the *FNS* yields no reliable estimates for neither translational nor affine motion parameters. Standard TLS and *SMP* perform similar for small noise variances. For noise with standard deviation $\sigma_n > 20$ *SMP* is not able to estimate reliable motion parameters any more. Estimates of OLS and standard TLS yield similar angular errors. However, *FOR III* estimates show lowest angular errors for noise with standard deviation $\sigma_n > 20$. The affine parameters are best estimated with OLS and *FOR III*. Angular errors of *SMP* and standard TLS are more than one order of magnitude higher compared to *FOR III* for $\sigma_n > 20$.
We conclude that the scale, i.e., selected units, of local coordinates has significant effect on estimation results. Using the *FOR III* scheme or OLS the effect is negligible, but standard TLS, the *SMP* scheme and the *FNS* scheme may yield highly degraded estimation results, if the scaling is not appropriate. The *FOR III* scheme yields overall highest accuracy for all scaled and unscaled parameters.

2.3.4 Robust Estimation

The presented estimation approaches assume that motion within a local neighborhood Λ can be determined by a single parameter set, e.g., a single motion vector, if the constancy assumption (2.3) is used. This motion assumption is violated in common situations due to transparency, depth discontinuities, shadows or noise. Robust estimation approaches relax the assumption of single motions. Black and Anandan [9] introduced a robust least squares estimation framework for optical flow to handle discontinuities and to estimate multiple motion in a spatial neighborhood. A robust ρ-function is used to decrease the influence of outliers in the estimation process. In order to estimate a second motion another estimation process may be initialized, using only the outliers of the first estimation process. The use of a robust ρ-function is detailed in the following example.
The OLS parameter estimation problem for optical flow is to find a motion vector $\mathbf{u}$

which minimizes energy function E

$$\begin{aligned} E(\mathbf{u}) = & \sum_{(\mathbf{x}\in\Lambda)} \left(I_x(\mathbf{x},t)u_x + I_y(\mathbf{x},t)u_y + I_t(\mathbf{x},t)\right)^2 \\ = & \sum_{(\mathbf{x}\in\Lambda)} \left((\nabla_{\mathbf{x}}^{\mathrm{T}} I)\mathbf{u} + I_t\right)^2 \\ = & \sum_{(\mathbf{x}\in\Lambda)} \rho_{quad}\left((\nabla_{\mathbf{x}}^{\mathrm{T}} I)\mathbf{u} + I_t\right) \end{aligned} \tag{2.35}$$

with the quadratic ρ-function $\rho_{quad}(x) = x^2$ and neighborhood Λ. This ρ-function is optimal, when errors in $\nabla_{\mathbf{x}} I$ are normally distributed. However, the quadratic ρ-function assigns high weights to outlying measurements. The so-called influence function of a particular ρ-function is used to demonstrate this behavior. The influence function characterizes the bias that a particular measurement has on the solution and is proportional to the derivative, Ψ, of the ρ-function. In the least squares case, the influence of data points increases linearly and without bound. To increase robustness a ρ-function should reduce the weight of outliers. The quadratic ρ-function and three common robust ρ-functions with the corresponding Ψ-functions are depicted in Fig. 2.7.
The truncated quadratic

$$\rho_{tq}(x,\alpha,\lambda) = \begin{cases} \lambda x^2 & \text{if } |x| < \frac{\sqrt{\alpha}}{\sqrt{\lambda}} \\ \alpha & \text{else} \end{cases} \tag{2.36}$$

weights errors quadratically up to a fixed threshold. Beyond that threshold errors receive a constant value α. The truncated quadratic is similar to the Huber function [30]. There errors increase linearly beyond the threshold. Other robust functions are the Geman and McClure [20]

$$\rho_{gem}(x,\sigma) = \frac{x^2}{\sigma + x^2} \tag{2.37}$$

and the Lorentzian

$$\rho_{lor}(x,\sigma) = \log\left(1 + \frac{1}{2}\left(\frac{x}{\sigma}\right)^2\right). \tag{2.38}$$

These robust function have differentiable Ψ-functions

$$\Psi_{gem}(x,\sigma) = \frac{2x\sigma}{(\sigma + x^2)^2} \tag{2.39}$$

and

$$\Psi_{lor}(x,\sigma) = \frac{2x}{2\sigma^2 + x^2}, \tag{2.40}$$

which provide a more gradual transition between inliers and outliers.
To make (2.35) more robust, it is reformulated to account for outliers by using a

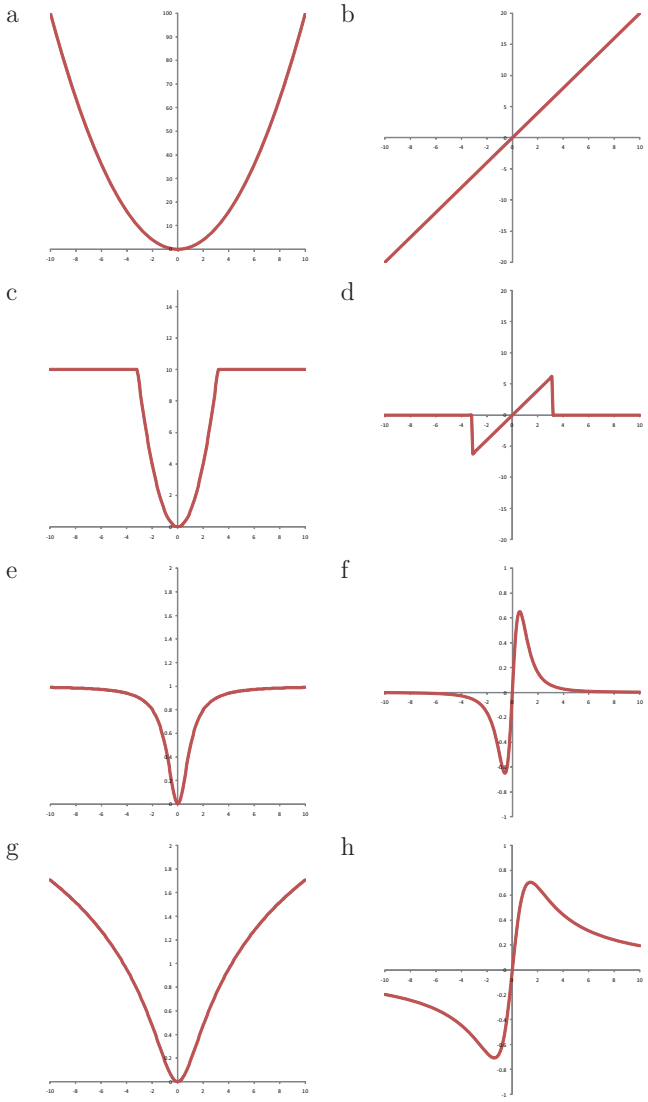


Figure 2.7: Common weighting ρ-functions (left) and their derivatives Ψ (right). From top to bottom: Quadratic, truncated quadratic, Geman-McClure and Lorentzian.

robust ρ-function. We get

$$E(\mathbf{u}) = \sum_{(\mathbf{x}\in\Lambda)} \rho\left((\nabla_{\mathbf{x}}^{\mathrm{T}} I)\mathbf{u} + I_t, \sigma\right) \tag{2.41}$$

with robust function ρ.

Robust Least Squares

Black and Anandan [9] use successive over-relaxation (SOR, cmp. [12]) to find the minimum of (2.41). The iterative update equation for one motion parameter u_i at step $n+1$ is

$$u_i^{n+1} = u_i^n - \kappa \frac{1}{T(u_i)} \frac{\partial E}{\partial u_i} \tag{2.42}$$

where κ is an over-relaxation parameter. When $0 < \kappa < 2$ the method can be shown to converge, but the rate of convergence is sensitive to the exact value of κ. The term $T(u_i)$ is an upper bound on the second partial derivative of E,

$$T(u_i) \geq \frac{\partial^2 E}{\partial u_i^2} \ . \tag{2.43}$$

Typically the objective function is non-convex. To find the globally optimal solution Black and Anandan propose to use a robust ρ-function with a parameter which controls the influence of the outliers, e.g., parameter σ. They start with minimizing a convex approximation of the objective function, i.e., high σ, thus outliers are not down-weighted. Then they successively minimize better approximations of the true objective function, i.e., lowering σ, starting from the previous estimated minimum, so that influence of outliers on the estimation process gets less. This approach works well in practice, but is not guaranteed to converge to the global minimum, because the initial estimate may be too far away from the global minimum.

Robust Total Least Squares

Equation (2.41) may be solved in total least squares sense analog to Sec. 2.3.2. The approximated maximum likelihood function (2.26) is combined with the robust ρ-function by

$$E_{AML,Rob} = \sum_{i}^{N} \rho\left(\frac{\mathbf{p}^{\mathrm{T}}\mathbf{A}_i\mathbf{p}}{\mathbf{p}^{\mathrm{T}}\mathbf{C}_i\mathbf{p}}, \sigma\right) \ . \tag{2.44}$$

The minimum of the energy $E_{AML,Rob}$ may be found numerically with the schemes presented in Sec. 2.3.2, e.g., *FOR III* or *FNS*. Analog to [9] the control parameter σ of the robust function should be successively minimized to find the global minimum.

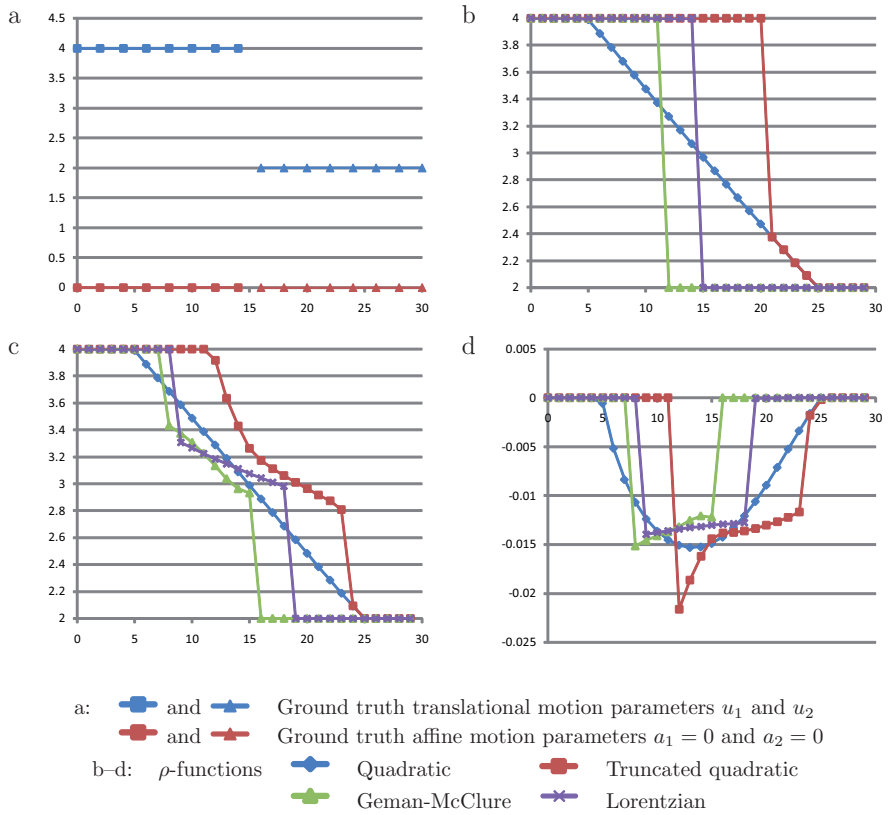


Figure 2.8: Robust estimation. a: Ground truth values, b: robust estimation with translational motion model, c and d: robust estimation with affine motion model. c: Translational motion estimates and d: affine motion estimates.

Robust Estimation for Parametric Motion Models

Finding the global minimum of a robust energy function highly depends on the choice of the ρ-function and its control parameter σ. The proposed minimization schemes work well in practice for constant motion models. Difficulties arise when using more complex motion models, e.g., an affine motion model (2.6).
An experiment investigating the influence of ρ-functions on parameter estimation for one dimensional optical flow is shown in Fig. 2.8. Figure 2.8a shows the ground truth optical flow parameters. Translational motion parameters $u_1 = 4$ and $u_2 = 2$ are depicted in blue and affine motion parameters $a_1 = a_2 = 0$ in red, i.e., pure translational motion. The data term is given by $\mathbf{d}_1 = [5, -20]^{\mathrm{T}}$ and $\mathbf{d}_2 = [7, -14]^{\mathrm{T}}$. A motion discontinuity occurs at $x = 15$. The control parameter for ρ_{gem} and ρ_{lor} is

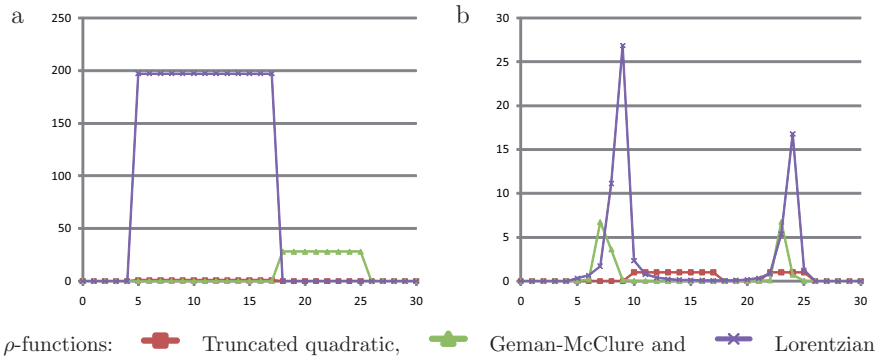


Figure 2.9: Robust estimation. Weights for estimate at $x = 23$, a: robust estimation with translational motion model and b: robust estimation with affine motion model.

lowered from $\sigma = 15$ to $\sigma = 0.1$. The threshold for the truncated quadratic is $\alpha = 3$ and $\lambda = 1$. Size of the neighborhood is $N = 21$.
Figure 2.8b shows estimation results using robust total least squares and a constant motion model. It clearly demonstrates the smoothing effect using the quadratic ρ-function, due to high weighting of outliers. The truncated quadratic ρ-function recovers a discontinuity, but it is shifted by approx. 6 values and still the motion parameters are slightly smoothed. Using robust functions ρ_{gem} or ρ_{lor} yield clearly separated motion parameters. However, using the ρ_{gem}-function proposed by Geman and McClure also leads to a shift of the discontinuity. The shift of the discontinuity depends on the difference of the energy functions around the discontinuity, the ρ-function and its control parameter, which determines when a parameter is declared as an outlier. Figures 2.8c and d show estimation results when using an affine motion model. Results for translational motion parameters are depicted in Fig. 2.8c, and for affine motion parameters in Fig. 2.8d. Simultaneous estimation of all parameters of a parametric model leads to erroneous estimates at the motion discontinuity, because the iterative estimation process does not converge to the global energy minimum. A local minimum solution is not a solely translational motion at one side and outliers at the other side, but a divergent motion. Figure 2.9 shows weights for $x = 23$, i.e., near the discontinuity, calculated by the different ρ-functions for the translational and the affine motion model. Using the translational motion model, the data on the left side gets high weights and the outliers on the right side are suppressed. Figure 2.9b demonstrates that high weights are assigned to data on both side of the discontinuity, when an affine motion model is used. This effect occurs regardless of which ρ-function is used.
Black and Anandan avoid this kind of erroneous estimates for parametric models by successive estimation of translational and affine model parameters.

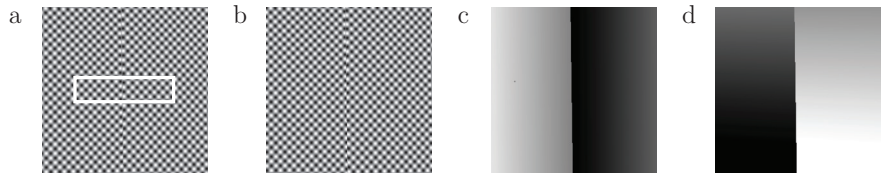


Figure 2.10: a and b: First and last frame of a sinusoidal sequence with two affine motions. The white rectangle depicts the area, where the average angular error is evaluated. c and d: gray value coded ground truth of translational motion parameters in x- and y-direction respectively.

Experiments

In this section we compare the robust estimators *Robust TLS*, *Robust Affine OLS* and *Robust Affine TLS*. Sinusoidal sequences are generated with the *Sequence Generator* (App. B). Wavelength of the sinusoidal pattern is $\lambda_x = \lambda_y = 8$ pixel and sequence size is $301 \times 301 \times 9$. In the first experiment two patches move with translational motion. In the second experiment both patches have translational and affine motion parameters. The maximum motion in both sequences is less than one pixel per frame. Figure 2.10 shows two frames of the sequence for translational and affine motion and the gray value coded ground truth optical flow in x- and y- direction. Optical flow values in x-direction are in the range $[-0.1, 0.3]$ and in y-direction in the range $[-0.3, 0.3]$. Standard deviation of additive i.i.d. Gaussian noise is increased and for each standard deviation experiments are repeated 10 times. The size of the neighborhood is 65 pixel in x- and y-direction and 5 frames in time and is weighted with a Gaussian with standard deviation $19 \times 19 \times 1$. All robust estimators use the Lorentzian (2.38), initialized with $\sigma = 5$ and lowered to $\sigma = 0.1$. Iterations are stopped after 50 iterations or when change of parameters is below $1E-08$. The average angular error is calculated for a rectangular patch of 151×11 pixel around the discontinuity (see Fig. 2.10a). Translational and affine angular errors for both sequences are depicted in Fig. 2.11.
Figure 2.11a demonstrates that all estimators perform similar for noise with standard deviation up to approx. $\sigma_n = 3$. The ordinary least squares estimator performs slightly better compared to the total least squares estimators for higher noise values. In the case of no noise the standard TLS performs best. Affine angular errors are similar for both affine estimators, when noise is present. In the noise-free case the OLS approach yields lower affine angular errors.
In Fig. 2.11c is shown that the affine TLS approach performs best for low noise ($\sigma_n < 3$), when a sequence contains affine motion parameters. Standard TLS performs better than affine OLS for noise with $\sigma_n < 1$. All estimators yield similar results for higher noise variances. Figure 2.11d demonstrates that affine angular errors of the affine TLS is up to one orders of magnitude lower than for the affine OLS in the

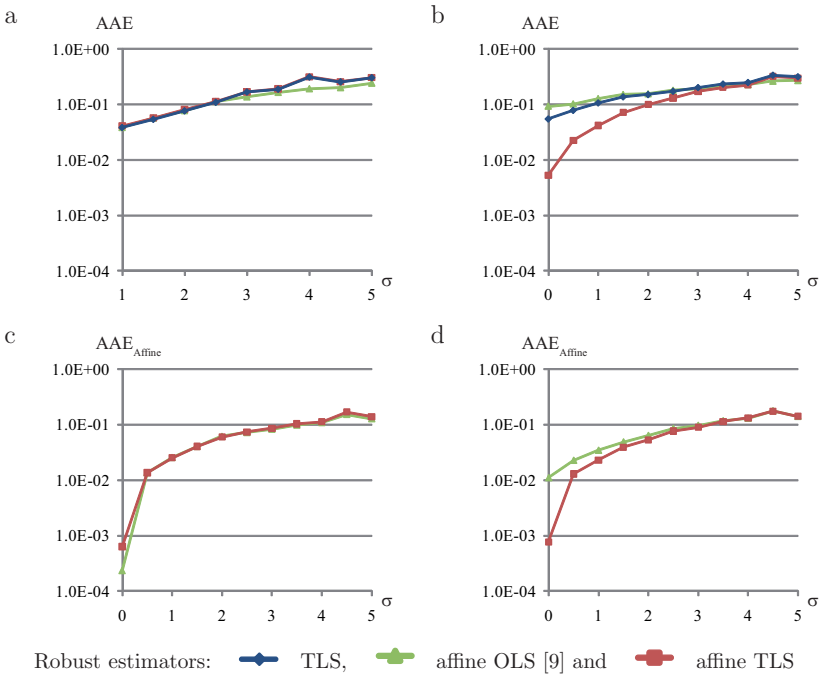


Figure 2.11: Average translational (top) and affine (bottom) angular errors of motion estimates near a discontinuity caused by two translational (left) and two affine (right) moving surface patches (see Fig. 2.10).

noise free case. The TLS estimator yields still slightly better results for noise with $\sigma_n < 3$, for higher standard deviations of noise OLS and TLS perform similar.

2.4 Conclusions

In this section parameter estimation techniques applied to motion estimation based on optical flow have been presented. A data model based on the brightness constancy constraint and constant, affine and quadratic motion models for optical flow have been introduced. Optimized filters used to calculate image derivatives have been proposed in Sec. 2.2. Standard local estimation techniques and more sophisticated local estimators incorporating covariance distributions and handling outliers have been discussed and compared in Sec. 2.3.
Synthetic experiments (cmp. Fig. 2.11) showed that affine modeling of the flow field does only improve estimation results when an appropriate estimator is used. An affine TLS estimator yields better results in the low noise case, whereas the least squares estimator proposed by Black and Anandan [9] yields only similar or worse

results than standard TLS. However, in order to ensure reliable estimates the size of the local neighborhood has to be increased for affine motion models, because more motion parameters have to be estimated.
Incorporating known measurement errors yields better motion estimates for sequences heavily corrupted by noise or when the influence of (affine) parameters is high due to inappropriate scaling of parameters. In this case the *First Order Renormalization Scheme Version III* (*FOR III*) showed best performance. Standard TLS yields similar performance for input data with low noise ($\sigma_n < 20$) and appropriate scaling of parameters, i.e., scaling factor $k = 1$.
Robust estimation should be used when discontinuities (e.g., caused by noise or multiple motions) occur in the data. The Lorentzian ρ-function best separated different parameter near discontinuities. Combinations of parametric motion models and robust estimation techniques may lead to corrupt estimates at motion discontinuities. Successive robust estimation of translational and affine motion parameters can avoid convergence to local minimum solutions. In synthetic experiments the robust affine TLS outperformed standard TLS and the affine OLS [9] for translational and affine motion estimation in sequences with discontinuities.

Chapter 3

Approaches to 3D Motion Estimation

In literature many models have been introduced for 3D structure and 3D motion estimation. 3D motion is sometimes called *Scene Flow*, to point out the relationship to optical flow, or *Range Flow*, that is 3D motion calculated from range data. Both motion estimation techniques, namely optical flow-based and correspondence-based (cmp. Sec. 2), have been used for 3D structure and 3D motion estimation. However, optical flow-based estimation schemes have been shown to yield higher accuracy for dedicated setups [26]. In this work the focus is on optical flow-based estimation. The so-called *Scene Flow* technique [74] is one of the most popular 3D motion estimation frameworks. In general this technique may be used for estimation of 3D motion from one or more cameras and for computing scene structure. However, the flow-based reconstruction algorithm proposed in [76] can only recover structure where the scene is actually moving. The 3D motion estimation algorithm has been improved by [40], recently. They incorporate probability distributions from 2D optical flow to make the estimation of 3D motion more reliable. *Range Flow* estimation techniques are based on range image sequences. Spies et al. [67] presented a framework where range constraints on range sequences are combined with optical flow on intensity sequences. *Scene Flow* and *Range Flow* techniques solely estimate 3D motion based on already available 3D structure information. In contrast [57] presented a multi-camera framework for simultaneous estimation of 3D structure and 3D motion. Therefore the parameters of an affine optical flow-based model (cmp. Sec. 2.1.1) are used to solve for depth, surface slopes and translational motion. The framework has been generalized in [56] and has been made robust against brightness changes in the data [60]. Further improvements in [62] made the model more robust and incorporated constraints from *Range Flow*.

In this chapter we review *Scene Flow*, *Range Flow* and the *Affine Model*. We evaluate estimation of depth and slopes with the *Affine Model* on synthetic test sequences and on real data. Finally the three motion estimation techniques are compared on synthetic sequences and real data.

3.1 *Scene Flow*

Vedula et al. [74] define *Scene Flow* as the 3D motion field of points in the world, just as optical flow is the 2D motion field of points in the image (cmp. Sec. 2.1). Optical flow is the projection of 3D motion onto the image plane of a camera.

Given a 3D point $\mathbf{X} = (X, Y, Z)^{\mathrm{T}}$, the projection onto the image plane in camera c is denoted by $\mathbf{x}_c = (x_c, y_c)^{\mathrm{T}}$. The 2D components of $\mathbf{x}_c$ are

$$x_c = \frac{[\mathbf{P}_c]_1(X, Y, Z, 1)^{\mathrm{T}}}{[\mathbf{P}_c]_3(X, Y, Z, 1)^{\mathrm{T}}}\,, \qquad y_c = \frac{[\mathbf{P}_c]_2(X, Y, Z, 1)^{\mathrm{T}}}{[\mathbf{P}_c]_3(X, Y, Z, 1)^{\mathrm{T}}} \tag{3.1}$$

where $[\mathbf{P}_i]_j$ is the j^{th} row of the camera projection matrix P_c [23]. If the camera is not moving, then the 2D optical flow $\mathbf{u} = \frac{\mathrm{d}\mathbf{x}_c}{\mathrm{d}t}$ is uniquely determined by

$$\mathbf{u} = \frac{\partial \mathbf{x}_c}{\partial \mathbf{X}} \frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t}\,. \tag{3.2}$$

This equation denotes how optical flow $\mathbf{u}$ can be computed for known 3D motion $\mathbf{f} = \frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t}$. Computing 3D motion $\mathbf{f}$ from optical flow $\mathbf{u}$ requires inverting (3.2). The problem is, that $\mathbf{X}$ depends not only on the optical flow but also on time, indirectly through the 3D structure, i.e., $\mathbf{X} = \mathbf{X}(\mathbf{x}_c(t), t)$. 3D motion $\mathbf{f}$ is given by the total differential of $\mathbf{X}$ with respect to time

$$\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} = \frac{\partial \mathbf{X}}{\partial \mathbf{x}_c} \frac{\mathrm{d}\mathbf{x}_c}{\mathrm{d}t} + \left.\frac{\partial \mathbf{X}}{\partial t}\right|_{\mathbf{x}_c}\,. \tag{3.3}$$

This means that the motion of a point cannot be calculated solely from optical flow. The second term determines the change of a point on the surface imaged by a fixed pixel, i.e., the motion of $\mathbf{X}$ along the ray corresponding to $\mathbf{x}_c$. In order to solve for 3D motion $\mathbf{f}$ with (3.3) this motion has to be known. If structure data is not available for additional time steps more than one camera viewing the scene is needed to invert (3.2). Assuming that optical flow is known we have two linear constraints and three unknowns in $\mathbf{f}$. Analog to optical flow (cmp. Sec. 2.1), more than two equations are needed to solve for 3D motion $\mathbf{f}$, i.e., at least two cameras are needed. The final equation system is

$$\mathbf{B}\mathbf{f} = \mathbf{U}\,, \text{ with } \mathbf{B} = \begin{pmatrix} \frac{\partial x_{c,1}}{\partial X} & \frac{\partial x_{c,1}}{\partial Y} & \frac{\partial x_{c,1}}{\partial Z} \\ \frac{\partial y_{c,1}}{\partial X} & \frac{\partial y_{c,1}}{\partial Y} & \frac{\partial y_{c,1}}{\partial Z} \\ \frac{\partial x_{c,2}}{\partial X} & \frac{\partial x_{c,2}}{\partial Y} & \frac{\partial x_{c,2}}{\partial Z} \\ \vdots & \vdots & \vdots \\ \frac{\partial x_{c,N}}{\partial X} & \frac{\partial x_{c,N}}{\partial Y} & \frac{\partial x_{c,N}}{\partial Z} \end{pmatrix} \text{ and } \mathbf{U} = \begin{pmatrix} \frac{\partial x_{c,1}}{\partial t} \\ \frac{\partial y_{c,1}}{\partial t} \\ \frac{\partial x_{c,2}}{\partial t} \\ \vdots \\ \frac{\partial x_{c,N}}{\partial t} \end{pmatrix} \tag{3.4}$$

for N calibrated cameras. Analog to optical flow, (3.4) may be solved with any technique presented in Sec. 2.3. Vedula et al. [75] propose a singular value decompo-

sition of **B** to find the solution that minimizes the sum of least squares of the error obtained by re-projecting the 3D motion onto each of the optical flows.

3.2 *Range Flow*

The *Range Flow* model is based on two motion constraints: one for range data and the other one for intensity data. Following [69], we briefly review these two constraints.

3.2.1 The Range Constraint

Let a surface be described by its depth $Z = Z(X, Y, t)$ as a function of space and time, where X, Y and Z are spatial coordinates and t denotes time. Without loss of generality, we define X and Y to be aligned with the camera sensor coordinates x and y, respectively. The Z-axis is the optical axis of the camera, which is assumed to be projective. The total derivative of Z with respect to time then yields the so-called range flow motion constraint equation

$$\frac{\mathrm{d}Z}{\mathrm{d}t} = \partial_X Z \frac{\mathrm{d}X}{\mathrm{d}t} + \partial_Y Z \frac{\mathrm{d}Y}{\mathrm{d}t} + \partial_t Z \tag{3.5}$$

where partial derivatives are denoted by $\partial_X Z := \frac{\partial Z}{\partial X}$ and so on. The range flow is now defined as $\mathbf{f} = [U, V, W]^{\mathrm{T}} := [\frac{\mathrm{d}X}{\mathrm{d}t}, \frac{\mathrm{d}Y}{\mathrm{d}t}, \frac{\mathrm{d}Z}{\mathrm{d}t}]^{\mathrm{T}}$. Some range sensors like those used in [67], and 3D structure from motion algorithms (cf. [60]) produce range data as data sets $X = X(x, y, t)$, $Y = Y(x, y, t)$ and $Z = Z(x, y, t)$ over the sensor coordinates x, y and time t. Rewriting the range flow constraint (3.5) as in [67] allows to compute partial derivatives directly on the sensor grid rather than on world coordinate data, thus avoiding interpolation artifacts and expensive preprocessing steps. Range flow, i.e., the total derivatives of the world coordinates with respect to time, may then be calculated as

$$U = \frac{\mathrm{d}X}{\mathrm{d}t} = \partial_x X \dot{x} + \partial_y X \dot{y} + \partial_t X \tag{3.6}$$

$$V = \frac{\mathrm{d}Y}{\mathrm{d}t} = \partial_x Y \dot{x} + \partial_y Y \dot{y} + \partial_t Y \tag{3.7}$$

$$W = \frac{\mathrm{d}Z}{\mathrm{d}t} = \partial_x Z \dot{x} + \partial_y Z \dot{y} + \partial_t Z \tag{3.8}$$

where total derivatives with respect to time are indicated by a dot. Not being interested in the changes on the sensor grid, i.e., the optical flow, $\dot{x}$ and $\dot{y}$ in (3.6)-(3.8) can be eliminated. This yields

$$\frac{\partial(Z, Y)}{\partial(x, y)} U + \frac{\partial(X, Z)}{\partial(x, y)} V + \frac{\partial(Y, X)}{\partial(x, y)} W + \frac{\partial(X, Y, Z)}{\partial(x, y, t)} = 0 \tag{3.9}$$

where

$$\frac{\partial(Z,Y)}{\partial(x,y)} = \begin{vmatrix} \partial_x Z & \partial_x Y \\ \partial_y Z & \partial_y Y \end{vmatrix} = \partial_x Z \partial_y Y - \partial_y Z \partial_x Y \tag{3.10}$$

is the Jacobian of Z, Y with respect to x, y, and so on. Equation (3.9) depends only on derivatives in sensor coordinates, and can be calculated easily using derivative kernels. Assuming aligned world and sensor coordinate systems ($\partial_y X = \partial_x Y = 0$), (3.9) reduces to

$$\begin{aligned}(\partial_y Y \partial_x Z)U + (\partial_x X \partial_y Z)V - (\partial_x X \partial_y Y)W \\ + (\partial_x X \partial_y Y \partial_t Z - \partial_x X \partial_t Y \partial_y Z - \partial_t X \partial_y Y \partial_x Z) = 0 \,.\end{aligned} \tag{3.11}$$

3.2.2 The Intensity Constraint

The range constraint is based solely on range data, and, analog to optical flow (cmp. Sec. 2.3.2) the full flow can only be estimated where 3 or more distinct depth planes intersect. Surfaces are sometimes nearly planar, smooth surfaces resulting in aperture problems almost everywhere when using the range flow constraint only. As proposed in [67], intensity data should therefore be incorporated in addition. Let the intensity of a point remain constant over the observation time interval. Then the so-called brightness constancy constraint equation often used for optical flow estimation (see, e.g., [4]) holds. Linearization of this constraint yields

$$\frac{\mathrm{d}I}{\mathrm{d}t} = \partial_x I \dot{x} + \partial_y I \dot{y} + \partial_t I = 0 \,. \tag{3.12}$$

Eliminating optical flow $(\dot{x}, \dot{y})$ using (3.6) and (3.7) yields

$$\frac{\partial(I,Y)}{\partial(x,y)}U + \frac{\partial(X,I)}{\partial(x,y)}V + \frac{\partial(X,Y,I)}{\partial(x,y,t)} = 0 \,. \tag{3.13}$$

The estimated range flow $\mathbf{f} = [U,V,W]^{\mathrm{T}}$ has to fulfill both the range flow constraint (3.9) and the intensity constraint (3.13). The intensity constraint is more reliable to provide point-to-point correspondences, and therefore often solves the aperture problem. Together with the range constraint, it allows to solve for $\mathbf{f}$ in places where the range constraint alone is insufficient.

3.2.3 *Range Flow* Estimation

To estimate parameters within a total least squares framework, we closely follow [69]. The range constraint (3.11) yields for every pixel an equation of the form $\mathbf{d}_{rc}^{\mathrm{T}}\mathbf{p} = 0$ with

$$\mathbf{d}_{rc} = \left[\frac{\partial\,(Z,Y)}{\partial\,(x,y)}, \frac{\partial\,(X,Z)}{\partial\,(x,y)}, \frac{\partial\,(Y,X)}{\partial\,(x,y)}, \frac{\partial\,(X,Y,Z)}{\partial\,(x,y,t)}\right]^{\mathrm{T}} \tag{3.14}$$

and parameter vector

$$\mathbf{p} = [U, V, W, 1]^{\mathrm{T}} \ . \tag{3.15}$$

To solve this equation standard total least squares is used (cmp. Sec. 2.3.2), but any other estimator proposed in Sec. 2.3 could be used as well. As described in Sec. 3.2.2 we have more than one constraint for the U and V components of *Range Flow*. In the same way as expressing the range constraint by $\mathbf{d}_{rc}^{\mathrm{T}}\mathbf{p} = 0$, the intensity constraint may be expressed by $\mathbf{d}_I^{\mathrm{T}}\mathbf{p} = 0$, where $\mathbf{d}_I$ is computed from the observed data according to

$$\mathbf{d}_I = \left[\frac{\partial\left(I, Y\right)}{\partial\left(x, y\right)}, \frac{\partial\left(X, I\right)}{\partial\left(x, y\right)}, 0, \frac{\partial\left(X, Y, I\right)}{\partial\left(x, y, t\right)}\right]^{\mathrm{T}} \ . \tag{3.16}$$

As shown in [69], combining the different constraints yields a new structure tensor which is simply the weighted sum of the tensor $\mathbf{J_{rc}}$ from the range constraint (3.9), and the tensor $\mathbf{J_I}$ from the intensity-dependent constraint (3.13). With the weight β, the overall tensor thus is

$$\mathbf{J} = \mathbf{J_{rc}} + \beta\mathbf{J_I} \ . \tag{3.17}$$

The weight β may be used to account for different signal-to-noise-ratios in the structure tensors. Furthermore, the data channels should be scaled to the same mean and variance before they are combined.
The optimal estimate of the sought parameter vector is then given by the eigenvector $\mathbf{b}$ corresponding to the lowest eigenvalue of $\mathbf{J}$ (cmp. Sec. 2.3.2). As the eigenvector is only defined up to a scaling factor, the range flow is finally computed by the normalization (see (3.15))

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \frac{1}{b_4}\begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \ . \tag{3.18}$$

3.3 *Affine Model*

Scharr and Küsters [57] introduced a multi-camera extension of the brightness constancy constraint model for simultaneous local estimation of 3D motion and 3D position. This model has been extended and evaluated for 4 special cases: estimation of depth only using 1D and 2D camera grids, estimation of depth and 3D motion, and estimation of depth and normals in Scharr [55]. Scharr and Schuchert [56] presented a general framework for this *Affine Model*. However, in [56] the model is restricted to surfaces inclined in only one direction, i.e., surface normal $\mathbf{n} = (Z_X, 0, -1)^{\mathrm{T}}$ or $\mathbf{n} = (0, Z_Y, -1)^{\mathrm{T}}$. We derive a model handling arbitrarily inclined surfaces and a simultaneous estimation technique for all parameters.
An image sequence can be interpreted as data in a 3D x-y-t-space. For optical flow

estimation in this space a BCCE defines a linear model for the changes of gray values due to local object motion. The result of the calculation is a displacement vector field. Assuming the data comes from a single fixed camera displacements come from object motion. Assuming a pinhole camera looking at a fixed scene, moving orthogonal to its viewing direction, displacements (then usually called disparities) are anti-proportional to local depth. This is known as structure from camera motion (e.g., [43]).
The basic idea for this estimation technique is to interpret camera position $\mathbf{s} = (s_x, s_y)$ as additional dimensions of the data. Hence all image sequences acquired by a 1D camera grid can be combined to sample a 4D-Volume in x-y-s_x-t-space. If a 2D camera grid is used (as e.g., in [45, 77]) we get a 5D-Volume in x-y-s_x-s_y-t-space. Brightness constancy can be defined in this space as vanishing total differential of the intensity data. A camera model is used to project a dynamic surface patch into the image. This yields equations for x- and y-position of the patch as a function of camera position and time. Plugging the linearized total differentials of x and y into the total differential of the intensity results in the sought model. It boils down to be an affine optical flow model with 3 dimensions (s_x, s_y, t) behaving like the time dimension in an usual affine optical flow model (cmp. (2.6) in Sec. 2.1). A detailed derivation can be found below in Sec. 3.3.1.
As this linear data model has the same form as usual BCCEs (cmp. Sec. 2.1) no special estimation framework has to be established. We use the structure tensor method (cmp. Sec. 2.3.2 and Sec. 3.3.2), but other methods can be applied as well (e.g., the ones in [4, 26]). But as parameters in the model are mixed motion-, normals- and disparity-parameters, they have to be disentangled. Especially when 2D camera grids are used multiple independent measurements are present in each model equation. These not only have to be disentangled, but also recombined respecting their error estimates (cmp. Sec. 3.3.2).

3.3.1 Derivation of the BCCE

In this section we derive a constraint equation describing local brightness changes in the data. To do so we project a geometric model of a moving surface patch onto a camera sensor plane using a pinhole camera model. This geometric description of moving projected surface elements is then combined with a brightness constancy assumption forming the sought constraint equation.

Surface Patch Model

For each pixel at a given point in time we want to estimate from a local neighborhood depth, motion and surface normals of a 3D patch imaged at that location. Thus we use a surface patch valid for the whole neighborhood as object/motion model. This surface element has its center at world coordinate position (X_0, Y_0, Z_0) and is modeled as a function of time t and local world coordinates $(\Delta X, \Delta Y)$ with X- and

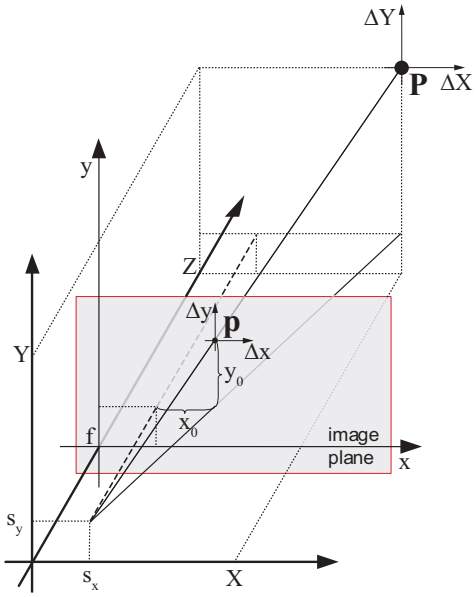


Figure 3.1: Pinhole projection of a point P using a camera with center at (s_x, s_y).

Y-slopes Z_X and Z_Y. It moves with velocity $\mathbf{U} = (U_X, U_Y, U_Z)^{\mathrm{T}}$:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} X_0 + U_X \Delta t + \Delta X \\ Y_0 + U_Y \Delta t + \Delta Y \\ Z_0 + U_Z \Delta t + Z_X \Delta X + Z_Y \Delta Y \end{pmatrix}. \tag{3.19}$$

The surface normal is then $\mathbf{n} = (Z_X, Z_Y, -1)^{\mathrm{T}}$. Velocity $\mathbf{U}$ could be further modeled using translation and rotation for rigid patches (cmp. Chap. 5) or using an affine transform for deformable patches. We omit this step here for simplicity.

Camera Model

We use a pinhole camera at world coordinate position $(s_x, s_y, 0)$, looking into Z-direction (cmp. Fig. 3.1)

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} X - s_x \\ Y - s_y \end{pmatrix} \tag{3.20}$$

where x and y are sensor coordinates and camera position space is sampled equidistantly using a camera grid.
A local neighborhood for parameter estimation is defined using neighboring pixels. However, we are interested in the local neighborhood given on the 3D surface patch by means of ΔX and ΔY. We therefore need to derive a relation between local 3D

coordinates ΔX and ΔY and local sensor coordinates Δx and Δy. To do so we solve the pinhole camera model (3.20) of the central camera, i.e., $s_x = s_y = 0$, for a 3D point (X, Y, Z) and substitute $Z = Z_{0,0}/(1 - Z_X \frac{x_0}{f} - Z_Y \frac{y_0}{f})$ with depth $Z_{0,0}$ at the principal point $(x, y) = (0, 0)$ (cmp. [23]). We get

$$\begin{pmatrix} X_{x_0,y_0} \\ Y_{x_0,y_0} \\ Z_{x_0,y_0} \end{pmatrix} = \frac{Z_{0,0}}{f} \begin{pmatrix} x_0 \\ y_0 \\ f \end{pmatrix} \frac{1}{1 - Z_X \frac{x_0}{f} - Z_Y \frac{y_0}{f}} \tag{3.21}$$

with (x_0, y_0) being the pixel coordinates where (X, Y, Z) is projected to. Assuming local coordinates ΔX constant in time, the total differential of $\mathbf{X}$ is given by

$$\Delta \mathbf{X} = \frac{\partial \mathbf{X}}{\partial x} \Delta x + \frac{\partial \mathbf{X}}{\partial y} \Delta y \; . \tag{3.22}$$

Inserting (3.21) in (3.22) yields

$$\begin{pmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{pmatrix} = \frac{Z_0}{f c_{x_0,y_0}} \begin{pmatrix} \left(1 - Z_Y \frac{y_0}{f}\right) \Delta x + Z_Y \frac{x_0}{f} \Delta y \\ \left(1 - Z_X \frac{x_0}{f}\right) \Delta y + Z_X \frac{y_0}{f} \Delta x \\ Z_X \Delta x + Z_Y \Delta y \end{pmatrix} \tag{3.23}$$

with

$$c_{x,y} = 1 - Z_X \frac{x}{f} - Z_Y \frac{y}{f} \tag{3.24}$$

and local neighborhood operators $\Delta x = x - x_0$ and $\Delta y = y - y_0$ on the image plane. In the case of a fronto-parallel surface, i.e., $Z_X = Z_Y = 0$, (3.23) boils down to

$$\begin{pmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{pmatrix} = \frac{Z_0}{f} \begin{pmatrix} \Delta x \\ \Delta y \\ 0 \end{pmatrix} \; . \tag{3.25}$$

There neighboring points in ΔX and ΔY directions are projected independent of each other and of the pixel position (x_0, y_0) onto the image plane. Figure 3.2 demonstrates the difference between projection of a fronto-parallel surface and a sloped surface.

