

3D Motion Analysis via Energy Minimization

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Andreas Wedel

aus

Siegburg

Bonn 2009

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der
Rheinischen Friedrich-Wilhelms-Universität Bonn.

1. Gutachter: Prof. Dr. Daniel Cremers
2. Gutachter: Prof. Dr. Bodo Rosenhahn

Tag der Promotion: 16. Oktober 2009

Diese Dissertation ist auf dem Hochschulserver der ULB Bonn unter
http://hss.ulb.uni-bonn.de/diss_online elektronisch publiziert.

Erscheinungsjahr: 2009

Acknowledgments



© creative commons, air-on, www.flickr.com

This work would not have been possible without the support and encouragement of my supervisors, Prof. Dr. Daniel Cremers and Dr. Uwe Franke. It was them, who motivated and prepared me for a good start into the fascinating journey of computer vision. They challenged me on my way to think outside of the box and to stay focused on a state-of-the-art research path. They will also be the ones to always stand up for me and advocate my work. I want to express my deep gratefulness for all their scientific, supervisory and professional mentoring. I also want to express my gratitude to the members of the audit committee, Prof. Dr. Bodo Rosenhahn, Prof. Dr.-Ing. Wolfgang Förstner, and Prof. Dr. Reinhard Klein.

Special thanks goes to Dr. Thomas Pock and Dr. Thomas Brox, who greatly influenced my work. I have rarely met people with such clear and profound

ability to explain even the most complex mathematical relationships as these fine men. Thomas Pock was the one who showed me the path of total variation denoising for optical flow, a path he himself has been traveling on. I thank Thomas Brox for his patience in explaining small and simple, though very essential, mathematical basics to me.

Many thanks also to my co-authors Tobi Vaudrey, Jürgen Braun, Dr. Clemens Rabe, Dr. Jens Klappstein, Dr. Hernán Badino, Dr. Thomas Schoenemann and Prof. Dr. Reinhard Klette. Especially, I want to thank Tobi Vaudrey and Clemens Rabe, whose support has been of great value and who contributed with numerous illustrations to this thesis.

Concerning support related to beverage and food nourishment during my PhD journey, I would like to thank Fridtjof and Heidi for espresso and cappuccino, Mario for steaks, Clemens for Pizzas and Dave for doughnuts. For my spiritual nutrition, I was helped along the way by the SYMs group at IBC Stuttgart and the Baptist church in Böblingen with a special thanks to Hans-Martin Beutel. Through all times on my journey, I have learned to think positive and put my faith in God.

On my journey I experienced both sunny and cloudy days. I have met many people who pointed out directions, supported and prayed for me, and helped me over the deep canyons in troubled times. There are many more people that have helped along the way who I owe a huge dept of gratitude, but they are too numerous to name. With some people I have lost sight during my journey, much to my regret.

It is all you people, who you have build the pathways and bridges I walked to write this thesis. You all are part of this work and without every single one of you, I would not have be able to successfully manage the challenges of a PhD thesis.

Abstract

This work deals with 3D motion analysis from stereo image sequences for driver assistance systems. It consists of two parts: the estimation of motion from the image data and the segmentation of moving objects in the input images. The content can be summarized with the technical term *machine visual kinesthesia*, the sensation or perception and cognition of motion.

In the first three chapters, the importance of motion information is discussed for driver assistance systems, for machine vision in general, and for the estimation of ego motion. The next two chapters delineate on motion perception, analyzing the apparent movement of pixels in image sequences for both a monocular and binocular camera setup. Then, the obtained motion information is used to segment moving objects in the input video. Thus, one can clearly identify the thread from analyzing the input images to describing the input images by means of stationary and moving objects. Finally, I present possibilities for future applications based on the contents of this thesis. Previous work in each case is presented in the respective chapters.

Although the overarching issue of motion estimation from image sequences is related to practice, *there is nothing as practical as a good theory* (Kurt Lewin). Several problems in computer vision are formulated as intricate energy minimization problems. In this thesis, motion analysis in image sequences is thoroughly investigated, showing that splitting an original complex problem into simplified sub-problems yields improved accuracy, increased robustness, and a clear and accessible approach to state-of-the-art motion estimation techniques.

In Chapter 4, optical flow is considered. Optical flow is commonly estimated by minimizing the combined energy, consisting of a data term and a smoothness term. These two parts are decoupled, yielding a novel and iterative approach to optical flow. The derived Refinement Optical Flow framework is a clear and straight-forward approach to computing the apparent image motion vector field. Furthermore this results currently in the most accurate motion estimation techniques in literature. Much as this is an engineering approach of fine-tuning precision to the last detail, it helps to get a better insight into the problem of motion estimation. This profoundly contributes to state-of-the-art research in motion analysis, in particular facilitating the use of motion estimation in a wide range of applications.

In Chapter 5, scene flow is rethought. Scene flow stands for the three-dimensional motion vector field for every image pixel, computed from a stereo image sequence. Again, decoupling of the commonly coupled approach of estimating three-dimensional position and three dimensional motion yields an approach to scene flow estimation with more accurate results and a considerably lower computational load. It results in a dense scene flow field and enables additional applications based on the dense three-dimensional motion vector field, which are to be investigated in the future. One such application is the segmentation of moving objects in an image sequence. Detecting moving objects within the scene is one of the most important features to extract in image sequences from a dynamic environment. This is presented in Chapter 6.

Scene flow and the segmentation of independently moving objects are only first steps towards machine visual kinesthesia. Throughout this work, I present possible future work to improve the estimation of optical flow and scene flow. Chapter 7 additionally presents an outlook on future research for driver assistance applications. But there is much more to the full understanding of the three-dimensional dynamic scene. This work is meant to inspire the reader to think outside the box and contribute to the vision of *building perceiving machines*.