Brightness Change Model

Using a camera grid means sampling (s_x, s_y)-space using several cameras each of which acquires an image sequence. We combine all of these sequences into one 5D data set sampling the continuous intensity function $I(x, y, s_x, s_y, t)$, i.e., intensity lives in a 5D space. We assume that the acquired brightness of a surface element is constant under camera translation, meaning we are looking at a Lambertian surface. In addition this brightness shall be constant in time, i.e., we need temporally constant illumination, which can be relaxed easily using the approach shown in Chap. 4. We

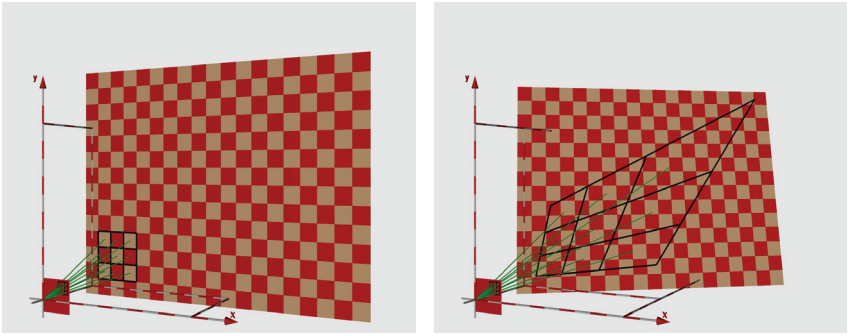


Figure 3.2: Pinhole projection of a patch. Left: fronto-parallel surface, right: Surface not fronto-parallel. We selected huge pixel sizes to demonstrate the effect.

see that brightness is constrained in one temporal and two spatial coordinates. In other words there is a 3D manifold in our 5D space in which I does not change. Thus the total differential $\mathrm{d}I$ vanishes in this manifold. The brightness model therefore is

$$\mathrm{d}I = I_x \mathrm{d}x + I_y \mathrm{d}y + I_{s_x} \mathrm{d}s_x + I_{s_y} \mathrm{d}s_y + I_t \mathrm{d}t = 0 \,. \tag{3.26}$$

Please note that all derivatives and differentials in this equation have physical units. Image coordinates x and y are given in pixel, i.e., the physical length of a sensor element on the camera chip, typically several μm. Camera coordinates s_x and s_y are given in camera to camera displacements and time t normalized to frames (depending on the measurement that is typically a fraction of seconds or minutes). This is important when interpreting parameters estimated from the model equations.

Combination of the 3 Models

In order to derive a single linear equation from the above models, we first project the moving surface element (3.19) to the sensor plane using a pinhole camera (3.20)

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} X_0 + U_X \Delta t + \Delta X - s_x \\ Y_0 + U_Y \Delta t + \Delta Y - s_y \end{pmatrix} \,. \tag{3.27}$$

Consequently we can calculate the differentials $\mathrm{d}x$ and $\mathrm{d}y$ for a *fixed surface location* (i.e., for constant ΔX and ΔY cmp. Sec. 3.3.1)

$$\begin{pmatrix} \mathrm{d}x \\ \mathrm{d}y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} (U_X - U_Z \frac{x}{f}) \mathrm{d}t - \mathrm{d}s_x \\ (U_Y - U_Z \frac{y}{f}) \mathrm{d}t - \mathrm{d}s_y \end{pmatrix} \,. \tag{3.28}$$

This equation is nonlinear in the sought 3D parameters. From (3.19) we know that

$$\begin{aligned} Z &= Z_0 + U_Z\Delta t + Z_X\Delta X + Z_Y\Delta Y \\ &= Z_0 + U_Z\Delta t + \tfrac{Z_0}{fc}\left(Z_X\Delta x + Z_Y\Delta y\right) \ . \end{aligned} \tag{3.29}$$

We linearize (3.29) via the assumption that $U_Z\Delta t$ is small compared to the overall depth $|Z_0| \gg |U_z\Delta t|$ and omit $U_Z\Delta t$ in the denominator. Further we can approximate

$$\frac{f}{Z} \approx \frac{f}{Z_0} - \frac{Z_X}{Z_0 c}\Delta x - \frac{Z_Y}{Z_0 c}\Delta y \tag{3.30}$$

assuming $|Z_X\Delta x/(fc) + Z_Y\Delta y/(fc)| \ll 1$. This is usually well fulfilled for not too large slopes, i.e., Z_X and Z_Y smaller than 5 corresponding to a slope of approximately 80° degrees, and normal angle lenses because then $c \approx 1$ and $\Delta x \ll f$. In our Sony XC-55 camera pixel size is 7.4μm and focal length $f = 12.5$mm, i.e., $n\Delta x/f \approx n0.6 \cdot 10^{-3}$ for the n^{th} neighbor pixel.
Plugging (3.30) into (3.28), using local coordinates $x = x_0 + \Delta x$ and $y = y_0 + \Delta y$, sorting by differentials and Δ-terms, and ignoring higher order Δ-terms we get

$$\begin{aligned} \mathrm{d}x &= \frac{f}{Z}\left[\left(U_X - \frac{x}{f}U_Z\right)\mathrm{d}t \ - \mathrm{d}s_x\right] \\ &= \frac{f}{Z_0}\left(U_X - \frac{x_0}{f}U_Z\right)\mathrm{d}t \\ &- \left[\frac{Z_X}{Z_0 c}\left(U_X - \frac{x_0}{f}U_Z\right) + \frac{U_Z}{Z_0}\right]\Delta x\mathrm{d}t \\ &- \left[\frac{Z_Y}{Z_0 c}\left(U_X - \frac{x_0}{f}U_Z\right)\right]\Delta y\mathrm{d}t \\ &- \frac{f}{Z_0}\mathrm{d}s_x + \frac{Z_X}{Z_0 c}\Delta x\mathrm{d}s_x + \frac{Z_Y}{Z_0 c}\Delta y\mathrm{d}s_x \end{aligned} \tag{3.31}$$

and

$$\begin{aligned} \mathrm{d}y &= \frac{f}{Z}\left[\left(U_Y - \frac{y}{f}U_Z\right)\mathrm{d}t \ - \mathrm{d}s_y\right] \\ &= \frac{f}{Z_0}\left(U_Y - \frac{y_0}{f}U_Z\right)\mathrm{d}t \\ &- \left[\frac{Z_X}{Z_0 c}\left(U_Y - \frac{y_0}{f}U_Z\right)\right]\Delta x\mathrm{d}t \\ &- \left[\frac{Z_Y}{Z_0 c}\left(U_Y - \frac{y_0}{f}U_Z\right) + \frac{U_Z}{Z_0}\right]\Delta y\mathrm{d}t \\ &- \frac{f}{Z_0}\mathrm{d}s_y + \frac{Z_X}{Z_0 c}\Delta x\mathrm{d}s_y + \frac{Z_Y}{Z_0 c}\Delta y\mathrm{d}s_y \ . \end{aligned} \tag{3.32}$$

We may now rename the mixed 3D parameters in (3.31) with

$$\begin{aligned} \mathrm{d}x &= u_x\mathrm{d}t + a_{11}\Delta x\mathrm{d}t + a_{12}\Delta y\mathrm{d}t \\ &+ \nu\mathrm{d}s_x + b_1\Delta x\mathrm{d}s_x + b_2\Delta y\mathrm{d}s_x \end{aligned} \tag{3.33}$$

and (3.32) with

$$\begin{aligned} \mathrm{d}y &= u_y\mathrm{d}t + a_{21}\Delta x\mathrm{d}t + a_{22}\Delta y\mathrm{d}t \\ &+ \nu\mathrm{d}s_y + b_1\Delta x\mathrm{d}s_y + b_2\Delta y\mathrm{d}s_y \ . \end{aligned} \tag{3.34}$$

Substituting (3.33) and (3.34) into the brightness change model (3.26) we get the sought affine optical flow-like model for dynamic surface reconstruction

$$\begin{aligned} &\nabla I\left[\begin{pmatrix} u_x\mathrm{d}t + \nu\mathrm{d}s_x \\ u_y\mathrm{d}t + \nu\mathrm{d}s_y \end{pmatrix} + \begin{pmatrix} a_{11}\mathrm{d}t + b_1\mathrm{d}s_x & a_{12}\mathrm{d}t + b_2\mathrm{d}s_x \\ a_{21}\mathrm{d}t + b_1\mathrm{d}s_y & a_{22}\mathrm{d}t + b_2\mathrm{d}s_y \end{pmatrix}\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}\right] \\ &+I_{s_x}\mathrm{d}s_x + I_{s_y}\mathrm{d}s_y + I_t\mathrm{d}t = 0 \ . \end{aligned} \tag{3.35}$$

We observe that the components of this model are known from the standard affine model (2.6) for optical flow, i.e., translations u_x, u_y, and affine components a_{11}, a_{12}, a_{21}, and a_{22} in the well known form [18]. Further there are disparity ν and its affine parameters b_1 and b_2 representing the projections of slopes Z_X and Z_Y. They may be grouped into two affine optical flow like submodels for depth estimation, i.e., one submodel for each camera displacement direction.
For a 2D camera grid both depth-submodels, i.e., the submodel with terms containing $\mathrm{d}s_x$ and the one containing terms with $\mathrm{d}s_y$ are present. When only a 1D camera grid is available, say oriented in s_x-direction, then terms containing $\mathrm{d}s_y$ can not be evaluated and are removed by setting $\mathrm{d}s_y = 0$. In the remainder of this work we call models operating on data of a 2D camera grid a *2D model*, the ones referring to a 1D camera grid *1D models*.

Local Equations for Flow Components

We can now derive the equations connecting flow components with surface patch parameters for the three "time"-like dimensions $\mathrm{d}t$, $\mathrm{d}s_x$ and $\mathrm{d}s_y$. Flow parameters coming with $\mathrm{d}t$ are

$$\begin{aligned} u_x &= \tfrac{f}{Z_0}\left(U_X - \tfrac{x_0}{f}U_Z\right) & u_y &= \tfrac{f}{Z_0}\left(U_Y - \tfrac{y_0}{f}U_Z\right) \\ a_{11} &= -\tfrac{Z_X}{Z_0 c}\left(U_X - \tfrac{x_0}{f}U_Z\right) - \tfrac{U_Z}{Z_0} & a_{21} &= -\tfrac{Z_X}{Z_0 c}\left(U_Y - \tfrac{y_0}{f}U_Z\right) \\ a_{12} &= -\tfrac{Z_Y}{Z_0 c}\left(U_X - \tfrac{x_0}{f}U_Z\right) & a_{22} &= -\tfrac{Z_Y}{Z_0 c}\left(U_Y - \tfrac{y_0}{f}U_Z\right) - \tfrac{U_Z}{Z_0} \end{aligned} \tag{3.36}$$

The ones coming with $\mathrm{d}s_x$ or $\mathrm{d}s_y$ are

$$\nu = \tfrac{f}{Z_0}\ , \quad b_1 = \tfrac{Z_X}{Z_0 c} \ \text{ and } \ b_2 = \tfrac{Z_Y}{Z_0 c}\ . \tag{3.37}$$

All patch parameters are evaluated at a fixed pixel position and point in time.

3.3.2 Parameter Estimation

In this section we present a way to solve for the parameters using two standard least squares estimators. We will start by estimating affine flow parameter vectors based on the structure tensor method (cmp. Sec. 2.3.2). Linear combinations of the solution vectors are used to disentangle depth and motion parameters. When using a 2D camera grid, estimates of depth and surface normals are combined according their covariance matrices. Finally motion parameters are estimated from affine motion parameters and 3D structure via least squares.

Estimation via Structure Tensor Method

In this parameter estimation method a model with linear parameters p_j has to be given in the form $\mathbf{d}^{\mathrm{T}}\mathbf{p} = 0$ with data depending vector $\mathbf{d}$ and parameter vector $\mathbf{p}$ (cmp. Sec. 2.3.2). Equation 3.35 is of this form when identifying

$$\begin{aligned} \mathbf{d} = \;& (I_x,\, I_y,\, I_x\Delta x,\, I_x\Delta y,\, I_y\Delta x,\, I_y\Delta y,\, I_{s_x},\, I_{s_y},\, I_t)^{\mathrm{T}} \\ \mathbf{p} = \;& (u_x\mathrm{d}t + \nu\mathrm{d}s_x\,,\, u_y\mathrm{d}t + \nu\mathrm{d}s_y\,,\, a_{11}\mathrm{d}t + b_1\mathrm{d}s_x\,,\, a_{12}\mathrm{d}t + b_2\mathrm{d}s_x\,, \\ & a_{21}\mathrm{d}t + b_1\mathrm{d}s_y\,,\, a_{22}\mathrm{d}t + b_2\mathrm{d}s_y\,,\, \mathrm{d}s_x\,,\, \mathrm{d}s_y\,,\, \mathrm{d}t\,)^{\mathrm{T}}\;. \end{aligned} \tag{3.38}$$

The parameters for using a 1D camera grid oriented in s_x-direction are obtained by setting $\mathrm{d}s_y = 0$

$$\begin{aligned} \mathbf{d} = \;& (I_x,\, I_y,\, I_x\Delta x,\, I_x\Delta y,\, I_y\Delta x,\, I_y\Delta y,\, I_{s_x},\, I_t)^{\mathrm{T}} \\ \mathbf{p} = \;& (u_x\mathrm{d}t + \nu\mathrm{d}s_x\,,\, u_y\mathrm{d}t\,,\, a_{11}\mathrm{d}t + b_1\mathrm{d}s_x\,,\, a_{12}\mathrm{d}t + b_2\mathrm{d}s_x\,, \\ & a_{21}\mathrm{d}t\,,\, a_{22}\mathrm{d}t\,,\, \mathrm{d}s_x\,,\, \mathrm{d}t\,)^{\mathrm{T}}\;. \end{aligned} \tag{3.39}$$

For standard optical flow the null-space is 1D, here it is 2D or 3D, depending on the model used. Therefore solution vectors are combined to find a minimum solution vector for the flow in the corresponding direction, i.e., $\mathrm{d}t$, $\mathrm{d}s_x$ or $\mathrm{d}s_y$.

Solving for Flow Components

For our models the solution space of (2.17) is spanned by eigenvectors $\tilde{\mathbf{p}}_\mathbf{i}$ of $\mathbf{J}$ (where $i \in \{1, 2\}$ using a 1D camera grid or $i \in \{1, 2, 3\}$ using a 2D camera grid). The components $(\tilde{p}_i)_j$ of these eigenvectors always represent two flow components (cmp. (3.38)), one depending on time and coming together with $\mathrm{d}t$ and the other depending on a camera displacement direction ($\mathrm{d}s_x$ or $\mathrm{d}s_y$).

We solve for time depending flow parameters (u_x, u_y, a_{11}, a_{12}, a_{21}, a_{22}) by linearly combining solution vectors $\tilde{\mathbf{p}}_\mathbf{i}$ such that $\mathrm{d}t = 1$ and $\mathrm{d}s_x = \mathrm{d}s_y = 0$, where $\mathrm{d}s_x$, $\mathrm{d}s_y$,

and $\mathrm{d}t$ are components 9 to 11, respectively. Components 1 to 8 of the resulting vector then correspond to u_x, u_y, a_{11}, a_{12}, a_{21}, and a_{22} respectively.

When using a 1D camera grid $\mathrm{d}s_y$ already vanishes and we only need to linearly combine solution vectors $\tilde{\mathbf{p}}_\mathbf{i}$ such that $\mathrm{d}t = 0$ and $\mathrm{d}s_x = 1$. Depth related flow parameters ν, b_1, b_2 then correspond to components 1, 3, and 4, respectively.

When using a 2D camera grid, 2 linear independent eigenvectors with $\mathrm{d}t = 0$ can be derived. One with $\mathrm{d}s_x = 1$ and $\mathrm{d}s_y = 0$, and the other with $\mathrm{d}s_x = 0$ and $\mathrm{d}s_y = 1$. Thus we could estimate v, b_1, and b_2 from either of the 2 eigenvectors, but unfortunately the two estimates for each flow parameter are neither identical, nor independent. This becomes clear considering the following example. Assume our cameras are looking at a striped pattern oriented in y-direction, i.e., parallel to s_y. Then a 1D camera grid shifting along x-direction will easily determine depth from disparity ν, but a camera grid shifting along y-direction suffers from the aperture problem and can not determine depth. A 2D camera grid also can only resolve depth from its s_x-movement, but not from s_y-movement. In this example the two estimates are independent. Now think of a striped pattern not oriented along x- or y-direction. The two ν-estimates are now coupled and we need to combine them according to their error covariance matrix $\mathbf{C}$. We do so by calculating $\mathbf{C}$ following Nestares and Fleet [46], and rotating the representation of s_x-s_y-space such that $\mathbf{C}$ becomes diagonal (cmp. Fig. 3.3a). Then the resulting single estimate for ν is a sum of the independent estimates weighted by the inverse of their individual errors

$$\nu = \frac{\nu_1 e_1 + \nu_2 e_2}{e_1 + e_2} \tag{3.40}$$

where ν_1 and ν_2 are the two independent disparity estimates and errors $e_1 = C_{11}^{-0.5}$ and $e_2 = C_{22}^{-0.5}$ with diagonal entries C_{11} and C_{22} of the diagonalized covariance matrix $\mathbf{C}$.

Scharr [55] demonstrates the advantages of this approach. On the one hand this approach solves the aperture problem in one camera displacement direction. In this case the error of one estimate becomes infinity and is therefore neglected. On the other hand the estimation accuracy slightly improves, because two estimates are used instead of one. Figure 3.3b shows a histogram of disparities ν_x, ν_y and ν estimated from a synthetic, sinusoidal test pattern with 10% Gaussian noise added (true value $\nu = 1$). Although there is no aperture problem and the same data has been used for all 3 estimates the histogram of ν (solid line) has less variance than the other two curves.

Disentangling Surface Patch Parameters

Disentangling the flow parameters into parameters describing the local surface patch means solving the nonlinear system of (3.36) and (3.37). We have 6 unknowns (U_X, U_Y, U_Z, Z_0, Z_X, Z_Y) and one equation per estimated flow component, i.e., 12 constraints. We have an overdetermined system of equations and consequently some

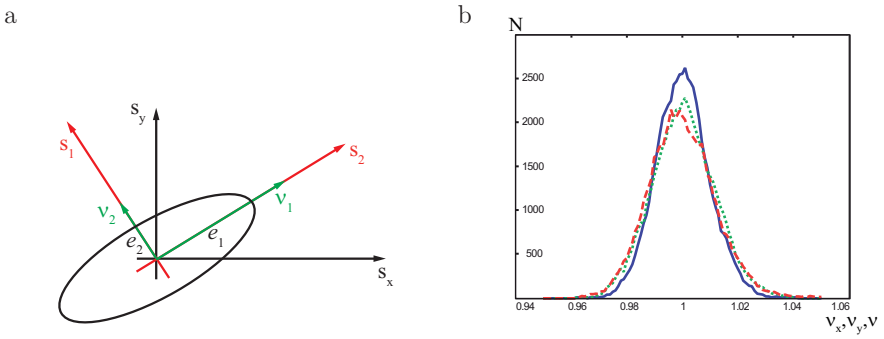


Figure 3.3: Estimation of multiple occurring parameters from [55]. a: Rotation of coordinate system by error covariance. ν_x and ν_y are the estimates for camera displacement in s_x- and s_y-direction, respectively. b: Accuracy gain by combination of the estimates.

choices how to solve it.

The first step is solving for depth and normals. From the estimates for the flow components ν, b_1 and b_2 (cmp. Sec. 3.3.2) and (3.37) and (3.24) we first derive Z_0 using $Z_0 = f/\nu$ and then solve for Z_X, Z_Y, and c. Focal length f has to be known, e.g., from a calibration step.

The second step is solving for motion components. To do so we substitute (3.37) into (3.36) and get

$$\begin{aligned} u_x &= -\nu \left(U_X - \tfrac{x_0}{f} U_Z\right) & u_y &= -\nu \left(U_Y - \tfrac{y_0}{f} U_Z\right) \\ a_{11} &= -b_1 \left(U_X - \tfrac{x_0}{f} U_Z\right) + \tfrac{\nu U_Z}{f} & a_{21} &= -b_1 \left(U_Y - \tfrac{y_0}{f} U_Z\right) \\ a_{12} &= -b_2 \left(U_X - \tfrac{x_0}{f} U_Z\right) & a_{22} &= -b_2 \left(U_Y - \tfrac{y_0}{f} U_Z\right) + \tfrac{\nu U_Z}{f} \end{aligned} \tag{3.41}$$

Equation 3.41 may be solved using any linear estimator, e.g., least squares or total least squares as proposed in Sec. 2.3. We use standard least squares.

3.4 Experiments

In this section we compare performance of *Scene Flow*, *Range Flow* and the 1D and 2D camera grid model for simultaneous estimation of motion, depth and normals (*Affine Model*). First we analyze estimation of depth and normals using the *Affine Model* proposed in Sec. 3.3. Estimates of 3D structure and (affine) optical flow are then used as input data for the motion estimation techniques. We show experiments on sinusoidal patterns for evaluation of systematic errors. A rendered sequence showing a moving cube is used to test performance under more realistic conditions

and with ground truth available. Estimation results on plant leafs demonstrate the performance of the models on real data.

3.4.1 Sinusoidal Pattern

In the first experiment we compare the influence of the additional estimation of normals to the other parameters in the 1D case (via (3.39)) using synthetic data. In a second experiment we test the benefit of a 2D camera grid (3.38) compared to the 1D camera grid (3.39). The estimated 3D structure is then used to compare the motion estimation techniques.

3D Structure

Scharr [55] investigated properties of basic elements of the *Affine Model* with respect to filter choices, linearization effects, and noise influence. We use the results here by choosing 5 cameras, i.e., sampling points in s_x, for the 1D grid and 5×5 cameras for the 2D grid, because the s_x- and s_y-derivatives can be calculated with low systematic error with the filter-sets proposed in [54] (cmp. Sec. 2.2). Noise increases the absolute error of the estimated components of the parameter vectors in the structure tensor. Thus image sequences should be acquired such that the estimable signal is maximal. Especially absolute values of displacements, i.e., optical flow components u_x and u_y as well as disparity v, should be as large as possible. For reasonable camera distances in the grid disparity is much larger than 1 pixel. Therefore we define a working depth with an integer disparity ν_0 and preshift our data in s_x- and s_y-directions such that this disparity becomes zero. Then we can estimate disparity reliably in a depth interval around the working depth, where extremal disparities should not exceed ± 1 pixel. We select our acquisition configuration accordingly. Please note that this restriction is not inherent to the model but is due to the use of the structure tensor and can be relaxed using warping or multi-scale estimators (see e.g., [14, 49]).

As synthetic data we calculate sinusoidal patterns with the *Sequence Generator* presented in App. B. Two example intensity images and the corresponding depth maps can be found in Fig. 3.4.

1D Grid with and without Normals In order to show the influence of the additional estimation of surface normals in the 1D camera grid model, we generate noise free, synthetic sinusoidal images (see above) with varying Z_X. The other parameters were set to $\lambda_x{=}\lambda_y{=}\lambda$, $\alpha{=}0°$, $Z_0{=}100$ mm, $f{=}10$mm, $Z_Y{=}0$ and width of pixels is $7.4\,\mu$m. From these sequences with and without additive Gaussian noise we estimate local depth Z for both 1D grid models, and Z_X for the model additionally handling surface normals. The respective plots can be found in Fig. 3.5 and Fig. 3.6. There Z_X is given in degrees, i.e., $Z_X[°]{=}\arctan(Z_X)$. Looking at Fig. 3.5 we observe that left and right plots are almost identical. Thus the model with the normals performs as stable as the simpler model without normals. This coincides with

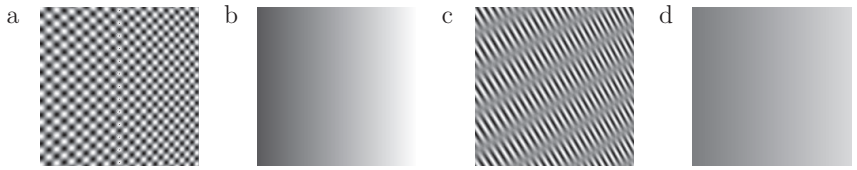


Figure 3.4: Sinusoidal images for the central camera s_x=s_y=0, origin (X_0, Y_0) in the image center, no motion, Z_0=100 mm, f=10 mm, Z_Y=0 using a: λ_x=λ_y=16 pixel=0.1184 mm, α=0°, Z_X=1.73≙60° and c: λ_x=8 pixel, λ_y=80 pixel, α=30°, Z_X=1≙45°. b and d are the corresponding ground truth depth maps.

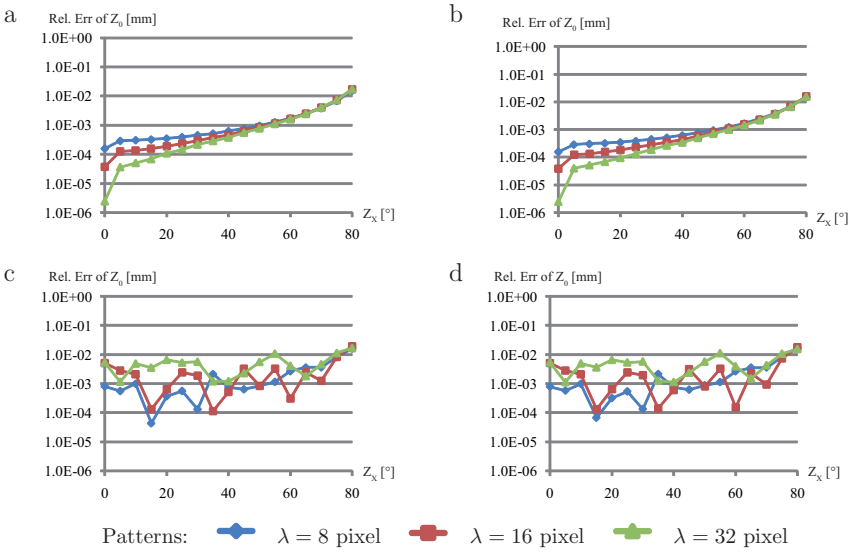


Figure 3.5: Comparison of 1D camera grid models. Relative errors from sinusoidal images. Left: model without normals, right: model with normals. a and b: noise free data, c and d: 2.5% Gaussian noise added.

experiments in Sec. 2.3. There we have seen that an affine optical flow model, i.e., the model with normals, may outperform a standard optical flow model, i.e., the model without normals. Here we are looking at disparity estimates and very small affine motions ($b_1 = 0.0001$ and $\nu \approx 0.27$, cmp. (3.37)). The improvements due to the *Affine Model* are therefore not visible. In the case of noise free synthetic data, estimates are better for larger wavelengths. For noisy data the influence of the wavelength is negligible. Figure 3.6 demonstrates that the absolute error of the estimated Z_X is in the range of 0.001° to 0.01° in the noise free case and in the range of 0.01° to 10° when 2.5% Gaussian noise is added to the data.

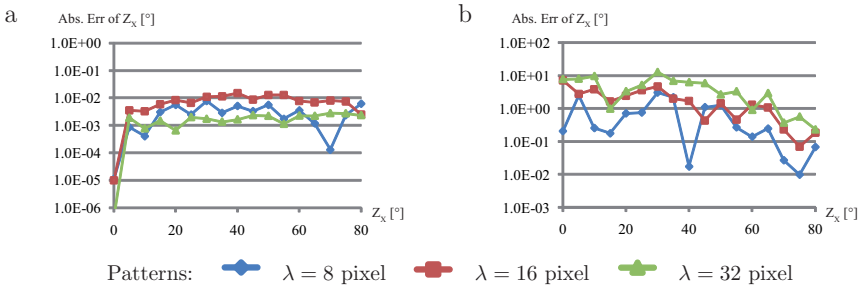


Figure 3.6: 1D camera grid model with normals. Absolute errors of normals calculated from sinusoidal images. a: no noise, b: 2.5% Gaussian noise.

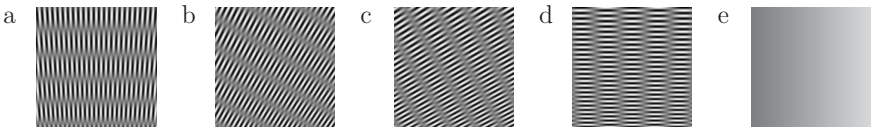


Figure 3.7: a–d: Input patterns with different orientations ($\alpha \in [0, 30, 60, 90]^\circ$ degrees), and e: gray-value coded depth map.

1D Grid versus 2D Grid The benefit of a second camera-shift direction is expected to lie in the reduction of a potential aperture problem. This means, looking at a strongly oriented pattern, a 1D camera grid can not resolve, e.g., depth if the pattern is oriented along the s_x-axis. A 2D camera grid should not encounter this problem as depth can be estimated easily in the camera direction normal to the pattern orientation. To test this behavior we generate noisy (σ_n=2.5%) and noise free sinusoidal patterns with λ_x=8 pixel, λ_y=80 pixel and varying orientation angle α. The other parameters are fixed to be Z_0=100 mm, f=10 mm, Z_Y=1$\hat{=}$45°, Z_X=0. Figure 3.7 shows the four different surface patterns and the gray value coded depth map. From these sequences we estimate local depth Z, and surface slope Z_Y and tabulate their standard deviation (see Tab. 3.1).

For the standard deviations we observe that the 1D grid model performs worse for rotated patterns with $\alpha = 30^\circ$ and $\alpha = 60^\circ$ in the noise free case. The estimation of Z_Y is more sensitive (its a first order derivative), therefore the errors of Z_Y are even higher. In the noisy data case the aperture problem becomes more clearly visible in the 1D model: estimation of Z_0 gets worse for increasing α, Z_Y is not estimable. The 2D model still gives Z_0 reliably, Z_Y has a high but limited variance.

3D Motion

For systematic error analysis of motion estimates and comparison of the different motion models, we generate image sequences showing a moving surface patch with

Table 3.1: Comparison 1D and 2D camera grid. Standard deviations calculated from sinusoidal images. Standard deviations which are significantly higher for the 1D model than for the 2D model are marked red.

α	noise	Z_0: std. dev. [mm]		Z_Y: std. dev. [°]	
		1D	2D	1D	2D
0°	0%	0.104	0.104	0.35	0.29
30°	0%	0.119	0.108	5.40	0.23
60°	0%	0.140	0.108	5.37	0.11
90°	0%	0.128	0.104	0.001	0.01
0°	2.5%	0.44	0.43	28.1	26.9
30°	2.5%	2.07	0.41	70.1	24.9
60°	2.5%	2.80	0.31	73.3	17.9
90°	2.5%	3.71	0.32	76.7	19.1

a sinusoidal pattern. The parameters of the surface patch are: $Z_0 = 100$ mm, $Z_X = 0.6\hat{=}30.96°$, $Z_Y = 0.5\hat{=}26.56°$, $U_X = 0.00733$ mm/frame, $U_Y = -0.00367$ mm/frame and $U_Z = 0.06$ mm/frame. As we use a simple total least squares estimator we kept the optical flow values well below 1 pixel/frame to avoid errors coming from the estimator. The synthetic sensor contains 501×501 pixels with width $4.4\,\mu$m. The focal length of the projective camera is set to $f = 12$ mm.
We compare the performance of the models for surface patches with different inclinations and velocities. Furthermore we investigate the effect of additive Gaussian noise with increasing standard deviation σ_n, and of two baselines with $d_1 = 0.01$ mm (near baseline) and $d_2 = 0.5$ mm (wide baseline). Near baseline needs no preprocessing of the input sequences as the optical flow between camera positions is smaller than 1 pixel. For wide baseline we shift the input sequences in camera displacement direction towards the central camera to get the smallest image motion for the central pixel (here we have a preshift of 14 pixel). Due to this preprocessing the effective image size is cropped to 301×301 pixel. For our analysis we compare the average angular error (App. A) for 3D motion estimates over all pixel N at a minimum distance of 70 pixel from the nearest image border. To reduce systematic errors we use the optimized $5 \times 5 \times 5$ filter sets presented in [54] and a 3D structure tensor weighting matrix $\mathbf{W}$ with spatio-temporal size of 65×65 pixel and 5 frames and standard deviation of $\sigma_W = 19$ in space and $\sigma_W = 1$ in time direction. Figure 3.8 shows average angular errors for increasing U_Z, Z_X, and σ_n for both near (left) and wide (right) baselines. For increasing depth motion U_Z and near baseline, errors of the *Affine Model* and *Scene Flow* are up to one order of magnitude lower than errors of *Range Flow*. For wide baseline only errors of the *Affine Model* increase. Errors of the other models stay in the same range as for near baseline. In case of large

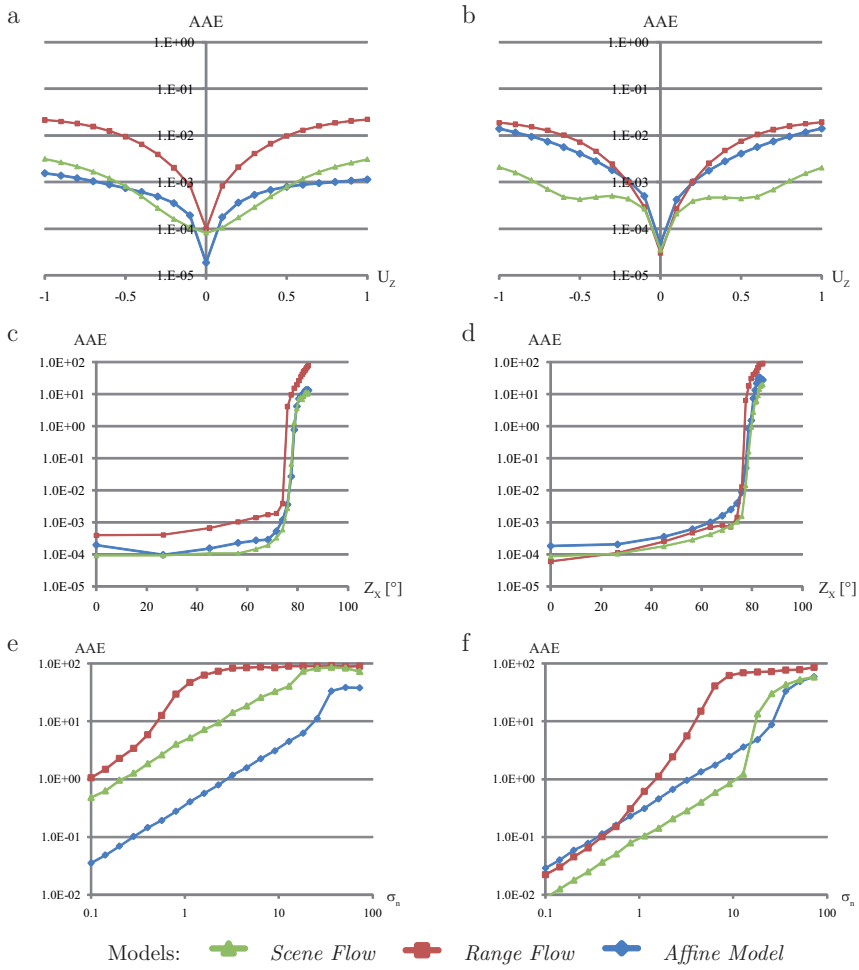


Figure 3.8: Average angular error versus U_Z (top), Z_X (central) and σ_n (bottom) for near (left) and wide (right) baseline.

slopes (up to approx. $Z_X = 75°$ degrees) a near baseline setup yields more reliable estimates compared to wide baseline setups using *Scene Flow* or the *Affine Model*. For increasing noise and near baseline the *Affine Model* outperforms *Scene Flow* and *Range Flow* by one order of magnitude. Using a wide baseline, errors of *Scene Flow* and *Range Flow* are reduced by one order of magnitude. The errors for the *Affine Model* are similar for both baselines.

For near baseline the *Affine Model* and *Scene Flow* perform significantly better than *Range Flow*. For wide baseline *Scene Flow* yields estimates with lowest errors

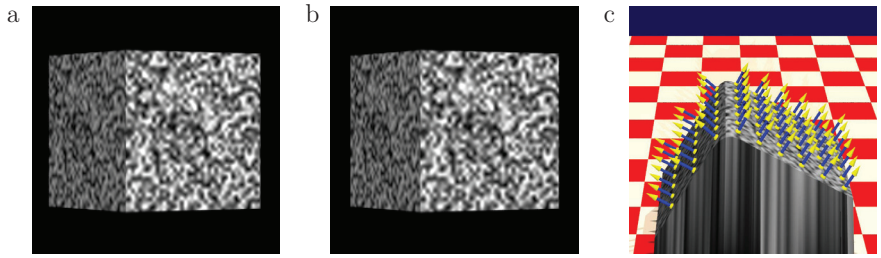


Figure 3.9: Synthetic cube sequence. a and b: Images taken from first and last camera. c: 3D reconstruction with surface normals.

in all cases. Compared to near baseline, the *Affine Model* yields worse results for increasing U_Z. *Range Flow* estimates are better in all cases for wide baseline than for near baseline.

3.4.2 Synthetic Cube

In order to test estimation of 3D structure, surface normals and motion on more realistic data with ground truth available, we rendered a cube covered with a constant noise pattern using POV-Ray [15]. Illumination remains constant for all camera positions and time frames to fulfill the brightness constancy constraint. We use a 3D structure tensor weighting matrix $\mathbf{W}$ with spatio-temporal size of 65×65 pixel and 5 frames and standard deviation of $\sigma_W = 19$ in space and $\sigma_W = 1$ in time direction. The size of the sequence is 640×480 pixel and we render data for 5 cameras with a displacement of 5 mm. The baseline results in a preshift of 14 pixel. The cube is rotated by 60° degrees and moves with $U_X = -0.2$ mm/frame, $U_Y = 0$ mm/frame and $U_Z = -2$ mm/frame. Figure 3.9 shows two images of the cube sequence and estimates of depth and normals rendered with POV-Ray [15]. The two images in Figs. 3.9a and b are from the first and last camera for $t = 0$. The images are preshifted, so that disparity is smaller than 1 pixel per camera displacement. Figure 3.9c demonstrates high accuracy of depth and surface slopes estimates. The border of the cube pointing towards the camera grid is smoothed, because of the large smoothing filter $\mathbf{W}$ in the standard structure tensor method. A robust estimator like the one presented in Sec. 2.3.4 could reduce this smoothing effect. Figures 3.10a and b show two time frames of the cube sequence from the central camera. The ground truth motion is depicted in Fig. 3.10c. Velocity and depth estimates of the proposed models are compared in Figs. 3.10d-f. For comparison of the models, errors are amplified by 100 in U_X-and U_Y-directions and 10 in U_Z-direction. *Scene Flow* estimates the motion best. Near the right border of the cube velocity U_Z is underestimated. Velocity estimates of *Range Flow* are distorted at all borders of the cube. Moreover estimates on the left side of the cube are less accurate than *Scene Flow*. The *Affine Model* yields more smooth estimates than *Range Flow*, but at the borders and on the left

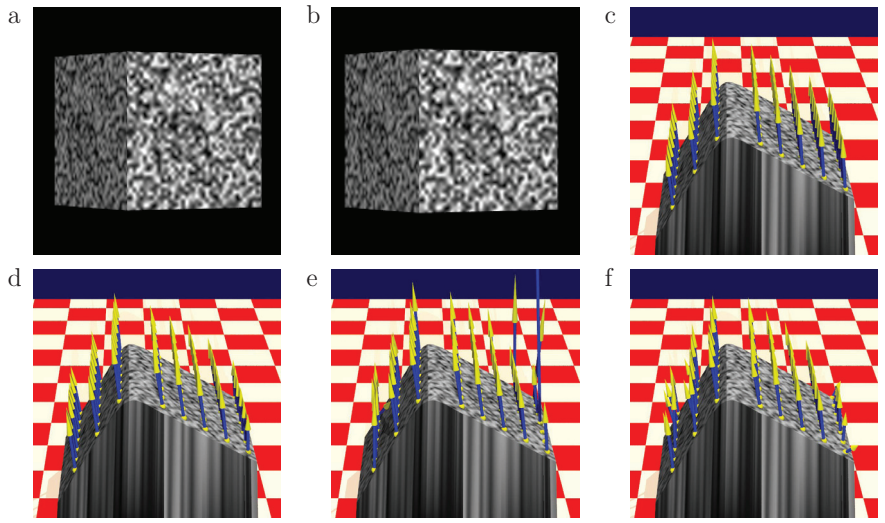


Figure 3.10: Motion estimation of cube moving towards camera. a and b: First and last image taken with central camera. c: Ground truth motion. d–f: Scaled motion estimates with amplified errors for the models *Scene Flow*, *Range Flow* and the *Affine Model* are depicted, respectively.

side worse results than *Scene Flow*. A robust estimator may improve estimation results at borders. *Range Flow* is more sensible to borders, because it incorporates derivatives of estimated depth sequences, which already suffer from border problems.

3.4.3 Plant Sequences

In this work the main focus is on plant leaf motion estimation. We show experiments on two different plants, namely *Kalanchoe* and *Castor Oil Plant* (Ricinus).

Kalanchoe

The input sequence contains images of 49 camera positions, i.e., a 7×7 camera grid with 0.5 mm camera displacement. Figure 3.11a and b show an image of the central camera and depth estimates rendered by POV-Ray [15]. The rendered view demonstrates that positions and surfaces of leafs are well reconstructed. Large smoothing filters ($\mathbf{W}$ has spatial size 128×128 with $\sigma_{\mathbf{W}} = 37$) have been used, because structures on the leafs are rare. This leads to strong smoothing of the 3D reconstruction. Structure of small leafs in the center of the plant cannot be estimated. Figure 3.11c and d show a close-up view of the large leaf positioned in the lower part in Fig. 3.11a. The rendered view and estimated surface slopes are depicted in

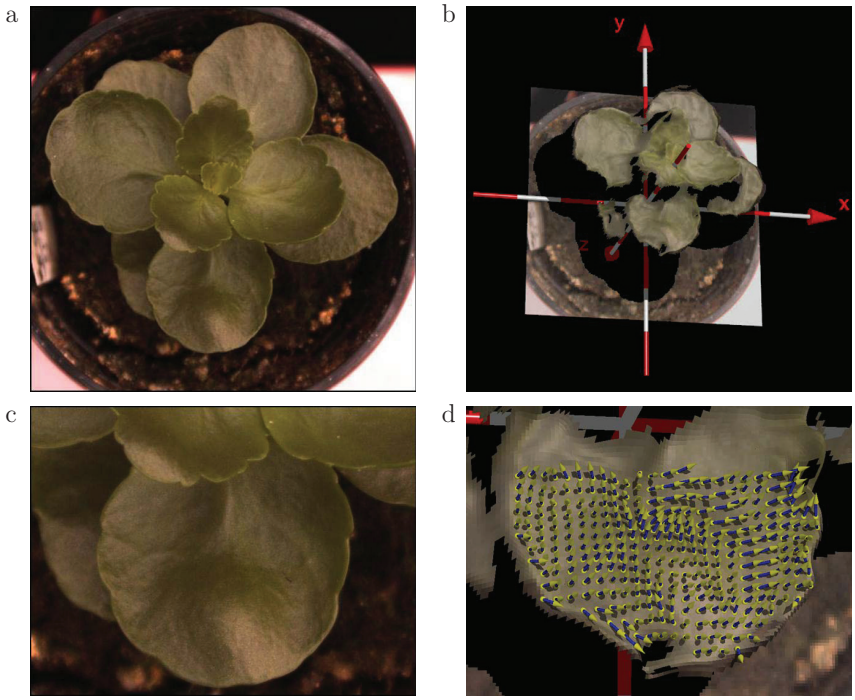


Figure 3.11: Reconstruction of plant using 49 camera positions. a: central image of input sequence, b: rendered depth estimates, c: close-up view on leaf and d: estimated depth with surface slopes

Fig. 3.11d. Estimated normals fit to the recovered structure of the leaf. The bump in the lower part of the leaf is clearly visible.

Castor Oil Plant

We are interested in motion estimation of plant leafs. Figure 3.12a shows one image of a *Castor Oil* plant leaf under lab conditions. A close up view of the region marked by the white rectangle is depicted in Fig. 3.12b. The scene is illuminated by directed infrared light emitting diodes from the top right causing shadows on the leaf of interest. A camera grid with 9 cameras is replaced by a camera on a moving stage. This is feasible because movement of the plant is negligible during acquisition time of approx. 30 seconds. Furthermore this virtual camera grid allows very small camera displacements, which are not realizable with real cameras. Here the camera displacement is 4 mm. Depth of the leaf is around $Z_0 = 250$ mm and its surface size is approx. 15×35 mm. For motion estimation, a sequence of nine frames with a sampling rate of one frame per 2 minutes was taken, i.e., acquisition time for all

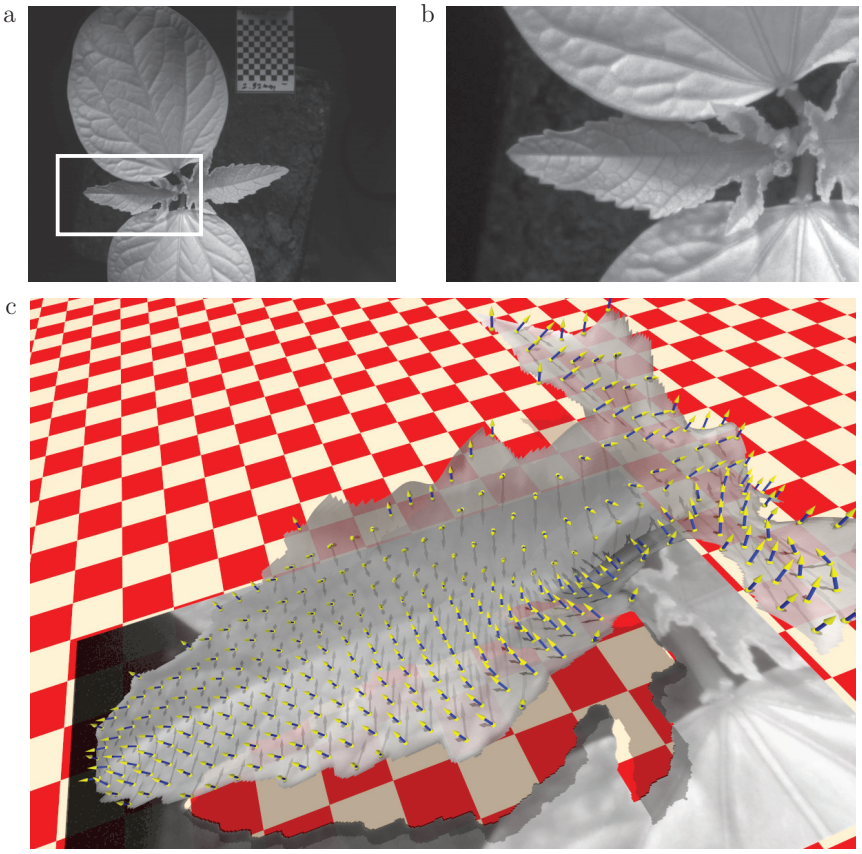


Figure 3.12: *Castor Oil Plant* sequence. a: Full size image taken with central camera. b: Close up view of plant leaf denoted by white rectangle in a. c: 3D reconstruction and surface slopes of plant leaf.

images was 16 minutes. We use a 4D structure tensor weighting matrix $\mathbf{W}$ with size of 77×77 pixel and 5 images in time and camera displacement direction and standard deviation of $\sigma_W = 23$ in space and $\sigma_W = 1$ in time and camera displacement direction. The unfolding leaf rotates around the node where it is attached to the stem. This results in a visible motion towards the camera and to the right while the shadow area caused by the top leaf decreases. Depth and normal estimates are depicted in Fig. 3.12c. Depth estimates seem to recover the true structure quite well. Estimated normals clearly show the curves of the leafs, but seem to be distorted at borders, especially at the right end of the leaf. There inclination of the leaf is very high and structures are too small to be estimated reliably. Figure 3.13 depicts three

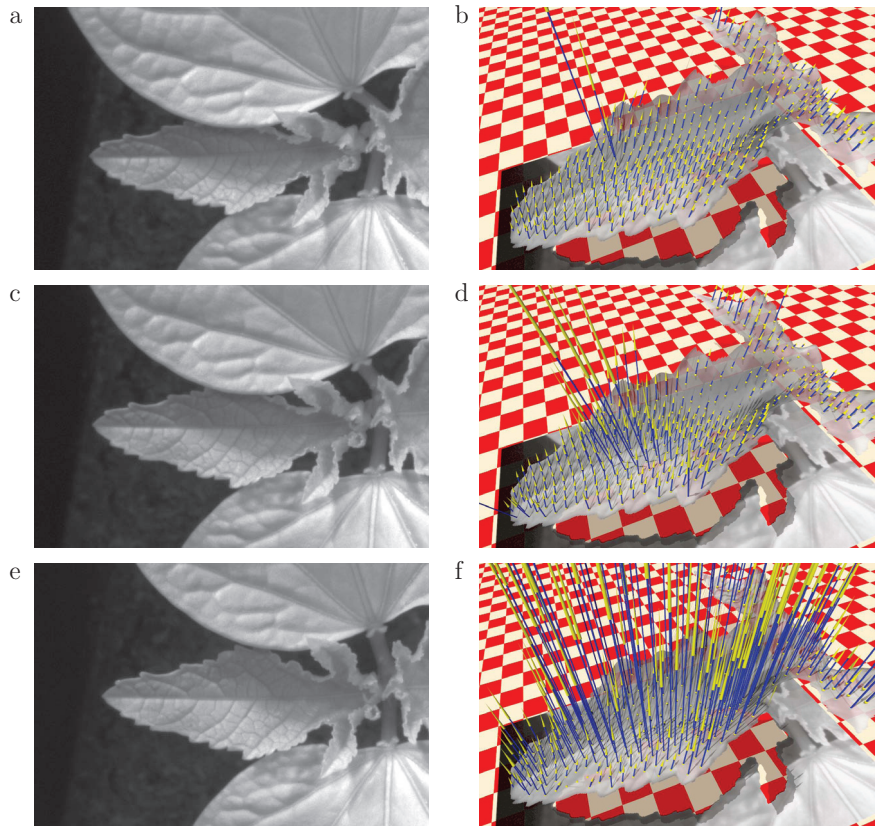


Figure 3.13: *Castor Oil Plant* sequence. Left: Three images taken with central camera. Right: Motion estimates for models *Scene Flow* (b), *Range Flow* (d) and the *Affine Model* (f) are depicted, respectively.

time frames of the *Castor Oil Plant* sequence (left side) and motion estimates of the three models (right side). *Scene Flow* best estimates motion of the leaf. In the upper part of the leaf, where brightness changes occur estimates are distorted. As expected for an almost rigid motion, which can be assumed because of the relatively high temporal resolution, we obtain a smoothly varying vector field. *Range Flow* suffers more from brightness changes and estimates at borders are heavily distorted. This effect coincides with *Range Flow* estimation results near borders in the cube sequence (Sec. 3.4.2). The *Affine Model* yields heavily corrupted estimates. This effect does not coincide with the results on the cube sequence. The main causes of the poor performance of the *Affine Model* may depend on its sensitivity to brightness changes and its surface patch model, which handles only translational motion.

Table 3.2: Comparison of motion estimation approaches

	Scene Flow	*Range Flow*	*Affine Model*
estimated parameters	3D motion	3D motion	3D structure & motion
input for intensity data	all cameras	central camera	all cameras
input for depth data	central time point	all time points	all time points

3.5 Conclusions

In this chapter we described three optical flow based approaches for 3D motion estimation. The *Affine Model* is also capable of structure estimation. The three models use different sources and models for motion estimation. A comparison of these sources is depicted in Tab. 3.2. The main goal of all methods is to use 2D optical flow to estimate 3D motion. Whereas pure 3D motion in X- and Y-direction may be easily explained by optical flow estimates, motion estimation in Z-direction is not uniquely defined by 2D optical flow estimates only. Therefore the 3D motion estimation approaches incorporate additional constraints to solve for 3D motion. *Scene Flow* combines optical flow estimates in all cameras with depth data for $t = 0$ to estimate 3D motion. On the other hand *Range Flow* combines optical flow in the central camera with the change in depth data seen in this camera. The *Affine Model* uses information of optical flow in the spatial neighborhood (e.g., divergence) to solve for 3D motion.
However, *Scene Flow* and *Range Flow* use two separate estimators for depth and flow estimation, i.e., depth data is estimated independent from optical flow. The *Affine Model* combines both estimation processes in one parameter estimation and solves then for motion in time or camera displacement by combining solution vectors of this parameter estimation. The benefit of this estimation approach is that all available input data influences estimation of the parameters, whereas the other approaches use only subsets of the data (cmp. Tab. 3.2). The general structure of *Scene Flow* and the *Affine Model* are quite similar. First optical flow components and depth are derived. In a second step these parameters are explained by 3D parameters. The difference is that *Scene Flow* incorporates information of translational optical flow, i.e., two parameters, of different cameras, whereas the *Affine Model* is based on more than two optical flow parameters in the central camera.
Synthetic experiments on sinusoidal patterns, the synthetic cube sequence and on plant leaf sequences showed that estimation of depth and slopes with the *Affine Model* yields reliable results for 1D and 2D camera grids. In the experiments we

compared the 1D model without normals from [55] to the full model. All parameters can be estimated as well or better with the full model. The additional parameters do not lead to instabilities. In addition the 2D model is more robust with respect to the aperture problem.
Scene Flow yields best 3D motion estimates for wide baseline setups. Sinusoidal experiments showed that for small baselines estimation accuracy of the *Affine Model* is comparable to the accuracy of *Scene Flow*. In case of noisy data and small baselines the *Affine Model* clearly yields best results. Estimates using *Range Flow* have highest accuracy only for special cases, e.g., fronto-parallel surface or no depth motion for wide baseline setups.
Experiments on the synthetic cube sequence and on plant sequences confirmed results from the sinusoidal sequences. In addition experiments on the plant leaf sequence showed that *Scene Flow* is quite robust to brightness changes, although it is based on the brightness constancy constraint. Estimates of *Range Flow* and especially of the *Affine Model* are highly corrupted by brightness changes. The *Affine Model* seems to be more sensitive to the rotational motion of the leaf. This may come from model assumptions of translating surface patches. We conclude that *Scene Flow* works best on real data and is less sensitive in regions where the brightness constancy constraint is not fulfilled. However, in order to get more reliable estimates we have to derive a new approach to handle brightness changes in the data which do not come from motion. This will be addressed in the following chapter.

Chapter 4

Modeling Brightness Changes

Motion estimation approaches as presented in the previous chapter yield good results for objects under homogeneous, diffuse illumination. Problems occur for directed, inhomogeneous illumination, because the brightness constancy constraint (2.3) is not sufficiently well satisfied anymore. A setup for *Range Flow* may include an illumination independent range sensor, such as a laser range sensor. This means the range constraint is still fulfilled. However, range data estimated by a structure from motion approach may also be corrupted. In this chapter we present different approaches to handle illumination changes. First we focus on prefiltering the intensity data, in order to reduce brightness changes coming from changing illumination. Then we will present two brightness constraints, which are more robust to brightness changes compared to the brightness constancy constraint (see (2.3)), and one novel constraint, which models brightness changes based on reflectance physics. In the second part of this chapter we will integrate these constraints in *Range Flow* and in the *Affine Model* and show experimental results for different combinations of prefilters and constraints. We do not explicitly test brightness handling approaches with *Scene Flow* because, as showed in the previous chapter, *Scene Flow* is basically a specific estimation method and therefore integration would not yield additional insights. Finally the performance of the three models is compared on synthetic sequences with illumination changes and we show improvements on the plant leaf sequence.

4.1 Prefiltering

Prefiltering is a well-known technique for illumination change suppression, making image data more or less illumination invariant.

Temporal and/or spatial highpass filtering approximately eliminates slow or low-frequent brightness changes in the data. However, faster illumination changes in both spatial and temporal domain still remain present in the data. In our experiments,

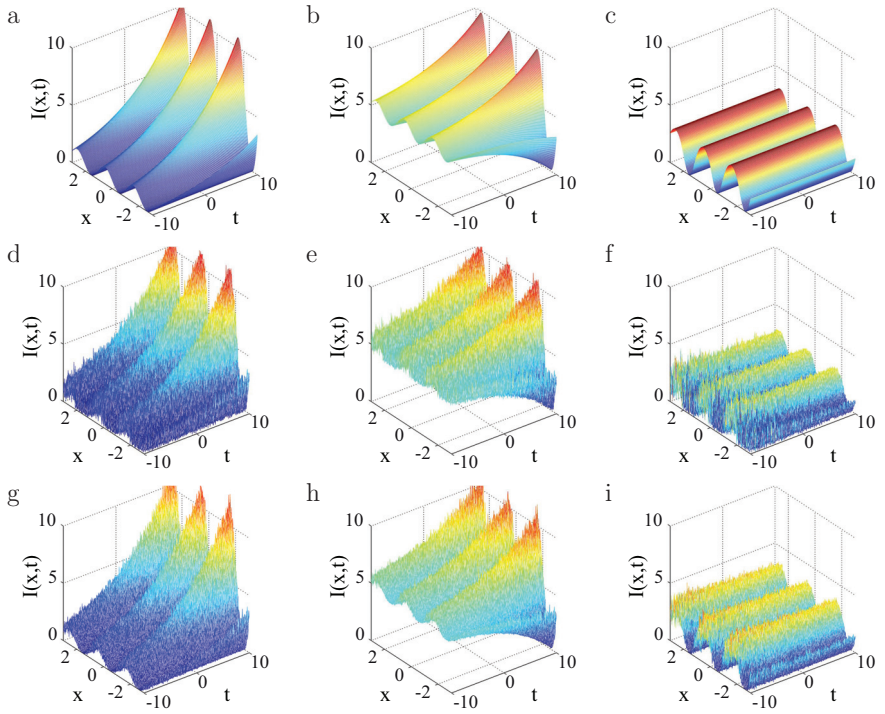


Figure 4.1: Sine wave with increasing illumination, parameter $a_1 = 0.1$, $a_{1,x} = 0$ (left), after highpass (middle) and homomorphic (right) prefiltering (lowpass with $\sigma_{pre} = 35$). Noise added: (top) no noise, (middle) additive Gaussian noise ($\sigma_n = 0.5$), and (bottom) shot noise ($k = 0.5$, see (4.31)).

we evaluate highpass filtering according to

$$\tilde{I} = (\mathbb{1} - G_{\sigma_{pre}}) * I \tag{4.1}$$

where $G_{\sigma_{pre}}$ denotes a spatial Gaussian filter (see, e.g., [32]) with standard deviation σ_{pre}, and where $\mathbb{1}$ is an identity filter whose impulse response equals one everywhere, and where $*$ is convolution.

A more sophisticated approach uses homomorphic filtering [48, 73]. Following [73], we briefly derive a straightforward implementation of homomorphic filtering, which proved to be very successful in suppressing illumination changes. Homomorphic filtering is based on modeling the image intensity $I(x, y)$ as being determined by illumination $L(x, y)$ reflected towards the camera by the surfaces of the objects present in the scene. The reflectance of the surfaces is denoted by $R(x, y)$. Note that in this model, both the illumination and the object reflectance are already

projected onto the image plane, i.e., they depend on the sensor coordinates x, y. For Lambertian surfaces, the image intensity can thus be modeled by

$$I(x, y) \propto L(x, y) \cdot R(x, y) \; . \tag{4.2}$$

The structure of the scene is represented by the reflectance $R(x, y)$, from which motion needs to be estimated. A logarithmic point transform converts the multiplicative relation[1] between illumination L and reflectance R into an additive one according to

$$\log(I(x, y)) \propto \log(L(x, y)) + \log(R(x, y)) \; . \tag{4.3}$$

In a rough approximation, $\log(L)$ and $\log(R)$ may be considered separated in frequency domain, as $L(x, y)$ is assumed to be low-frequent (such that even its harmonics generated by the nonlinear point transform can be neglected), while R and hence $\log(R)$ is regarded as predominantly high-frequent such that it can be extracted by a highpass filter. In practice, though, the two components overlap. The design of the highpass filter thus implies a trade off between suppressing brightness changes and loss of relevant signal. After highpass filtering, exponentiation returns an approximation of the sought reflectance component. Note that the nonlinear log-operation turns the camera noise, which, neglecting quantum noise, is often modeled as being signal-independent, into signal-dependent noise, which may affect motion parameter estimation. Figure 4.1 illustrates the effects of the proposed prefilters on a 2D signal $I(x, t)$ with exponential temporal brightness changes. Results for linear highpass and homomorphic prefiltering of the signal for noise-free data (top), for added Gaussian noise (middle) and for intensity-dependent noise (bottom) are shown (see Sec. 4.4.2 for details on intensity dependent noise). Evidently, homomorphic prefiltering removes all brightness changes for noise-free data, whereas highpass prefiltering only removes signal offset. Figure 4.1f shows how the nonlinear log-operation leads to increased noise in regions with small intensity values, especially for $t < 0$. If the noise is intensity-dependent like shot noise, homomorphic prefiltering may reduce this effect (Fig. 4.1i). A general possibility to reduce the signal dependence of shot or quantum noise is to subject the data first to a square-root transform [2]. We will, though, not consider this any further here.

4.2 Brightness Constraints

The brightness constancy constraint (2.3) can be replaced by an illumination invariant or illumination insensitive constraint, which can be applied instead of or together with prefiltering the data. In this section we present two standard and one novel constraint.

[1] Note that this multiplicative relation is also preserved for cameras with a nonlinear conversion from image irradiance to intensity, if the nonlinearity follows a γ-curve.

4.2.1 The Gradient Constancy Constraint

For the estimation of optical flow, a successful such constraint is to assume that the 2D image intensity *gradient* remains constant along the motion trajectory [49]. As gradient computation corresponds to derivative filtering, which is a highpass operation, this constraint therefore attenuates illumination changes, as discussed in Sec. 4.1. This leads to two linearized gradient constancy constraints

$$\frac{\mathrm{d}I_x}{\mathrm{d}t} = \partial_x I_x u_x + \partial_y I_x u_y + \partial_t I_x = 0 \tag{4.4}$$

$$\frac{\mathrm{d}I_y}{\mathrm{d}t} = \partial_x I_y u_x + \partial_y I_y u_y + \partial_t I_y = 0 \tag{4.5}$$

where lower indices indicate partial derivatives, e.g., $I_x := \partial_x I$.

4.2.2 Combined Intensity and Gradient Constancy Constraint

A known drawback of the gradient constancy constraint is that it noticeably reduces structure in the images and leads to aperture problems. Using both the intensity constraint and the gradient constraint simultaneously reduces this effect in optical flow estimation [49]. Doing so leads to three constraint equations ((2.3), (4.4) and (4.5)), which should be satisfied simultaneously.

4.2.3 A Physics-Based Brightness Change Model

A different approach to handle brightness changes is to model these explicitly, and to then estimate both optical flow and brightness change parameters based on this model. Haussecker and Fleet [25] proposed a generalized formulation of optical flow estimation based on models of brightness variation that are caused by time-dependent physical processes. Brightness changes along a temporal trajectory $\mathbf{x}(t) = (x(t), y(t))^{\mathrm{T}}$. This is described by a parameterized function h_I

$$I(\mathbf{x}(t), t) = h_I(I_0, t, \mathbf{g}) \tag{4.6}$$

where $I_0 = I(\mathbf{x}(0), 0)$ denotes image intensity at time $t = 0$ and $\mathbf{g} = [g_1, ..., g_n]^{\mathrm{T}}$ contains n brightness change parameters. Taking the total derivative on both sides yields

$$\partial_x I u_x + \partial_y I u_y + \partial_t I = \frac{\mathrm{d}}{\mathrm{d}t} h_I(I_0, t, \mathbf{g}) \; . \tag{4.7}$$

Assuming brightness constancy, i.e., $h_I(I_0, t, \mathbf{g}) = c$, (4.7) reduces to (2.3). Figure 4.2a shows constant, linear and nonlinear changes of intensity due to brightness changes. Given a physical model h for brightness changes, both the optical flow (u_x, u_y) and the parameter vector $\mathbf{g}$ need to be estimated. Several time-dependent

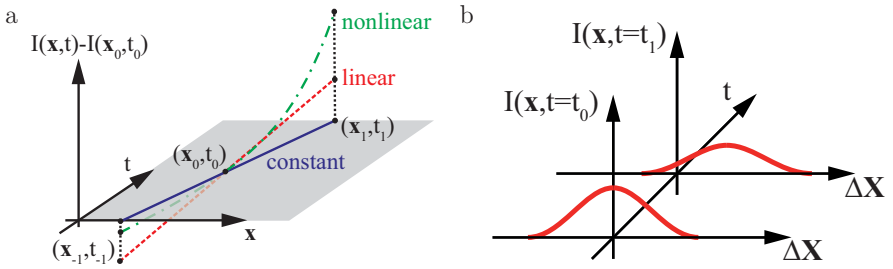


Figure 4.2: a: Change of intensity values caused by brightness changes (cf. [25]) and b: temporal intensity changes caused by illumination changes within a spatial neighborhood with local 3D coordinates ΔX.

brightness change models are proposed in [25], i.e., changing surface orientation, motion of the illuminant, and physical models of heat transport in infrared images. We derive a novel brightness change model [60], which describes spatially varying time-dependent illumination changes caused by directed, inhomogeneous illumination and changing surface orientation. Figure 4.2b illustrates spatially varying intensity changes in a neighborhood Λ. ΔX and ΔY are local coordinates of Λ. This model is able to also describe illumination by spotlights, whereas the models presented in [25] assume that the intensity change of pixels is constant over a local neighborhood.

The cameras convert light intensity L into image intensities I (i.e., gray values). In order to derive a model for $\mathrm{d}I\,/\mathrm{d}t$, the temporal changes visible in the data, we look into the dependencies of L. We assume that a translating surface patch is illuminated by a spatially smoothly varying, translating and rotating light source. Direction $\mathbf{n}_i$ of incident irradiance E may vary smoothly with time and space but reflectance direction $\mathbf{n}_r$ is kept constant.[2] Visible light intensity i.e., reflected radiance L depends on incident irradiance E and on the patch's bidirectional reflectance distribution function (BRDF) B (cmp. e.g., [24]) according to

$$L(\mathbf{X}(\Delta X, \Delta Y, t), t, \mathbf{n}_r) = B(\mathbf{X}(\Delta X, \Delta Y, t), \mathbf{n}_i(t), \mathbf{n}_r) E(\Delta X, \Delta Y, t, \mathbf{n}_i(t)) \tag{4.8}$$

and the BRDF depends on the material and hence on the position on the surface patch as well as the directions of incidence $\mathbf{n}_i$ and reflectance $\mathbf{n}_r$. We assume that the material does not change with time and therefore

$$B(\mathbf{X}(\Delta X, \Delta Y, t), \mathbf{n}_i(t), \mathbf{n}_r) = B(\mathbf{X}(\Delta X, \Delta Y, 0), \mathbf{n}_i(t), \mathbf{n}_r)\ . \tag{4.9}$$

If the BRDF is smooth enough, which is typically given at sufficient angular distance from specularities, changes due to smoothly changing incidence direction $\mathbf{n}_i(t)$ can

[2] Reflectance direction $\mathbf{n}_r$ obviously also varies with pixel position in the cameras, but we do not use this extra information.

be modeled using a smooth function $h_B(t)$ with $h_B(0) = 1$

$$B(\mathbf{X}(\Delta X, \Delta Y, t), \mathbf{n}_i(t), \mathbf{n}_r) = B(\mathbf{X}(\Delta X, \Delta Y, 0), \mathbf{n}_i(0), \mathbf{n}_r) h_B(t) \; . \tag{4.10}$$

Being spatially inhomogeneous the moving irradiance E changes not only by a time dependent factor, but by a factor also varying smoothly in space

$$E(\Delta X, \Delta Y, t, \mathbf{n}_i(t)) = E(\Delta X, \Delta Y, 0, \mathbf{n}_i(0)) h_E(\Delta X, \Delta Y, t) \; . \tag{4.11}$$

Here again $h_E(\Delta X, \Delta Y, t)$ is a smooth function with $h_E(\Delta X, \Delta Y, 0) \equiv 1$. Plugging (4.10) and (4.11) in (4.8) the reflected radiance L becomes

$$L(\mathbf{X}(\Delta X, \Delta Y, t), t) = L(\mathbf{X}(\Delta X, \Delta Y, 0), 0) h_B(t) h_E(\Delta X, \Delta Y, t) \; . \tag{4.12}$$

We assume image intensities I to be proportional to the radiance L, i.e., the characteristic curve of the used camera to be linear, and therefore

$$I(\mathbf{X}(\Delta X, \Delta Y, t), t) = I(\mathbf{X}(\Delta X, \Delta Y, 0), 0) \exp(h_I(\Delta X, \Delta Y, t)) \tag{4.13}$$

where $h_I(\Delta X, \Delta Y, t) := \ln(h_B(t) h_E(\Delta X, \Delta Y, t))$. The sought temporal derivative of (4.13) is thus

$$\begin{aligned} \tfrac{\mathrm{d}}{\mathrm{d}t} I &= I(\mathbf{X}(\Delta X, \Delta Y, 0), 0) \exp(h_I(\Delta X, \Delta Y, t)) \tfrac{\mathrm{d}}{\mathrm{d}t} h_I(\Delta X, \Delta Y, t) \\ &= I(\mathbf{X}(\Delta X, \Delta Y, t), t) \tfrac{\mathrm{d}}{\mathrm{d}t} h_I(\Delta X, \Delta Y, t) \; . \end{aligned} \tag{4.14}$$

The most common assumption in optical flow-like approaches is brightness constancy, boiling down to $h_I(\Delta X, \Delta Y, t) \equiv 0$. Haussecker and Fleet [25] derive models for changing surface orientation and a moving illumination envelope approximating h_I as a second order power series respecting temporal changes only

$$h_I(\Delta X, \Delta Y, t) \approx h_{HF}(t, \mathbf{g}) := \sum_{i=1}^{2} g_i t^i \tag{4.15}$$

where g_1 and g_2 are treated as local constants in the estimation process. Looking at Figs. 4.8f and g we observe that for highest accuracy this is not sufficient. Therefore we introduce a more accurate approximation of h_I explicitly modeling spatial variations still respecting $h_I(\Delta X, \Delta Y, 0) \equiv 0$

$$h_I(\Delta X, \Delta Y, t) \approx h(\Delta X, \Delta Y, t, \mathbf{g}) := \sum_{i=1}^{2} (g_i + g_{i,x} \Delta X + g_{i,y} \Delta Y)\, t^i \; . \tag{4.16}$$

The temporal derivative of h is then

$$f(\Delta X, \Delta Y, t, \mathbf{g}) := \frac{\mathrm{d}}{\mathrm{d}t} h(\Delta X, \Delta Y, t, \mathbf{g}) = \sum_{i=1}^{2} i \left(g_i + g_{i,x}\Delta X + g_{i,y}\Delta Y\right) t^{i-1} \quad (4.17)$$

using the notation

$$\mathbf{g} = [g_1, g_2, g_{1,x}, g_{1,y}, g_{2,x}, g_{2,y}]^{\mathrm{T}} . \quad (4.18)$$

Insertion of $f(\Delta X, \Delta Y, t, \mathbf{g})$ into the generalized brightness change constraint (4.7) yields the sought brightness constraint.

$$I_x u_x + I_y u_y + I_t - I \sum_{i=1}^{2} i \left(g_i + g_{i,x}\Delta X + g_{i,y}\Delta Y\right) t^{i-1} = 0 . \quad (4.19)$$

4.3 Extension of 3D Motion Estimation Models

In this section we briefly present, how the three models proposed in Chap. 3, namely *Scene Flow*, *Range Flow* and the *Affine Model* are extended to handle the brightness constraints different to the brightness constancy constraint (2.3).

4.3.1 *Scene Flow*

The extension of *Scene Flow* to handle brightness changes is straightforward. The constraints proposed in Sec. 4.2.1, Sec. 4.2.2 or Sec. 4.2.3 replace the brightness constancy constraint in (3.4). The equation system is then solved analog to Sec. 3.1.

4.3.2 *Range Flow*

In this section we show how to extend *Range Flow* for varying illumination [61] and how the choice of prefilters and brightness constraints influence estimation results of *Range Flow*.

Adaption of Constraints

In order to adapt the gradient constraints in (4.4) and (4.5) for *Range Flow* the optical flow u_x, u_y is eliminated using (3.6) and (3.7) (cmp. derivation of the intensity constraint (3.13)), yielding

$$\frac{\partial(I_x, Y)}{\partial(x, y)} U + \frac{\partial(X, I_x)}{\partial(x, y)} V + \frac{\partial(X, Y, I_x)}{\partial(x, y, t)} = 0 \quad (4.20)$$

$$\frac{\partial(I_y, Y)}{\partial(x, y)} U + \frac{\partial(X, I_y)}{\partial(x, y)} V + \frac{\partial(X, Y, I_y)}{\partial(x, y, t)} = 0 . \quad (4.21)$$

The combination of gradient and intensity constraint proposed in Sec. 4.2.2 leads to three constraint equations, viz. (3.13), (4.20) and (4.21), which should be satisfied simultaneously by the horizontal and vertical *Range Flow* components U and V. Adaption of the brightness change model is handled analog. Eliminating the optical flow (u_x, u_y) in (4.19) using (3.6) and (3.7) leads to

$$\frac{\partial(I,Y)}{\partial(x,y)}U + \frac{\partial(X,I)}{\partial(x,y)}V + \frac{\partial(X,Y,I)}{\partial(x,y,t)} - Ig_1 - Ig_{1,x}\Delta X - Ig_{1,y}\Delta Y \\ - 2Ig_2 t - 2Ig_{2,x}\Delta X t - 2Ig_{2,y}\Delta Y t = 0\,. \quad (4.22)$$

Estimation of Motion and Brightness Parameters

Estimating the 3D motion from the input data then corresponds to fitting the selected model to the original or, if applicable, prefiltered data. The parameters to be estimated comprise the motion parameters U, V, W for all models and, for the physics-based brightness change model from Sec. 4.2.3, the brightness change parameters $g_1, g_{1,x}, g_{1,y}, g_2, g_{2,x}, g_{2,y}$ in the Taylor series (4.16).

As described in Secs. 3.2.3, 4.2.1, 4.2.2 and 4.2.3 we have more than one constraint for the U and V components of the *Range Flow*. In the same way as expressing the range constraint by $\mathbf{d}_{rc}^{\mathrm{T}}\mathbf{p} = 0$, the intensity constraint (3.13) and the gradient constancy constraints (4.20) and (4.21) may be expressed by $\mathbf{d}_Q^{\mathrm{T}}\mathbf{p} = 0$, where $\mathbf{d}_Q$ is computed from the observed data according to

$$\mathbf{d}_Q = \left[\frac{\partial(Q,Y)}{\partial(x,y)}, \frac{\partial(X,Q)}{\partial(x,y)}, 0, \frac{\partial(X,Y,Q)}{\partial(x,y,t)}\right]^{\mathrm{T}} \quad (4.23)$$

with $Q \in \{I, I_x, I_y\}$ selected as appropriate. We adapt all constraints to identical dimensions by inserting zeros at those positions into the data vector $\mathbf{d}$, which correspond to positions of parameters in the parameter vector $\mathbf{p}$ which are not part of the respective constraint. In particular, the physics-based brightness change model in Sec. 4.2.3 contains motion *and* brightness change parameters, thus leading to the enlarged parameter vector

$$\mathbf{p} = [U, V, W, 1, \mathbf{g}^{\mathrm{T}}]^{\mathrm{T}}\,. \quad (4.24)$$

The data vectors for both the range constraint and the brightness change constraint have hence to be enlarged correspondingly by appropriate insertion of zeros. As shown in [69], combining the different constraints yields a new structure tensor which is simply the weighted sum of the tensor $\mathbf{J_{rc}}$ from the range constraint (3.14), and the tensors $\mathbf{J_i}$, $i = 1, \ldots, 4$ from the intensity-dependent constraints (3.13), (4.20),

(4.21) and (4.22). With the weights β_i, the overall tensor thus is

$$\mathbf{J} = \mathbf{J_{rc}} + \sum_{i=1}^{4} \beta_i \mathbf{J_i} \; . \tag{4.25}$$

The weights β_i may even be used to switch between the models, and to account for different signal-to-noise-ratios in the structure tensors. Analog to standard *Range Flow*, data channels should be scaled to the same mean and variance before they are combined. Parameter estimation is done analog to Sec. 3.2.3.

4.3.3 *Affine Model*

In this section we will adapt the brightness change model (4.19) to the *Affine Model* presented in [60] and Sec. 3.3. Combining the brightness change model (4.19) with the 5D affine flow model (3.26) yields

$$I_x \mathrm{d}x \; + I_y \mathrm{d}y \; + I_{s_x} \mathrm{d}s_x \; + I_{s_y} \mathrm{d}s_y \; + I_t \mathrm{d}t \; - I f \mathrm{d}t \; = 0 \; . \tag{4.26}$$

Inserting differentials $\mathrm{d}x$ and $\mathrm{d}y$ in (4.26) analog to Sec. 3.3 leads to data vector $\mathbf{d}$ and parameter vector $\mathbf{p}$:

$$\begin{aligned}
\mathbf{d} = \; & (I_x, \, I_y, \, I_x \Delta x, \, I_x \Delta y, \, I_y \Delta x, \, I_y \Delta y, \\
& I_{s_x}, \, I_{s_y}, \, I_t, \, I, \, I \Delta x, \, I \Delta y, \, I t, \, I t \Delta x, \, I t \Delta y)^{\mathrm{T}} \\
\mathbf{p} = \; & (u_x \mathrm{d}t \; + \nu \mathrm{d}s_x \; , \; u_y \mathrm{d}t \; + \nu \mathrm{d}s_y \; , \; a_{11} \mathrm{d}t \; + b_1 \mathrm{d}s_x \; , \; a_{12} \mathrm{d}t \; + b_2 \mathrm{d}s_x \; , \\
& a_{21} \mathrm{d}t \; + b_1 \mathrm{d}s_y \; , \; a_{22} \mathrm{d}t \; + b_2 \mathrm{d}s_y \; , \; \mathrm{d}s_x \; , \; \mathrm{d}s_y \; , \; \mathrm{d}t \; , \\
& \tilde{g}_1 \mathrm{d}t \; , \; \tilde{g}_{1,x} \mathrm{d}t \; , \; \tilde{g}_{1,y} \mathrm{d}t \; , \; \tilde{g}_2 \mathrm{d}t \; , \; \tilde{g}_{2,x} \mathrm{d}t \; , \; \tilde{g}_{2,y} \mathrm{d}t \;)^{\mathrm{T}} \; .
\end{aligned} \tag{4.27}$$

where f has been substituted by the novel brightness change model from (4.17) and the brightness change parameters are

$$\begin{aligned}
\tilde{g}_1 &= -g_1 & \tilde{g}_2 &= -g_2 \\
\tilde{g}_{1,x} &= -\frac{Z_0}{fc} \left(g_{1,x}(1 - Z_Y \frac{y_0}{f}) + g_{1,y} Z_X \frac{y_0}{f} \right) & \tilde{g}_{1,y} &= -\frac{Z_0}{fc} \left(g_{1,x} Z_Y \frac{x_0}{f} + g_{1,y}(1 - Z_X \frac{x_0}{f} \right) \\
\tilde{g}_{2,x} &= -\frac{Z_0}{fc} \left(g_{2,x}(1 - Z_Y \frac{y_0}{f}) + g_{2,y} Z_X \frac{y_0}{f} \right) & \tilde{g}_{2,y} &= -\frac{Z_0}{fc} \left(g_{2,x} Z_Y \frac{x_0}{f} + g_{2,y}(1 - Z_X \frac{x_0}{f} \right)
\end{aligned} \tag{4.28}$$

For simpler brightness models or when a 1D camera grid is used (i.e., $\mathrm{d}s_y = 0$) terms with non-existing parameters are simply omitted.
Parameter estimation is done analog to the standard *Affine Model* (cmp. Sec. 3.3).

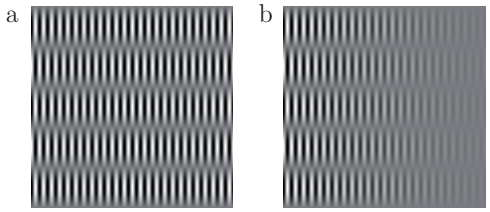


Figure 4.3: Sinusoidal pattern data with intensity values scaled to range $[0; 255]$. a and b: First and last image taken at central camera position, $g_1 = g_{1,x} = 0$, $g_2 = -0.2$ and $g_{2,x} = -0.002$ (cmp. (4.16)).

4.4 Experiments

In this section we validate the performance of the proposed brightness handling techniques, namely prefilters and more robust brightness constraints and the novel brightness change constraint. We first evaluate the novel brightness change model presented in Sec. 4.2.3 with the *Affine Model* and compare results to the brightness change model proposed by Haussecker and Fleet [25]. Then we show influence of prefilters and brightness constraints on *Range Flow* in Sec. 4.4.2. Finally we compare the three 3D motion estimation models presented in Chap. 3 with and without brightness change modeling.

4.4.1 Evaluation of Brightness Change Models for the *Affine Model*

Synthetic sinusoidal sequences are used to examine the influence of the different brightness change parameters (4.19) onto motion estimation. Performances of the standard brightness constancy model, the brightness change model presented by Haussecker and Fleet [25] and the novel brightness change model are then compared on synthetic data in Sec. 4.4.1.

Sinusoidal Pattern

Sinusoidal pattern data is used to evaluate systematic errors and noise dependence of the estimation process. In Fig. 4.3 two images of such a sequence are shown. The wavelengths are 8 pixel in x- and 80 pixel in y-direction and amplitude changing according to (4.16). We generated data sets for different values of brightness change parameters g_1, $g_{1,x}$, g_2, and $g_{2,x}$, but not for $g_{1,y}$ and $g_{2,y}$ as they work like the respective x-parameters. The other parameters are $U_X = U_Y = 0$ mm/frame, $U_Z = 0.1$ mm/frame, $Z_X = Z_Y = 0$, $Z_0 = 100$ mm and $f = 10$ mm. As performance measure for a parameter Q we use the mean absolute value either of the relative

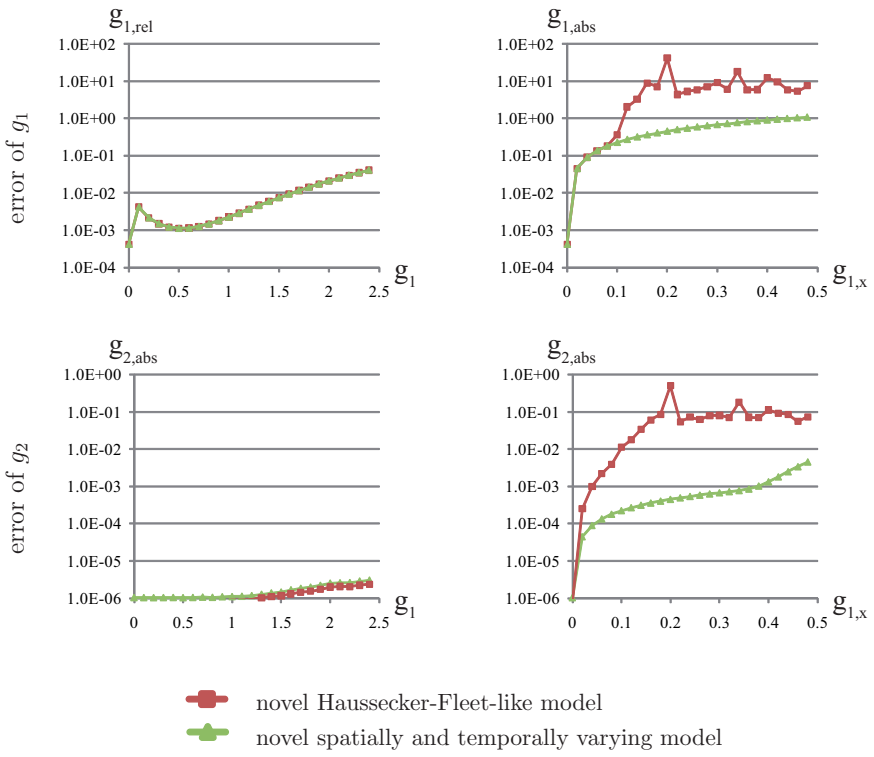


Figure 4.4: Mean absolute value of relative or absolute error of brightness change parameters g_1 (top) and g_2 (bottom) versus the brightness change parameters g_1 and $g_{1,x}$. Noise free data.

error if $Q_{ref} \neq 0$ or of the absolute error if $Q_{ref} = 0$

$$Q_{rel} = \frac{1}{N}\sum_{i}^{N}\frac{|Q_{est}(i) - Q_{ref}(i)|}{|Q_{ref}(i)|} \qquad Q_{abs} = \frac{1}{N}\sum_{i}^{N}|Q_{est}(i) - Q_{ref}(i)| \quad (4.29)$$

where the sum runs over all pixels not suffering from border effects and the lower indices *rel* stand for 'relative error', *abs* for 'absolute error', *est* for 'estimated' and *ref* for 'reference'. Parameter estimation was done according to (3.3.2), with weighting matrix $\mathbf{W}$ implemented via a 65-tab Gaussian with standard deviation $\sigma_W = 16$.