Contents

Acknowledgments	ii
Abstract	iii
Introduction	1
1 Introduction	1
1.1 Eyes for Intelligent Vehicles	1
1.2 Intelligent Vehicle Safety Systems	3
1.3 Motion Analysis via Visual Kinesthesia	4
1.4 Energy Minimization Problems	6
1.5 Contributions of this Thesis	7
1.5.1 Visual Kinesthetic Perception (Theory)	7
1.5.2 Visual Kinesthetic Cognition (Application)	8
2 Vision in Natural Sciences	9
2.1 Biologically Inspired Perception	10
2.2 History of Motion Analysis	11
2.3 Categorization of Perception	12
3 Ego Motion Estimation	13
3.1 Image Formation and Coordinate Systems	14
3.1.1 Image Formation	14
3.1.2 Coordinate Systems	15
3.2 Fundamental Matrix Geometry	17
3.3 Robust Ego-Motion Estimation	19
3.3.1 The Linear Criterion	19
3.3.2 The Non-Linear Criteria	20
3.3.3 Re-weighted Least Squares and Parametrization	20
I Visual Kinesthetic Perception (Theory)	21
4 Optical Flow	22
4.1 Approaches in Literature	24
4.1.1 Census Based Optical Flow	25
4.1.2 The Optical Flow Constraint	25
4.1.3 Total Variation Optical Flow	28
4.1.4 Other Optical Flow Approaches	30
4.2 A General Flow Refinement Framework	30
4.2.1 Data Term Optimization	30
4.2.2 Smoothness Term Evaluation	36
4.2.3 Implementation Details	40
4.3 Increasing Robustness to Illumination Changes	42
4.4 Experimental Results	46
4.4.1 Quantitative Evaluation	46
4.4.2 Results for Traffic Scenes	53
4.5 Ideas for Future Research	56
5 Scene Flow	57
5.1 Introduction and Related Work	58

5.2	Formulation and Solving of the Constraint Equations	59
5.2.1	Stereo Computation	60
5.2.2	Scene Flow Motion Constraints	61
5.2.3	Solving the Scene Flow Equations	62
5.2.4	Visualizing the 3D Velocity Field	63
5.3	Comparison of Results for Different Stereo Inputs	63
5.4	Derivation of a Pixel-wise Accuracy Measure	66
5.5	Ideas for Future Research	75
II	Visual Kinesthetic Cognition (Application)	76
6	Flow Cut - Moving Object Segmentation	77
6.1	Segmentation Algorithm	79
6.1.1	Energy Functional	79
6.1.2	Graph Mapping	80
6.2	Deriving the Motion Metrics	81
6.2.1	Monocular Motion Analysis	81
6.2.2	Stereo Motion Analysis	84
6.3	Experimental Results and Discussion	88
6.3.1	Robust Segmentation	89
6.3.2	Comparing Monocular and Binocular Segmentation	89
6.4	Ideas for Future Research	90
7	Research Ideas using Visual Kinesthesia	93
7.1	Extending Warp Cut	94
7.2	Motion from Motion	96
7.3	Dynamic Free Space	97
	Epilogue	99
III	Appendix	100
A	Data Terms for Refinement Optical Flow	101
B	Quadratic Optimization via Thresholding	103
B.1	Karush-Kuhn-Tucker (KKT) Conditions	103
B.2	Single Data Term	103
B.3	Two Data Terms	105
C	Variational Image Flow Framework	109
D	Space-Time Multi-Resolution Cut	111
D.1	Literature Overview	111
D.2	Multi-Resolution Graph-Cut	113
	Bibliography	115

Introduction



We're teaching cars to see, because mum can't be everywhere.

© Daimler active safety campaign

Contents

1.1	Eyes for Intelligent Vehicles	1
1.2	Intelligent Vehicle Safety Systems	3
1.3	Motion Analysis via Visual Kinesthesia	4
1.4	Energy Minimization Problems	6
1.5	Contributions of this Thesis	7
1.5.1	Visual Kinesthetic Perception (Theory)	7
1.5.2	Visual Kinesthetic Cognition (Application)	8

1.1 Eyes for Intelligent Vehicles

“Keep your eyes on the road” is one of the first rules taught to a new driver [87]. But as with many rules, we do not always practice what we preach. Answering cell phones, paying attention to route guidance systems, applying make up and filing complicated paperwork often catch our attention. We are frequently distracted from keeping our eyes focused on the road. Even for experienced drivers, the monotonous task of driving on the interstate highway can be very tiresome. Our eyes are getting tired and our time of reaction is slowing down; the risk of accidents increases.

The human eye has its limitations, which can have deadly consequences. Nine out of ten car crashes are caused by human error. In more than 20 percent of those crashes, sleep is the culprit [87, 23].

Over the last decades, vehicles have become indispensable in our everyday life. They have developed from a pure means of transportation to a central piece of our lifestyle, be it a sports car or a luxury car, or even a camper van. The development of cars is ever-progressing and hopefully many accidents can “be prevented in the future if vehicles are equipped with suitable assistance systems. A worthy goal, if ever there was one” [23]...



Figure 1.1: The *left image* illustrates the camera setup in the demonstrator car. The *middle image* shows the Night Vision System [17] and the *right image* the Speed Limit Recognition, two computer vision based assistance applications in the new Daimler E class (W212 model).

The environment perception group at Daimler “is exploring new ways to give cars their own eyes” [87]. A pair of video cameras, mounted about 30 centimeter apart, monitors the road in front of the vehicle. The video input signal is transferred to a computer. Here, the signal is analyzed algorithmically and relevant information about the image content is obtained. Typical applications based on the visual perception of the environment range from low level image processing to higher level pattern recognition and image understanding.

Image quality enhancement (e. g. remove noise or increase contrast) serves for better perception of the vehicle environment by presenting the processed image to the vehicle operator. One example of such a technique is the *Night Vision* assistant developed by Bosch in cooperation with Mercedes-Benz (see Figure 1.1 for a sample image). The Night Vision system provides increased visibility by illuminating the scene with infrared light and showing an image to the driver which is increased in contrast.

High level image processing goes a step further, analyzing the camera images to extract scene information. This includes vehicle and lane detection, traffic sign recognition, scene model reconstruction, and the perception of motion. The whole process of information extraction from images is called machine vision or computer vision. This information may again be passed onto a situation analysis level which warns the driver or takes over the control of the vehicle. Intelligent vehicles with such cognitive skills will help to prevent crashes.

Some applications such as lane keeping or vehicle following are well-known applications which every driver performs on a daily basis. Detecting lane markings and other traffic participants is a preliminary requirement to automate these rather simple tasks.

In order to automate driving maneuvers, simply detecting other traffic participants in the camera images often is not sufficient. In a driving vehicle, the image content is moving although objects may be static. If, additionally, image contents move in different directions the task may get rather complex. Intelligent vehicles have to cope with different object motions and with different types of uncertainties. In traffic environments, the knowledge about the dynamic movement in the scene can make the difference between a good and a bad choice when it comes to decision making.

Eyes for intelligent vehicles do not blink. They stay focused on the road and do not get tired. These eyes need to be trained to perceive the environment with high precision and to develop cognitive skills. Such skills will one day help intelligent vehicles to make autonomous decisions and to increase traffic safety. This thesis acquires novel ways of “teaching cars to see, because mum can’t be everywhere”¹.

¹Slogan taken from the Daimler active safety campaign.



Figure 1.2: In 2003 outside of cities 28% of all vehicle fatalities were car-to-car head-on collisions. If either car knew the exact dynamic movement of the approaching car, such collisions might be avoided by taking reactive actions. Images and text © Torsten Wiche, available on-line at <http://www.crashtec.de/>.

1.2 Intelligent Vehicle Safety Systems

Modern vehicles already support the operator in many ways. Active safety systems, such as the Electronic Stability Program (ESP) or the Active Brake Assistant (ABA), aim at reducing the extent of losing control of the vehicle. They take corrective action by directly controlling the vehicle. Most systems solely rely on the actual state of the surveyed vehicle, neglecting the surrounding environment and other traffic participants. Although the value of such information is unquestioned and strategies implying it exist, the problem of surveying the vehicle environment is not yet solved. The challenge to be solved becomes obvious when considering that already small changes of the environment and the involved vehicles may lead to completely different autonomous strategies. Hence, the required accuracy is beyond current state-of-the-art vehicle environment perception systems. In this thesis, I will develop novel concepts yielding optimal solutions (with respect to a given objective function) to motion estimation and object detection and solving the environment surveying task with unprecedented accuracy.

Although the benefit of active safety systems is evident, intervening in the vehicle's operation could irritate drivers who like to be in control. The problem with such systems is that "You don't want a system in New York City that's slamming on brakes all the time when someone's in front of you, because someone will slam into the back of you," as Francis Memole, vice president of sales and marketing at Iteris Inc., a developer of vehicle safety systems, points out [87]. A different way to support the driver is a non-intervening safety system. Non-intervening alert systems warn the driver of approaching hazards but do not take over control of the vehicle's operation.

However, driver acceptance of such intelligent safety systems is not guaranteed. "Unless these things are made very much second nature and don't need technological interaction, I think they're going to be hard to adjust to" [87]. Safety systems for intelligent vehicles need to integrate naturally into the driving process. The false alarm rate of such systems has to be very low, hence, they need accurate and reliable environment perception systems. This thesis tackles the accuracy and robustness issue, by evaluating information obtained in multiple images to perceive scene structure and motion with high precision.

Intelligent vehicles are equipped with a range of sensors for environment perception. A comparison of different sensors for intelligent vehicles is found in [5]. In this thesis, video cameras are used to monitor the road in front of the vehicle. Compared to other sensors, video cameras represent the solution with least hardware cost and most spatial resolution. While other

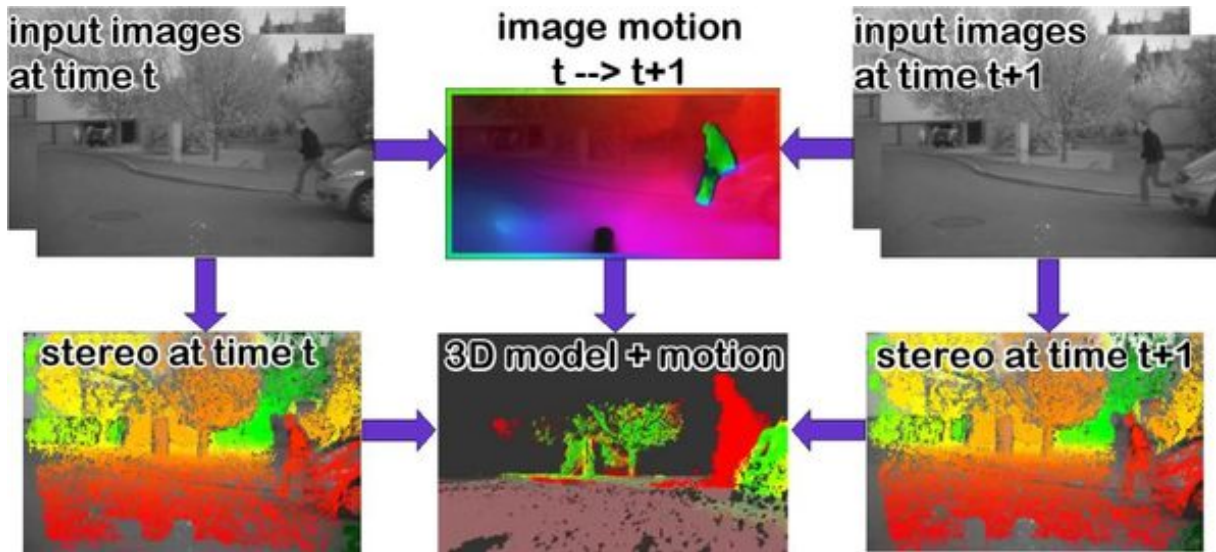


Figure 1.3: Reconstruction of the full scene motion from two stereo image pairs. For both image pairs, the stereo is estimated. The color map from red to green corresponds to distance (close to far). With known image motion (see border of image for direction), points can be registered and a three dimensional motion concluded. In the three dimensional scene model, green corresponds to stationary points and red to moving points.

sensors, such as time of flight or radio wave sensors, aim at directly measuring the distance to objects, they show weaknesses in direct shape representations and tracking over time. Such tracking however is crucial when reconstructing the three dimensional scene motion. When decisions need to be made, having the knowledge of such movement within a scene could make the difference between a good decision or a bad one. See Figure 1.2 for an example. In 2003, 16,932 car-to-car head-on collisions occurred in Germany with 1,504 fatalities. In situations with front-end accidents, both cars had a chance to *see the hazard coming*. With the knowledge of the scene reconstruction and the scene dynamics, reactive actions could have been triggered to, at least, reduce the risk of impact. Using the principles developed in this thesis, the visual sensor is upgraded into an accurate means of distance and motion perception which opens the door to novel safety systems.