The first experiment evaluates systematic error of and cross talk between brightness change parameters. Figures 4.4 and 4.5 show errors of g_1 and g_2 versus brightness change parameters g_1 and $g_{1,x}$, and g_2 and $g_{2,x}$ respectively. We observe that the

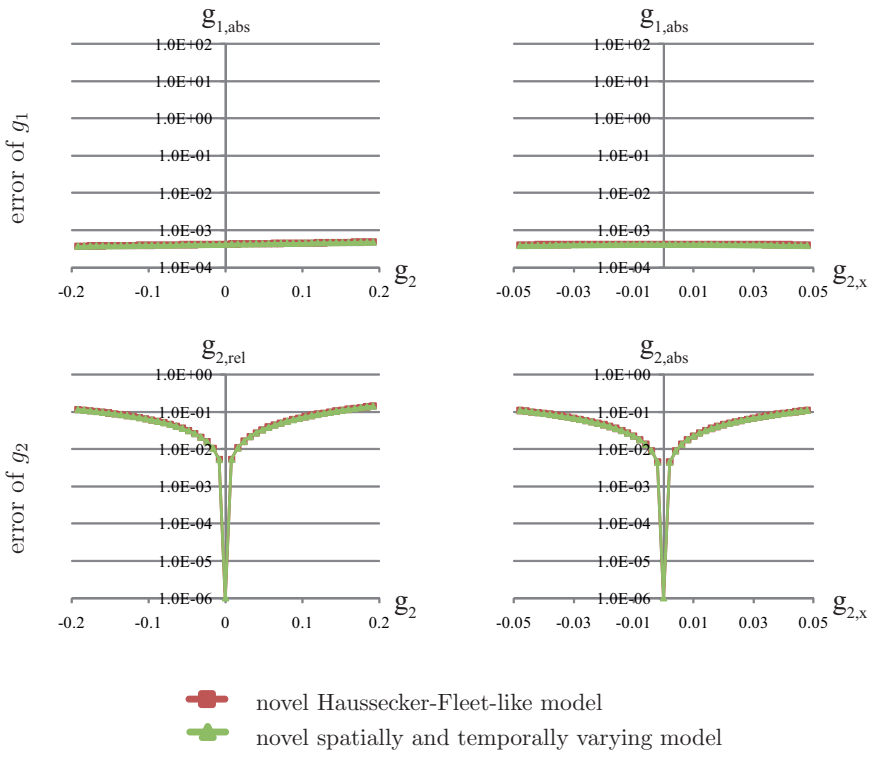


Figure 4.5: Mean absolute value of relative or absolute error of brightness change parameters g_1 (top) and g_2 (bottom) versus the brightness change parameters g_2, and $g_{2,x}$. Noise free data.

relative error of g_1 is well below 0.5% if $g_1 < 1$ and then moderately raises. This is due to the fact that temporal derivatives of the data I_t become less and less accurate when exponential behavior of the data becomes more and more prominent. The same explanation holds for the linear error increase of g_2 with increasing g_2. And as local brightness changes due to $g_{1,x}$ come close to changes due to g_1 if $g_1 = g_{1,x}\Delta X$ for the same local patch, we expect and observe severe cross talk between $g_{1,x}$ and g_1, more severe for the model not containing $g_{1,x}$. This is also true for $g_{2,x}$ and g_2, but there the cross talk is the same for both models, thus modeling $g_{2,x}$ is of no advantage here (cmp. Fig. 4.5 bottom). Further g_1 is almost independent of g_2 and $g_{2,x}$, as well as g_2 of g_1. But while g_2 does depend on $g_{1,x}$ if $g_{1,x}$ is not modeled, the error of g_2 is about 1 to 2 orders of magnitude smaller if $g_{1,x}$ is modeled (cmp. Fig. 4.4 lower right). The positive effect on accuracy of the method if $g_{1,x}$ is modeled is even higher for U_Z, as we see next.

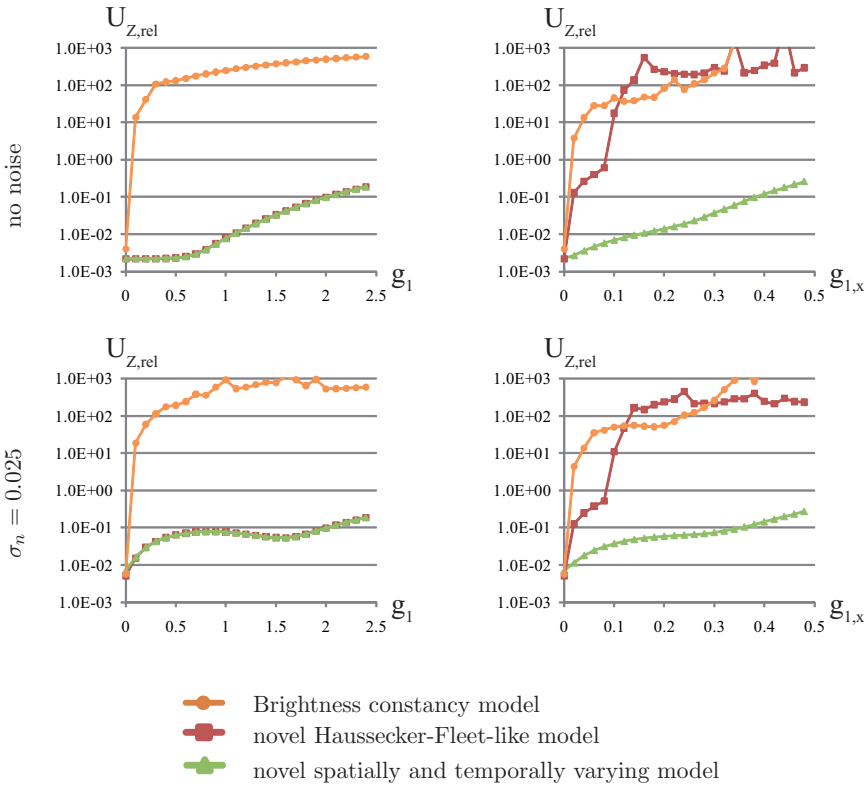


Figure 4.6: Mean absolute value of relative error of U_Z versus the brightness change parameters g_1 and $g_{1,x}$. Noise free (top row) and noisy data (bottom).

In Figs. 4.6 and 4.7 results for $U_{Z,rel}$ versus brightness change parameters are shown, using noise free data and data with Gaussian noise of standard deviation $\sigma_n = 0.025$ being 2.5% of the amplitude of the signal at $t = 0$. As before all parameters except the one on the ordinate have been kept fix. U_Z is the most relevant motion parameter, because errors in U_Z directly also influence U_X and U_Y (see the components of the parameter vector $\mathbf{p}$ in (3.36)). Let us first look at the noise free case. As soon as g_1 is significantly larger than 0 the brightness constancy model immediately breaks down, errors get unacceptably high. For the two other models U_Z does not react on small g_1 and only weak for larger values of g_1. When brightness changes due to $g_{1,x}$ are present only the model containing spatial changes remains stable, brightness constancy and Haussecker-Fleet-like models have severe problems. Looking at $U_{Z,rel}$ with changes due to g_2 or $g_{2,x}$ we observe that g_2 and $g_{2,x}$ cause similar errors in U_Z. This is in complete consistency with our earlier

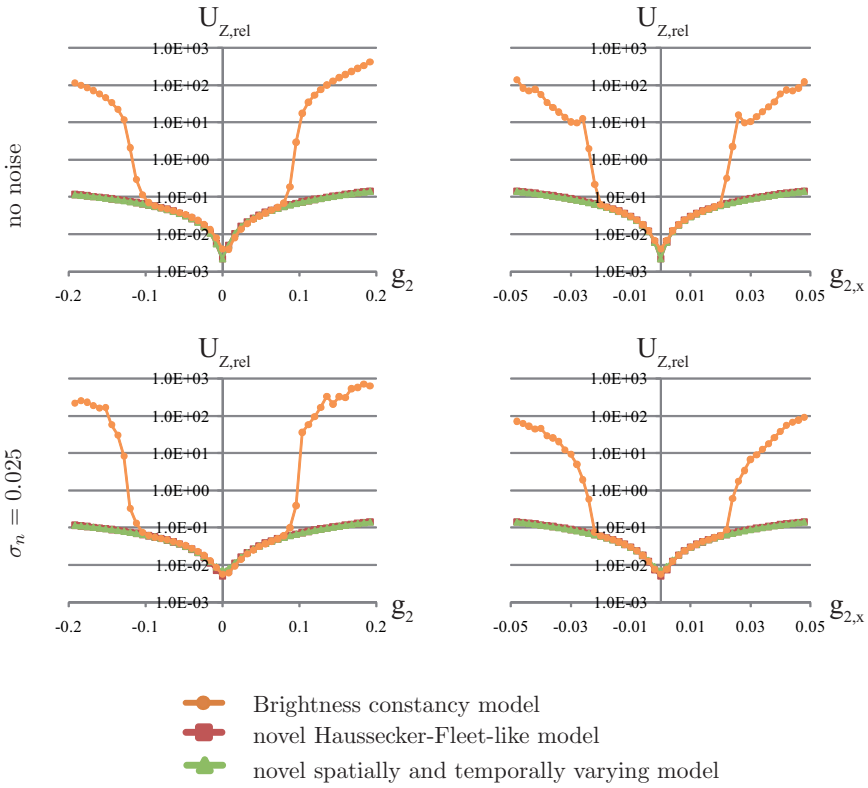


Figure 4.7: Mean absolute value of relative error of U_Z versus the brightness change parameters g_2, and $g_{2,x}$. Noise free (top row) and noisy data (bottom).

observation in Fig. 4.5. While all models behave the same for small absolute values of g_2 or $g_{2,x}$, the brightness constancy model rapidly breaks down at $|g_2| \approx 0.1$ or $|g_{2,x}| \approx 0.02$. Comparing errors of U_Z for noise free and noisy data sets, we see only a small effect when g_2 or $g_{2,x}$ are close to 0. For larger g_2 or $g_{2,x}$ the plots for noisy and noise free data look almost identical. Also for large g_1 and $g_{1,x}$ errors remain unchanged. But for smaller g_1 and $g_{1,x}$ the influence of noise can be quite high. We observe that errors increase from well below $U_{Z,rel} = 0.01$ up to nearly $U_{Z,rel} = 0.1$. We conclude that modeling $g_{1,x}$ is worth the effort while $g_{2,x}$ does not really help. Noise may be an issue, thus it has to be kept as low as possible.

Synthetic Cube

Temporal image sequences with 9 images were created at 25 positions of a 2D 5×5 camera grid using POV-Ray [15]. For the whole cube ground truth is $U_X = U_Y = 0$

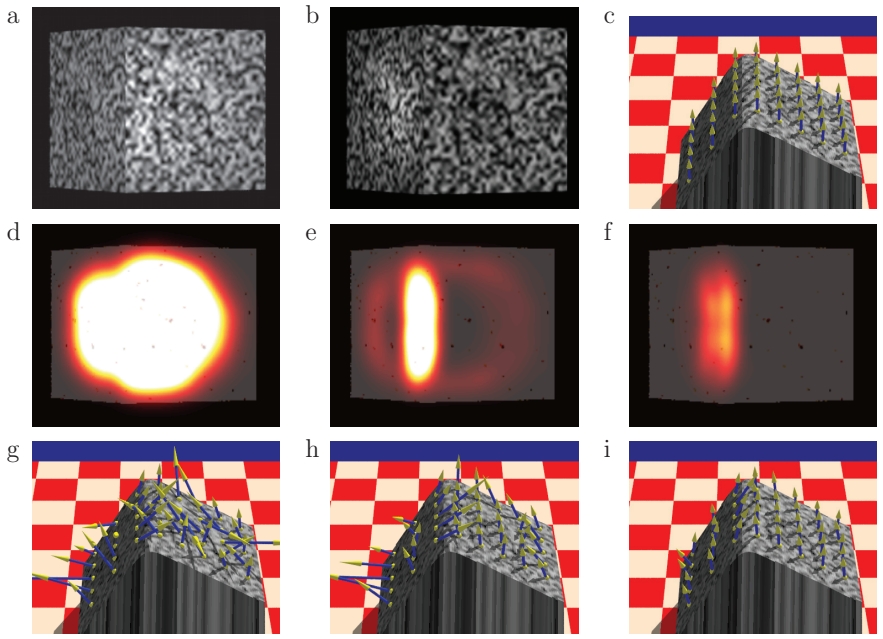


Figure 4.8: Motion estimation of cube moving towards camera with spot light moving around cube center. a and b: first and last image taken with central camera. c: Ground truth motion, d–f: color coded model errors (projected on contrast reduced cube) for models without (d), constant temporal (e), and spatially varying temporal brightness change (f). Below the model errors, scaled motion estimates for the models are depicted, respectively (g–i).

mm/frame, $U_Z = 2$ mm/frame, and $Z_Y = 0$. At the left side $Z_X \approx 1.73 \hat{=} 60°$ and on the right $Z_X \approx 0.577 \hat{=} 30°$. As one can see in Figs. 4.8a and b, a noise texture with high contrast is mapped on the sides of the cube and in addition to the ambient illumination a spot light rotates around the center of the cube such that it moves from right to left. Figure 4.8c depicts ground truth motion. In Figs. 4.8d-f the numerical model error, i.e., the largest of the 3 smallest eigenvalues of the structure tensor is depicted as color overlay on the central input image. For the brightness constancy model (Fig. 4.8d) error is highest. Modeling spatially constant brightness changes (Fig. 4.8e) errors reduce, but at the edge of the cube and at the border of the spotlight they are still high. With spatially varying temporal changes errors again become smaller, visible only at the edge of the cube. The components U_X and U_Y of the motion vectors shown in Figs. 4.8g-i are scaled by a factor 135 relatively to U_Z in order to visualize estimation errors (U_X and U_Y should be 0). Even with this large accentuation of errors motion vectors estimated with the richest model

point in the correct direction almost everywhere. The other models yield much less accurate vector fields.

4.4.2 Evaluation of Prefilters and Brightness Change Models for *Range Flow*

The models compared are combinations of the range constraint with

1. the intensity constancy constraint (3.13), i.e., $\beta_2 = \beta_3 = \beta_4 = 0$ (INT),
2. the gradient constancy constraint (4.20) and (4.21), i.e., $\beta_1 = \beta_4 = 0$ (GRAD),
3. the combined intensity and gradient constancy constraint (3.13), (4.20) and (4.21), i.e., $\beta_4 = 0$ (INTGRAD) and
4. the intensity constraint with modeling of brightness changes by Taylor series (4.22), i.e., $\beta_1 = \beta_2 = \beta_3 = 0$ (TAYLOR).

Model INT without prefiltering is the original *Range Flow* as introduced by Spies et al. [69]. As shown in Sec. 4.4.1, we may drop $g_{2,x}$ and $g_{2,y}$ for model TAYLOR, as these have only a negligible effect on the estimation. We investigate the influence of temporal and spatial varying temporal brightness changes for noise-free input sequences as well as for data corrupted by additive Gaussian noise or by intensity dependent noise. Furthermore, accuracy of the 3D motion estimates is evaluated on a rendered cube illuminated by a directed light source. For the synthetic sinusoidal sequences, ground truth range data is used, while for all the cube experiments, we generate reference range data by the *Affine Model*. Using multi-camera data allows for comparison with [31] and has no further effect on range flow.

To compute the intensity derivatives, we used the optimized $5 \times 5 \times 5$ filter sets described in [54] for all experiments. Motion and brightness change parameters are determined as the minimizer of the model error given in (2.17), as described in Sec. 2.3.2. Spies et al. [69] use $\beta_0 = 1$ for their experiments (cmp. (4.25)). We scale all structure tensors from the different models and prefilters to have same variance as standard *Range Flow*, i.e., the variance of model INT without prefiltering. In [61], structure tensors were not scaled, therefore leading to slightly different results.

Sinusoidal Patterns

For a systematic error analysis of the different models, we use sinusoidal patterns under varying illumination generated using the *Sequence Generator*(see App. B). The varying illumination is generated according to (4.6), (4.13) and (4.16). Three frames of a typical test sequence are shown in Fig. 4.9. Its parameters were set as follows: Rendered surfaces translate with $U = 0.0073\,\text{mm/frame}$, $V = 0\,\text{mm/frame}$ and $W = 0.5\,\text{mm/frame}$ and rotate around the Y-axis with an angular velocity of $\omega = 0.002\,\text{radians/frame}$. For $t = 0$, the surface normal of the patch is $\mathbf{n} = (1, 2, -1)^{\mathrm{T}}$,

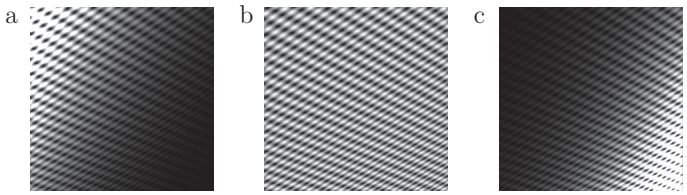


Figure 4.9: Scaled first (a), central (b) and last (c) frame of sinusoidal sequence with illumination parameters $g_1 = 0$ and $g_{1,x} = 0.06$.

and the distance of the patch center to the camera is $Z_0 = 100\,\text{mm}$. The synthetic sensor contains 301×301 pixels of size $(0.0044\,\text{mm})^2$. The focal length of the synthetic projective camera is $f = 12\,\text{mm}$. For each experiment, we evaluate the mean absolute value of the relative error of U

$$ERR_U = \frac{1}{N} \sum_{i}^{N} \frac{|U_{estimated}(i) - U_{reference}(i)|}{|U_{reference}(i)|} \tag{4.30}$$

over all pixels i at a minimum distance of 60 pixels from the nearest image border. The structure tensor weighting matrix $\mathbf{W}$ is realized by a large, 65-tab Gaussian with standard deviation $\sigma_{\mathbf{W}} = 19$ in order to reduce systematic errors stemming from the phase of the sinusoidal pattern. Prefilters are designed as described in Sec. 4.1, i.e., σ_{pre} denotes the standard deviation of the lowpass $G_{\sigma_{pre}}$ in (4.1). As in [60], we compare estimation errors of U for increasing illumination parameters $g_1|_{g_{1,x}=0}$ and $g_{1,x}|_{g_1=0}$ to simulate brightness changes. We provide here the errors for U only, as errors of V and W exhibited very similar behavior. Furthermore, we investigate the influence of adding Gaussian noise with $\sigma_n = 0.025$ or shot noise to the intensity data. The Poisson-distributed shot noise is approximated by adding Gaussian noise with intensity-dependent standard deviation

$$\sigma_{sn} = k\sqrt{\frac{I(x,t)}{\alpha}} \tag{4.31}$$

with $k = 0.025$ and $\alpha = 1$. The sinusoidal pattern for $t = 0$ has intensity values in the range of $0 \leq I \leq 2$. To facilitate the evaluation of the effects of prefiltering, we show the errors for all tested models *with* and *without* prefiltering in Fig. 4.10-Fig. 4.13.

In Fig. 4.10, the errors of U for the described models with and without highpass and homomorphic prefiltering are shown over the parameter g_1. Highpass prefiltering reduces estimation errors for models INT and INTGRAD. For models GRAD and TAYLOR, using no prefilter yields results as good as using a highpass with standard deviation $\sigma_{pre} > 15$ (see (4.1)). Strong filtering, i.e., small σ_{pre}, degrades estimation results for all models. Overall, the model TAYLOR yields the best results for most cases, only for $g_1 \approx 0.3$ the model GRAD performs slightly better. Grooves in the

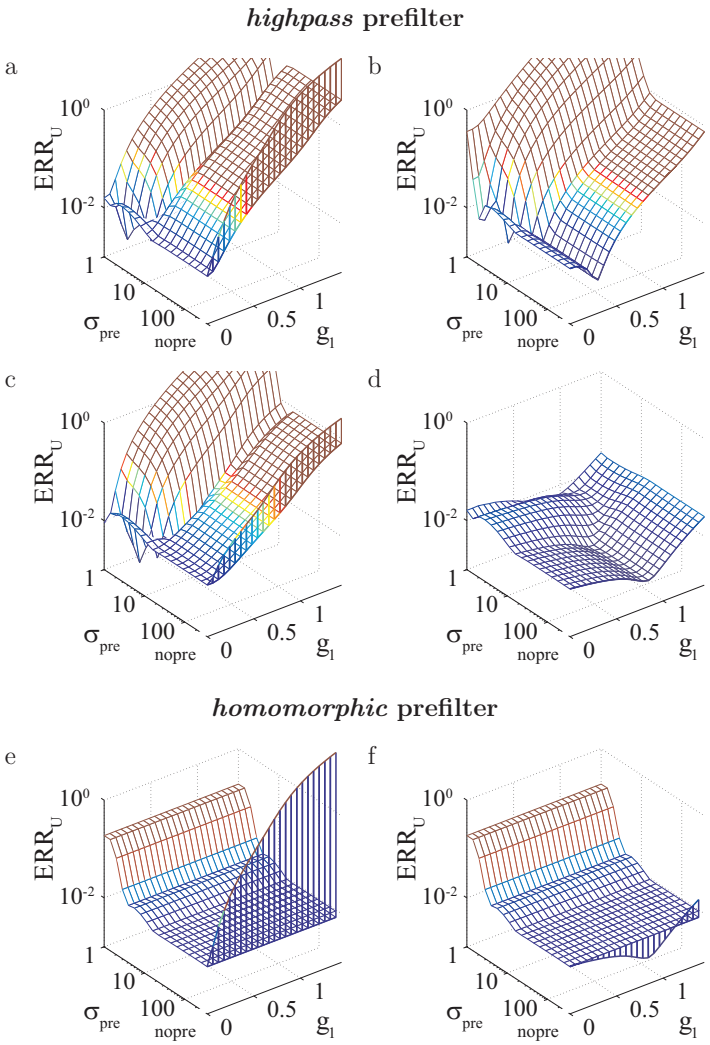


Figure 4.10: Error of motion estimates U versus increasing *temporal* brightness changes without prefiltering ("nopre") and with *highpass* (a - d) or *homomorphic* (e and f) prefiltering. *Temporal* brightness changes are modeled by $g_1 \neq 0$. All other parameters of $\mathbf{g}$ in (4.18) are zero. Standard deviation σ_{pre} according to (4.1). Models with *highpass* prefiltering are a: INT, b: GRAD, c: INTGRAD and d: TAYLOR. Models with *homomorphic* prefiltering are e: INT and f: TAYLOR.

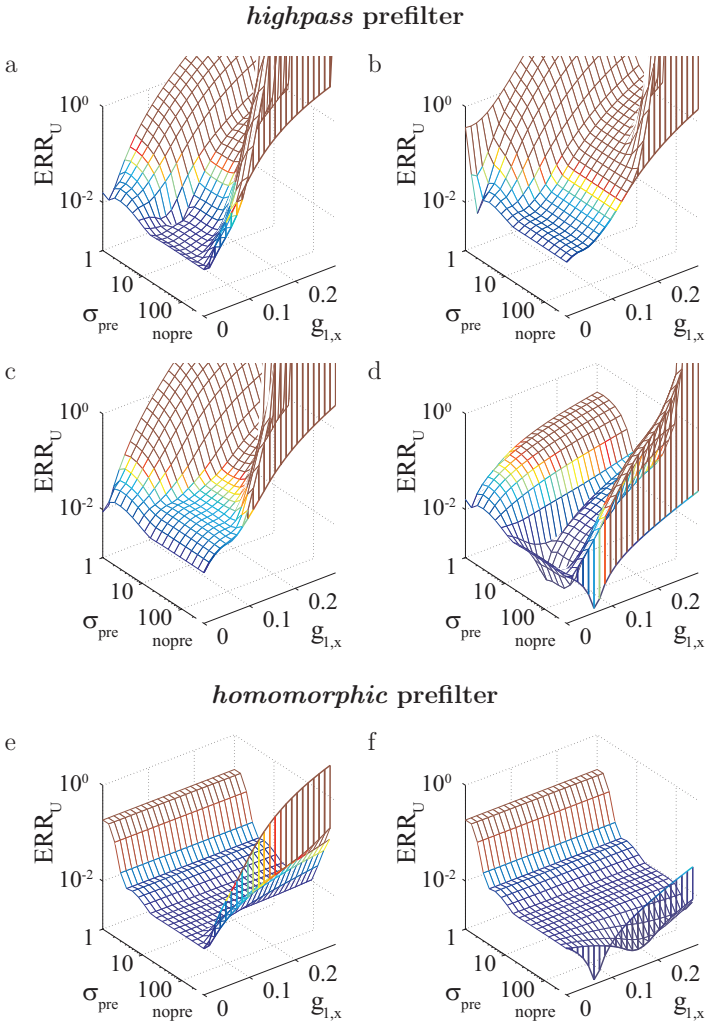


Figure 4.11: Error of motion estimates U versus increasing *spatially varying temporal* brightness changes without prefiltering ("nopre") and with *highpass* (a - d) or *homomorphic* (e and f) prefiltering. *Spatially varying temporal* brightness changes are modeled by $g_{1,x} \neq 0$. All other parameters of $\mathbf{g}$ in (4.18) are zero. Standard deviation σ_{pre} according to (4.1). Models with *highpass* prefiltering are a: INT, b: GRAD, c: INTGRAD and d: TAYLOR. Models with *homomorphic* prefiltering are e: INT and f: TAYLOR.

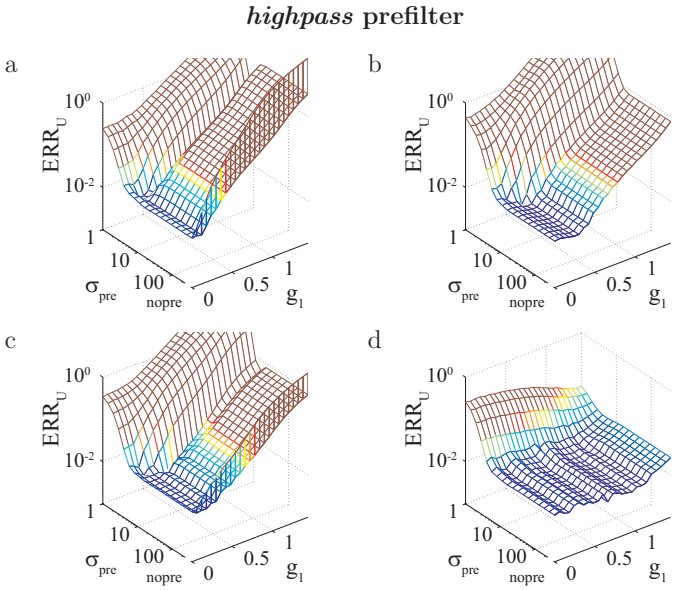


Figure 4.12: Same as Figs. 4.10a-d but for data with additive Gaussian noise of standard deviation $\sigma_n = 0.025$. Error of motion estimates U versus increasing *temporal* brightness changes and *highpass* prefiltering or without prefiltering ("nopre"). Models are a: INT, b: GRAD, c: INTGRAD and d: TAYLOR.

error surfaces where errors become low, e.g., for model INT at $a_1 \approx 0.1$ and $\sigma_{pre} < 10$, may be the result of improved gradients due to brightness changes and are highly pattern dependent. All models perform comparable when applying homomorphic prefiltering. Therefore, we show errors for the models INT and TAYLOR only (Figs. 4.10e and f). Homomorphic prefiltering with $\sigma_{pre} > 10$ reduces the brightness changes efficiently for all values of a_1, while prefiltering with $\sigma_{pre} < 10$ degrades the signal. For small brightness changes, model TAYLOR performs even better without prefiltering.

Figure 4.11 repeats the experiments shown in Fig. 4.10, but with brightness changes now being governed by the parameter $g_{1,x}$ instead of g_1. This means that brightness changes vary now not only temporally, but also spatially.

Evidently, the errors shown in Fig. 4.11 depend more strongly on σ_{pre} than those in Fig. 4.10. Prefiltering with standard deviation of $12 \leq \sigma_{pre} \leq 50$ yields the best results. As before, models GRAD and TAYLOR perform well without prefilter. Figure 4.11e and f show similar behavior for homomorphic prefiltering. Prefiltering with $\sigma_{pre} > 50$ degrades the signal for all models but model TAYLOR. For model TAYLOR, using no prefilter yields best results if $g_{1,x} \lesssim 0.07$.

Figs. 4.12 and 4.13 illustrate the effects of the proposed prefilters on noisy data.

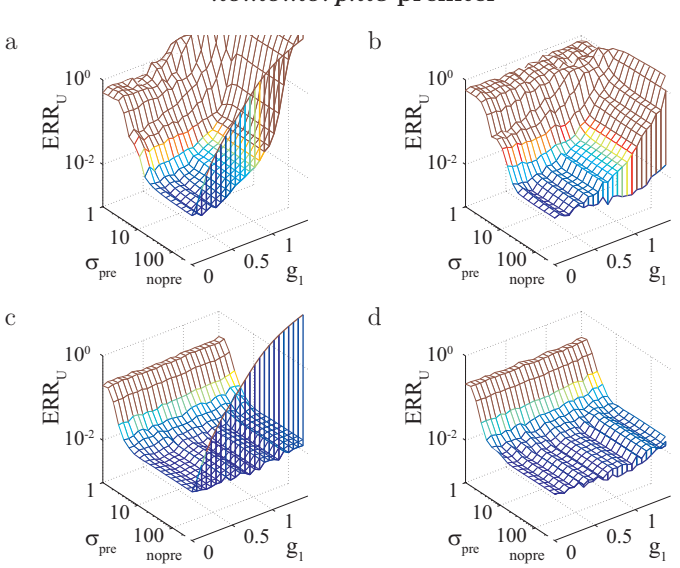


Figure 4.13: Same as Figs. 4.10e and f but for input data with added Gaussian noise with $\sigma_n = 0.025$ (a and b) or shot noise with $k = 0.025$ (c and d). Error of motion estimates U versus increasing *temporal* brightness changes and *homomorphic* prefiltering or without prefiltering ("nopre"). Models are INT (a and c), and TAYLOR (b and d).

In Sec. 4.1, highpass filtering was shown to have no effect on noise distributions in contrast to homomorphic prefiltering. We present errors of U for highpass prefiltering for added Gaussian noise with standard deviation $\sigma_n = 0.025$ only (Fig. 4.12), because results for shot noise are almost the same. In comparison to the noise-free case, the errors increased and the grooves in the error surfaces vanished. Overall, the model TAYLOR yields best results for $g_1 > 0$ and similar results as the other models for $g_1 \approx 0$.

Figure 4.13 shows how homomorphic prefiltering affects estimation results for added Gaussian noise and simulated shot noise. As stated above, the nonlinear log-operation makes originally signal-independent noise signal-dependent. Therefore, homomorphic prefiltering of noisy intensity data with brightness changes $g_1 > 0.5$ causes heavily degraded signals. If the noise itself is signal-dependent, such as shot noise, homomorphic prefiltering performs much better. In both noise experiments, model TAYLOR without prefilter yields the best and most reliable results.

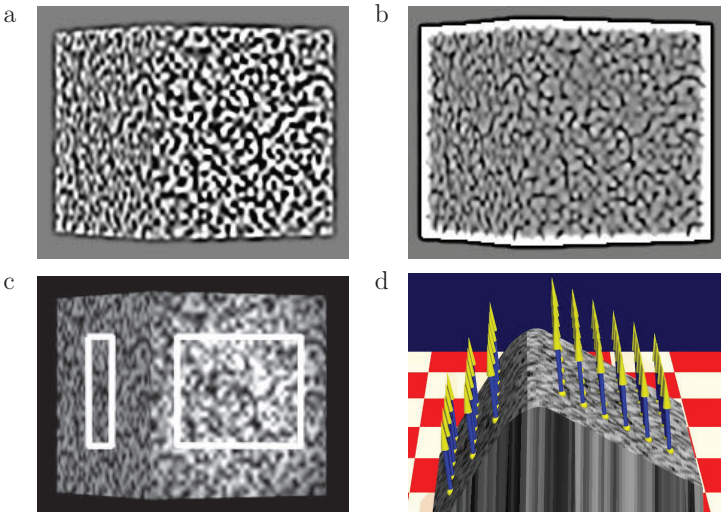


Figure 4.14: Cube experiment data. a: Central frame of cube sequence after highpass and b: homomorphic prefiltering, c: regions of cube used for error analysis, and d: rendering of ground truth.

Synthetic Cube

The synthetic cube sequence allows to test the models on more realistic data, but still with ground truth available. The cube moves with $U = -0.2\,\text{mm/frame}$, $V = 0\,\text{mm/frame}$ and $W = -2\,\text{mm/frame}$. In addition to ambient light, the cube is illuminated by a fixed spotlight from the right. The input data consists of sequences with nine frames each, acquired by five synthetic cameras positioned on the x-axis. This setup allows us to use the optimized filter sets proposed in [54]. We compute range data with the stereo estimation algorithm presented in [60]. Figure 4.14 shows two frames of the cube sequence, a frame after highpass prefiltering, and one after homomorphic prefiltering. Also shown are the regions investigated on the left and right side of the cube, as well as a rendering of the ground truth. Both prefilters have a standard deviation of $\sigma_{pre} = 2$, and the weighting matrix $\mathbf{W}$ is implemented using a 31-tab Gaussian with standard deviation $\sigma_{\mathbf{W}} = 11$. In Fig. 4.15, motion estimates of the original *Range Flow*[69] (i.e., model INT) are compared to the following three models: GRAD with highpass prefilter, INTGRAD with homomorphic prefilter, and TAYLOR without prefilter. For all models, the errors are too small to be visible without amplification. Thus *errors are amplified* for U and V by 100 and for W by 50.

Standard *Range Flow* estimation [69], i.e., model INT without prefilter (Fig. 4.15a), yields highly corrupted estimation results on the right side of the cube where illumination changes because of the fixed spotlight. The other side does not suffer from

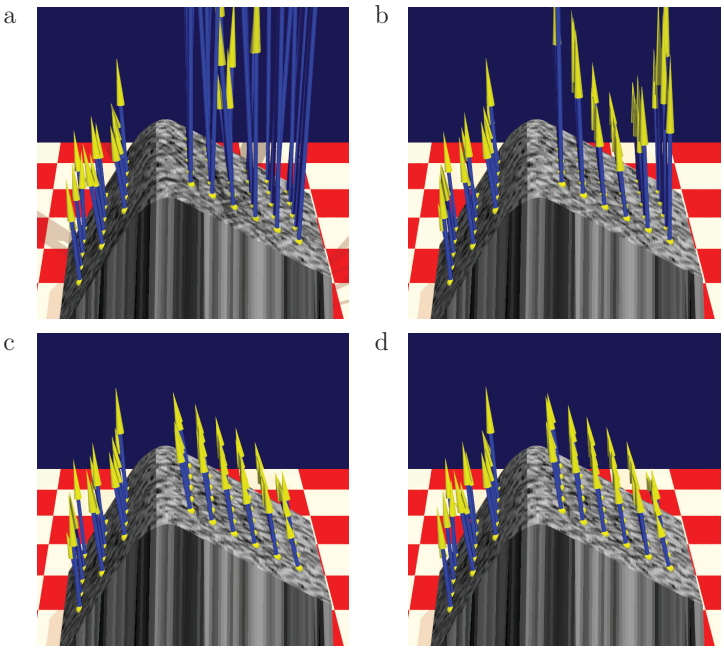


Figure 4.15: Scaled motion estimates with amplified errors for different models: a: INT without prefiltering, b: GRAD with highpass prefiltering, c: INTGRAD with homomorphic prefiltering, d: TAYLOR without prefiltering. The ground truth is shown in Fig. 4.14d.

illumination changes. *Range Flow* estimates there are thus much more accurate. As expected, estimates are substantially improved by the other models where brightness changes are present.

On the left side of the cube all combinations of models and prefilters yield more or less the same results as original *Range Flow*.

On its right side, where the brightness changes dominate, applying a highpass prefilter, for instance together with model GRAD (Fig. 4.15b), improves the motion estimates where spatial brightness changes are small, i.e., in the middle of the light spot. Motion estimates are visibly worse at the borders of the light spot. Model TAYLOR (Fig. 4.15d) and model INTGRAD yield more uniform and more accurate motion vectors for the right side of the cube.

Table 4.1 shows numerical errors of the different models for the regions on the left and right side of the cube (see Fig. 4.14c). We show the average angular error (see App. A) and, since our target application is plant growth estimation, the average relative growth rate and their standard deviation. In this experiment, the cube does not grow, thus estimated relative growth rates should be zero. According to [70],

Table 4.1: Average angular error (AAE) in degrees, average relative growth rate (RGR) in percent per frame and their standard deviations of regions on left and right side of the cube (see Fig. 4.14c). Errors or standard deviations above 1° (AAE) or 1%/frame (RGR) are indicated in red, below 0.1° (AAE) or 0.01%/frame (RGR) in green.

model	prefilter	left region		right region	
		AAE	RGR	AAE	RGR
INT	NO	0.148 ± 0.074	0.093 ± 0.092	4.311 ± 4.024	4.372 ± 9.211
	HP	0.136 ± 0.071	0.108 ± 0.096	0.255 ± 0.278	0.137 ± 0.195
	HOM	0.132 ± 0.069	0.111 ± 0.099	0.061 ± 0.015	-0.007 ± 0.054
GRAD	NO	0.146 ± 0.075	0.096 ± 0.095	0.681 ± 0.719	0.352 ± 0.627
	HP	0.139 ± 0.070	0.111 ± 0.098	0.425 ± 0.455	0.236 ± 0.321
	HOM	0.131 ± 0.067	0.115 ± 0.100	0.062 ± 0.015	-0.007 ± 0.055
INTGRAD	NO	0.150 ± 0.075	0.095 ± 0.094	4.764 ± 4.309	5.055 ± 10.029
	HP	0.143 ± 0.071	0.110 ± 0.097	0.671 ± 0.726	0.395 ± 0.552
	HOM	0.132 ± 0.068	0.113 ± 0.099	0.062 ± 0.015	-0.006 ± 0.053
TAYLOR	NO	0.148 ± 0.074	0.090 ± 0.094	0.056 ± 0.014	-0.009 ± 0.052
	HP	0.136 ± 0.071	0.108 ± 0.096	0.046 ± 0.015	-0.012 ± 0.051
	HOM	0.132 ± 0.069	0.111 ± 0.099	0.061 ± 0.015	-0.007 ± 0.054
[31]	NO	4.941 ± 1.812	-0.739 ± 11.59	2.216 ± 1.223	-3.36 ± 8.694
	HP	8.977 ± 3.714	-3.20 ± 15.51	4.041 ± 4.078	-4.19 ± 1.557
	HOM	8.907 ± 2.995	-1.83 ± 13.02	2.093 ± 0.784	0.347 ± 7.358

$$dA = \frac{|s(x+1,y)+f(x+1,y) - (s(x,y)+f(x,y))| \times |s(x,y+1) + f(x,y+1) - (s(x,y)+f(x,y))|}{|s(x+1,y) - s(x,y)| \times |s(x,y+1) - s(x,y)|} \tag{4.32}$$

the relative area change dA of a local surface s may be calculated by (4.32), when $s(x,y) = [X(x,y), Y(x,y), Z(x,y)]^{\mathrm{T}}$ is parameterized in sensor coordinates x and y, and when the 3D displacement vector field $f(x,y) = [U(x,y), V(x,y), W(x,y)]^{\mathrm{T}}$ is given. The relative growth rate is determined by $RGR = (dA - 1) \cdot 100\%$.

For the left side of the cube, where no brightness changes are present, all presented *Range Flow* models perform comparably. This coincides with the observations in Fig. 4.15. On the right side of the cube, where illumination changes are present, homomorphic prefiltering results in excellent estimates for all models. Model TAYLOR is most robust with respect to prefiltering, whereas results for the other models are much more prefilter dependent.

Additionally, we show results obtained by the recent scene flow approach of Huguet and Devernay [31]. This is essentially a warping technique for stereo camera sequences analogous to the optical flow approach of Papenberg et al. [49]. We apply

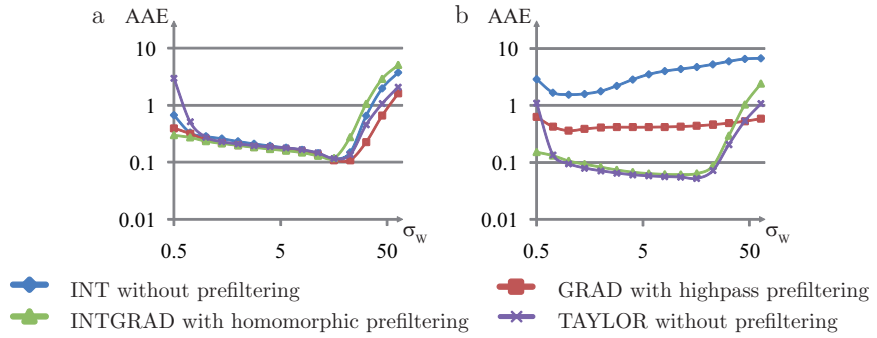


Figure 4.16: Average angular error of the motion estimates in the left and right region, respectively, of the cube for increasing σ_W of weighting matrix $\mathbf{W}$ for different models.

the algorithm as provided by Huguet and Devernay using parameter settings of the *rotating sphere* experiment. Only the weighting parameter γ of the gradient constraint was increased to $\gamma = 30$ in order to reduce the effect of the severe brightness variations in the data. The algorithm can handle 4 input images, i.e., two for each camera in a stereo setup. We chose the first and the last frame of the two outer cameras from the cube data set.

Compared to the original *Range Flow* of [69], and on the right side of the cube, errors are reduced by approximately a factor of 2. However, on the stronger tilted left side of the cube, results are drastically worse than for all other models. In contrast to the other models, prefiltering does not or not significantly improve results. Probably incorporating more input data, i.e., more than four images, may significantly improve results. This would confirm that using an elaborate estimator alone does not necessarily help, instead, the whole estimation framework including appropriate constraints, discretizations and the estimator needs to be optimized.

The cube experiment showed that growth estimates are accurate when homomorphic prefiltering or model TAYLOR are applied. On the right side of the cube, best growth estimates are well below 0.1%/frame and accurate enough for plant growth studies where a relative measurement error of approximately 10% is acceptable. However, variances are high, such that spatial resolution is not yet in the desired range. In addition, results for the accuracy on the left, stronger tilted side of the cube are not yet sufficient.

In Fig. 4.16, we compare the performance of the different models for different sizes of the neighborhood Λ used in the estimation process. As stated in Sec. 4.3.2, the neighborhood Λ is defined by a normalized Gaussian filter $\mathbf{W}$ with standard deviation $\sigma_{\mathbf{W}}$. On the left side of the cube, all models yield similar results for most sizes of $\mathbf{W}$. Only for higher values of σ_W, model INTGRAD with homomorphic prefiltering yields slightly higher errors. For neighborhoods with $\sigma_W > 25$, edges in the data degrade

estimation results. On the right side of the cube, model INT without prefiltering performs worst, as expected. Also, model GRAD with highpass prefiltering yields unsatisfying results. Model INTGRAD with homomorphic prefiltering and model TAYLOR without prefilter perform similar and significantly better than the other models. The optimal size of filter $\mathbf{W}$ is $\sigma_W \lesssim 15$.