1.3 Motion Analysis via Visual Kinesthesia

The scope of this thesis can be summarized with the key expression *machine visual kinesthesia*, the sensation or perception and cognition of motion. Visual kinesthesia is one important part of the human perception, which we all rely on. Visual kinesthesia has been thoroughly studied in medical and psychological research [118]. This thesis investigates the analysis of motion perception in the field of computer vision, particularly for driver assistance systems.

Visual kinesthesia represents both, the perception of the motion of one's own body and a spectator's perception of the motion of a performer [28]. In vehicle applications, these two steps refer to ego vehicle motion and the motion estimation of other traffic participants.

Typically both tasks need to be combined. The ego vehicle moves at a certain speed and other traffic participants move as well. An important feature to extract from a moving scene is dynamic movement within the scene. Safe autonomous maneuvers and reliable situation analysis are only possible if the movement within the scene is accurately known.

The stated problem for computer vision algorithms can be summarized as follows. Given the left and right images of a binocular camera at two successive time instances t and $t+1$, one has to solve the following two objectives:

**Objective one: Reconstruct the dynamic scene motion
between time t and $t+1$ for every pixel.**

Figure 1.3 shows an example of this task. The stereo disparity is estimated for both stereo input image pairs. With known disparities, a three dimensional model of the world can be computed via stereo triangulation. To reconstruct the three dimensional motion (*scene flow* or *scene motion*) for every image point, points must be registered one to another amongst the two time instances. One way to register the image points is to estimate the image motion from time t to time $t+1$ for every image pixel. The scene motion then computes as the difference between the two triangulated points at the two time instances. This yields the relative motion of the scene in the camera coordinate system. If the camera has moved between the two time instances, one has to additionally compensate for the camera motion (also known as camera ego motion, see Figure 1.4). The described procedure yields perfect results, if no errors occur in the disparity estimation, the image registration is correct, and the camera motion is exactly known. Obviously, these assumptions do not hold in practice and one has to deal with occluded areas and error-prone disparity data. This makes the reliable and accurate computation of scene flow quite challenging and complex.

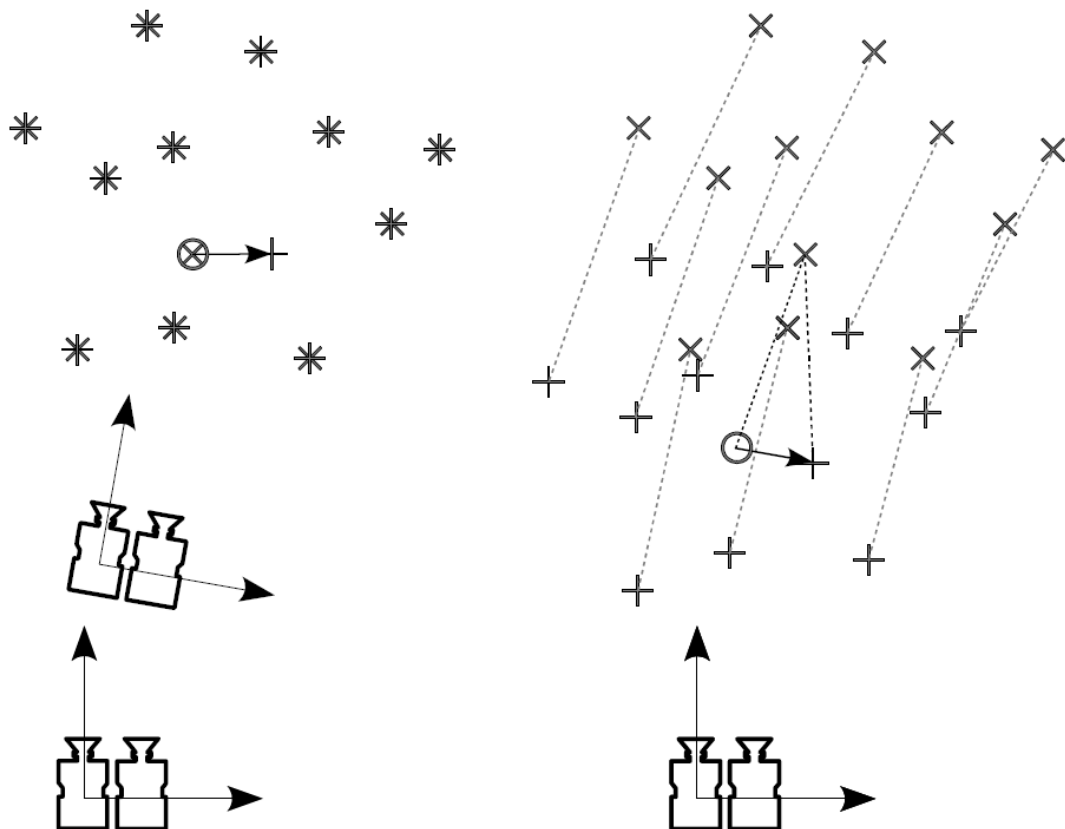


Figure 1.4: Motion of a cloud of points observed from the environment (*left*, fixed world coordinate system) and observed from the camera (*right*, camera coordinate system). Points corresponding to static objects move according to the camera translation and rotation. Independently moving points deviate from the expected position of a stationary point (denoted with a circle). The difference between expected position and observed position is the proper motion of the observed point. © Hernán Badino

Objective two: Analyze the scene motion and perform cognition tasks, such as object segmentation.

Figure 1.5 illustrates the second objective, the motion cognition and analyzing task. A perfect solution detects and segments every individual moving or stationary object in the images as well as the free space, the space free of obstacles and available to maneuver the vehicle.

All stationary objects have to be separated from the free space. Moving objects have to be segmented in the image and in the three dimensional scene reconstruction. Their movement direction and their velocity have to be estimated and a possible time to collision has to be computed for every object. Cognitive tasks which assign object categories to objects (classification algorithms) are not in the focus of this thesis. In this thesis, the problem of motion analysis is tackled for arbitrary types of objects. This general approach to cognitive motion analysis is a yet unsolved problem in computer vision.

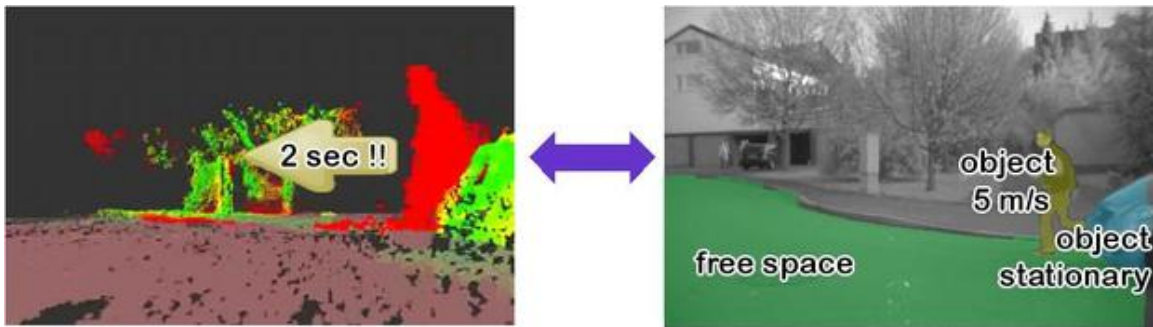


Figure 1.5: Segmentation and analyzing of free space and objects with their motion parameters in the three dimensional scene (*left*) and in the image (*right*). The left image shows the movement direction of the running person and the time left until collision, assuming a constant velocity of both, the ego vehicle and the running person.

1.4 Energy Minimization Problems

The problems addressed in the two objectives can be mathematically formalized as follows [117]: given a gray scale image $I : \Omega \rightarrow \mathbb{R}$ on the domain $\Omega \subset \mathbb{R}^2$, each point in the image is assigned a discrete segment $\{1, \dots, L\}$ (note, that connected regions should be favored) or a motion vector $\mathbf{u} \in \mathbb{R}^2$ (resp. $\mathbf{u} \in \mathbb{R}^3$ for 3D motion). The challenge is to assign suitable segments or motion vectors to a given input. In this thesis, the concept of energy minimization is used:

- (1) define a cost function, also called quality or energy function, which assigns an input-dependent cost to a conceivable solution and
- (2) find the solution of optimal quality (i. e. minimal cost or minimal energy).

If one discretizes the domain Ω into N pixels in the segmentation task given above, there are L^N candidate solutions to consider. For the case of $L=2$ and an image of 16×16 pixels, this yields roughly 10^{77} possible solutions. Provided, that the processing of a single candidate solution takes one machine cycle, a 10 GHz computer would take 10^{59} years to evaluate every single solution [117]. For such discrete topologies, efficient solution strategies to finding the global optimum are known based on dynamic programming or graph cuts. In Chapter 6, graph cuts are used for the segmentation of independently moving objects from image motion data.

If the solution space is continuous, as in the above motion estimation task, the solution space is infinite-dimensional and cannot be discretized as to consider all possible candidate solutions. In this case, one reverts to continuous optimization algorithms. The continuous optimization algorithms used in chapter 4 and 5 for motion estimation in image sequences are quadratic programming methods and variational methods, derived from partial differential equations. These are particularly suitable where accuracy (in a continuous domain) is of utmost importance.

1.5 Contributions of this Thesis

The content of this thesis is based on my work presented in the literature overview on page 115. Following the subdivision of visual kinesthesia into *perception* and *cognition*, the thesis is divided into a perception part and a segmentation part. The first part derives methods to *extract* the information about three-dimensional motion from the image sequence. The second part presents an application which uses the obtained motion information for the segmentation of independently moving objects. In the following, the main contributions of the specific chapters are outlined:

Chapter 2: Vision in Psychology and Natural Sciences In this chapter I take a glimpse into the fascinating world of biologically inspired perception and the history of motion analysis in computer vision. The term *visual kinesthesia* is set in relation to *structure from motion* and the *kinetic depth effect*, an effect thoroughly studied in the area of psychology to investigate the human visual perception system.

Chapter 3: Ego Motion Estimation This chapter serves as an introduction to the used coordinate systems. The notations are defined here and it rounds up this thesis by providing necessary algorithms for the ego motion estimation. The mathematics that are presented are similar to [69]. It is recommended to read this chapter in order to become familiar with notation conventions and the camera transformation matrices used throughout this thesis.

1.5.1 Visual Kinesthetic Perception (Theory)

Chapter 4: Optical Flow – The Image Motion Field Real-time methods to estimate the apparent motion of image pixels over time (optical flow) are commonly performed feature based because this allows one to concentrate the computational effort on a few image pixels. A second advantage of feature based approaches is the robustness because only distinct, good-to-track, image features are used. Both advantages seem to have vanished lately as dense energy minimization approaches became more and more powerful; they yield the most accurate flow fields in terms of sub pixel accuracy. Furthermore, recent computer hardware enables the computation of such flow fields in real time (e. g. below 40 ms).

In Chapter 4, I will investigate dense optical flow in more detail. Most dense optical flow algorithms compute flow fields by minimizing an energy made of a data and a regularity term. This thesis derives a novel concept for optical flow based on iterative refinement of an approximate flow field. The concept of regularization is decoupled from the data term. Within the optical flow framework, a fundamental matrix prior favoring rigid body motion (if supported by the image data) is introduced as an additional data term. This improves robustness and inherently estimates the camera motion. The general framework for optical flow is outlined for an arbitrary number of data terms and implementation details are given. The proposed framework relies on sequential convex optimization, is real-time capable and outperforms all previously published algorithms on the Middlebury optic flow benchmark. A discussion section concludes the optical flow chapter.