We conclude that, if the brightness constancy assumption is violated, homomorphic prefiltering rather than linear highpass prefiltering should be used. Furthermore, the size of the prefilter has a great impact on estimation results and has to be chosen carefully. Nevertheless, even for small neighborhoods modeling brightness changes yields slightly more accurate results than using a model merely attenuating the effects of brightness changes.

4.4.3 Comparison of Models

So far we extended the three models to handle brightness changes coming from varying illumination and tested the influence of prefilters and different brightness constraints Sec. 4.4.2 and Sec. 4.4.1. In this section we compare performance of the novel brightness change model combined with *Scene Flow*, *Range Flow* and the *Affine Model*. Again we do not model brightness change parameters $g_{2,x}$ and $g_{2,y}$, as these have only a negligible effect on the estimation (cmp. Sec. 4.4.1). Analog to the experiments in Sec. 3.4 we show experiments on sinusoidal patterns for evaluation of systematic errors. A rendered sequence showing a moving cube is used to test performance under more realistic conditions and with ground truth available. Estimation results on plant leafs demonstrate the improved performance of the models on real data.

Sinusoidal Pattern

In the first experiment we compare the influence of the brightness change parameters g_1, $g_{1,x}$ and g_2 on the average angular error (cmp. App. A). The sinusoidal sequences are generated with the *Sequence Generator* (cmp. App. B). The basic parameters for all experiments are the same as in Sec. 3.4 and we use a wide baseline setup. Brightness change parameters are set to $g_1 = 0.05$, $g_{1,x} = 0.01$ and $g_2 = 0.02$. Figure 4.17 demonstrates the performance of the three models for increasing brightness change parameters, while all other parameters are fixed. In general we see that *Range Flow* yields lowest angular errors for small brightness changes and for increasing g_2. For larger values of g_1 or $g_{1,x}$ *Scene Flow* and the *Affine Model* perform better. Figure 4.18 depicts average angular errors for fixed brightness changes and increasing U_Z (left) and σ_n (right). For all values $U_Z \neq 0$ *Range Flow* performs best, whereas *Scene Flow* and the *Affine Model* perform up to one order of magnitude worse. This holds also for low noise sequences. In case of increasing noise *Scene Flow* yields lowest errors and *Range Flow* highest.

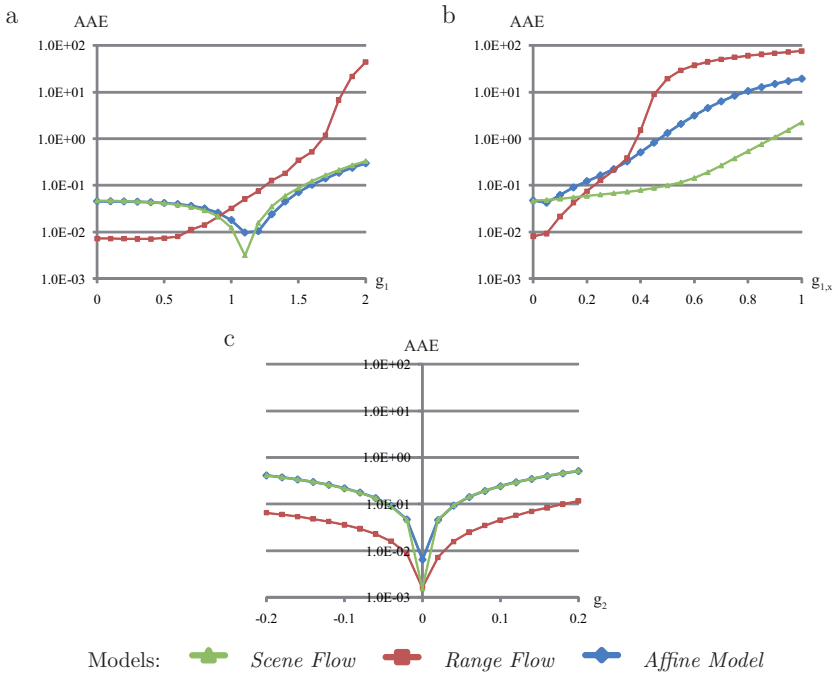


Figure 4.17: Average angular error versus illumination parameters g_1, $g_{1,x}$ and g_2.

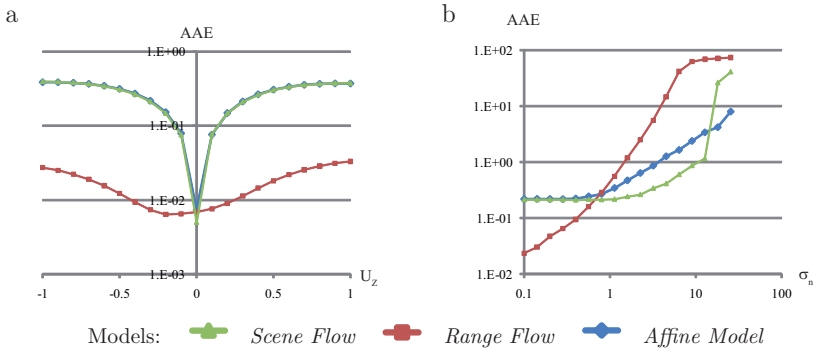


Figure 4.18: Average angular error versus U_Z and σ_n.

Synthetic Cube

Analog to Sec. 3.4 we test estimation of 3D motion on more realistic data with ground truth available. We use the cube sequence rendered with POV-Ray [15] and add varying illumination. Therefore the cube is additionally illuminated by a light spot

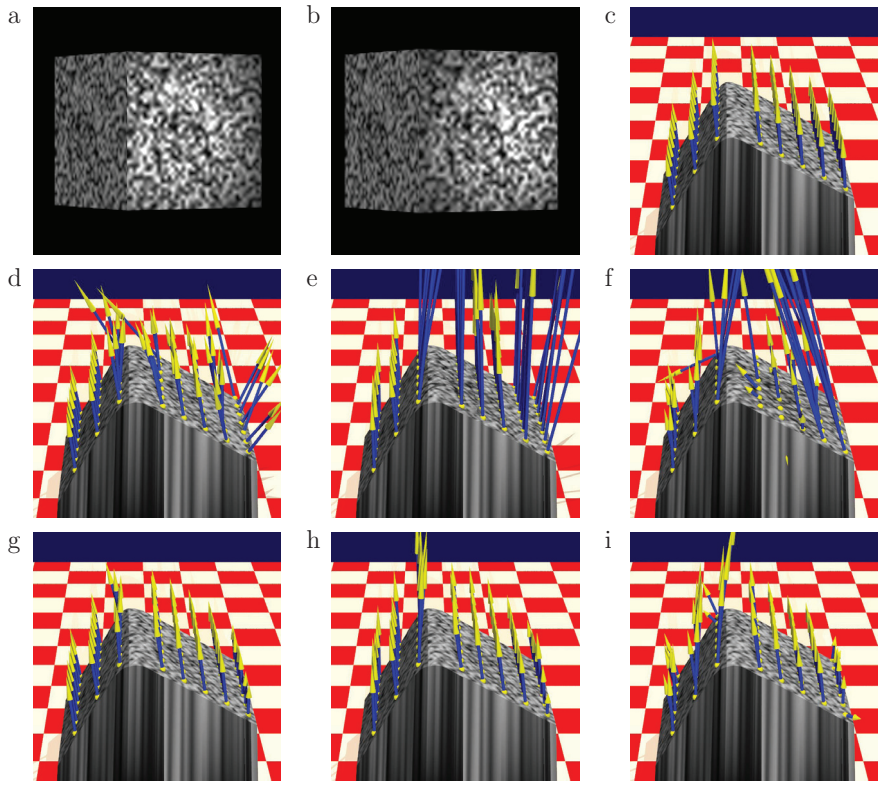


Figure 4.19: Motion estimation of cube moving towards camera illuminated by a spotlight. a and b: First and last frame of the cube sequence. c: ground truth motion estimates. d–i: Scaled motion estimates, middle row: models *without* brightness change modeling: *Scene Flow* (d), *Range Flow* (e) and *Affine Model* (f). Bottom row: Models *with* brightness change modeling: *Scene Flow* (g), *Range Flow* (h) and *Affine Model* (i).

from the right. This causes temporally and spatially varying intensities, as expected by our target application plant leaf motion estimation. We use the same 3D structure tensor weighting matrix $\mathbf{W}$ as in Sec. 3.4 with spatio-temporal size of 65×65 pixel and 5 frames and standard deviation of $\sigma_W = 19$ in space and $\sigma_W = 1$ in time direction. The cube is rotated by 60° degrees and moves with $U_X = -0.2$ mm/frame, $U_Y = 0$ mm/frame and $U_Z = -2$ mm/frame. The first and the last frame of the central camera are shown in Figs. 4.19a and b. Figure 4.19c depicts the ground truth motion. Figures 4.19d-f show estimation results with the three standard models presented in Chap. 3 and Figs. 4.19g-i show estimation results for the corresponding

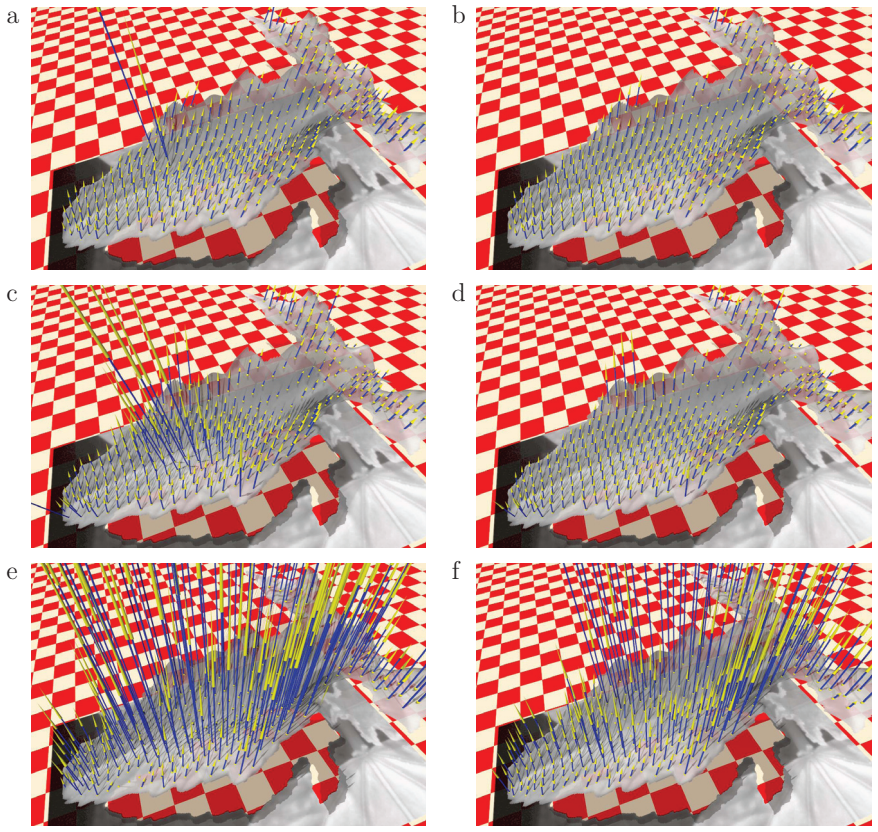


Figure 4.20: *Castor Oil Plant* sequence. Motion estimates for models *without* (left) and *with* (right) brightness change modeling. a and b: *Scene Flow*, c and d: *Range Flow*, e and f: *Affine Model*.

models with brightness change modeling. Errors of motion estimates are scaled by factor by 100 in U_X-and U_Y-directions and 10 in U_Z-direction. Motion estimates of models without brightness change modeling clearly have higher errors compared to the models with brightness change modeling. *Scene Flow* with brightness change modeling performs best, only at borders estimates are distorted. Brightness change modeling shows much more effect on *Range Flow* and the *Affine Model* compared to *Scene Flow*. Nevertheless estimates of *Range Flow* still suffer from errors and borders and overestimation of U_Z. Estimation errors of the *Affine Model* are higher in X- and Y-direction, but U_Z has no significant offset compared to *Range Flow*. However the *Affine Model* yields worst results for this data set.

Castor Oil Plant

The *Castor Oil Plant* sequence proposed in Sec. 3.4 heavily suffers from changing illumination. Especially at the top of the leaf, where the shadow area decreases the standard models yield no reliable estimates. We compare the standard models from Chap. 3 with the models with brightness change modeling in Fig. 4.20. Motion estimates of *Scene Flow* already yield almost everywhere reliable results without brightness change modeling. The novel version outperforms the standard version of *Scene Flow* and no outliers are visible and the estimated velocity field looks even more smooth. Also estimates of *Range Flow* and the *Affine Model* show significantly more smooth results with brightness change modeling. However, as for the results of the cube sequence, *Range Flow* yields distorted motion estimates near depth discontinuities and the *Affine Model* yields less reliable estimates.

4.5 Conclusions

In this section, we extended the three 3D motion estimation models described in Chap. 3 by different approaches to handle inhomogeneous illumination. We derived a novel brightness change constraint, where changes are approximated by a power series. This model allows to handle changes in reflected radiance due to (1) changes of illumination direction and (2) changes in incoming light intensity caused by moving inhomogeneous incident irradiance. While the first effect may be modeled by spatially constant temporal changes, the latter one causes spatially variant temporal changes. In order to test performance of the novel brightness change constraint, it was applied to the *Affine Model*. The sinusoidal pattern experiments reveal that modeling spatial variations of brightness changes results in increased motion estimation accuracy with respect to $g_{1,x}$, but not with $g_{2,x}$ (cmp. (4.16)). Motion vector fields of a translating cube illuminated by a moving spotlight have been estimated using brightness constancy assumption and brightness change model with or without spatial changes. The richest model yields significantly better results than the other ones.
Based on this conclusion we presented a detailed error analysis for the novel brightness change constraint and three different brightness constraints in combination with highpass or homomorphic prefiltering on synthetic image sequences with *Range Flow*. Prefiltering improved estimation results on data when illumination changes were present, however, the standard deviation σ_{pre} of the lowpass used in the filters should be large enough. If not, too much signal is lost, resulting in less accurate motion estimates. Highpass filtering performs well if Gaussian noise or shot noise is present. Homomorphic filtering works excellent for shot noise, but errors increase when Gaussian noise and strong brightness changes are present. We conclude that, except for this case, well tuned homomorphic prefiltering allows for accurate range flow estimation in conjunction with all models. However a wrongly tuned filter severely corrupts motion estimates. Furthermore, prefiltering increased uncertainty

of motion estimates, as relevant signal is then also affected by the filtering.

Modeling brightness changes in *Range Flow* by the models GRAD or TAYLOR instead of highpass prefiltering results in motion estimates of same or higher accuracy in almost all scenarios investigated. The model INTGRAD mostly showed results lying between the results of standard range flow (model INT) and the model GRAD. Generally, best results were achieved by model TAYLOR and, especially when noise was present, without prefiltering. Only for brightness changes with strong spatial variations, i.e., $g_{1,x} \gg 0$, and noise-free data homomorphic prefiltering is recommended. We conclude that suitably modeling brightness changes almost always outperforms other approaches and increases overall accuracy without tuning of prefilters.

A comparison of the three 3D motion estimation models using the novel brightness change model demonstrated, that for synthetic sinusoidal sequences *Range Flow* performs best for low noise and little brightness changes. *Scene Flow* showed best performance for higher noise or larger brightness changes. The *Affine Model* yields same results as *Scene Flow* for low noise and small brightness changes, but in other cases higher errors. On the cube and the plant leaf sequence *Scene Flow* performed best, while *Range Flow* suffered from outliers near borders. The *Affine Model* showed most unreliable results.
The *Affine Model* is based on a translating patch model. This is not true for the plant leaf sequence, because the leaf unfolds and rotates around the point where it is fixed to the stem. In the next section we will extend the *Affine Model* to handle rotation and introduce additional constraints to make 3D motion estimation more reliable.

Chapter 5

Modeling Rotation

The *Affine Model* presented in Sec. 3.3 handles translational motion only. This may lead to severe problems in sequences where objects of interest rotate (cmp. the *Castor Oil Plant* sequence in Fig. 4.20). In this chapter we extend the *Affine Model* to handle rotational motion. Furthermore we introduce two additional local dimensions to the standard 2D affine optical flow model, i.e., we model affine transformations not only in local image coordinates Δx and Δy, but also in local camera displacement Δs and time direction Δt. Analog to the 2D model we identify the additional parameters with terms depending on scene structure, scene motion, and camera displacement. This 4D affine optical flow model helps to make the second estimation process more reliable, i.e., estimation of 3D motion parameters. The derivation is shown for a 1D camera grid only, but extension for a 2D camera grid is analog[1].
As in Sec. 3.3 we interpret the camera position s as additional dimension of the data. Hence all image sequences acquired by a 1D camera grid can be combined to sample a 4D-Volume in x-y-s-t-space. Brightness changes in this space are modeled as total differential of the intensity data. A pinhole camera model is used to project a dynamic surface patch into the image. This yields equations for x- and y-position of the patch as a function of camera position and time. Plugging the linearized total differentials of x and y into the total differential of the intensity results in the sought model. A crucial point here is the correct handling of neighbor locations. We model it by back-projection of the pixel grid to the surface in the scene (see Secs. 5.1.3 and 5.1.4). A detailed derivation of the modified model can be found in Section 5.1.

In order to evaluate the model we use a parameter estimation procedure as proposed in Sec. 3.3.2. Adaptations needed here are presented in Section 5.2.

Quantitative experiments (Sec. 5.3) use synthetic data with ground truth available generated with the sequence generator presented in App. B. For more realistic scenes with ground truth available we use another cube sequence rendered by POV-Ray [15] in 8bit-integer accuracy. Finally we demonstrate improvements coming from modeling rotation on the *Castor Oil Plant* sequences and compare results to *Scene*

[1] In case of a 2D camera model we achieve a 5D affine optical flow model with local camera displacement coordinates Δs_x and Δs_y

Flow and *Range Flow*.

5.1 Model Derivation

We briefly review the derivation of the *Affine Model* for translational motion and its extension to 4D affine optical flow. Afterwards we will model rotational motion of the patch.

5.1.1 Surface Patch Model

Following Sec. 3.3 we model a surface patch at world coordinate position (X_0, Y_0, Z_0) as a function of time t by

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} X_0 + U_X t + \Delta X \\ Y_0 + U_Y t + \Delta Y \\ Z_0 + U_Z t + Z_X \Delta X + Z_Y \Delta Y \end{pmatrix} \tag{5.1}$$

where Z_X and Z_Y are surface slopes in X- and Y-direction for time $t = 0$ and $\mathbf{U} = (U_X, U_Y, U_Z)$ is the velocity of the patch. The surface normal is then $\mathbf{n} = (Z_X, Z_Y, -1)$.

5.1.2 Camera Model

We use pinhole cameras at world coordinate positions $(s, 0, 0)$, looking into Z-direction

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} X - s \\ Y \end{pmatrix} \tag{5.2}$$

Sensor coordinates x, y are aligned with world coordinates X, Y. Camera position space is sampled equidistantly using a 1D camera grid. Modeling steps taken in following can also be applied for a 2D camera grid (e.g., [45, 77]), cmp. [62]. Here we focus on 1D grids and combine data acquired by all cameras into one 4D data set equidistantly sampling the continuous intensity function $I(x, y, s, t)$.

5.1.3 Pixel-Centered View

Parameter estimation at a 4D pixel $\mathbf{x_0} = (x_0, y_0, s_0, t_0)$ is done using the acquired image data $I(\mathbf{x}) := I(x, y, s, t)$ in a local neighborhood Λ, with $\mathbf{x} := (x, y, s, t)^{\mathrm{T}}$. Consequently we need to know the surface position $\mathbf{X}$ for each 4D pixel, i.e., $\mathbf{X}(\mathbf{x})$. Using a pinhole camera (5.2) we know

$$\begin{pmatrix} X(\mathbf{x}) \\ Y(\mathbf{x}) \end{pmatrix} = \frac{Z(\mathbf{x})}{f} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} s \\ 0 \end{pmatrix} . \tag{5.3}$$

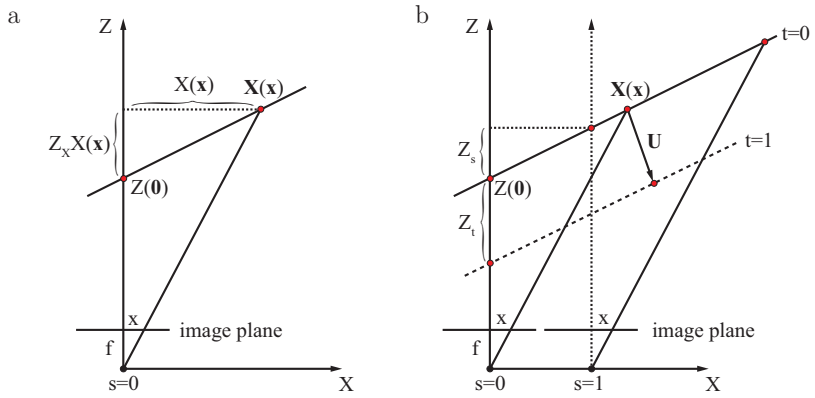


Figure 5.1: Relationship between $\mathbf{X}_0$, $Z(\mathbf{0})$ (left) and Z_t and Z_s (right).

In order to derive an expression for $Z(\mathbf{x})$ we fit a tangent plane with surface normal $\mathbf{n} = (Z_X, Z_Y, -1)^{\mathrm{T}}$ to the point $\mathbf{X}(\mathbf{x})$. The intersection of this tangent plane with the Z-axis is then $Z(0,0,0,t)$, and $Z(0,0,0,t) = Z(\mathbf{0}) + Z_t t$ for a constantly translating plane, where $\mathbf{0} := (0,0,0,0)^{\mathrm{T}}$. Consequently $Z(\mathbf{x})$ can be expressed as

$$\begin{aligned} & Z(\mathbf{x}) = \quad Z(\mathbf{0}) + Z_X X(\mathbf{x}) + Z_Y Y(\mathbf{x}) + Z_t t \\ \Leftrightarrow \quad & Z(\mathbf{x}) = \quad \tfrac{Z(\mathbf{0}) + Z_X s + Z_t t}{1 - Z_X \frac{x}{f} - Z_Y \frac{y}{f}} \end{aligned} \tag{5.4}$$

where we used (5.3) to substitute $X(\mathbf{x})$ and $Y(\mathbf{x})$. Combining (5.3) and (5.4) yields

$$\mathbf{X}(\mathbf{x}) \quad = \quad \tfrac{Z(\mathbf{0}) + Z_X s + Z_t t}{f - Z_X x - Z_Y y} \begin{pmatrix} x \\ y \\ f \end{pmatrix} + \begin{pmatrix} s \\ 0 \\ 0 \end{pmatrix} \quad . \tag{5.5}$$

Equation (5.5) extends the formulation in [62], where $\mathbf{X}$ only depends on local image coordinates x and y. Figure 5.1 depicts the relationship between $\mathbf{X}(\mathbf{x})$ and $Z(\mathbf{0})$ for a point in the central camera and frame (Fig. 5.1a) and for varying s and t (Fig. 5.1b). Obviously depth seen at the principal point of a camera changes due to motion of the camera or the patch. These changes are addressed by Z_s and Z_t.

5.1.4 Projecting the Pixel Grid to the Surface

A pixel $\mathbf{x}$ in the local neighborhood Λ used for parameter estimation is given by $\mathbf{x} = \mathbf{x_0} + \Delta\mathbf{x} = (x_0 + \Delta x, y_0 + \Delta y, s_0 + \Delta s, t_0 + \Delta t)^{\mathrm{T}}$. The surface patch center $\mathbf{X_0}$ by definition is the back projection of the neighborhood center point $\mathbf{x_0}$ to the surface. Consequently neighbor points of $\mathbf{X_0}$ on the surface given by $\Delta\mathbf{X} = (\Delta X, \Delta Y, \Delta Z)$ are projections of the cameras pixel grids to the moving surface and we need to

derive $\Delta\mathbf{X}(\Delta\mathbf{x})$. In order to stay on the surface, we model $\Delta Z = Z_X \Delta X + Z_Y \Delta Y$, cmp. (5.1). Further we know from (5.1) that

$$\Delta\mathbf{X} = \mathbf{X} - \mathbf{X}_0 - \mathbf{U}t \tag{5.6}$$

where $\mathbf{X}$ is a point on the surface at a given point in time t, $\mathbf{X}_0$ is the surface patch center at time $t_0 = 0$, and $\Delta\mathbf{X}$ is the distance between $\mathbf{X}$ and the point $\mathbf{X}_0 + \mathbf{U}t$ where the patch center moved to. The distance between $\mathbf{X}$ and $\mathbf{X_0}$ can expressed by the linearization

$$\mathbf{X} - \mathbf{X}_0 = \frac{\partial\mathbf{X}}{\partial x}\Delta x + \frac{\partial\mathbf{X}}{\partial y}\Delta y + \frac{\partial\mathbf{X}}{\partial s}\Delta s + \frac{\partial\mathbf{X}}{\partial t}\Delta t \ . \tag{5.7}$$

The partial derivatives of $\mathbf{X}(\mathbf{x})$ are (cmp. (5.5))

$$\begin{aligned} \frac{\partial\mathbf{X}}{\partial x} &= \frac{Z}{fc}\begin{pmatrix} 1 - Z_Y\frac{y}{f} \\ Z_X\frac{y}{f} \\ Z_X \end{pmatrix}, \quad \frac{\partial\mathbf{X}}{\partial y} = \frac{Z}{fc}\begin{pmatrix} Z_Y\frac{x}{f} \\ 1 - Z_X\frac{x}{f} \\ Z_Y \end{pmatrix} \\ \frac{\partial\mathbf{X}}{\partial s} &= \frac{1}{c}\begin{pmatrix} c + Z_X\frac{x}{f} \\ Z_X\frac{y}{f} \\ Z_X \end{pmatrix}, \quad \frac{\partial\mathbf{X}}{\partial t} = \frac{Z_t}{fc}\begin{pmatrix} x \\ y \\ f \end{pmatrix} \end{aligned} \tag{5.8}$$

where we used the notation

$$c = 1 - Z_X\frac{x}{f} - Z_Y\frac{y}{f} \ . \tag{5.9}$$

Substituting (5.7) and (5.8) in (5.6) yields the sought expression for $\Delta\mathbf{X}(\Delta\mathbf{x})$.

5.1.5 Brightness Change Model

We model intensity changes analog to Chap. 4, i.e., $\mathrm{d}I$ of a surface element for such a multi-dimensional data set is

$$\begin{aligned} \mathrm{d}I &= I_x\mathrm{d}x + I_y\mathrm{d}y + I_s\mathrm{d}s + I_t\mathrm{d}t \\ &= I(g_1 + g_{1,x}\Delta X + g_{1,y}\Delta Y + g_2 t)\mathrm{d}t \ . \end{aligned} \tag{5.10}$$

We denote $I_* = \frac{\partial I}{\partial *}$ for partial derivatives of the image intensities I. In the following we use notation $g = (g_1 + g_{1,x}\Delta X + g_{1,y}\Delta Y + g_2 t)$ for the intensity change function. This brightness change constraint equation (BCCE) models spatially constant and spatially varying brightness changes stemming from varying illumination.

5.1.6 Combination of Models

Following Sec. 3.3 we project the moving surface patch model to the sensor plane using the pinhole camera model and calculate the differentials $\mathrm{d}x$ and $\mathrm{d}y$ for a *given*

surface location (i.e., for constant ΔX and ΔY)

$$\begin{pmatrix} \mathrm{d}x \\ \mathrm{d}y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} (U_X - U_Z\frac{x}{f})\mathrm{d}t - \mathrm{d}s \\ (U_Y - U_Z\frac{y}{f})\mathrm{d}t \end{pmatrix} . \tag{5.11}$$

This equation is nonlinear in the sought for 3D parameters. Furthermore this equation depends on ΔX and ΔY in Z. We simplify it by rephrasing f/Z via the surface patch model (5.1), i.e., $Z = Z_0 + U_Z\Delta t + Z_X\Delta X + Z_Y\Delta Y$. Inserting Eqs. (5.6)–(5.9) leads to

$$\frac{f}{Z} = -\nu - b_1\Delta x - b_2\Delta y - b_3\Delta s - b_4\Delta t \tag{5.12}$$

$$\text{with} \quad \nu = -\tfrac{f}{Z_0}\ , \quad b_1 = \tfrac{Z_X}{Z_{0c}}\ , \quad b_2 = \tfrac{Z_Y}{Z_{0c}}\ , \quad b_3 = \tfrac{f}{Z_0}\tfrac{Z_X}{Z_c}\ , \quad b_4 = \tfrac{f}{Z_0}\tfrac{Z_t}{Z_c} \tag{5.13}$$

Combination of the proposed models is done analog to [62]. In the first step Equations (5.12), (5.1) and (5.6)–(5.9) are plugged into (5.11), using local coordinates $\mathbf{x} = \mathbf{x_0} + \Delta\mathbf{x}$. Sorting by differentials and Δ-terms, and ignoring higher order Δ-terms we substitute $\mathrm{d}x$ and $\mathrm{d}y$ in the brightness change model (5.10). This yields

$$\begin{aligned}
\mathrm{d}x &= \tfrac{f}{Z}\left[\left(U_X - \tfrac{x}{f}U_Z\right)\mathrm{d}t - \mathrm{d}s\right] \\
&= -\nu\left(U_X - \tfrac{x}{f}U_Z\right)\mathrm{d}t \\
&- \left[b_1\left(U_X - \tfrac{x_0}{f}U_Z\right) + \tfrac{U_Z}{Z_0}\right]\Delta x\mathrm{d}t \\
&- \left[b_2\left(U_X - \tfrac{x_0}{f}U_Z\right)\right]\Delta y\mathrm{d}t \\
&- \left[b_3\left(U_X - \tfrac{x_0}{f}U_Z\right)\right]\Delta s\mathrm{d}t \\
&- \left[b_4\left(U_X - \tfrac{x_0}{f}U_Z\right)\right]\Delta t\mathrm{d}t \\
&+ \nu\mathrm{d}s + b_1\Delta x\mathrm{d}s + b_2\Delta y\mathrm{d}s \\
&+ b_3\Delta s\mathrm{d}s + b_4\Delta t\mathrm{d}s
\end{aligned} \tag{5.14}$$

and

$$\begin{aligned}
\mathrm{d}y &= \tfrac{f}{Z}\left(U_Y - \tfrac{y}{f}U_Z\right)\mathrm{d}t \\
&= -\nu\left(U_Y - \tfrac{y_0}{f}U_Z\right)\mathrm{d}t \\
&- \left[b_1\left(U_Y - \tfrac{y_0}{f}U_Z\right)\right]\Delta x\mathrm{d}t \\
&- \left[b_2\left(U_Y - \tfrac{y_0}{f}U_Z\right) + \tfrac{U_Z}{Z_0}\right]\Delta y\mathrm{d}t \\
&- \left[b_3\left(U_Y - \tfrac{y_0}{f}U_Z\right)\right]\Delta s\mathrm{d}t \\
&- \left[b_4\left(U_Y - \tfrac{y_0}{f}U_Z\right)\right]\Delta t\mathrm{d}t
\end{aligned} \tag{5.15}$$

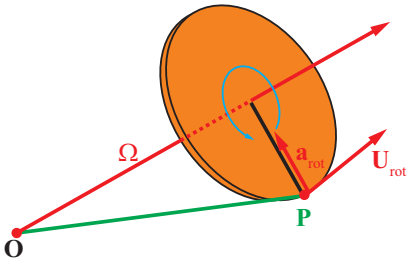


Figure 5.2: Rotation $\mathbf{U}_{rot}$ at distance $\mathbf{P}$ due to angular velocity vector $\mathbf{\Omega}$.

5.1.7 Rotation

We define rotation of a surface patch by angular velocity vector $\mathbf{\Omega} = (\Omega_X, \Omega_Y, \Omega_Z)$ located at its central point $\mathbf{X}_0 = (X_0, Y_0, Z_0)$. Figure 5.2 shows general parameters of rotational motion, namely angular velocity vector Ω, rotation center $\mathbf{O}$, distance to the rotation axis $\mathbf{P}$, velocity $\mathbf{U}_{rot}$ and angular acceleration a_{rot}. Rotational motion $\mathbf{U}_{rot}$ may be obtained by

$$\mathbf{U}_{rot} = \mathbf{\Omega} \times \mathbf{P}. \tag{5.16}$$

Overall translational and rotational velocity $\mathbf{U}$ of points on the surface patch, i.e., $\mathbf{P} = \Delta\mathbf{X}$, is then determined by

$$\mathbf{U} = \mathbf{N} + \mathbf{U}_{rot} = \mathbf{N} + \mathbf{\Omega} \times \Delta\mathbf{X} \tag{5.17}$$

with translational velocity $\mathbf{N} = (N_X, N_Y, N_Z)^{\mathrm{T}}$, distance to the rotation center $\Delta\mathbf{X}$ and angular velocity $\mathbf{\Omega}$.

Equation (5.17) defines translation of and rotation around the surface patch center. For general rotational motion this model is not sufficient, as the true center of rotation may not coincide with the patch center. This leads to accelerated motion of the patch center. We approximate this non-constant motion by acceleration $\mathbf{A}$

$$\mathbf{U} = \mathbf{N} + \mathbf{A}t + \mathbf{\Omega} \times \Delta\mathbf{X}. \tag{5.18}$$

We address only constant acceleration, whereas acceleration coming from rotation is non-constant. This could be modeled by estimation of the true rotation center introducing 3 additional parameters analog to (5.17).

5.1.8 The Range Constraint, Z_t, b_4, and why (5.5) still holds under Rotation

Flow parameter b_4 (5.13) linearly depends on Z_t, i.e., the partial derivative of Z with respect to time. We are not explicitly interested in Z_t, therefore we want to

express it using parameters we are interested in. We know that $U_Z := dZ/dt$. The time derivative of the first equation in (5.4) therefore yields

$$Z_t = U_Z - Z_X U_X - Z_Y U_Y \tag{5.19}$$

for *translating* planes, i.e., for constant surface slopes Z_X and Z_Y. This is the so-called *range constraint* known from [69] (cmp. (3.5)). Obviously surface slopes change when a surface rotates and the range constraint becomes

$$Z_t = U_Z - Z_X U_X - Z_Y U_Y - X Z_{X,t} - Y Z_{Y,t} \tag{5.20}$$

$Z_{X,t}$ and $Z_{Y,t}$ are first order time derivatives of Z_X and Z_Y.

Equation (5.4) was derived for translating surfaces. We model the change in (5.4) due to rotation by replacing Z_X and Z_Y by their time depending counterparts $Z_X(t)$ and $Z_Y(t)$. We approximate the change of the surface normals by first order Taylor expansions $Z_X(t) = Z_X(0) + Z_{X,t}t$ and $Z_Y(t) = Z_Y(0) + Z_{Y,t}t$ and derive for (5.4)

$$\begin{aligned} Z(\mathbf{x}) &= Z(\mathbf{0}) + Z_X(t)X(\mathbf{x}) + Z_Y(t)Y(\mathbf{x}) + Z_t t \\ \Leftrightarrow Z(\mathbf{x}) &- Z_X(\mathbf{0})Z(\mathbf{x})\tfrac{x}{f} - Z_Y(\mathbf{0})Z(\mathbf{x})\tfrac{y}{f} \\ &= Z(\mathbf{0}) + Z_X s + (Z_t + Z_{X,t}X(\mathbf{x}) + Z_{Y,t}Y(\mathbf{x}))t \\ \Leftrightarrow Z(\mathbf{x}) &= \tfrac{Z(\mathbf{0}) + Z_X(\mathbf{0})s + (Z_t + Z_{X,t}X(\mathbf{x}) + Z_{Y,t}Y(\mathbf{x}))t}{c} \end{aligned} \tag{5.21}$$

Substituting Z_t using (5.20) yields

$$\begin{aligned} Z(\mathbf{x}) &= \tfrac{Z(\mathbf{0}) + Z_X s + (U_Z - Z_X U_X - Z_Y U_Y)t}{c} \\ \Leftrightarrow\ Z(\mathbf{x}) &= \tfrac{Z(\mathbf{0}) + Z_X s + \tilde{Z}_t t}{c} \end{aligned} \tag{5.22}$$

where now $\tilde{Z}_t$ is *defined* by the standard range constraint, i.e.,

$$\tilde{Z}_t := U_Z - Z_X U_X - Z_Y U_Y \ . \tag{5.23}$$

We conclude that (5.5) still holds for a first order model of rotational motion, if $\tilde{Z}_t$ ignores surface slope changes due to rotation. Consequently Z_t in (5.13) also becomes $\tilde{Z}_t$.

5.1.9 A 4D-*Affine Model*

Collecting terms and equations derived so far leads to the sought 4D affine model. We derive it by inserting (5.6) in (5.18), substitution of velocity vector $\mathbf{U}$ in (5.14) and (5.15). Ignoring higher order terms yields representations of the elements of the 4 dimensional optical flow model (5.24) with parameters (5.25). The partial derivatives of world coordinates in (5.25) are given in (5.8).

$$\nabla^{\mathrm{T}} I \left[\begin{pmatrix} u_x \mathrm{d}t + \nu \mathrm{d}s \\ u_y \mathrm{d}t \end{pmatrix} + \begin{pmatrix} a_{11}\mathrm{d}t + b_1 \mathrm{d}s & a_{12}\mathrm{d}t + b_2\mathrm{d}s & a_{13}\mathrm{d}t + b_3\mathrm{d}s & a_{14}\mathrm{d}t + b_4\mathrm{d}s \\ a_{21}\mathrm{d}t & a_{22}\mathrm{d}t & a_{23}\mathrm{d}t & a_{24}\mathrm{d}t \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \\ \Delta t \end{pmatrix} \right]$$
$$+ I_s \mathrm{d}s + I_t \mathrm{d}t = I\mathbf{g}\mathrm{d}t \tag{5.24}$$

$$\begin{aligned}
u_x &= -\nu \left(N_X - \tfrac{x_0}{f} N_Z \right) \\
u_y &= -\nu \left(N_Y - \tfrac{y_0}{f} N_Z \right) \\
a_{11} &= -\nu \left[\Omega_Y \tfrac{\partial Z}{\partial x} - \Omega_Z \tfrac{\partial Y}{\partial x} - \tfrac{x_0}{f} \left(\Omega_X \tfrac{\partial Y}{\partial x} - \Omega_Y \tfrac{\partial X}{\partial x} \right) - \tfrac{N_Z}{Z_0} \right] && -b_1 \left(N_X - \tfrac{x_0}{f} N_Z \right) \\
a_{12} &= -\nu \left[\Omega_Y \tfrac{\partial Z}{\partial y} - \Omega_Z \tfrac{\partial Y}{\partial y} - \tfrac{x_0}{f} \left(\Omega_X \tfrac{\partial Y}{\partial y} - \Omega_Y \tfrac{\partial X}{\partial y} \right) \right] && -b_2 \left(N_X - \tfrac{x_0}{f} N_Z \right) \\
a_{13} &= -\nu \left[\Omega_Y \tfrac{\partial Z}{\partial s} - \Omega_Z \tfrac{\partial Y}{\partial s} - \tfrac{x_0}{f} \left(\Omega_X \tfrac{\partial Y}{\partial s} - \Omega_Y \tfrac{\partial X}{\partial s} \right) \right] && -b_3 \left(N_X - \tfrac{x_0}{f} N_Z \right) \\
a_{14} &= -\nu \left[\Omega_Y \tfrac{\partial Z}{\partial t} - \Omega_Z \tfrac{\partial Y}{\partial t} - \tfrac{x_0}{f} \left(\Omega_X \tfrac{\partial Y}{\partial t} - \Omega_Y \tfrac{\partial X}{\partial t} \right) + \left(A_X - \tfrac{x_0}{f} A_Z \right) \right] && -b_4 \left(N_X - \tfrac{x_0}{f} N_Z \right) \\
a_{21} &= -\nu \left[\Omega_Z \tfrac{\partial X}{\partial x} - \Omega_X \tfrac{\partial Z}{\partial x} - \tfrac{y_0}{f} \left(\Omega_X \tfrac{\partial Y}{\partial x} - \Omega_Y \tfrac{\partial X}{\partial x} \right) \right] && -b_1 \left(N_Y - \tfrac{y_0}{f} N_Z \right) \\
a_{22} &= -\nu \left[\Omega_Z \tfrac{\partial X}{\partial y} - \Omega_X \tfrac{\partial Z}{\partial y} - \tfrac{y_0}{f} \left(\Omega_X \tfrac{\partial Y}{\partial y} - \Omega_Y \tfrac{\partial X}{\partial y} \right) - \tfrac{N_Z}{Z_0} \right] && -b_2 \left(N_Y - \tfrac{y_0}{f} N_Z \right) \\
a_{23} &= -\nu \left[\Omega_Z \tfrac{\partial X}{\partial s} - \Omega_X \tfrac{\partial Z}{\partial s} - \tfrac{y_0}{f} \left(\Omega_X \tfrac{\partial Y}{\partial s} - \Omega_Y \tfrac{\partial X}{\partial s} \right) \right] && -b_3 \left(N_Y - \tfrac{y_0}{f} N_Z \right) \\
a_{24} &= -\nu \left[\Omega_Z \tfrac{\partial X}{\partial t} - \Omega_X \tfrac{\partial Z}{\partial t} - \tfrac{y_0}{f} \left(\Omega_X \tfrac{\partial Y}{\partial t} - \Omega_Y \tfrac{\partial X}{\partial t} \right) + \left(A_Y - \tfrac{y_0}{f} A_Z \right) \right] && -b_4 \left(N_Y - \tfrac{y_0}{f} N_Z \right)
\end{aligned} \tag{5.25}$$

5.2 Parameter Estimation

We estimate parameters in three steps. **1.** We solve for 4D affine optical flow parameters ν, b_1, …, b_4, u_x, u_y, a_{11}, …, a_{24} and brightness change parameters g_1, $g_{1,x}$, $g_{1,y}$ and g_2 using the total least squares method described in [62]. **2.** We solve for depth Z_0, and surface normals Z_X and Z_Y, and Z_t using (5.13), where focal length f has to be known e.g., from a calibration step. This allows to calculate c from (5.9) and partial derivatives of world coordinates from (5.8). **3.** We solve for translation $\mathbf{N}$, and rotation $\mathbf{\Omega}$ if desired, using the equations in (5.25) or – as reference methods – using *Scene Flow* from Sec. 3.1 or *Range Flow* from Sec. 3.2. Acceleration $\mathbf{A}$ can only be estimated up to 1 degree of freedom as we have only 2 equations (the ones for a_{14} and a_{24}) for 3 parameters A_X, A_Y, A_Z. Equations (5.25) and (5.13) being an overdetermined system of equations, there are several ways to solve for $\mathbf{N}$ and $\mathbf{\Omega}$ using a standard least squares estimation scheme. For our experiments we select the following submodels by selecting some or all equations from (5.25) and (5.13), or by removing terms when parameters like rotation or acceleration are not estimated. This is equivalent to not modeling these parameters or setting them to zero.