Chapter 5: Scene Flow – The World Motion Field I will address the key component of visual kinesthesia, the perception of the three dimensional structure and the associated motion, in this chapter. The two-dimensional optical flow for monocular image sequences is taken into a third dimension, estimating additionally the change of disparity. To this end, the optical flow framework from Chapter 4 is utilized to estimate a consistent scene flow field in two consecutive stereo image pairs.

A presented novelty is the decoupling of position and motion, enabling the computation of dense scene flow for every non-occluded image pixel in real-time. Via triangulation, the scene flow data is then transformed into the three-dimensional world flow for every projected point

within the scene. The presented motion estimation algorithm is being evaluated in artificial scenes where the ground truth motion is known, and in real scenes showing accurate perception of the scene dynamics. In chapter 5 also an insight into the accuracy of the estimated scene flow field between two consecutive stereo image pairs is given. An outlook on how to gain further improvements in the field of kinesthetic environment perception concludes this chapter.

1.5.2 Visual Kinesthetic Cognition (Application)

Chapter 6: Flow Cut – Segmentation of Independently Moving Objects In this chapter a method to segment independently moving objects using the graph cut method is presented. The motion of the ego vehicle greatly complicates the problem of motion detection because simple background subtraction of successive images yields no feasible result. Therefore, the information used for the segmentation process consists of the motion vectors derived in Chapters 4 and 5. The key idea to detect moving objects is to evaluate the hypothesis “the object is not stationary.” This is done by virtually reconstructing the three-dimensional translation vector for every flow vector within the image domain.

The detection of moving objects using a monocular camera has the inherent handicap that the two images are taken at different time instances and the observed (moving) object has moved between the two camera capture times. Hence, triangulation and distance estimation is impossible without knowing the exact object’s movement. On the other hand, object motion estimation is not possible without knowing the exact distance of objects. However, the movement of stationary image points can be restricted by the epipolar geometry reviewed in Chapter 3. Chapter 6 uses the error measures provided by the epipolar geometry for the segmentation of moving objects within the image.

The same idea of differentiating between stationary and moving image points is also used to segment dense scene flow fields derived in Chapter 5. Here, the full three dimensional translation vectors for all image pixels are analyzed to find and segment moving objects. The chapter concludes with an experimental comparison on monocular and binocular segmentation.

Chapter 7 Outlook – Research Ideas using Visual Kinesthesia This thesis is concluded by presenting ideas for further research in chapter 7. For many computer vision tasks, motion clues have been ignored for the most part. Examples are the detection and segmentation of stationary objects, the estimation of road surfaces, or the detection of the free space. However, a quick look into nature reveals that animals utilize (image) motion as one of the main cues for environment perception.

The segmentation process for stationary objects can greatly benefit from motion analysis. Although this does sound unconventional, I will show that detecting stationary objects in monocular image sequences has been successfully solved employing motion cues. Combining these ideas with the segmentation approach presented in Chapter 6 has the potential to improve the segmentation of independently moving and also stationary objects significantly.

Another research idea building on the scene flow derived in this thesis is to model free space dynamics. Humans have a very simple understanding of free space dynamics; algorithmically, these free space dynamics are a yet unaddressed research topic in computer vision. Concepts are sketched on how to estimate a dynamic free space boundary.

Examiner: “What can’t you identify on this microscope picture of a cell lying in front of you?”

Resigned student: “A tram car”

Jens Borum,

citation from “How to write consistently boring scientific literature” [115]

Vision in Psychology and Natural Sciences



Like beauty and color, motion is in the eye of the beholder.

Andrew B. Watson and Albert J. Ahumada, 1985

© under the creative commons license, Ranoush, www.flickr.com

Contents

2.1	Biologically Inspired Perception	10
2.2	History of Motion Analysis	11
2.3	Categorization of Perception	12

In this chapter the relations between 3D motion analysis using stereo sequences and monocular variants of structure and motion estimation is established. The term *visual kinesthesia* is set in relation to the *kinetic depth effect*, an effect thoroughly studied in the area of psychology to investigate the human visual perception system.

In the beginning, I will take a glimpse into the fascinating world of biologically inspired vision. More precisely, research work is presented examining the issue of monocular vs. binocular vision. It is certainly true that a binocular camera always yields superior results for the estimation of scene depth and scene motion. At the same time it is most interesting to discover the limits of monocular vision – if such limitations exist at all.

In the second part, a short insight into the history of motion analysis in computer vision is presented. Along the historical overview pioneering work in this research area is mentioned; state-of-the-art methods for the estimation of apparent motion and for succeeding applications will be presented in the follow-on chapters.

From this research work in the field of motion perception, a simple categorization into static vs. dynamic for both, the camera and the observed objects (the three dimensional scene), is derived. This categorization niftily illustrates the relation between structure from motion, the kinetic depth effect, and visual kinesthesia.

2.1 Biologically Inspired Perception

Since the discovery of the process of stereo-vision (Wheatstone 1838 [143]) its importance for environment perception has been a disputable issue. Some researchers have argued, that insects use stereo vision to perceive distance and structure in their visual world. Studying the praying mantis, researchers have even claimed that monocular vision essentially lacks depth perception [96]. Interestingly, the laws of physics prohibits accurate depth measurement beyond 10 mm distance for mantis larvae with an eye distance of 1 mm [111].

Others have shown, that grasshoppers first carefully peer at a target in order to prepare their jumps (see Figure 2.1). This allows the conclusion that “correct distance perception in the [monocular] grasshopper depends essentially on movement parallax” [60]. Investigating the jumping behaviour of monocular (one eye was covered with paint) and binocular grasshoppers indicated only insignificant differences in the target hit probability.

We see that different studies lead to significantly different conclusions. A reason for this might be that a mantis requires two fully intact eyes for jumping towards a target. Locusts and grasshoppers, unlike mantis, also precisely jump towards a target when one eye is fully occluded (see [111] and references therein).

However, a survey of biologically inspired vision clearly goes beyond the scope of this thesis. Even the compound eyes of insects, consisting of multiple ommatidia, are not yet fully understood and a clear conclusion about the importance of binocular vision cannot be drawn. Most researchers today agree though, that motion information is most valuable for insects to perform structure and depth perception. In [71], for instance, the authors show that cells in the eyes of a fly fire at rates up to 300 Hz when objects are close. They claim, that such sensitive small-field motion neurons enable the segregation of object and background. In fact, several experiments have shown that flies make use of the apparent movement of texture in the visual field to perform navigation. Insect flight strategies like *Forward Focus*, *Landing*, *Hovering*, *Saccading*, or *Collision Response* can be explained based on motion analysis. A survey on “Biomimetic Visual Sensing and Flight Control” is presented in [38].

Considering human vision, a nice historical survey on “Understanding Human Motion” is found in [80]. The importance of motion for the human visual perception is beyond dispute since J. J. Gibsons work “The Perception of the Visual World” [66] and David Marrs influential book “Vision: A Computational Investigation into the Human Representation and Processing of Visual Information” [97]. Although the role of motion is unquestioned, some researchers argued that “like beauty and color, motion is in the eye of the beholder” [140]. While this is certainly true to some extent, such notion of motion analysis implies that no *true motion perception* exists. Thus, it is better taken care of in psychophysics than in computer vision.

In computer vision, motion analysis is commonly referred to as the analysis of apparent motion via optical flow. From this optical flow point of view, motion analysis is well defined as to measure the motion in the two-dimensional projection of the three-dimensional world. In the following section, I will give a short historical overview of motion analysis in computer vision in order to put the work of this thesis into a greater context.

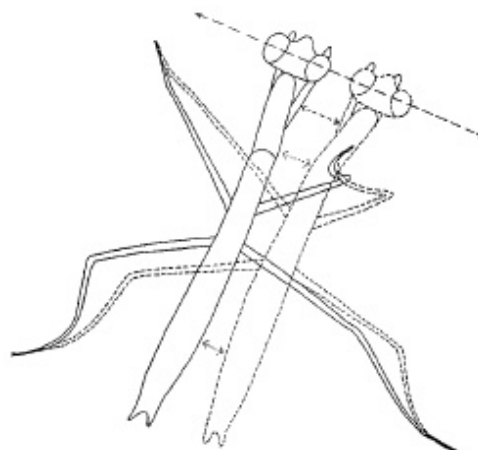


Figure 2.1: Schematic drawing of the peering movement. Video analysis shows that peering begins as an accelerated movement. The head turns forward from a slightly sideways position, continues with a uniform translatory movement and ends with a delayed movement with the head turned slightly sideways [111]

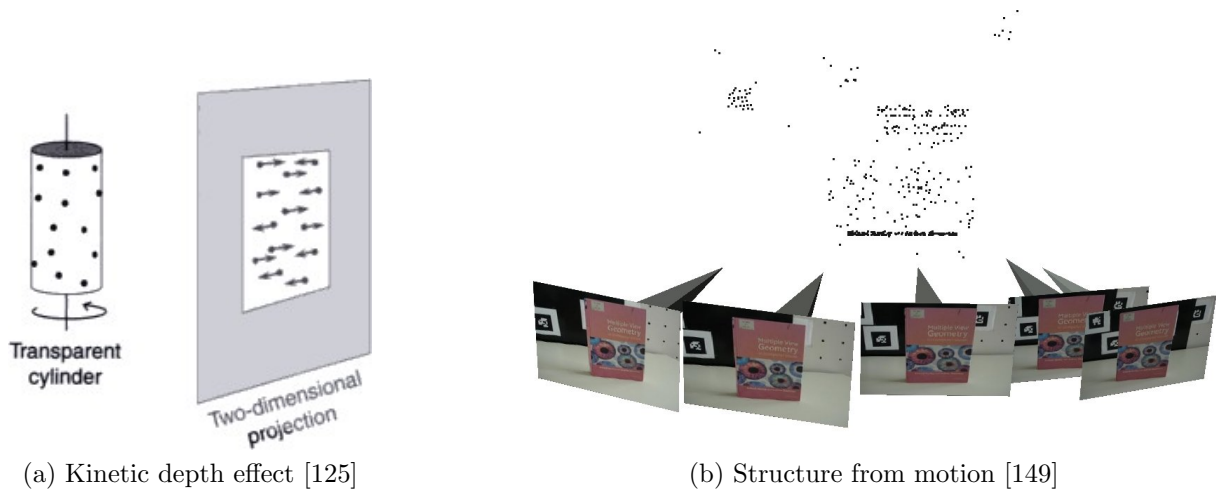


Figure 2.2: The kinetic depth effect allows a three-dimensional reconstruction of the cylinder from the projected motion of the dots. Structure from motion allows a three-dimensional reconstruction of the scene from corresponding pixels in multiple images.

2.2 History of Motion Analysis

The history of motion pictures can be traced back to the Belgian researcher Plateau and the Austrian scientist von Stampfer, who simultaneously invented the *phenakistoscope* in 1832 [27]. The word *phenakistoscope* originates from Greek and means *to cheat*, as it deceives the eye by making pictures look like an animation. The pictures are located on a spinning disc and the basic principle is similar to a flip-book (which first appeared in 1868 [25]). This was the first time that 3D motion was *captured* on a two-dimensional manifold. In this thesis we are interested in the counter direction, the perception of motion from animated pictures.

First studies of apparent motion are found in 1875 by Exner and in 1912 by Wertheimer [29]. In the first half of the last century, motion and perception were investigated in psychology, physiology, and cognitive sciences. Physiology analyzes the organization of the brain and body. “For their discoveries concerning information processing in the visual system” Hubel and Wiesel won the Nobel price in 1981 [22]. In this thesis, I focus on the perception of motion and depth from images rather than perceiving the images themselves. This has been addressed in psychology (e. g. learning and information processing). In the mid-1950s it was proclaimed that

“building perceiving machines would take about a decade” [84].

An ambitious goal which was never achieved until even today.

It was back in these days when Wallach and O’Connel described their *kinetic depth effect*, the phenomenon whereby the three-dimensional structural form of an object, viewed in projection, can be perceived when the object is only rotating (see Figure 2.2a). Since then many researchers have examined the effect of stimulus parameters, frame timing, occlusion, detection of non-rigidity, and veridicality (see [122] and references therein).