We use the following submodels

2D OF trans. estimates $\mathbf{N}$ only, using equations for $u_x, u_y, a_{11}, a_{12}, a_{21}, a_{22}$.

2D OF rot. estimates $\mathbf{N}$ and $\mathbf{\Omega}$ using equations for $u_x, u_y, a_{11}, a_{12}, a_{21}, a_{22}$.

2D OF trans. and .../2D OF rot. and ... using additional equations indicated by dots (...)

4D OF trans. estimates $\mathbf{N}$ only, using all 11 equations containing motion information, 10 from (5.25) and 1 from (5.13), i.e., the one for b_4.

4D OF rot. estimates $\mathbf{N}$ and $\mathbf{\Omega}$ only, using the 11 equations.

4D OF estimates $\mathbf{N}$, $\mathbf{\Omega}$, and $\mathbf{A}$ using the 11 equations.

Parameters that are not solved for are set to zero. *4D OF* uses two equations more than *2D OF rot. and* a_{13}, a_{23}, b_4 but estimates $\mathbf{A}$ in addition. We therefore get identical results for $\mathbf{N}$ and $\mathbf{\Omega}$ using the two models. Thus we do not show results for *4D OF* in the experiments below.

5.3 Experiments

In a first experiment we use synthetic sinusoidal sequences for systematic error analysis. Then performance of different models is compared on more realistic data with ground truth available, i.e., a translating and rotating cube rendered with POV-Ray [15]. Finally we show estimation results of a sequence showing a rotating plant leaf. In all experiments we minimize systematic discretization errors coming from derivatives by using optimized 5-tab derivative filter sets presented in [54].

5.3.1 Sinusoidal Pattern

For systematic error analysis we render a surface patch with sinusoidal pattern with the *Sequence Generator* (cmp. App. B), where geometry and intensities mimic typical settings used in our actual lab experiments with plants. The 32-bit float intensity values are in the range $[50; 150]$. Figure 5.3 shows first and last frame of such an input sequence with surface patch parameters $Z_0 = 100$ mm, $Z_X = 0.6$, and $Z_Y = -0.5$, and motion parameters $\mathbf{N} \approx (0.0073, -0.0037, 0.06)^{\mathrm{T}}$ mm/frame and $\omega = 0.003$ degree/frame around rotational axis $\mathbf{v} = (2, 3, 2)^{\mathrm{T}}$, i.e., $\Omega = \omega\mathbf{v}$. In each experiment we vary only one of these parameters. The synthetic sensor contains 501×501 pixels with width and height 0.0044 mm. The focal length of the projective camera is set to $f = 12$ mm. We generate data for 9 cameras, positioned horizontally as a 1D, equidistantly spaced camera grid with displacement of 0.5 mm. In order to keep optical flow in camera displacement direction below 1 pixel/displacement, we preshift the data by 13 pixel/displacement. The effective image size shrinks to 301×301

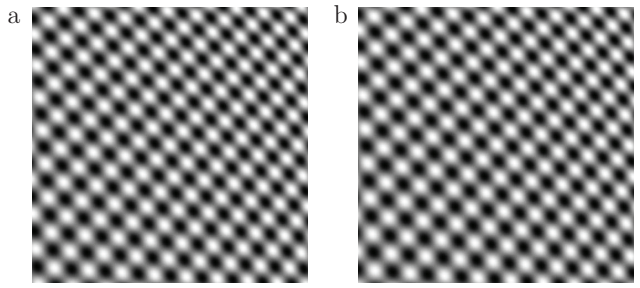


Figure 5.3: a: First and b: last frame of the input sequence with translation and rotation.

pixel due to border effects. Neighborhood Λ is implemented by a Gaussian filter with size $65 \times 65 \times 5 \times 5$ and standard deviations $19 \times 19 \times 1 \times 1$ in x, y, s, t-directions. In order to compare performance of models, we use the average angular error (see App. A)

$$AAE = \frac{1}{N} \sum_{i=1}^{N} \arccos\left(\mathbf{r}_t(i)^{\mathrm{T}} \mathbf{r}_e(i)\right) \tag{5.26}$$

for N pixel with a minimum distance of 60 pixel to the nearest border, true motion $\mathbf{r}_t$ and estimated motion $\mathbf{r}_e$, with $\mathbf{r} = (\mathbf{N}^{\mathrm{T}}, 1)^{\mathrm{T}}$ for translation and $\mathbf{r} = (\mathbf{\Omega}^{\mathrm{T}}, 1)^{\mathrm{T}}$ for rotation.

Figure 5.4 shows average angular errors of translational motion estimates for sequences without (top) and with (bottom) rotation. We show errors for increasing translational motion N_Z in Figs. 5.4a and c and for increasing standard deviation of noise σ_n in Figs. 5.4b and **d**. Figures 5.4a and b demonstrate that all models perform almost equally well for sequences without rotation. *Scene Flow* performs best in almost all cases. Affine models perform better when using more affine parameters (*4D OF trans./rot.*, and *2D OF rot. and* a_{13}, a_{23}, b_4). *Range Flow* performs only slightly better than affine models for low noise sequences. In case of rotation (Figs. 5.4c and d) *Range Flow* and the translational models yield high errors compared to rotational models, *Scene Flow* again performs best. However, comparing rotational models, *4D OF rot.* performs worst. This indicates that modeling $\mathbf{A}$ in the equations for a_{14} and a_{24} ((5.25)) or not using a_{14} and a_{24} (i.e., *2D OF rot. and* a_{13}, a_{23}, b_4) is beneficial. In case of noise *Scene Flow* shows best performance up to $\sigma_n = 10$. Rotational models using 4D affine terms (*2D OF rot. and* a_{13}, a_{23}, b_4 and *4D OF rot.*) yield better results than the 2D model (*2D OF rot.*). *4D OF trans.* shows considerable better performance than *Range Flow*, despite for $N_Z = 0$ and no noise, and performs as good as the best rotational models for $1 < \sigma_n < 10$. In Fig. 5.5 we compare average angular errors of translational (Fig. 5.5a) and rotational (Figs. 5.5b, c and d) motion parameters for different rotation models. Translational models and *Range Flow* are shown for reference in Fig. 5.5a. Figures 5.5c and d show angular errors of rotational

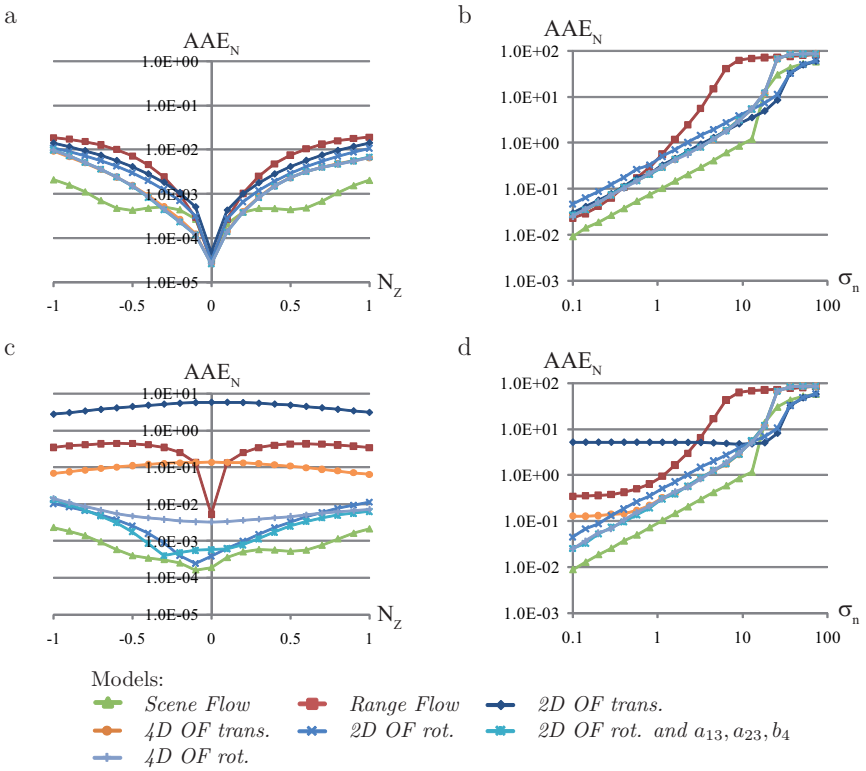


Figure 5.4: Average angular error (AAE) of $\mathbf{N}$ versus increasing N_Z (left) and σ_n (right). a and b: without rotation, c and d: with rotation.

parameters for a rotating sequence and increasing N_Z and σ_n, respectively, i.e., the $\mathbf{\Omega}$ counterparts of Figs. 5.4c and d.

The figures demonstrate that incorporating the affine parameters a_{14} and a_{24} in *4D OF rot.* without modeling of acceleration significantly increases errors. Figures 5.5a and b show average angular errors of translational and rotational parameters for sequences with increasing ω. Rotational models without a_{14} and a_{24} perform similar and up to three orders of magnitude better than *Range Flow* and the translational models, only *Scene Flow* yields lower errors.

We conclude that modeling rotation yields almost always significantly lower or at least similar errors as the translational models and *Range Flow*. *Scene Flow* performs best in all cases, but is not able to estimate angular velocity. Using a_{14} and a_{24} without modeling acceleration $\mathbf{A}$ should be avoided.

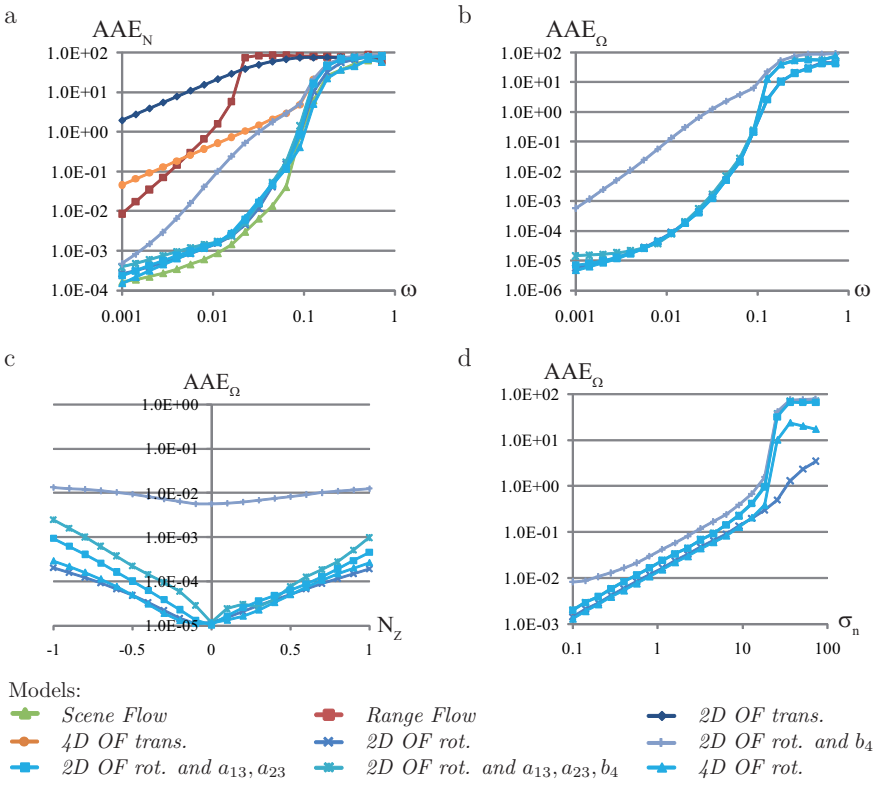


Figure 5.5: a: AAE of $\mathbf{N}$ versus ω, and AAE of $\mathbf{\Omega}$ versus b: ω, c: N_Z and d: σ_n.

5.3.2 Synthetic Cube

The synthetic cube sequence allows us to compare models on more realistic data with ground truth available. The cube is moving with $\mathbf{N} = (-0.2, 0, -1)^{\mathrm{T}}$ mm/frame and rotates around its Y-axis with $\omega = 0.4$ degrees/frame. The cube is covered with a noise pattern in order to make local estimation reliable. Neighborhood Λ is the same as for the sinusoidal sequences. The 1D camera grid contains 9 cameras with a displacement of 5 mm. Figure 5.6a and b show first and last frame of the central camera with regions where errors are evaluated. Ground truth motion is depicted in Fig. 5.6c.

Figures 5.6d–i show estimation results of *Scene Flow*, *Range Flow*, the translational models *2D OF trans.* and *4D OF trans.* and the rotational models *2D OF rot.* and *2D OF rot. and* a_{13}, a_{23}, b_4. The errors are amplified by a factor of 5 for better comparison of the models. The estimates of *2D OF trans.* clearly show large errors, where estimates on the right side of the cube point in Z-direction, estimates on

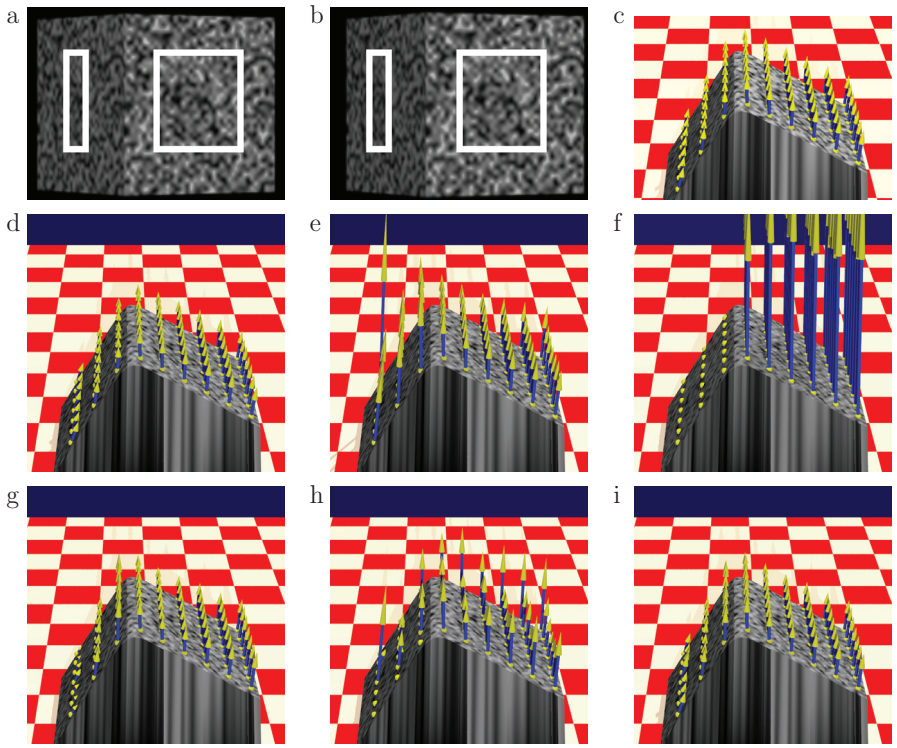


Figure 5.6: Cube moving towards camera with rotation. a and b: First and last input frame with evaluation areas and c: ground truth motion. Motion estimates $\mathbf{N}$ with amplified errors. d: *Scene Flow*, e: *Range Flow*, f: *2D OF trans.*, g: *4D OF trans.*, h: *2D OF rot.*, and i: *2D OF rot. and* a_{13}, a_{23}, b_4.

the left side of the cube are not visible because they point inwards the cube. Also estimates of the rotational model *2D OF rot.* suffer from high and non-uniform errors. Estimates of *Range Flow* are more accurate, but distorted near borders of the cube. Models *4D OF trans.* and *2D OF rot. and* a_{13}, a_{23}, b_4 yield more accurate results. Estimates of the translational model are still distorted, mainly on the left side of the cube. Estimation results of *Scene Flow* best match the ground truth.

Figure 5.7 shows a rendered top view of the cube with ground truth of the rotational parameters and estimation results using models *2D OF rot.* and *2D OF rot. and* a_{13}, a_{23}, b_4. The estimates clearly recover the true motion, where estimates of *2D OF rot. and* a_{13}, a_{23}, b_4 are more uniform.

Table 5.1 shows angular errors for the regions depicted in Figs. 5.6a and b which quantitatively confirm the visual impression of the rendered results. *2D OF trans.* performs better when b_4, a_{13} and a_{23}, or all three terms are additionally used

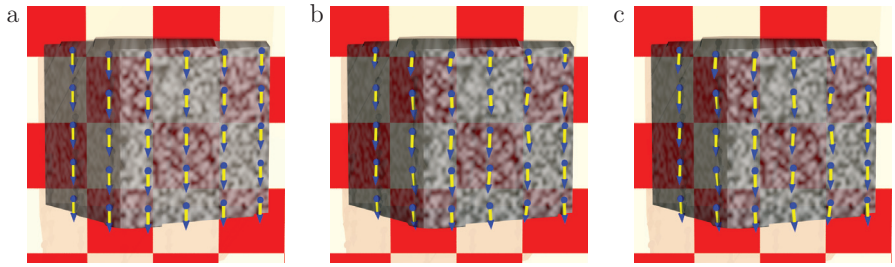


Figure 5.7: Rotational motion $\mathbf{\Omega}$. a: Ground truth, b: estimated via *2D OF rot.* and c: estimated via *2D OF rot. and* a_{13}, a_{23}, b_4.

Table 5.1: Average angular error (AAE) and standard deviations in degrees of translational and rotational motion parameters of regions on left and right side of the cube (see Figs. 5.6a and b). Errors or standard deviations above 1° (AAE) are indicated in red, below 0.1° (AAE) in green.

motion model	affine parameters	AAE left region		AAE right region	
		translation	rotation	translation	rotation
Scene Flow		0.36 ± 0.09	n/a	0.12 ± 0.04	n/a
Range Flow		8.86 ± 0.69	n/a	1.77 ± 0.15	n/a
translation	2D OF	112 ± 1.01	n/a	19.6 ± 0.70	n/a
	2D OF + b_4	1.22 ± 0.43	n/a	0.26 ± 0.15	n/a
	2D OF + a_{13},a_{23}	1.72 ± 1.28	n/a	0.84 ± 0.55	n/a
	2D OF + a_{13},a_{23},b_4	0.91 ± 0.64	n/a	0.51 ± 0.35	n/a
	4D OF	0.91 ± 0.64	n/a	0.51 ± 0.35	n/a
translation and rotation	2D OF	6.85 ± 6.24	0.018 ± 0.01	1.81 ± 1.42	0.004 ± 0.01
	2D OF + b_4	0.52 ± 0.19	0.017 ± 0.01	0.25 ± 0.16	0.004 ± 0.01
	2D OF + a_{13},a_{23}	0.93 ± 0.37	0.017 ± 0.01	0.20 ± 0.10	0.004 ± 0.01
	2D OF + a_{13},a_{23},b_4	0.67 ± 0.29	0.017 ± 0.01	0.22 ± 0.12	0.004 ± 0.01
	4D OF	0.67 ± 0.29	0.017 ± 0.01	0.22 ± 0.12	0.004 ± 0.01

for estimation. Otherwise estimates are heavily distorted. The same is true for translation estimates with models also estimating rotation. Rotation estimates are equally well for all rotation models. Errors of *Range Flow* are lower than for *2D OF trans.*, but significantly higher than for models incorporating more affine terms. *Scene Flow* shows best performance for translational motion estimation.

5.3.3 *Castor Oil Plant*

Figure 5.8 shows estimation results for the *Castor Oil Plant* sequence (cmp. Sec. 3.4). Neighborhood Λ is implemented using a Gaussian filter with size $77 \times 77 \times 5 \times 5$

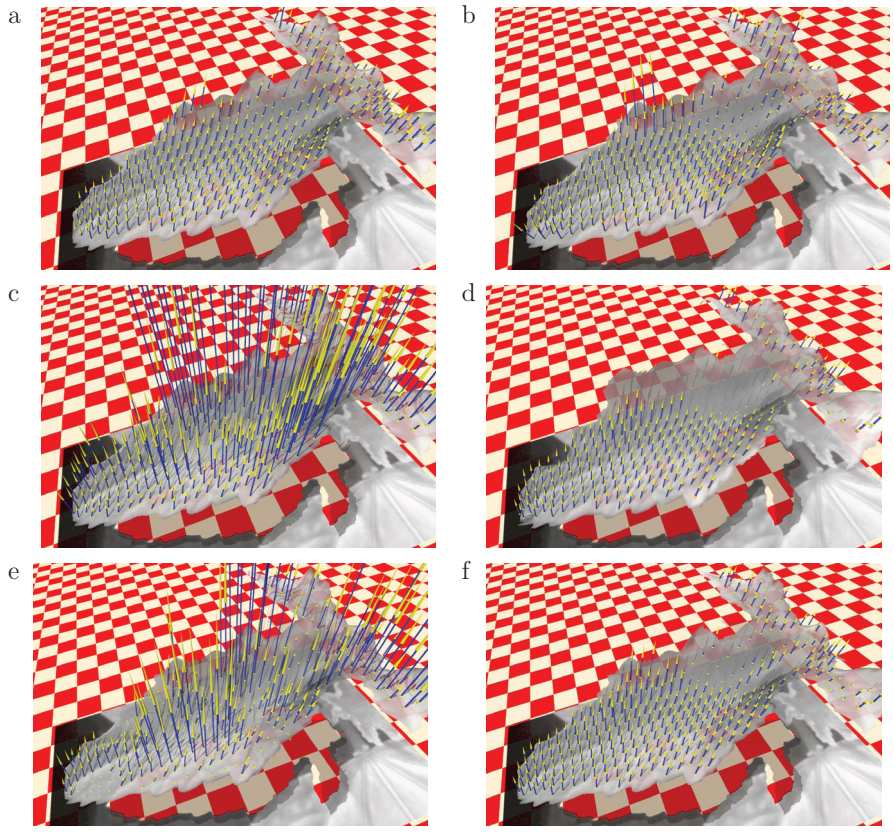


Figure 5.8: Motion estimation results for the plant leaf sequence. a *Scene Flow*, b *Range Flow* for varying illumination, c: *2D OF trans.*, d: *4D OF trans.*, e: *2D OF rot.* and f: *2D OF rot and* a_{13},a_{23} *and* b_4.

and standard deviation $23 \times 23 \times 1 \times 1$ in x, y, s, t-direction.

The leaf rotates around its node where it is attached to the stem. This results in a visible motion towards the camera and to the right, while the shadow area caused by the top leaf decreases. We compare estimation results of the translational model (*4D OF trans.* and of the rotational models with the estimates from Sec. 4.4, i.e., *Scene Flow*, *Range Flow* and the standard translational model (*2D OF trans.*). Figure 5.8 clearly shows the improvements in motion estimation, when more affine parameters (*4D OF trans.* and *2D OF rot and* a_{13},a_{23} *and* b_4) are incorporated in the estimation. Both models perform visibly better than *Range Flow*, because they produce less outliers. Using the rotational model estimation results are smoother, but *Scene Flow* still yields more homogeneous results especially in the shadow area

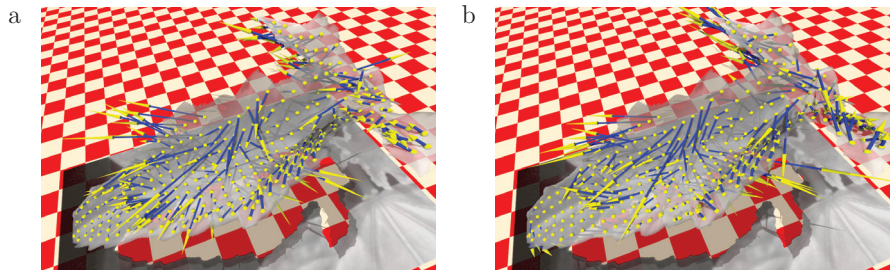


Figure 5.9: Estimates of angular velocities for the plant leaf sequence. a: *2D OF rot.* and b: *2D OF rot and* a_{13},a_{23} *and* b_4.

of the leaf.

Figure 5.9 shows estimation results for angular velocity Ω. Estimates of both models look heavily distorted. Ground truth is not available, but we assume that motion and unfolding of the leaf does not lead to a smooth angular vector field. Nevertheless, outliers are present and demonstrate that the rotational model is not robust enough for reliable estimation of angular velocity in plant leaf sequences.

5.4 Conclusions

In this chapter we presented a 4D affine optical flow model and how the parameters of this model can be explained by real world parameters. Based on a rigid surface patch we modeled translation, acceleration and rotation. The rotational model improves estimation results in almost all cases and additionally allows to estimate rotational parameters. Synthetic experiments showed that modeling acceleration is not sufficient to estimate rotation reliably and should therefore not be used if rotation occurs in the sequence. The 4D *Affine Model* and its explanation of real world parameters improved accuracy of motion estimates on synthetic and real data compared to *Range Flow*. *Scene Flow* still yields more reliable results. Therefore the estimation method of *Scene Flow* should be adapted to the *Affine Model*. We will do this in the next section.

Chapter 6

5D-*Affine Scene Flow Model*

In the previous sections we extended the *Affine Model* to handle (1) sloped surfaces (Sec. 3.3), (2) brightness changes (Sec. 4.3.3), and (3) rotational motion (Chap. 5). The 4D-*Affine Model* already implicitly includes the range constraint (Sec. 5.1.8) and showed better performance than *Range Flow*, although rotation/acceleration could be modeled for a neighborhood (Sec. 5.2) only. In this section we will combine the *Affine Model* with the estimation technique of *Scene Flow* and derive a formulation for the complete 5D-*Affine Scene Flow Model*. Towards this end, we extend the brightness change model proposed in Chap. 4 to handle brightness changes coming from camera motion. This leads to an estimation technique which allows estimation of all parameters including acceleration and/or rotation of patch origins.

6.1 5D-*Affine Model*

In this section we will extend the *Affine Model* presented in Sec. 5.1 to handle camera displacement in s_x and s_y direction. In a setup where a spotlight is mounted on the camera, brightness changes due to motion of the camera occur. We will derive a brightness change model according to Chap. 4 in order to handle brightness changes caused by motion of cameras with fixed spotlights.

In case of camera motion the direction of reflectance changes, i.e., $\mathbf{n}_r$ becomes $\mathbf{n}_r(s_x, s_y)$. This leads to the visible light intensity

$$L(\mathbf{X}(\Delta X, \Delta Y, t), s_x, s_y, t) = B(\mathbf{X}(\Delta X, \Delta Y, t), \mathbf{n}_i(t), \mathbf{n}_r(s_x, s_y))E(\Delta X, \Delta Y, t, \mathbf{n}_i(t)) \tag{6.1}$$

with the patch's bidirectional reflectance distribution function (BRDF) B and incident light irradiance E (cmp. (4.8)). We model changes of the BRDF due to smoothly changing reflectance direction $\mathbf{n}_r$ by smooth functions depending on camera displacement, i.e., $h_{B,s_x}(s_x)$ with $h_{B,s_x}(0) = 1$ and , $h_{B,s_y}(s_y)$ with $h_{B,s_y}(0) = 1$. We

obtain

$$B(\mathbf{X}(\Delta X, \Delta Y, s_x, s_y, t), \mathbf{n}_i(t), \mathbf{n}_r(s_x, s_y)) = \\ B(\mathbf{X}(\Delta X, \Delta Y, 0, 0, 0), \mathbf{n}_i(0), \mathbf{n}_r(0, 0)) h_{B,s_x}(s_x) h_{B,s_y}(s_y) h_{B,t}(t) \; . \tag{6.2}$$

BRDF B depends on three smooth functions $h_{B,s_x}(s_x)$, $h_{B,s_y}(s_y)$ and $h_{B,t}(t)$ instead of $h_{B,t}(t)$ when modeling only temporal changes as in Chap. 4. However, derivation of the visible intensity steps is analog to Chap. 4. This leads to

$$I(\mathbf{X}(\Delta X, \Delta Y, t), s_x, s_y, t) = I(\mathbf{X}(\Delta X, \Delta Y, 0), 0, 0, 0) \exp(h_I(\Delta X, \Delta Y, s_x, s_y, t)) \tag{6.3}$$

where

$$h_I(\Delta X, \Delta Y, s_x, s_y, t) := \ln(h_{B,s_x}(s_x) h_{B,s_y}(s_y) h_{B,t}(t) h_E(\Delta X, \Delta Y, t)) \; . \tag{6.4}$$

The sought temporal derivative of (6.3) is thus

$$\begin{aligned} \mathrm{d}I &= I(\mathbf{X}(\Delta X, \Delta Y, 0), 0, 0, 0) \exp(h_I(\Delta X, \Delta Y, s_x, s_y, t)) \mathrm{d}h_{\,I}(\Delta X, \Delta Y, s_x, s_y, t) \\ &= I(\mathbf{X}(\Delta X, \Delta Y, t), s_x, s_y, t) \mathrm{d}h_{\,I}(\Delta X, \Delta Y, s_x, s_y, t) \; . \end{aligned} \tag{6.5}$$

We introduce an approximation of h_I explicitly modeling spatial variations due to camera displacement and time still respecting $h_I(\Delta X, \Delta Y, 0, 0, 0) \equiv 0$. The multiplicative behavior between h_{B,s_x}, h_{B,s_y} and $h_{B,t}$ in (6.4) leads to a sum of power series

$$h_I(\Delta X, \Delta Y, s_x, s_y, t) \approx \\ h(\Delta X, \Delta Y, s_x, s_y, t, \mathbf{g}) := \sum_{m \in [s_x, s_y, t]} \sum_{i=0}^{2} \left(g_{m,i} + g_{m,i,x} \Delta X + g_{m,i,y} \Delta Y\right) m^i \; . \tag{6.6}$$

The total derivative of h is then

$$\begin{aligned} f(\Delta X, \Delta Y, s_x, s_y, t, \mathbf{g}) &:= \mathrm{d}h\,(\Delta X, \Delta Y, s_x, s_y, t, \mathbf{g}) \\ &= \sum_{m \in [s_x, s_y, t]} \sum_{i=1}^{2} i \left(g_{m,i} + g_{m,i,x} \Delta X + g_{m,i,y} \Delta Y\right) m^{i-1} \mathrm{d}m \; . \end{aligned} \tag{6.7}$$

We substitute $s_x = s_x - s_{x,0} = \Delta s_x$, $s_y = s_y - s_{y,0} = \Delta s_y$ and $t = t - t_0 = \Delta t$ in (6.7) and omit higher order terms. We get

$$f(\Delta X, \Delta Y, s_x, s_y, t, \mathbf{g}) = \\ g_1 + g_{1,x} \Delta X + g_{1,y} \Delta Y + 2 \left(g_{s_x,2} \Delta s_x \mathrm{d}s_x + g_{s_y,2} \Delta s_y \mathrm{d}s_y + g_{t,2} \Delta t \mathrm{d}t\right) \tag{6.8}$$

with

$$\begin{aligned} g_1 &= g_{s_x,1}\mathrm{d}s_x + g_{s_y,1}\mathrm{d}s_y + g_{t,1}\mathrm{d}t \\ g_{1,x} &= g_{s_x,1,x}\mathrm{d}s_x + g_{s_y,1,x}\mathrm{d}s_y + g_{t,1,x}\mathrm{d}t \\ g_{1,y} &= g_{s_x,1,y}\mathrm{d}s_x + g_{s_y,1,y}\mathrm{d}s_y + g_{t,1,y}\mathrm{d}t \ . \end{aligned} \tag{6.9}$$

Inserting (6.8) in (5.24) and adding camera displacement directions s_x and s_y, we obtain the full *Affine Model* (6.10), where $\tilde{\nabla} I = (I_x, I_y, I)^{\mathrm{T}}$ and affine parameters are given by (6.12), (6.13). Brightness change parameters can be derived by (6.9)

$$\tilde{\nabla}^{\mathrm{T}} I \left[\begin{pmatrix} u_x\mathrm{d}t + \nu\mathrm{d}s_x \\ u_y\mathrm{d}t + \nu\mathrm{d}s_y \\ -\left(g_{t,1}\mathrm{d}t + g_{s_x,1}\mathrm{d}s_x + g_{s_y,1}\mathrm{d}s_y\right) \end{pmatrix} + \mathbf{A} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s_x \\ \Delta s_y \\ \Delta t \end{pmatrix} \right] + I_{s_x}\mathrm{d}s_x + I_{s_y}\mathrm{d}s_y + I_t\mathrm{d}t = 0 \tag{6.10}$$

$$\mathbf{A} = \begin{pmatrix} a_{11}\mathrm{d}t + b_1\mathrm{d}s_x & a_{12}\mathrm{d}t + b_2\mathrm{d}s_x & a_{13}\mathrm{d}t + b_3\mathrm{d}s_x & a_{14}\mathrm{d}t + b_4\mathrm{d}s_x & a_{15}\mathrm{d}t + b_5\mathrm{d}s_x \\ a_{21}\mathrm{d}t + b_1\mathrm{d}s_y & a_{22}\mathrm{d}t + b_2\mathrm{d}s_y & a_{23}\mathrm{d}t + b_3\mathrm{d}s_y & a_{24}\mathrm{d}t + b_4\mathrm{d}s_y & a_{25}\mathrm{d}t + b_5\mathrm{d}s_y \\ g_{\Delta x} & g_{\Delta y} & g_{\Delta s_x} & g_{\Delta s_y} & g_{\Delta t} \end{pmatrix} \tag{6.11}$$

$$\begin{aligned} u_x &= -\nu\left(N_X - \tfrac{x_0}{f}N_Z\right) \\ u_y &= -\nu\left(N_Y - \tfrac{y_0}{f}N_Z\right) \\ a_{11} &= -\nu\left[\Omega_Y\tfrac{\partial Z}{\partial x} - \Omega_Z\tfrac{\partial Y}{\partial x} - \tfrac{x_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial x} - \Omega_Y\tfrac{\partial X}{\partial x}\right) - \tfrac{N_Z}{Z_0}\right] && -b_1\left(N_X - \tfrac{x_0}{f}N_Z\right) \\ a_{12} &= -\nu\left[\Omega_Y\tfrac{\partial Z}{\partial y} - \Omega_Z\tfrac{\partial Y}{\partial y} - \tfrac{x_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial y} - \Omega_Y\tfrac{\partial X}{\partial y}\right)\right] && -b_2\left(N_X - \tfrac{x_0}{f}N_Z\right) \\ a_{13} &= -\nu\left[\Omega_Y\tfrac{\partial Z}{\partial s_x} - \Omega_Z\tfrac{\partial Y}{\partial s_x} - \tfrac{x_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial s_x} - \Omega_Y\tfrac{\partial X}{\partial s_x}\right)\right] && -b_3\left(N_X - \tfrac{x_0}{f}N_Z\right) \\ a_{14} &= -\nu\left[\Omega_Y\tfrac{\partial Z}{\partial s_y} - \Omega_Z\tfrac{\partial Y}{\partial s_y} - \tfrac{x_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial s_y} - \Omega_Y\tfrac{\partial X}{\partial s_y}\right)\right] && -b_4\left(N_X - \tfrac{x_0}{f}N_Z\right) \\ a_{15} &= -\nu\left[\Omega_Y\tfrac{\partial Z}{\partial t} - \Omega_Z\tfrac{\partial Y}{\partial t} - \tfrac{x_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial t} - \Omega_Y\tfrac{\partial X}{\partial t}\right) + \left(A_X - \tfrac{x_0}{f}A_Z\right)\right] && -b_5\left(N_X - \tfrac{x_0}{f}N_Z\right) \\ a_{21} &= -\nu\left[\Omega_Z\tfrac{\partial X}{\partial x} - \Omega_X\tfrac{\partial Z}{\partial x} - \tfrac{y_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial x} - \Omega_Y\tfrac{\partial X}{\partial x}\right)\right] && -b_1\left(N_Y - \tfrac{y_0}{f}N_Z\right) \\ a_{22} &= -\nu\left[\Omega_Z\tfrac{\partial X}{\partial y} - \Omega_X\tfrac{\partial Z}{\partial y} - \tfrac{y_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial y} - \Omega_Y\tfrac{\partial X}{\partial y}\right) - \tfrac{N_Z}{Z_0}\right] && -b_2\left(N_Y - \tfrac{y_0}{f}N_Z\right) \\ a_{23} &= -\nu\left[\Omega_Z\tfrac{\partial X}{\partial s_x} - \Omega_X\tfrac{\partial Z}{\partial s_x} - \tfrac{y_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial s_x} - \Omega_Y\tfrac{\partial X}{\partial s_x}\right)\right] && -b_3\left(N_Y - \tfrac{y_0}{f}N_Z\right) \\ a_{24} &= -\nu\left[\Omega_Z\tfrac{\partial X}{\partial s_y} - \Omega_X\tfrac{\partial Z}{\partial s_y} - \tfrac{y_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial s_y} - \Omega_Y\tfrac{\partial X}{\partial s_y}\right)\right] && -b_4\left(N_Y - \tfrac{y_0}{f}N_Z\right) \\ a_{25} &= -\nu\left[\Omega_Z\tfrac{\partial X}{\partial t} - \Omega_X\tfrac{\partial Z}{\partial t} - \tfrac{y_0}{f}\left(\Omega_X\tfrac{\partial Y}{\partial t} - \Omega_Y\tfrac{\partial X}{\partial t}\right) + \left(A_Y - \tfrac{y_0}{f}A_Z\right)\right] && -b_5\left(N_Y - \tfrac{y_0}{f}N_Z\right) \end{aligned} \tag{6.12}$$

$$\begin{aligned} \nu &= -\tfrac{f}{Z_0}, \quad b_1 = \tfrac{Z_X}{Z_0 c}, \quad b_2 = \tfrac{Z_Y}{Z_0 c} \\ b_3 &= \tfrac{f}{Z_0}\tfrac{Z_X}{Z c}, \quad b_4 = \tfrac{f}{Z_0}\tfrac{Z_Y}{Z c}, \quad b_5 = \tfrac{f}{Z_0}\tfrac{Z_t}{Z c} \end{aligned} \tag{6.13}$$

$$c = 1 - Z_X\tfrac{x_0}{f} - Z_Y\tfrac{y_0}{f} \tag{6.14}$$

and the new variables $g_{\Delta i}$ for $i \in [x, y, s_x, s_y, t]$ by appropriate substitution of ΔX and ΔY in 6.8. The partial derivatives of world coordinates can be obtained by (5.8). This notation demonstrates, that the brightness change model has a similar behavior as the standard affine model, but instead of the gradients of the image intensity the intensity itself is used as input data. Moreover, the novel brightness change model introduces only two new affine parameters ($g_{s_x,2}$ and $g_{s_y,2}$) into the estimation process and a new interpretation of the other brightness change parameters (g_1, $g_{1,x}$ and $g_{1,y}$) compared to the old brightness change model.

6.2 Parameter Estimation

Scene Flow combines translational optical flow estimates from several cameras in order to solve the aperture problem of *Scene Flow* (cmp. (3.4)). We extend (3.4) for a 2D camera grid to handle affine optical flow parameters, i.e., parameter b_5 and $\mathbf{a} = (a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{21}, a_{22}, a_{23}, a_{24}, a_{25})^{\mathrm{T}}$. Additional equations constraining *Scene Flow* then allow to solve for acceleration $(A_X, A_Y, A_Z)^{\mathrm{T}}$. We obtain for a camera grid with N calibrated cameras

$$\mathbf{BV} = \mathbf{U} \text{ ,with } \mathbf{V} = \begin{pmatrix} N_X \\ N_Y \\ N_Z \\ \Omega_X \\ \Omega_Y \\ \Omega_Z \\ A_X \\ A_Y \\ A_Z \end{pmatrix} \text{ and } \mathbf{U} = \begin{pmatrix} u_{x,1} \\ u_{y,1} \\ \mathbf{a}_1 \\ b_{5,1} \\ u_{x,2} \\ u_{y,2} \\ \mathbf{a}_2 \\ \vdots \\ b_{5,N} \end{pmatrix} \tag{6.15}$$

with affine parameters $\mathbf{a}_i$ of the optical flow in camera i. Matrix $\mathbf{B}$ is created out of appropriate entries of matrix $\mathbf{A}$ and parameters u_x, u_y as defined in (6.12).
Not all parameters of the *Affine Model* may be available in all cameras. We compute the full model, i.e., with affine parameters containing Δs_x and Δs_y (a_{13},a_{14},a_{23},a_{24} and b_5) only for the central camera. For all other cameras we omit these parameters and get 8 optical flow parameters each.

6.3 Experiments

In a first experiment we use synthetic sinusoidal sequences for systematic error analysis. Then performance of different models is compared on more realistic data with ground truth available, i.e., a translating and rotating cube with varying illumination in time and camera displacement direction, rendered with POV-Ray [15]. Finally we show estimation results on the previous introduced *Castor Oil*

Plant sequence and another sequence showing a rotating tobacco plant leaf. In all experiments we minimize systematic discretization errors coming from derivatives by using optimized 5-tab derivative filter sets presented in [54].
In order to test performance of models and influence of acceleration we compare *Scene Flow* and *Range Flow* with the models[1]

Affine Model Trans. estimates $\mathbf{N}$ using equations u_x,u_y,a_{11},a_{12},a_{21} and a_{22}, i.e., standard *Affine Model*,

Affine Model Trans. Stab. estimates $\mathbf{N}$ using all equations from (6.12), and b_5,

Affine Model Rot. estimates $\mathbf{N}$ and $\mathbf{\Omega}$ using equations u_x,u_y,a_{11},a_{12},a_{21} and a_{22},

Affine Model Rot. Stab. estimates $\mathbf{N}$ and $\mathbf{\Omega}$ using all equations from (6.12), and b_5, except a_{15} and a_{25},

Affine Scene Flow Model extends the *Affine Model Rot. Stab.* by additionally using equations u_x,u_y,a_{11},a_{12},a_{21} and a_{22} of all other cameras and

Affine Scene Flow Model Acc. extends the *Affine Scene Flow Model* by additionally using equations a_{15} and a_{25} of all cameras and estimates $\mathbf{A}$.

6.3.1 Sinusoidal Pattern

For systematic error analysis we render a surface patch with sinusoidal pattern using the *Sequence Generator* (cmp. App. B). The parameters of the sequence are similar to the parameters used in previous experiments (cmp. Sec. 5.3). Additionally we add acceleration of the patch center $\mathbf{A} = (0.002, -0.001, 0.003)^{\mathrm{T}}$ mm/frame^2. In each experiment we vary only one of the parameters. Neighborhood Λ is implemented by a Gaussian filter with size $65 \times 65 \times 5 \times 5$ and standard deviations $19 \times 19 \times 1 \times 1$ in x, y, s, t-directions. In order to compare performance of models, we use the average angular error (see App. A) for translational (AAE_N), rotational (AAE_Ω) and acceleration (AAE_A) parameters.

Figure 6.1 shows average angular errors for sequences with increasing translational velocity N_Z (left) and with increasing acceleration A_Z (right). Figures 6.1a and b demonstrate that all models based on *Scene Flow*, i.e., *Scene Flow* and the *Affine Scene Flow Model* with and without acceleration, perform similar and best of all models almost everywhere. Only in case of small patch motion towards the camera the *Affine Model Rot. Stab.* performs slightly better. The *Affine Model Rot.* yields best results when acceleration is large. *Range Flow*, the *Affine Model Trans. Stab.* and the *Affine Model Trans.* yield highest errors in both cases. Models based on *Scene Flow* show high performance also for estimation of rotational parameters (cmp. Figs. 6.1c and d). The *Affine Scene Flow Model Acc.* performs slightly worse than the

[1] If not stated differently, all models are extended to handle varying illumination as derived in Chap. 4.

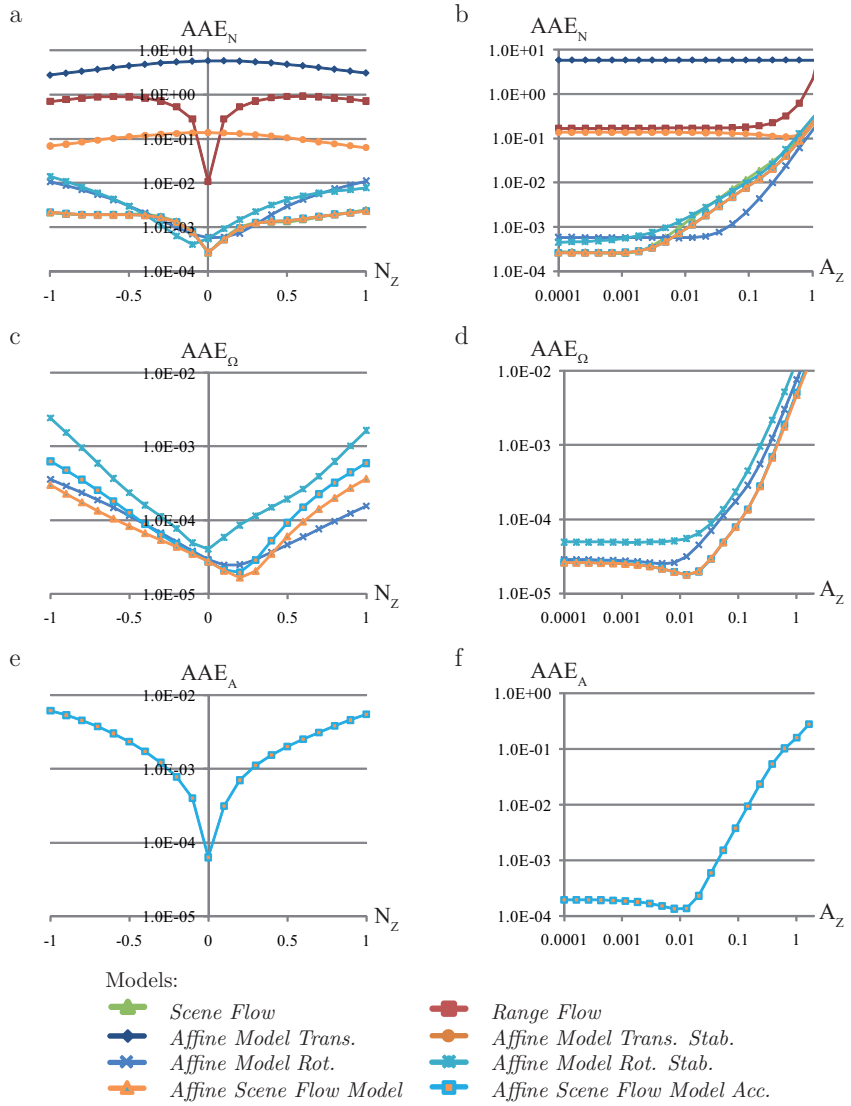


Figure 6.1: Average angular error of a and b: translational (AAE_N), c and d: rotational (AAE_Ω) and e and f: acceleration (AAE_A) parameters versus increasing N_Z (left) and A_Z (right).

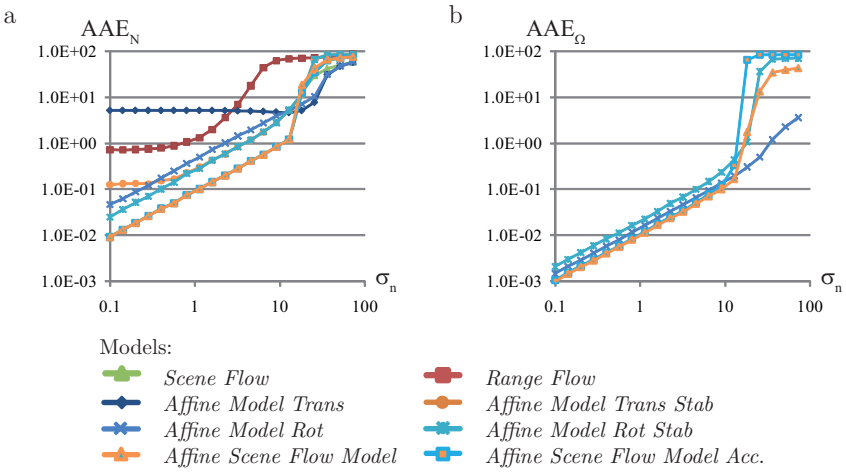


Figure 6.2: Average angular error of translational (AAE_N) and rotational (AAE_Ω) parameters versus σ_n.

model without acceleration. The *Affine Model Rot.* performs best for high N_Z, the *Affine Model Rot. Stab.* yields worst results. In case of increasing acceleration both models using the estimation technique of *Scene Flow* perform better than the *Affine Model*. Figures 6.1 e and f show that the *Affine Scene Flow Model Acc.* estimates acceleration reliably for $A_Z < 0.1$.

Figures 6.2a and b demonstrate that the increased accuracy of models using the estimation technique based on *Scene Flow* also holds under noise. Translational as well as rotational parameters are best estimated with the *Affine Scene Flow Model* for typical noise values $\sigma_n < 10$. Errors of the other models are similar to the errors obtained for rotational motion in Chap. 5. The model error is highest for the *Affine Model Trans.* and *Range Flow*. The *Affine Model Trans. Stab.* yields lower errors and same as the models with rotation for noise values $\sigma_n > 1$. Also the *Affine Model Rot. Stab.* performs better than the *Affine Model* without stabilization. In case of rotational motion it is the other way around. There the *Affine Model Rot.* yields lower errors, but the model error is lowest for the *Affine Scene Flow Model*.
We conclude that the *Affine Scene Flow Model* in general increases stability due to noise and estimates are more reliable compared to the *Affine Model*. Estimation of acceleration is possible and additional estimation of acceleration does not have a significant effect on the estimates.

6.3.2 Synthetic Cube

The synthetic cube sequence allows to compare models on more realistic data with ground truth available. We use a cube sequence similar to the one used in Sec. 5.3,

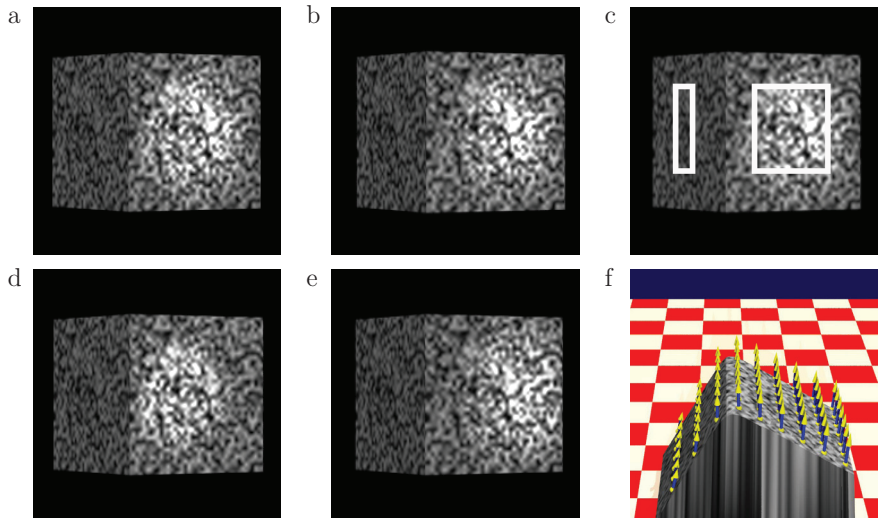


Figure 6.3: Motion estimation of cube moving towards camera with rotation and illumination changes. a and b: First and last frame of central camera, c: central frame of central camera with areas used for evaluation, d and e: central frame of first and last camera, and f: ground truth motion.

i.e., the cube is moving with $\mathbf{N} = (-0.2, 0, -1)^{\mathrm{T}}$ mm/frame and rotates around its Y-axis with $\omega = 0.4$ degrees/frame. Instead of a static spotlight (cmp. Chap. 4), we add a spotlight moving with the camera, causing illumination changes in camera displacement direction and time. Such a setup is frequently used in robotics, where cameras and illumination units are placed at a robot's end-effector. Neighborhood Λ is the same as for the sinusoidal sequences. The 1D camera grid contains 9 cameras with a displacement of 5 mm. Figures 6.4a and b show first and last frame of the central camera and Figs. 6.4d and e the central frame of the first and the last camera. The brightness changes in time and camera displacement direction are clearly visible. Regions where errors are evaluated are depicted in Fig. 6.3c and ground truth motion in Fig. 6.4f. Figures 6.4a-c show motion estimates of the different models using the brightness change model introduced in Chap. 4. Errors are amplified by a factor of 10 in all directions. The models depicted are the best from the previous experiments, namely *Scene Flow*, the *Affine Model Rot. Stab.* and the novel *Affine Scene Flow Model*. Results look similar for the *Affine Scene Flow Model* with and without rotation, therefore we depict the *Affine Scene Flow Model Acc.*, only. Figures 6.4d–f show the same models using the new brightness change model. The estimates improve for all models. The models based on *Scene Flow* yield better results compared to the *Affine Model*. The combined *Affine Scene Flow Model* shows the most homogeneous motion vector field. For all models most

Table 6.1: Average angular error of translational (AAE_N) and rotational (AAE_Ω) motion parameters in degrees, average relative growth rate (RGR) in percent per frame and their standard deviations of regions on left and right side of the cube (see Fig. 6.3c). Errors or standard deviations above 1° (AAE) or 1%/frame (RGR) are indicated in red, below or equal 0.1° (AAE) or 0.01%/frame (RGR) in green.

brightness model	motion model	left region			right region		
		AAE_N	AAE_Ω	RGR	AAE_N	AAE_Ω	RGR
BCCE (Chap. 4)	*Scene Flow*	0.33 ± 0.10	n/a	-0.0013 ± 0.041	0.344 ± 0.37	n/a	-0.07 ± 0.28
	Range Flow	7.16 ± 0.48	n/a	-0.303 ± 0.264	34.32 ± 44.20	n/a	$\gg 100$
	Affine Model Trans. Stab.	1.07 ± 0.73	n/a	0.536 ± 0.096	1.325 ± 0.853	n/a	-0.29 ± 0.72
	Affine Model Rot. Stab.	0.51 ± 0.18	0.0047 ± 0.0024	0.0094 ± 0.088	1.096 ± 0.910	0.045 ± 0.027	-0.23 ± 0.73
	Affine Scene Flow Model	0.32 ± 0.10	0.0035 ± 0.0017	0.0006 ± 0.043	0.410 ± 0.392	0.028 ± 0.021	-0.08 ± 0.28
	Affine Scene Flow Model Acc.	0.32 ± 0.10	0.0036 ± 0.0017	0.0006 ± 0.043	0.410 ± 0.392	0.027 ± 0.021	-0.08 ± 0.28
novel BCCE	*Scene Flow*	0.33 ± 0.10	n/a	-0.0013 ± 0.041	0.378 ± 0.40	n/a	-0.063 ± 0.280
	Range Flow	7.15 ± 0.49	n/a	-0.308 ± 0.260	4.63 ± 1.45	n/a	0.433 ± 0.972
	Affine Model Trans. Stab.	1.07 ± 0.73	n/a	0.537 ± 0.096	0.647 ± 0.477	n/a	-0.044 ± 0.588
	Affine Model Rot. Stab.	0.52 ± 0.18	0.0047 ± 0.0024	0.010 ± 0.088	0.585 ± 0.597	0.028 ± 0.019	0.009 ± 0.591
	Affine Scene Flow Model	0.32 ± 0.10	0.0035 ± 0.0017	0.0006 ± 0.043	0.371 ± 0.374	0.027 ± 0.021	-0.057 ± 0.275
	Affine Scene Flow Model Acc.	0.32 ± 0.10	0.0036 ± 0.0017	0.0006 ± 0.043	0.371 ± 0.374	0.027 ± 0.021	-0.057 ± 0.275

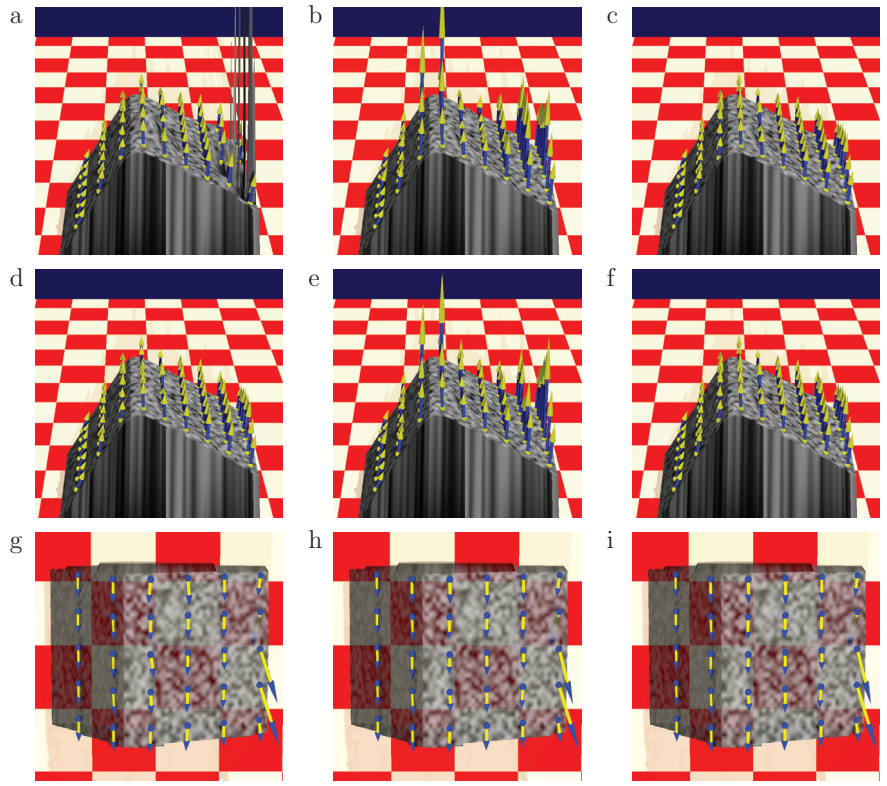


Figure 6.4: Scaled motion estimates of cube sequence with errors amplified. a–c: Motion estimates using the brightness change model proposed in Chap. 6 with models *Scene Flow* (a), the *Affine Model Rot. Stab.* (b), and the *Affine Scene Flow Model Acc.* (c). d–f: Motion estimates using the new brightness change model with models *Scene Flow* (d), the *Affine Model Rot. Stab.* (e), and the *Affine Scene Flow Model Acc.* (f). g–i: Scaled rotational motion estimates with amplified errors using the new brightness change model. The *Affine Model Rot. Stab.* (g), the *Affine Scene Flow Model* (h) and the *Affine Scene Flow Model Acc.* (i).

errors occur at borders. The depth reconstruction using the old brightness change model fails as expected (cmp. Fig. 6.4a). However, errors of the motion estimates of *Scene Flow* are lower as expected. This demonstrates that the *Scene Flow* estimation technique is more robust with respect to false depth estimates compared to the *Affine Model*. Using the *Affine Model* brightness changes are well compensated also with the old brightness change model. As discussed in Sec. 6.1 this may depend on the fact that the model for temporal brightness changes already compensates

parts of the brightness changes in camera displacement direction. The estimates of rotational motion (not amplified) are depicted in Figs. 6.4j–l for the *Affine Model* Rot. Stab., the *Affine Scene Flow Model* and the *Affine Scene Flow Model Acc.*. The ground truth is the same as for the cube sequence in Sec. 5.3, i.e., the vectors should point downwards. All three models show almost similar performance and significant differences to the ground truth only at the right border. It is most likely that border effects cause these outliers. An appropriate estimator should avoid the errors introduced by borders.
Quantitative results of the two regions on the left and the right side of the cube (cmp. Fig. 6.3c) are presented in Tab. 5.1. The new brightness change model does not influence estimates on the left side of the cube, where no brightness changes occur. On the right side of the cube the novel BCCE improves estimates of all models but *Scene Flow*. The novel combined *Affine Scene Flow Model* with the novel BCCE yields best results almost everywhere on the cube. Only *Scene Flow* with the BCCE proposed in Chap. 4 yields a slightly lower mean angular error for the right side of the cube. The *Affine Model Rot. Stab.* shows a very small mean angular error, but the high standard deviation depicts the uncertainty of this estimate. Also rotational estimates of the *Affine Scene Flow Model* are significantly better compared to the *Affine Model Rot. Stab.*. There is no significant difference visible between the *Affine Scene Flow Model* with and without acceleration.
We conclude that the novel BCCE and the *Scene Flow* estimation technique should be used in order to make estimation results more reliable. The *Affine Scene Flow Model* yields similar performance as *Scene Flow*, but additionally allows to estimate for rotational motion and acceleration.

6.3.3 *Castor Oil Plant*

We demonstrate the performance of the new estimation technique on the previous presented *Castor Oil Plant* sequence. We choose same parameters as in Chap. 5. Figures 6.5a–d show estimated translational motion vectors of models *Scene Flow*, *Range Flow*, the *Affine Model Rot. Stab.* and the *Affine Scene Flow Model Acc.*. Motion vector fields of the *Affine Scene Flow Model* and *Scene Flow* are smoother compared to *Range Flow* and the *Affine Model*. In the right part of the leaf estimates of the *Affine Scene Flow Model* are more directed to the outside of the leaf, therefore they are occluded by the leaf. The vector field estimated by *Scene Flow* seems to be more smooth in this region. However, as the leaf is unfolding, estimates of the *Affine Scene Flow Model* are the more likely ones. Figures 6.5e and f show estimates of rotational motion of the *Affine Model* and the *Affine Scene Flow Model*. The estimates are distorted in both cases, but the vector field of the *Affine Scene Flow Model* looks more smooth.

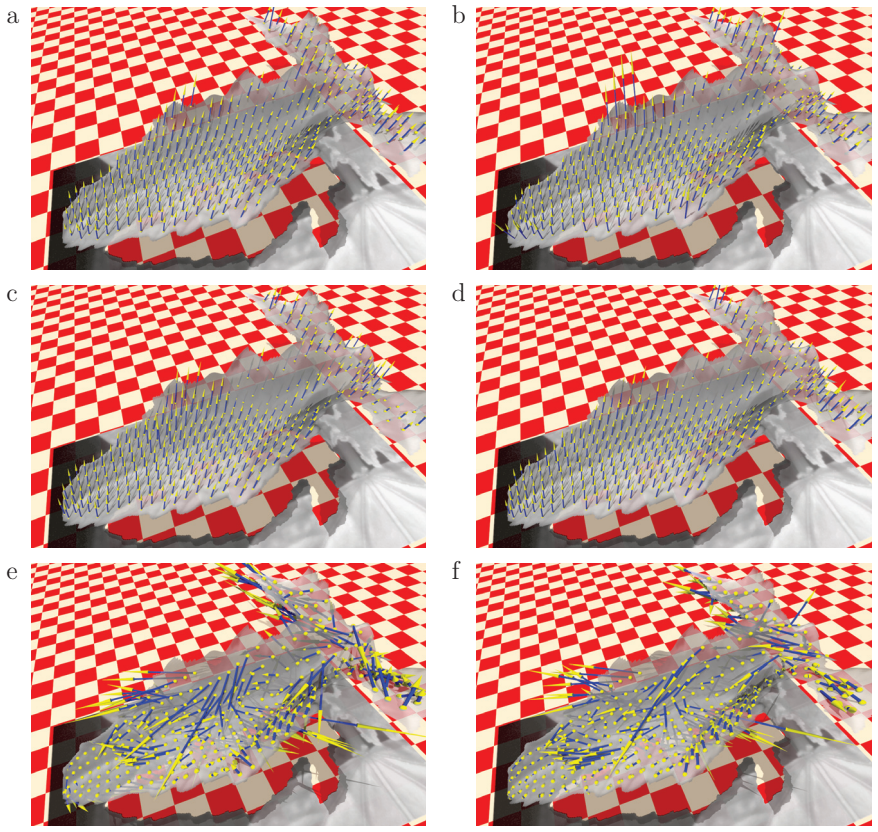


Figure 6.5: *Castor Oil Plant* Sequence. Translational motion estimates of models *Scene Flow* (a), *Range Flow* (b), the *Affine Scene Flow Model wo* Δt (c) and the *Affine Scene Flow Model with* Δt (d). Rotational motion estimates of the *Affine Model Rot. Stab.* (e) and the *Affine Scene Flow Model with* Δt (f).

6.3.4 *Tobacco Plant*

We show more estimation results using a second plant leaf sequence, imaging a tobacco plant leaf. Three frames of the sequence are shown in Figs. 6.6a, c and e. The leaf is marked with spots of watercolor to avoid errors coming from the aperture problem. The maximal width of the leaf is approximately 20 mm. The leaf rotates around the node where it is attached to the stem, causing a visible motion to the camera and to the left. Moreover it is folding, which leads to slight motion of the sides of the leaf to its center. In the left part of the leaf brightness changes occur due to shadowing by other leafs. We use a camera on a moving stage to obtain 9 images

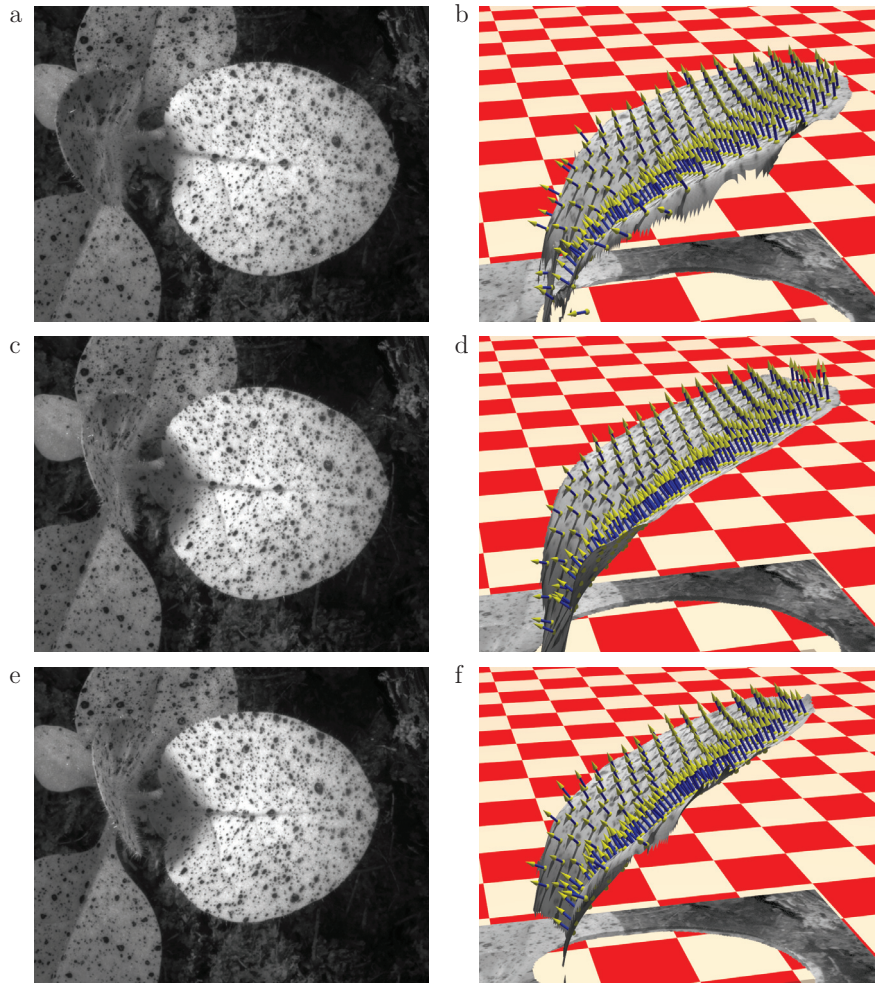


Figure 6.6: Tobacco leaf sequence. Left: Input frame of the central camera a: $t = t_0 - 20$ min, c: $t = t_0$ min and e: $t = t_0 + 20$ min. b, d and f: Corresponding rendered 3D structure and surface normals estimated with the *Affine Model*.

from positions with a camera displacement of 1 mm. This leads to a preshift of 26 pixel. The images are taken every 2 minutes. The size of the sensor is 1600×1200 pixel with width and height of 0.0044 mm. The focal length is 25 mm. Neighborhood Λ is realized by a normalized Gaussian with size $121 \times 121 \times 5 \times 5$ and standard deviation $\sigma_W = 41 \times 41 \times 1 \times 1$ in x-y-s_x-and t-direction. Figures 6.6b, d and f depict 3D reconstructions and surface slopes for times $t = t_0 - 20$ min, $t = t_0$ min

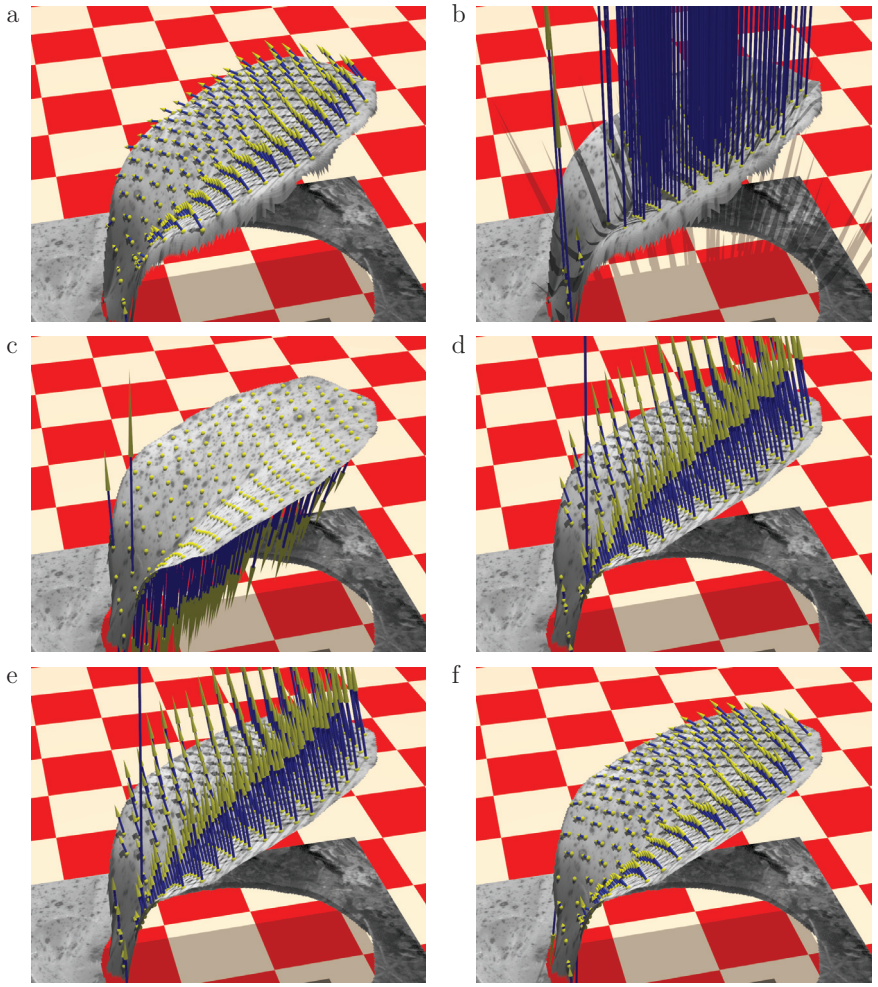


Figure 6.7: Tobacco leaf sequence. a–c: Motion estimates for models *Scene Flow*, *Range Flow* and the *Affine Model* without brightness modeling, d–f: Motion estimates for Affine model using $\Delta s_x dt$, using $\Delta s_x dt$ and $\Delta t d s_x$ and using $\Delta s_x dt$, $\Delta t d s_x$ and $\Delta t dt$ respectively.

and $t = t_0 + 20$ min obtained by the *Affine Model*. Reconstructed depth and surface slopes clearly recover the true position of the leaf. Furthermore the rotation of the leaf around its node, where it is attached to the stem, is visible.
Scaled translational motion estimates of different models are shown in Fig. 6.7. Rendered velocity fields seem most reliable for *Scene Flow* (a) and the *Affine Scene*

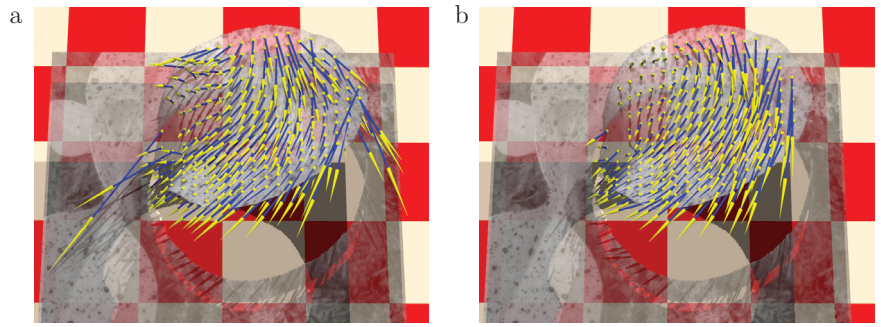


Figure 6.8: Tobacco leaf sequence. Rotational estimates of a: the *Affine Model Rot. Stab.* and b: the *Affine Scene Flow Model* with acceleration.

Flow Model (f). Estimates of *Range Flow* (b) and the *Affine Model* (c–e) are over estimated in particular in Z-direction. This behavior was already confirmed for motion estimation in the rotating cube sequence. *Range Flow* does not handle rotation explicitly, therefore correspondences of the range constraint may fail. We suggest that more appropriate choices of β_i, i.e., the weighting of the structure tensors, may reduce this effect (cmp. (4.25)). The standard *Affine Model* is based on a patch model with translational motion only, which estimates motion in Z-direction based on affine transformations, e.g., divergence. Rotation and folding of the leaf leads to visible shrinking of the projection of the leaf. The *Affine Model* interprets this shrinkage as motion away from the camera. The extended models using more affine parameters (cmp. Figs. 6.7d and e) yield similar and better approximations of the motion vector field.
Figures 6.8a and b show estimates of the rotational motion by the *Affine Model Rot. Stab.* and the *Affine Scene Flow Model Acc.*. We depict estimates for the *Affine Scene Flow Model Acc.* only, as the model without acceleration yields similar results. The vector field of the *Affine Scene Flow Model* is more reliable compared to the vector field estimated by the *Affine Model*. There are less outliers and the directions of the vectors recover the true rotation and folding of the leaf.

6.4 Conclusions

In this section we proposed a novel BCCE (6.10) and combined the estimation technique of *Scene Flow* with the *Affine Model*. The brightness change constraint handles also brightness changes coming from camera displacement and fits well into the general notation of the *Affine Scene Flow Model*. Estimation of optical flow parameters in all cameras and combining these parameters with the estimation of the *Affine Model* yields high accuracy and robustness to noise. Compared to *Scene Flow* the new model yields higher accuracy and moreover allows to estimate

rotational motion and acceleration parameters reliably. The combination with the novel BCCE and the *Affine Model*, i.e., simultaneous estimation of 3D structure, surface slopes and 3D motion, further ensures more accurate 3D surface structure and slope estimation. Experiments on real data demonstrate that the *Affine Scene Flow Model* yields superior performance compared to *Range Flow*, the *Affine Model* and in some regions also compared to *Scene Flow*. The tobacco plant leaf sequence shows that estimation of rotational parameters with the *Affine Scene Flow Model* is possible.

Chapter 7

Conclusions And Outlook

In this work we presented a highly accurate 3D motion estimation model based on optical flow, which allows to estimate plant motion reliably. Starting from three basic models, i.e., *Scene Flow*, *Range Flow* and the *Affine Model*, a new model for simultaneous estimation of surface structure, slopes and motion has been developed and thoroughly tested. We first summarize results and conclusions of the different models and modifications that have been presented. Finally we discuss possible future work.

7.1 Summary and Conclusions

Motion estimation of dynamic processes has been investigated in Chap. 2. The typical estimation process contains at least three important tasks: model derivation, discretization and parameter estimation. First an appropriate model of the data has to be derived. This model sets the accuracy limits of the estimation process, i.e., estimation results are only as good as the model fits to the data. In this work several motion models for optical flow have been proposed. In typical image sequences affine transformations of the motion vector field occur, therefore at least an affine motion model should be used. An appropriate discretization is needed in order to handle input data and to compute input parameters, e.g., image gradients. Finally an adequate estimator has to be chosen or developed. Chapter 2 presented several local least squares estimators with and without handling of covariance information and outliers. The novel robust total least squares estimator with handling of covariance information showed best performance in estimation of affine model parameters.
Chapter 3 reviews three 3D motion estimation models based on optical flow, namely *Scene Flow*, *Range Flow* and the *Affine Model*. *Scene Flow* and *Range Flow* need 3D structure information for one or more time points in order to estimate 3D motion. *Scene Flow* first computes optical flow in more than two cameras. A second estimation process yields 3D motion, which re-projections best fit to each of the optical flows. *Range Flow* constrains range data of several time points and optical

flow of one camera. A single estimation yields the 3D motion parameters. The *Affine Model* starts from multi-camera image sequences. It is based on an affine optical flow model and allows simultaneous estimation of flow parameters in camera displacement direction and time. 3D structure, slopes and 3D motion can be derived from these flow parameters using projective geometry. The *Affine Model* also allows estimation of parameters describing the 3D neighborhood, i.e., surface slopes. This makes the model appropriate for reliable estimation of highly accurate plant leaf motion. *Scene Flow* is based on a pinhole camera model similar to the *Affine Model*, but all parameters in a neighborhood are assumed constant. Synthetic experiments showed that the *Affine Model* yields best results for multi-camera grids with small baselines (≈ 0.01 % of depth). For plant measurement setups, where average depth typically is below 1m, these baselines are physically not realizable with multiple cameras, but with one camera on moving stages. For typical baselines (≈ 0.5 % of depth) *Scene Flow* yields best results. However, all models fail on plant leaf motion estimation. The main reason is that the models are based on standard optical flow, which implies brightness constancy. This is well fulfilled for undirected, diffuse light. In a laboratory setup the plant is illuminated by directed infrared LED-lights (cmp. Chap. 3). Therefore motion of the leaf, the light source or the camera causes brightness changes in the data.

In order to handle such brightness changes coming from varying illumination a novel brightness change constraint has been derived in Chap. 4. The brightness change constraint explicitly models temporal and spatially varying temporal brightness changes. Motion estimation using the *Affine Model* with the novel brightness change constraint on synthetic data with brightness changes coming from varying illumination demonstrated the improved performance. In order to compare the novel BCCE with other approaches an intensive evaluation and comparison with different prefilters (highpass and homomorphic) and brightness constraints using *Range Flow* has been carried out. It turns out that if the brightness constancy assumption is violated, homomorphic prefiltering rather than linear highpass prefiltering should be used. Nevertheless, modeling brightness changes yields slightly more accurate results. The novel brightness constraint in combination with each of the three estimation models yields improved results on the plant leaf sequence. *Range Flow* showed higher errors near borders, while the *Affine Model* yields worst results compared to the other models.

In Chap. 5 a 4D-*Affine Model* has been derived. This model is based on a higher dimensional affine optical flow, which allows estimation of affine transformations in four dimensions, i.e., in local image coordinates and in camera displacement direction and time. It is demonstrated that the affine transformation of the disparity in time boils down to the *Range Flow* constraint. Further the motion model of the 4D-*Affine Model* is extended to handle rotational motion and acceleration. Transformations of the 3D neighborhood as well as 3D motion are explained by affine optical flow parameters. Tests on synthetic data demonstrated that the new *Affine Model* outperforms *Range Flow*. However, *Scene Flow* yields lowest errors. Experiments

showed that affine parameters describing transformations of the optical flow in time should not be used for estimation, because acceleration parameters are not uniquely determined by the affine parameters.
A general derivation of a 5D-*Affine Model* is proposed in Chap. 6. This model handles affine transformations in 5D-data, i.e., 2D camera grid sequences, and also brightness changes coming from motion of the camera. Further the 5D-*Affine Model* is combined with the estimation technique used for *Scene Flow*. This leads to a high accuracy 3D motion estimation model (*Affine Scene Flow Model*), which shows high robustness. Synthetic experiments demonstrated that the novel *Affine Scene Flow Model* outperforms all other models, and allows estimation not only of translational motion parameters, but also rotational motion and acceleration parameters. The high performance of the model was demonstrated on two real plant leaf sequences.

7.2 Future Work

The 5D-*Affine Scene Flow Model* has been thoroughly tested on synthetic image data and on two real plant leaf sequences. A possible focus in future work may be on establishing an appropriate estimation framework and embed the algorithm into a measurement system. Apart from practical work we suggest future work on three fundamental parts: the model, the parameter estimation and data acquisition.
Although an accurate model based on a moving surface patch has been derived, the following extensions may further improve the accuracy.

- The affine optical flow model is based on first order approximations of surface structure and motion. Higher order affine models have been proposed for optical flow computation (cmp. Sec. 2.1.1). Including higher orders of optical flow allows to more accurately approximating 3D parameters, e.g., round surfaces or acceleration of a neighborhood. However, increasing order of the model also leads to more parameters which have to be estimated. This could be handled with a coarse-to-fine technique, where the order of the model is increased, if the residual of a simpler model is above a threshold (cmp. [47]).

- The *Affine Scene Flow Model* showed high accuracy in estimating smooth velocity vector fields on synthetic data. Growth is estimated based on 3D motion vectors. Modeling growth explicitly in the affine parameters could improve estimation results. Apart from growth other parameters of interest, e.g., reflectance, could be modeled as well.

In this work parameters of the *Affine Scene Flow Model* are estimated using two standard least squares estimators. Possible improvements of the estimation process include the following:

- The parameter estimation of the optical flow parameters is based on standard TLS. It was already shown in Chap. 2 that handling of outliers and incorporation

of covariance information improves estimation. The extension of the *Affine Scene Flow Model* using these estimators is straight forward. The main drawback of the more sophisticated estimators is their computational complexity. In order to speed up the estimation algorithm the implementation could be parallelized and implemented on graphics processing units (GPU), which showed high performance in optical flow computation recently [50].

- Total least squares estimation of optical flow implies motions well below one pixel per frame. A coarse-to-fine multi-grid and/or a warping technique [49, 47] should be integrated in the estimation process in order to allow larger motions.

- Most of the best estimators on the popular Middlebury-Benchmark [3] are global estimators. These estimators allow incorporation of additional constraints like smoothness of the motion vector field or the equivalence between the derivative of depth Z and surface normals Z_X and Z_Y. Furthermore direct estimation of 3D parameters without estimating optical flow as an intermediate result could be possible. Global estimators yield dense motion vector fields, but error analysis is more difficult compared to local estimators.

- Although the *Affine Model* allows simultaneous estimation of optical flow parameters in time and camera displacement direction, two estimation steps are needed for estimation of 3D parameters. First the 3D structure and surface slopes are estimated. 3D motion parameters are then obtained by solving an equation system build up out of the 3D structure information and the affine optical flow parameters. In order to solve for all parameters simultaneously a nonlinear estimator should be developed. This leads to an iterative estimation process, where iteratively one parameter set is kept fixed while the other parameters are estimated.

- The two estimation processes in the *Affine Scene Flow Model* are solved independently of each other. Error distributions should be forwarded to the next estimator or the next estimation process. There are two different errors to be handled. First the reliability of the different translational and affine flow parameters. In general the translational parameters are more reliable than the affine parameters, but also affine parameters in camera displacement direction have different errors compared to affine parameters in time or local coordinates. Reliability of parameters from different cameras in the *Scene Flow* estimation technique need to be estimated and used in an appropriate estimation scheme. The incorporation of error distributions may increase robustness of the estimator, e.g., in case of occlusions and shadowing, as shown for standard *Scene Flow* computation in [40].

- Experiments, where relative growth is estimated for fixed leafs (cmp. [63]), showed that large temporal averaging filters increase the reliability of growth estimation. Therefore the estimation framework should be extended to include

temporal averaging of parameters, e.g., plant leaf motion. This may be done in a variational framework, where smoothness constraints on parameter fields can be expressed explicitly.

Finally we discuss important issues concerning the acquisition setup:

- The developed 5D-*Affine Scene Flow Model* was tested for short sequences (5-9 frames). In future this algorithm should be used for analysis of long time leaf motion and/or growth, i.e., more than 100 frames. Therefore a system has to be developed, which ensures highly accurate acquisition of input data. Experiments with moving stages showed that accuracy of displacement of ≈ 10 μm leads to significant errors that can be clearly detected when using a highly accurate estimation scheme like the *Affine Scene Flow Model*. For instance when using a small baseline of 1 mm such inaccuracies lead to an error of 1 % which directly influences estimation of disparity and therefore depth Z. Recently 5×5 camera arrays have been released (see [77]), which allow small and constant baselines.

- Depending on the plant species, leafs may move, bend and fold in all possible positions. In order to get most reliable depth and motion estimates the leaf should be imaged fronto-parallel. Furthermore the leafs could move out of focus. Therefore it may be beneficial to track the leaf of interest. We suggest that a robot arm with a mounted camera array could follow the movements of the plant.

7.3 Closure

The main focus of this work was on derivation of a highly accurate model, allowing growth estimation of plants. The 5D-*Affine Scene Flow Model* showed highest accuracy on synthetic data compared to the three discussed 3D motion estimation models. However, a *method* needs not only a highly accurate model of the data, but also an appropriate discretization, estimation framework and acquisition setup. For instance, using a more sophisticated estimator than standard total least squares, can significantly improve the performance of the *method* (cmp. Chap. 6, where the estimation technique of *Scene Flow* is adapted to the *Affine Model*). In this context the derivation of the 5D-*Affine Scene Flow Model* is an important step in order to establish a *method* for plant leaf growth estimation.

Appendix A

Average Angular Error

The angular error is a common error measure used for optical flow evaluation (cmp. [4, 26]). A 2D motion vector $\mathbf{u} = [u_x, u_y]^{\mathrm{T}}$ can be represented as a normalized 3D vector $\mathbf{r} = [r_1, r_2, r_3]^{\mathrm{T}}/\sqrt{r_1^2 + r_2^2 + r_3^2}$ where $\mathbf{u} = [r_1, r_2]^{\mathrm{T}}/r_3$. The angular error describes the error as an angle in 3D

$$AE = \arccos\left(\mathbf{r}_t^{\mathrm{T}}\mathbf{r}_e\right) \tag{A.1}$$

with 3D vectors $\mathbf{r}_t$ and $\mathbf{r}_e$ denoting the true and the estimated flow respectively. The advantage of this error measure is, that it addresses directional and magnitude errors simultaneously. However, the angular error measure is biased [26], because the relation between relative and angular error depends on the magnitude of the motion vector. That means, that for constant relative error the angular error reaches a maximum for a displacement of 1 pixel per frame. Moreover the angular error yields different measures for positive and negative deviations from the true value. However, quantitative evaluation of synthetic experiments showed that the angular error yields a reasonable error measure. The average angular error is defined for a neighborhood of size N by

$$AAE = \frac{1}{N}\sum_{i}^{N} \arccos\left(\mathbf{r}_t(i)^{\mathrm{T}}\mathbf{r}_e(i)\right) \tag{A.2}$$

and may be used also for true 3D motion estimates, e.g., $\mathbf{r} = [U_X, U_Y, U_Z, 1]^{\mathrm{T}}$. Affine motion parameters are in general significantly smaller than translational parameters. Therefore angular errors of translational and affine motion parameters should be separated, because errors in translational parameters would dominate errors in affine motion parameters.

Appendix B

Synthetic Testsequences

Synthetic sequences are used for systematic error analysis of 2D optical flow and 3D scene flow estimation. We model a camera with perspective projection, but derivation of a sequence for a camera with orthogonal projection is analog. For each image sensor element the intersection between the line of sight and the object is calculated. Therefore the appearance of the object is specified as a translating and rotating plane.
Following [23] a point $\mathbf{X} = (X, Y, Z)^{\mathrm{T}}$ in 3D-space is represented in homogeneous coordinates as a 4D-vector $\mathbf{X} = (X_1, X_2, X_3, X_4)^{\mathrm{T}}$ with $X_4 \neq 0$ by

$$X = \frac{X_1}{X_4}\,, \quad Y = \frac{X_2}{X_4}\,, \quad Z = \frac{X_3}{X_4}\,. \tag{B.1}$$

Homogeneous coordinates are used to easily represent homogeneous transformations like affine transformations or perspective projections by a 3×4 matrix and to carry out calculations in the projective space. Homogeneous points with $X_4 = 0$ represent points at infinity. For further details on homogeneous representation see [23].
The image acquired by a camera is a projection of 3D points $\mathbf{X}$ onto a 2D image plane with points $\mathbf{x}$. This projection can be expressed by a matrix multiplication of 3D points $\mathbf{X}$ with a camera matrix $\mathbf{K}$. Using a projective camera model the corresponding point $\mathbf{x}$ on the image plane can be calculated by

$$\mathbf{x} = \mathbf{K} \cdot \mathbf{X}\,, \text{ with } \mathbf{K} = \begin{pmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{B.2}$$

with focal length f and coordinates of the principal point (p_x, p_y). The camera matrix used here is sufficient to model typical CCD cameras, where skew parameter is negligible and aspect ratio is one.

The projected 2D point is given by

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{f}{Z}X + p_x \\ \frac{f}{Z}Y + p_y \\ 1 \end{pmatrix} . \tag{B.3}$$

A rectangular surface patch is defined by a surface normal $\mathbf{n} = (Z_X, Z_Y, -1)^{\mathrm{T}}$ and its central position $\mathbf{X_0} = (X_0, Y_0, Z_0)^{\mathrm{T}}$. The patch moves with translational velocity $\mathbf{N}$ and rotates around a vector $\mathbf{v} = (v_X, v_Y, v_Z)^{\mathrm{T}}$ positioned at its center $\mathbf{X}_0$ with angular velocity ω. This yields a combined translational and angular velocity of $\mathbf{U} = \mathbf{N} + \mathbf{\Omega} \times \mathbf{R}$ for each point $\mathbf{x_P} = (x_P, y_P)^{\mathrm{T}}$ on the patch with $\mathbf{\Omega} = \omega\mathbf{v} = (\Omega_X, \Omega_Y, \Omega_Z)^{\mathrm{T}}$ and distance to the patch center $\mathbf{R}$.
The surface of a patch centered at the origin is defined by $\mathbf{P_0} = (x_P, y_P, 0, 1)^{\mathrm{T}}$, where $\mathbf{x_P} = (x_P, y_P)^{\mathrm{T}}$ denotes the local position on the patch. Rotation of $\mathbf{P}_0$ by two 3×3 rotation matrices and one translation yields coordinates for a patch with sought for position $\mathbf{X}_0$, surface normal $\mathbf{n}$ and motion $\mathbf{U}$. Rotation matrix $\mathbf{R_0}$ ensures surface normal $\mathbf{n}$ of the patch for $t = 0$. Rotation axis is given by $\mathbf{q} = (Z_X, -Z_Y, 0)^{\mathrm{T}}$ and rotation angle is $\omega_0 = \arccos(1/\sqrt{1 + {Z_X}^2 + {Z_Y}^2})$. The second matrix rotates the patch around a vector $\mathbf{v}$ fixed at patch origin $\mathbf{X}_0$ with rotational velocity ω_t. This rotation and translation of the patch are combined in one matrix $\mathbf{TR}(t)$. 3D points on the surface patch are then defined by

$$\mathbf{P_s} = \mathbf{TR}(t) \cdot \mathbf{R_0} \cdot \mathbf{P_0} \tag{B.4}$$

with rotation matrix

$$\mathbf{R_0} = \begin{pmatrix} \mathbf{R}(\mathbf{q}, \omega_0) & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{pmatrix} \tag{B.5}$$

where

$$\mathbf{R}(\mathbf{q}, \omega_0) = \begin{pmatrix} \cos(\omega_0) + q_X^2\,(1 - \cos(\omega_0)) & q_X q_Y\,(1 - \cos(\omega_0)) - q_Z \sin(\omega_0) & q_X q_Z\,(1 - \cos(\omega_0)) + q_Y \sin(\omega_0) \\ q_Y q_X\,(1 - \cos(\omega_0)) + q_Z \sin(\omega_0) & \cos(\omega_0) + q_Y^2\,(1 - \cos(\omega_0)) & q_Y q_Z\,(1 - \cos(\omega_0)) - q_X \sin(\omega_0) \\ q_Z q_X\,(1 - \cos(\omega_0)) - q_Y \sin(\omega_0) & q_Z q_Y\,(1 - \cos(\omega_0)) + q_X \sin(\omega_0) & \cos(\omega_0) + q_Z^2\,(1 - \cos(\omega_0)) \end{pmatrix} \tag{B.6}$$

and combined rotational and translational matrix

$$\mathbf{TR}(t) = \begin{pmatrix} \mathbf{R}(\mathbf{v}, \omega_t t) & \mathbf{X_0} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{pmatrix} . \tag{B.7}$$

Translational motion $\mathbf{N}$ and acceleration $\mathbf{A}$ of the patch and displacement $\mathbf{S} = [s_x, s_y]^{\mathrm{T}}$ of the camera is included in

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}(t) = \begin{pmatrix} X_0 + N_X t + 0.5 A_X t^2 - s_x s \\ Y_0 + N_Y t + 0.5 A_Y t^2 - s_y s \\ Z_0 + N_Z t + 0.5 A_Z t^2 \end{pmatrix} . \tag{B.8}$$

Figure B.1: a: Central image of a sinusoidal sequence, b: gray value coded depth map and c: rendered patch in 3D with ground truth velocities.

Projection of $\mathbf{P}_s$ onto the image plane yields the corresponding points $\mathbf{p}$

$$\mathbf{p} = \mathbf{K} \cdot \mathbf{TR}(t) \cdot \mathbf{R} \cdot \mathbf{P_0} \; . \tag{B.9}$$

Inversion of (B.9) returns coordinates and motion of the plane given at a certain pixel element on the sensor. Intensity values are mapped onto the surface by

$$I(x_P, y_P) = o + a \cos\left(\frac{2\pi \hat{x}_P}{\lambda_1}\right) \cos\left(\frac{2\pi \hat{y}_P}{\lambda_2}\right) \tag{B.10}$$

$$\hat{x}_P = x_P \cos(\alpha) + y_P \sin(\alpha) \tag{B.11}$$

$$\hat{y}_P = -x_P \sin(\beta) + y_P \cos(\beta) \tag{B.12}$$

Figure B.1 shows an intensity image of a sequence, the gray value coded depth map Z and a rendered 3D view of a patch with 3D motion vectors. The parameters of this inclined surface are $Z_0 = 100$ mm, $Z_X = 1$, $Z_Y = 0$, $N_X = 0.00733$ mm/frame, $N_Y = -0.00367$ mm/frame, $N_Z = -0.6$ mm/frame, $f = 12$ mm, $\lambda_1 = \lambda_2 = 0.3$ mm, $\alpha = \beta = 0°$, $o = 127.5$ and $a = 127.5$. Of course, other intensity data may be projected onto the surface, e.g., a radial sinus. Motion and acceleration of the plane for each pixel of the sensor is calculated for quantitative error analysis. 3D motion is given by

$$\frac{dX}{dt} = U_X = N_X + A_X t + \Omega_Y \Delta Z - \Omega_Z \Delta Y \tag{B.13}$$

$$\frac{dY}{dt} = U_Y = N_Y + A_Y t + \Omega_Z \Delta X - \Omega_X \Delta Z \tag{B.14}$$

$$\frac{dZ}{dt} = U_Z = N_Z + A_Z t + \Omega_X \Delta Y - \Omega_Y \Delta X \tag{B.15}$$

with $\Omega_i = v_i\omega$. Optical flow components u_x, u_y are calculated by projecting the 3D motion onto the image plane

$$u_x = \frac{f}{Z}\left(U_X - \frac{x_0}{f}U_Z\right) \tag{B.16}$$

$$u_y = \frac{f}{Z}\left(U_Y - \frac{y_0}{f}U_Z\right). \tag{B.17}$$

Affine components of the optical flow are derived by partial derivatives of the optical flow in the corresponding dimension. Affine flow parameters caused by camera displacement s_x and s_y in X-and Y-direction, i.e., disparity ν and its partial derivatives are

$$\begin{aligned} \nu &= -\frac{f}{Z}, \quad & \frac{\partial\nu}{\partial x_0} &= \frac{f}{Z^2}\frac{\partial Z}{\partial x_0}, \quad & \frac{\partial\nu}{\partial y_0} &= \frac{f}{Z^2}\frac{\partial Z}{\partial y_0}, \\ \frac{\partial\nu}{\partial s_x} &= \frac{f}{Z^2}\frac{\partial Z}{\partial s_x}, \quad & \frac{\partial\nu}{\partial s_y} &= \frac{f}{Z^2}\frac{\partial Z}{\partial s_y}, \quad & \frac{\partial\nu}{\partial t} &= \frac{f}{Z^2}\frac{\partial Z}{\partial t}. \end{aligned} \tag{B.18}$$

Bibliography

[1] G. Adiv, "Determining 3-d motion and structure from optical flow generated by several moving objects," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 7, no. 4, pp. 384–401, 1985.

[2] H. Arsenault and M. Denis, "Image processing in signal-dependent noise," *Canadian Journal of Physics*, vol. 61, pp. 309–317, 1983.

[3] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," in *Intern. Journal of Computer Vision*, 2007, pp. 1–8.

[4] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *Intern. Journal of Computer Vision*, vol. 12, pp. 43–77, 1994.

[5] M. Bertero, T. A. Pogcio, and V. Torre, "Ill-posed problems in early vision," *Proc. of the IEEE*, vol. 76, no. 8, pp. 869–889, August 1988.

[6] J. Bigün and G. H. Granlund, "Optimal orientation detection of linear symmetry," in *Intern. Conference on Computer Vision*, 1987, pp. 433–438.

[7] J. Bigün and G. H. Granlund, "Optical flow based on the inertia matrix of the frequency domain," in *Proceedings from SSAB Symposium on Picture Processing*. Lund University, Sweden: SSAB, March 1988, pp. 132–135, report LiTH-ISY-I-0934, Computer Vision Laboratory, Linköping University, Sweden, 1988.

[8] M. J. Black, "Robust incremental optical flow," Ph.D. dissertation, Yale University, 1992.

[9] M. J. Black and P. Anandan, "The robust estimation of multiple motions: parametric and piecewise-smooth flow fields," in *Computer Vision and Image Understanding*, vol. 63(1), 1996, pp. 75–104.

[10] M. J. Black, D. J. Fleet, and Y. Yacoob, "Robustly estimating changes in image appearence," *Computer Vision and Image Understanding*, vol. 7, no. 1, pp. 8–31, 2000.

[11] M. J. Black and A. Jepson, "Estimating optical flow in segmented images using variable-order parametric models with local deformations," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 972–986, 1996.

[12] A. Blake and M. Isard, *Active Contours*. Springer, 1998.

[13] T. Brox, J. Weickert, B. Burgeth, and P. Mrázek, "Nonlinear structure tensors," Saarland University, Saarbrücken, Germany, Revised version of Technical Report No. 113, 2004.

[14] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/kanade meets horn/schunck: Combining local and global optic flow methods," *Intern. Journal of Computer Vision*, vol. 61, no. 3, pp. 211–231, 2005.

[15] C. Cason, "Persistence of vision ray tracer (POV-Ray), version 3.6, Windows," www.povray.com, 2005.

[16] W. Chojnacki, M. J. Brooks, and A. van den Hengel, "Rationalising the renormalisation method of kanatani," *Journal of Mathematical Imaging and Vision*, vol. 14, pp. 200–1, 2001.

[17] W. Chojnacki, M. J. Brooks, A. van den Hengel, and D. Gawley, "A new constrained parameter estimator for computer vision applications," *Image and Vision Computing*, vol. 22, no. 2, pp. 85 – 91, 2004, statistical Methods in Video Processing.

[18] D. J. Fleet and Y. Weiss, "Optical flow estimation," in *Mathematical models for Computer Vision: The Handbook*, N. Paragios, Y. Chen, and O. F. (eds.), Eds. Springer, 2005.

[19] B. Galvin, B. Mccane, K. Novins, D. Mason, and S. Mills, "Recovering motion fields: An evaluation of eight optical flow algorithms," in *British Machine Vision Conference*, 1998, pp. 195–204.

[20] S. Geman and D. McClure, "Statistical methods for tomographic image reconstruction," *Bull. Intern. Statist. Inst.*, vol. LII-4, pp. 5–21, 1987.

[21] H. Gharavi and S. Gao, "3-d motion estimation using range data," *IEEE Trans. on Intelligent Transportation Systems*, vol. 8, no. 1, pp. 133–143, March 2007.

[22] Google, "Google street view," http://maps.google.com/help/maps/streetview/, 2005, google Inc.

[23] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[24] H. Haussecker, "Interaction of radiation with matter," in *Handbook of Computer Vision and Applications*, vol. 1. Academic Press, 1999, pp. 37–62.

[25] H. Haussecker and D. J. Fleet, "Computing optical flow with physical models of brightness variation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 661–673, June 2001.

[26] H. Haussecker and H. Spies, "Motion," in *Handbook of Computer Vision and Applications*, 1st ed., B. Jähne, H. Haussecker, and P. Geißler, Eds. Academic Press, 1999, vol. 2, ch. 13, pp. 309–396.

[27] B. Horn, *Robot Vision*. McGraw-Hill, 1986.

[28] B. Horn and B. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–204, 1981.

[29] Y. Hsu, H. Nagel, and G. Rekers, "New likelihood test methods for change detection in image sequences," *CVGIP: Graphical Models and Image Processing*, vol. 26, no. 1, pp. 73–106, April 1984.

[30] P. J. Huber, "Robust estimation of a location parameter," *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.

[31] F. Huguet and F. Devernay, "A variational method for scene flow estimation from stereo sequences," in *Intern. Conference on Computer Vision*, 2007.

[32] B. Jähne, *Digitale Bildverarbeitung*, 3rd ed. Springer, 1993.

[33] B. Jähne, *Spatio-temporal image processing*, ser. Lecture Notes in Computer Science. Springer Verlag, 1993.

[34] B. Jähne, "Performance characteristics of low-level motion estimators in spatiotemporal images," in *DAGM-Workshop Performance Characteristics and Quality of Computer Vision Algorithms, Braunschweig, September 18, 1997*, W. Foerstner, Ed., Univ. Bonn, 1997.

[35] B. Jähne, H. Haussecker, and P. Geissler, *Handbook of Computer Vision and Applications*, 1st ed. Academic Press, 1999.

[36] S. Ju, M. J. Black, and A. D. Jepson, "Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency," in *IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, June 1996, pp. 307–314.

[37] K. Kanatani, "Structure from motion without correspondence: general principle," in *Proc. Image Understanding Workshop*, 1985, pp. 10 711–6.

[38] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*. Elsevier, 1996.

[39] K. Krajsek and R. Mester, "Marginalized maximum a posteriori hyper-parameter estimation for global optical flow techniques," in *Bayesian Inference and Maximum Entropy Methods In Science and Engineering*, Paris, France, November 2006.

[40] R. Li and S. Sclaroff, "Multi-scale 3d scene flow from binocular stereo sequences," *Computer Vision and Image Understanding*, vol. 110, no. 1, pp. 75–90, 2008.

[41] H. Longuet-Higgins and K. Prazdny, "The interpretation of a moving retinal image," *Proc. of The Royal Society of London B, 208*, pp. 385–397, 1980.

[42] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Seventh Intern. Joint Conference on Artificial Intelligence*, Vancouver, Canada, August 1981, pp. 674–679.

[43] L. Matthies, R. Szeliski, and T. Kanade, "Kalman filter-based algorithms for estimating depth from image sequences," *Intern. Journal of Computer Vision*, vol. 3, pp. 209–236, 1989.

[44] M. Mühlich and T. Aach, "A theory for multiple orientation estimation," in *Proc. European Conference on Computer Vision*, ser. LNCS, H. Bischof and A. Leonardis, Eds., no. 3952. Springer, 2006, pp. (II) 69–82.

[45] Y. Nakamura, T. Matsuura, K. Satoh, and Y. Ohta, "Occlusion detectable stereo–occlusion patterns in camera matrix," in *IEEE Conference Computer Vision and Pattern Recognition*, 1996, pp. 371–378.

[46] O. Nestares, D. J. Fleet, and D. J. Heeger, "Likelihood functions and confidence bounds for Total Least Squares estimation," in *IEEE Conference Computer Vision and Pattern Recognition*, 2000.

[47] T. Nir, A. M. Bruckstein, and R. Kimmel, "Over-parameterized variational optical flow," *Intern. Journal of Computer Vision*, vol. 76, no. 2, pp. 205–216, 2008.

[48] A. Oppenheim, R. Schafer, and J. Stockham, T.G., "Nonlinear filtering of multiplied and convolved signals," in *Proc. of the IEEE*, vol. 56, 1968, pp. 1264 – 1291.

[49] N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert, "Highly accurate optic flow computation with theoretically justified warping," *Intern. Journal of Computer Vision*, vol. 67, no. 2, pp. 141–158, April 2006.

[50] T. Pock, M. Unger, D. Cremers, and H. Bischof, "Fast and exact solution of total variation models on the gpu," in *CVPR Workshop on Visual Computer Vision on GPU's*, Anchorage, Alaska, USA, Jun. 2008.

[51] S. Roth and M. J. Black, "On the spatial statistics of optical flow," *Intern. Journal of Computer Vision*, vol. 74, no. 1, pp. 33–50, 2007.

[52] P. D. Sampson, "Fitting conic sections to 'very scattered' data: An iterarive refinement of the bookstein algorithm," *Computer Graphics and Image Processing*, vol. 18, no. 1, pp. 97–108, January 1982.

[53] H. Scharr, "Optimal operators in digital image processing," Ph.D. dissertation, Interdisciplinary Center for Scientific Computing, University of Heidelberg, Germany, 2000.

[54] H. Scharr, "Optimal filters for extended optical flow," in *Complex Motion, 1. Intern. Workshop, Günzburg, Oct. 2004*, ser. Lecture Notes in Computer Science, B. Jähne, E. Barth, R. Mester, and H. Scharr, Eds., vol. 3417. Berlin: Springer Verlag, 2005.

[55] H. Scharr, "Towards a multi-camera generalization of brightness constancy," in *Complex Motion, 1. Intern. Workshop, Günzburg, Oct. 2004*, ser. Lecture Notes in Computer Science, B. Jähne, E. Barth, R. Mester, and H. Scharr, Eds., vol. 3417. Berlin: Springer Verlag, 2007, pp. 78–90.

[56] H. Scharr and T. Schuchert, "Simultaeous estimation of depth, motion and slopes using a camera grid," in *Vision Modeling and Visualization 2006*, L. Kobbelt, T. Kuhlen, T. Aach, and R. Westermann, Eds., Aachen, 2006, pp. 81–88.

[57] H. Scharr and R. Küsters, "A linear model for simultaneous estimation of 3d motion and depth," in *IEEE Workshop on Motion and Video Computing, Orlando, USA*, December 2002.

[58] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Intern. Journal of Computer Vision*, vol. 47, pp. 7–42, 2001.

[59] D. Schmundt and U. Schurr, "Plant-leaf growth," in *Handbook of Computer Vision and Applications*, B. Jähne, H. Haussecker, and P. Geißler, Eds. Academic Press, 1999.

[60] T. Schuchert and H. Scharr, "Simultaneous estimation of surface motion, depth and slopes under changing illumination," in *Pattern Recognition Symposium of the German Association for Pattern Recognition (DAGM)*, 2007.

[61] T. Schuchert, T. Aach, and H. Scharr, "Range flow for varying illumination," in *European Conference on Computer Vision*, vol. 5302/2008. Springer Berlin / Heidelberg, 2008, pp. 509–522.

[62] T. Schuchert and H. Scharr, "An affine optical flow model for dynamic surface reconstruction," in *Statistical and Geometrical Approaches to Visual Motion Analysis*, ser. Dagstuhl Seminar Proceedings, D. Cremers, B. Rosenhahn, and A. L. Yuille, Eds., no. 08291. Dagstuhl, Germany: Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, 2008. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2008/1620

[63] U. Schurr, A. Walter, S. Wilms, H. Spies, N. Kirchgeßner, H. Scharr, and R. Küsters, "Dynamics of leaf and root growth," in *12th Intern. Conference on Photosynthesis*, 2001.

[64] E. P. Simoncelli, "Design of multi-dimensional derivative filters," in *IEEE Intern. Conference on Image Processing*, Austin TX, 1994, pp. 790–793.

[65] E. P. Simoncelli, E. H. Adelson, and D. J. Heeger, "Probability distributions of optical flow," in *IEEE Conference on Computer Vision and Pattern Recognition*. Mauii, Hawaii: IEEE Computer Society, Jun 3-6 1991, pp. 310–315.

[66] Sony, "Eyetoy," www.eyetoy.com, 2005, sony Computer Entertainment Europe.

[67] H. Spies, H. Haussecker, B. Jähne, and J. Barron, "Differential range flow estimation," in *Pattern Recognition Symposium of the German Association for Pattern Recognition (DAGM)*, 1999, pp. 309–316.

[68] H. Spies and B. Jähne, "A general framework for image sequence processing," in *Fachtagung Informationstechnik*, 2001, pp. 125–132.

[69] H. Spies, B. Jahne, and J. Barron, "Range flow estimation," *Computer Vision and Image Understanding*, vol. 85, no. 3, pp. 209–231, March 2002.

[70] H. Spies, B. Jähne, and J. L. Barron, "Surface expansion from range data sequences," in *Pattern Recognition Symposium of the German Association for Pattern Recognition (DAGM)*. London, UK: Springer-Verlag, 2001, pp. 163–169.

[71] S.-C. Tai and S.-M. Yang, "A fast method for image noise estimation using laplacian operator and adaptive edge detection," in *Communications, Control and Signal Processing, 2008. ISCCSP 2008. 3rd Intern. Symposium on*, St Julians, 2008, pp. 1077–1081.

[72] D. Terzopoulos, "Image analysis using multigrid relaxation methods," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 8, no. 2, pp. 129–139, 1986.

[73] D. Toth, T. Aach, and V. Metzler, "Illumination-invariant change detection," in *4th IEEE Southwest Symposium on Image Analysis and Interpretation*. Austin, TX: IEEE Computer Society, April 2 - 4 2000, pp. 3–7.

[74] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Threedimensional scene flow," in *IEEE Intern. Conference on Computer Vision*, 1999, pp. 722–729.

[75] S. Vedula, S. Baker, S. Seitz, R. Collins, and T. Kanade, "Shape and motion carving in 6d," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2000, pp. 592–598.

[76] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 475–480, 2005.

[77] ViewPLUS and Point Grey, "ProFUSION 25, 5x5 Camera Array," http://www.viewplus.co.jp/products/profusion25/ProFUSION25-e.pdf, 2008.

[78] A. Walter, "Räumliche und zeitliche wachstumsmuster in wurzeln und blättern dikotyler pflanzen'," Ph.D. dissertation, University of Heidelberg, 2001.

[79] A. Waxman, B. Kamgar Parsi, and M. Subbarao, "Closed-form solutions to image flow equations for 3d structure and motion," *Intern. Journal of Computer Vision*, vol. 1, no. 3, pp. 239–258, October 1987.

[80] M. Yamamoto, P. Boulanger, J. A. Beraldin, and M. Rioux, "Direct estimation of range flow on deformable shape from a video rate range camera," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 15, no. 1, pp. 82–89, 1993.

[81] Y. Zhang and C. Kambhamettu, "Integrated 3d scene flow and structure recovery from multiview image sequences," in *IEEE Conference Computer Vision and Pattern Recognition*, 2000, pp. 2674–2681.

Curriculum Vitae

Personal Information:

Name	Tobias Schuchert
Date of Birth	July 23, 1980
Place of Birth	Essen, Germany

Qualification:

10/2000 - 10/2005	Diploma in Information Technology, TU Dortmund University
11/2005 - 05/2009	PhD Student, Forschungszentrum Jülich, ICG-3: Phytosphere, Image Processing Group and RWTH Aachen University, Institute of Imaging & Computer Vision
since 08/2009	Researcher, Fraunhofer IOSB, Karlsruhe

April 20, 2010

Acknowledgements

I would like to thank everyone who helped me during my thesis. In particular ...

... Prof. Til Aach for supervising my thesis and fruitful discussions,

... Prof. Uli Schurr for great support and interesting insights into plant physiology,

... Hanno Scharr for giving me this topic, endless discussions on perspective geometry, estimation processes and illumination modeling, and especially for his guidance and encouragement at all time,

... Richard Poiré and Andreas Fischbach for many discussions, many coffees and many other beverages...,

... Georg Dreissen and all other MaTAs for providing assistance with experiments and athletic challenges,

... Daniel Gorman for doing experiments with Fred I/II and Steve and his north American lifestyle,

... all at ICG-3, especially all PhD students, for interesting interdisciplinary discussions and a positive work environment,

... my friends and especially my family and Anja for their support and patience.

1. **Einsatz von multispektralen Satellitenbilddaten in der Wasserhaushalts- und Stoffstrommodellierung – dargestellt am Beispiel des Rureinzugsgebietes**
   von C. Montzka (2008), XX, 238 Seiten
   ISBN: 978-3-89336-508-1

2. **Ozone Production in the Atmosphere Simulation Chamber SAPHIR**
   by C. A. Richter (2008), XIV, 147 pages
   ISBN: 978-3-89336-513-5

3. **Entwicklung neuer Schutz- und Kontaktierungsschichten für Hochtemperatur-Brennstoffzellen**
   von T. Kiefer (2008), 138 Seiten
   ISBN: 978-3-89336-514-2

4. **Optimierung der Reflektivität keramischer Wärmedämmschichten aus Yttrium-teilstabilisiertem Zirkoniumdioxid für den Einsatz auf metallischen Komponenten in Gasturbinen**
   von A. Stuke (2008), X, 201 Seiten
   ISBN: 978-3-89336-515-9

5. **Lichtstreuende Oberflächen, Schichten und Schichtsysteme zur Verbesserung der Lichteinkopplung in Silizium-Dünnschichtsolarzellen**
   von M. Berginski (2008), XV, 171 Seiten
   ISBN: 978-3-89336-516-6

6. **Politikszenarien für den Klimaschutz IV – Szenarien bis 2030**
   hrsg.von P. Markewitz, F. Chr. Matthes (2008), 376 Seiten
   ISBN 978-3-89336-518-0

7. **Untersuchungen zum Verschmutzungsverhalten rheinischer Braunkohlen in Kohledampferzeugern**
   von A. Schlüter (2008), 164 Seiten
   ISBN 978-3-89336-524-1

8. **Inorganic Microporous Membranes for Gas Separation in Fossil Fuel Power Plants**
   by G. van der Donk (2008), VI, 120 pages
   ISBN: 978-3-89336-525-8

9. **Sinterung von Zirkoniumdioxid-Elektrolyten im Mehrlagenverbund der oxidkeramischen Brennstoffzelle (SOFC)**
   von R. Mücke (2008), VI, 165 Seiten
   ISBN: 978-3-89336-529-6

10. **Safety Considerations on Liquid Hydrogen**
    by K. Verfondern (2008), VIII, 167 pages
    ISBN: 978-3-89336-530-2

11. **Kerosinreformierung für Luftfahrtanwendungen**
von R. C. Samsun (2008), VII, 218 Seiten
ISBN: 978-3-89336-531-9

12. **Der 4. Deutsche Wasserstoff Congress 2008 – Tagungsband**
hrsg. von D. Stolten, B. Emonts, Th. Grube (2008), 269 Seiten
ISBN: 978-3-89336-533-3

13. **Organic matter in Late Devonian sediments as an indicator for environmental changes**
by M. Kloppisch (2008), XII, 188 pages
ISBN: 978-3-89336-534-0

14. **Entschwefelung von Mitteldestillaten für die Anwendung in mobilen Brennstoffzellen-Systemen**
von J. Latz (2008), XII, 215 Seiten
ISBN: 978-3-89336-535-7

15. **RED-IMPACT**
Impact of Partitioning, Transmutation and Waste Reduction Technologies on the Final Nuclear Waste Disposal
SYNTHESIS REPORT
ed. by W. von Lensa, R. Nabbi, M. Rossbach (2008), 178 pages
ISBN 978-3-89336-538-8

16. **Ferritic Steel Interconnectors and their Interactions with Ni Base Anodes in Solid Oxide Fuel Cells (SOFC)**
by J. H. Froitzheim (2008), 169 pages
ISBN: 978-3-89336-540-1

17. **Integrated Modelling of Nutrients in Selected River Basins of Turkey**
Results of a bilateral German-Turkish Research Project
project coord. M. Karpuzcu, F. Wendland (2008), XVI, 183 pages
ISBN: 978-3-89336-541-8

18. **Isotopengeochemische Studien zur klimatischen Ausprägung der Jüngeren Dryas in terrestrischen Archiven Eurasiens**
von J. Parplies (2008), XI, 155 Seiten, Anh.
ISBN: 978-3-89336-542-5

19. **Untersuchungen zur Klimavariabilität auf dem Tibetischen Plateau - Ein Beitrag auf der Basis stabiler Kohlenstoff- und Sauerstoffisotope in Jahrringen von Bäumen waldgrenznaher Standorte**
von J. Griessinger (2008), XIII, 172 Seiten
ISBN: 978-3-89336-544-9

20. **Neutron-Irradiation + Helium Hardening & Embrittlement Modeling of 9%Cr-Steels in an Engineering Perspective (HELENA)**
by R. Chaouadi (2008), VIII, 139 pages
ISBN: 978-3-89336-545-6

21. **in Bearbeitung**

22. **Verbundvorhaben APAWAGS (AOEV und Wassergenerierung) – Teilprojekt: Brennstoffreformierung – Schlussbericht**
von R. Peters, R. C. Samsun, J. Pasel, Z. Porš, D. Stolten (2008), VI, 106 Seiten
ISBN: 978-3-89336-547-0

23. **FREEVAL**
Evaluation of a Fire Radiative Power Product derived from Meteosat 8/9 and Identification of Operational User Needs
Final Report
project coord. M. Schultz, M. Wooster (2008), 139 pages
ISBN: 978-3-89336-549-4

24. **Untersuchungen zum Alkaliverhalten unter Oxycoal-Bedingungen**
von C. Weber (2008), VII, 143, XII Seiten
ISBN: 978-3-89336-551-7

25. **Grundlegende Untersuchungen zur Freisetzung von Spurstoffen, Heißgaschemie, Korrosionsbeständigkeit keramischer Werkstoffe und Alkalirückhaltung in der Druckkohlenstaubfeuerung**
von M. Müller (2008), 207 Seiten
ISBN: 978-3-89336-552-4

26. **Analytik von ozoninduzierten phenolischen Sekundärmetaboliten in *Nicotiana tabacum* L. cv Bel W3 mittels LC-MS**
von I. Koch (2008), III, V, 153 Seiten
ISBN 978-3-89336-553-1

27. **IEF-3 Report 2009. Grundlagenforschung für die Anwendung**
(2009), ca. 230 Seiten
ISBN: 978-3-89336-554-8

28. **Influence of Composition and Processing in the Oxidation Behavior of MCrAlY-Coatings for TBC Applications**
by J. Toscano (2009), 168 pages
ISBN: 978-3-89336-556-2

29. **Modellgestützte Analyse signifikanter Phosphorbelastungen in hessischen Oberflächengewässern aus diffusen und punktuellen Quellen**
von B. Tetzlaff (2009), 149 Seiten
ISBN: 978-3-89336-557-9

30. **Nickelreaktivlot / Oxidkeramik – Fügungen als elektrisch isolierende Dichtungskonzepte für Hochtemperatur-Brennstoffzellen-Stacks**
von S. Zügner (2009), 136 Seiten
ISBN: 978-3-89336-558-6

31. **Langzeitbeobachtung der Dosisbelastung der Bevölkerung in radioaktiv kontaminierten Gebieten Weißrusslands – Korma-Studie**
von H. Dederichs, J. Pillath, B. Heuel-Fabianek, P. Hill, R. Lennartz (2009), Getr. Pag.
ISBN: 978-3-89336-532-3

32. **Herstellung von Hochtemperatur-Brennstoffzellen über physikalische Gasphasenabscheidung**
von N. Jordán Escalona (2009), 148 Seiten
ISBN: 978-3-89336-532-3

33. **Real-time Digital Control of Plasma Position and Shape on the TEXTOR Tokamak**
by M. Mitri (2009), IV, 128 pages
ISBN: 978-3-89336-567-8

34. **Freisetzung und Einbindung von Alkalimetallverbindungen in kohlebefeuerten Kombikraftwerken**
von M. Müller (2009), 155 Seiten
ISBN: 978-3-89336-568-5

35. **Kosten von Brennstoffzellensystemen auf Massenbasis in Abhängigkeit von der Absatzmenge**
von J. Werhahn (2009), 242 Seiten
ISBN: 978-3-89336-569-2

36. **Einfluss von Reoxidationszyklen auf die Betriebsfestigkeit von anodengestützten Festoxid-Brennstoffzellen**
von M. Ettler (2009), 138 Seiten
ISBN: 978-3-89336-570-8

37. **Großflächige Plasmaabscheidung von mikrokristallinem Silizium für mikromorphe Dünnschichtsolarmodule**
von T. Kilper (2009), XVII, 154 Seiten
ISBN: 978-3-89336-572-2

38. **Generalized detailed balance theory of solar cells**
by T. Kirchartz (2009), IV, 198 pages
ISBN: 978-3-89336-573-9

39. **The Influence of the Dynamic Ergodic Divertor on the Radial Electric Field at the Tokamak TEXTOR**
von J. W. Coenen (2009), xii, 122, XXVI pages
ISBN: 978-3-89336-574-6

40. **Sicherheitstechnik im Wandel Nuklearer Systeme**
von K. Nünighoff (2009), viii, 215 Seiten
ISBN: 978-3-89336-578-4

41. **Pulvermetallurgie hochporöser NiTi-Legierungen für Implanat- und Dämpfungsanwendungen**
von M. Köhl (2009), XVII, 199 Seiten
ISBN: 978-3-89336-580-7

42. **Einfluss der Bondcoatzusammensetzung und Herstellungsparameter auf die Lebensdauer von Wärmedämmschichten bei zyklischer Temperaturbelastung**
von M. Subanovic (2009), 188, VI Seiten
ISBN: 978-3-89336-582-1

43. **Oxygen Permeation and Thermo-Chemical Stability of Oxygen Permeation Membrane Materials for the Oxyfuel Process**
by A. J. Ellett (2009), 176 pages
ISBN: 978-3-89336-581-4

44. **Korrosion von polykristallinem Aluminiumoxid (PCA) durch Metalljodidschmelzen sowie deren Benetzungseigenschaften**
von S. C. Fischer (2009), 148 Seiten
ISBN: 978-3-89336-584-5

45. **IEF-3 Report 2009. Basic Research for Applications**
(2009), 217 Seiten
ISBN: 978-3-89336-585-2

46. **Verbundvorhaben ELBASYS (Elektrische Basissysteme in einem CFK-Rumpf) - Teilprojekt: Brennstoffzellenabgase zur Tankinertisierung - Schlussbericht**
von R. Peters, J. Latz, J. Pasel, R. C. Samsun, D. Stolten
(2009), xi, 202 Seiten
ISBN: 978-3-89336-587-6

47. **Aging of ^{14}C-labeled Atrazine Residues in Soil: Location, Characterization and Biological Accessibility**
by N. D. Jablonowski (2009), IX, 104 pages
ISBN: 978-3-89336-588-3

48. **Entwicklung eines energetischen Sanierungsmodells für den europäischen Wohngebäudesektor unter dem Aspekt der Erstellung von Szenarien für Energie- und CO_2 - Einsparpotenziale bis 2030**
von P. Hansen (2009), XXII, 281 Seiten
ISBN: 978-3-89336-590-6

49. **Reduktion der Chromfreisetzung aus metallischen Interkonnektoren für Hochtemperaturbrennstofzellen durch Schutzschichtsysteme**
von R. Trebbels (2009), iii, 135 Seiten
ISBN: 978-3-89336-591-3

50. **Bruchmechanische Untersuchung von Metall / Keramik-Verbundsystemen für die Anwendung in der Hochtemperaturbrennstoffzelle**
von B. Kuhn (2009), 118 Seiten
ISBN: 978-3-89336-592-0

51. **Wasserstoff-Emissionen und ihre Auswirkungen auf den arktischen Ozonverlust**
Risikoanalyse einer globalen Wasserstoffwirtschaft
von T. Feck (2009), 180 Seiten
ISBN: 978-3-89336-593-7

52. **Development of a new Online Method for Compound Specific Measurements of Organic Aerosols**
by T. Hohaus (2009), 156 pages
ISBN: 978-3-89336-596-8

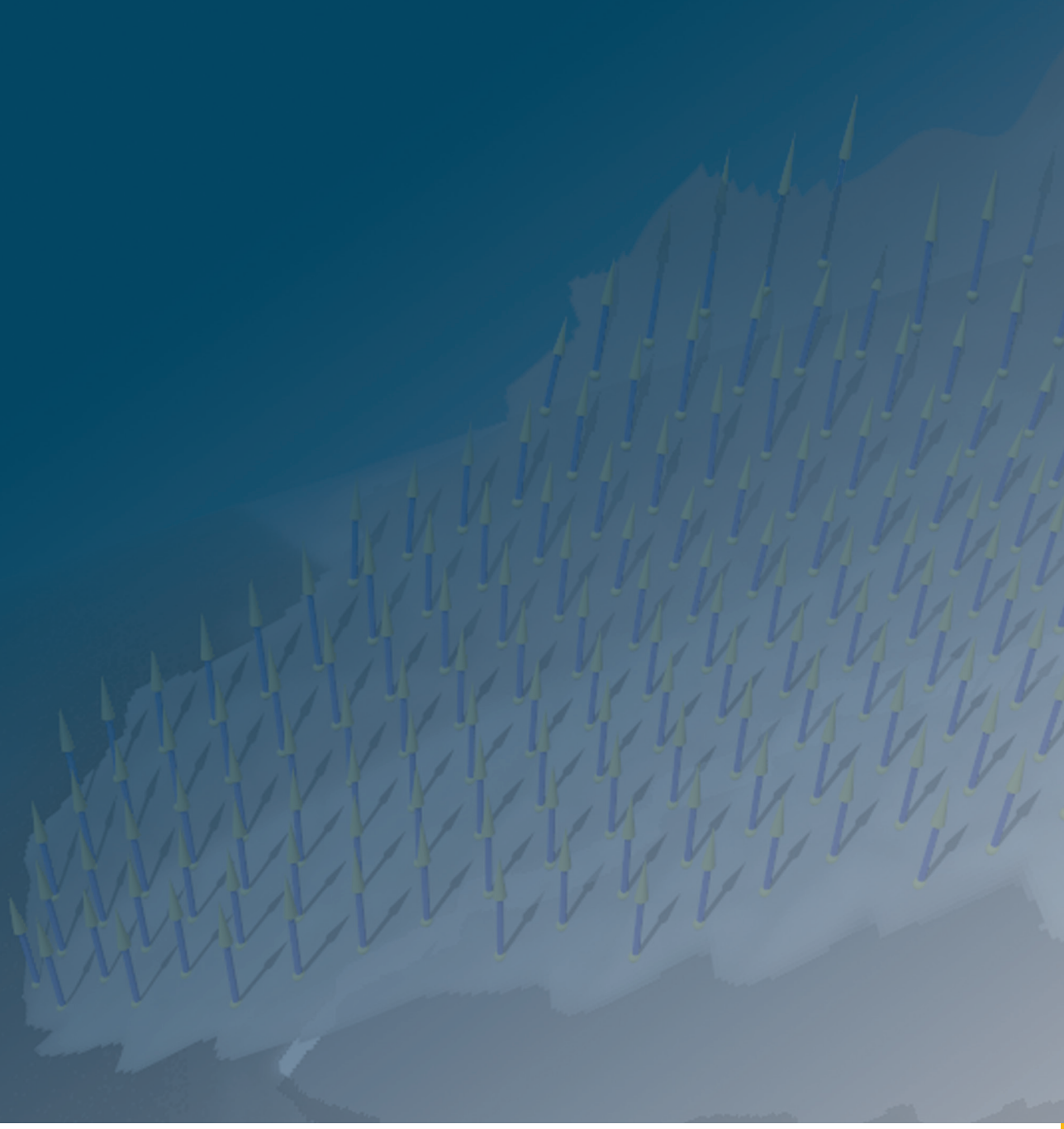
53. **Entwicklung einer FPGA basierten Ansteuerungselektronik für Justageeinheiten im Michelson Interferometer**
von H. Nöldgen (2009), 121 Seiten
ISBN: 978-3-89336-599-9

54. **Observation – and model – based study of the extratropical UT/LS**
by A. Kunz (2010), xii, 120, xii pages
ISBN: 978-3-89336-603-3

55. **Herstellung polykristalliner Szintillatoren für die Positronen-Emissions-Tomographie (PET)**
von S. K. Karim (2010), VIII, 154 Seiten
ISBN: 978-3-89336-610-1

56. **Kombination eines Gebäudekondensators mit H2-Rekombinatorelementen in Leichwasserreaktoren**
von S. Kelm (2010), vii, 119 Seiten
ISBN: 978-3-89336-611-8

57. **Plant Leaf Motion Estimation Using A 5D Affine Optical Flow Model**
by T. Schuchert (2010), X, 143 pages
ISBN: 978-3-89336-613-2



Energie & Umwelt / Energy & Environment
Band / Volume 57
ISBN 978-3-89336-613-2

JÜLICH
FORSCHUNGSZENTRUM