It was not until 1981, when Longuet-Higgins showed for the first time in his work “The Interpretation of a Moving Image” [89] that “in the case of an observer moving around a static textured environment, the 3D structure of the environment and the observer’s 3D motion can be recovered from the first and second order derivatives of the optical flow [apparent motion]. This landmark paper influenced practically all further work on structure from motion” [61] (see Figure 2.2b for an example).

For the kinetic depth effect, the camera is stationary and depth (structure) is inferred from the object dynamics; the object needs to be rigid. In the structure from motion setting, the scene is stationary and structure is inferred utilizing the camera's movement. None of these approaches is able to infer structure from a dynamic scene using a moving camera. Only recently, in 2004, Vidal et. al demonstrated an approach for two-view multi-body structure from motion [138]. The structure of multiple moving objects is reconstructed while the camera in the original setting is stationary. The stationarity of the camera motion does not generally limit the approach. There remains however an essential limitation: the rigidity assumption must be valid for each reconstructed object in order to infer the 3D structure (not employing prior knowledge).

This assumption can be dropped once a second camera is available, using binocular vision and stereo reconstruction. Then motion and structure estimation in dynamic scenes with a moving camera becomes possible. Pioneering work in this area is the joint disparity and motion field estimation presented by Patras et. al in 1996 [108]. It is the first approach to estimating consistent scene flow (3D structure and 3D motion) from two stereo image pairs. Scene flow solves the structure from motion problem directly, as stereo already provides the scene structure. It also takes away the limits of the kinetic depth effect (specifically the rigid object assumption) and enables the full motion perception of the environment. This includes the perception of the motion of the camera, the perception of the motion of multiple (non-rigid) objects, and the perception of the structure of the static scene.

2.3 Categorization of Perception

To summarize above areas of research, the perception of the stationary or dynamic world with either a stationary or a moving camera is categorized in Figure 2.3. It becomes clear, that the full perception of the static scene and dynamic motion, while the camera itself is moving, is only possible using a stereo camera system. This is the approach I pursue in this thesis. I call this full motion perception task *visual kinesthesia*, according to the definition of *kinesthesia* in [28].

The benefit of stereo vision over monocular vision for three-dimensional reconstruction is hard to specify precisely. Certainly, if moving objects would be removed from the images, this allows one to reconstruct the structure of the static scene from a monocular sequence. The extracted moving objects can further be investigated using a rigid body assumption. Thus, monocular vision might be enough. On the other hand, the limits of monocular motion perception have to be characterized. For obstacle detection, the knowledge of the three-dimensional structure from stereo vision is directly available at one time instance and, assuming a stationary world, might be sufficient. If the world is dynamic, combining stereo and motion into a consistent scene flow perception is the most appealing and most promising approach.

		Scene	
		Stationary	Dynamic
Camera	Stationary	Stereo Vision	Optical Flow Kinetic Depth Effect
	Moving	Optical Flow Structure From Motion	Stereo & Optical Flow Visual Kinesthesia 3D Structure & Motion

Figure 2.3: Perception in Computer Vision. Monocular methods using optical flow are limited to either a stationary camera or a stationary world. If a stereo camera is available, perception becomes possible in the two other cases: static world & stationary camera and dynamic world & moving camera.

Ego Motion Estimation



Sometimes the world moves faster than it appears.

Special thanks to Jan-Martin Will for taking this great picture.

Contents

3.1	Image Formation and Coordinate Systems	14
3.1.1	Image Formation	14
3.1.2	Coordinate Systems	15
3.2	Fundamental Matrix Geometry	17
3.3	Robust Ego-Motion Estimation	19
3.3.1	The Linear Criterion	19
3.3.2	The Non-Linear Criteria	20
3.3.3	Re-weighted Least Squares and Parametrization	20

Moving objects often are a hazard to drivers. These objects might be other vehicles, bikes, pedestrians, but also a closing barrier or an opening driver's door of a parked vehicle. In computer vision the challenge lies in differentiating between motion caused by camera movement and scene motion. Hence, estimating the camera motion is a key component of motion perception and visual kinesthesia, also referred to as *ego motion estimation*.

If the camera is rigidly connected to the car, the movement of the car and the movement of the camera coincide (but may be given in a different coordinate system). To this end, the camera movement can be derived from motion sensors, such as the speed sensor and sensors measuring the rotational forces of the vehicle. Such a mechanical perception of the ego motion is referred to as ego motion from inertial sensors.

For many applications motion estimation from common inertial sensors is not precise enough. While the synchronization of the sensors with the camera imager is a problem, the sensors itself also suffer from temperature dependant drifts and discretization artifacts. It is, hence, desirable to improve the ego motion estimation using the captured camera image itself.

This chapter deals with direct ego motion estimation from camera images. In Section 3.1, the image formation process is described. Section 3.2 derives the basics of the epipolar geometry, describing the mutual orientation of two cameras. This is then used in Section 3.3 to estimate the motion of the camera.

3.1 Image Formation and Coordinate Systems

The physical process of image formation is a well-researched area and a detailed discussion is beyond the focus of this thesis. Only the basic principles for the cameras used in the experiments are outlined in this section. The topic of this section is also covered in [69, 77]. Firstly, the image formation process and rectification process is outlined. In a second subsection, the involved two- and three-dimensional coordinate systems are described.

3.1.1 Image Formation

Images are the projection of the three-dimensional space onto a two-dimensional manifold, ideally a plane. It is evident, that cameras do not constitute an exact planar projection due to image plane distortions and lens distortions. The projection from the two-dimensional manifold onto a Euclidean plane, removing these distortion artifacts, is called rectification. Several algorithms to rectify images have been published. A well-known and widely used approach was published by Bouguet [18]. A good analogy in everyday life for the rectification process is the work of an optician, who produces glasses in order to remedy defects of human vision.

After the camera images are rectified, one might think of a simple pinhole camera model as illustrated in Figure 3.1. The projection plane of the camera is the Euclidean or retinal plane, the (perfect) lens is represented by a simple pinhole. It is well-known that a pinhole, instead of a lens, does not induce distortion artifacts; however, the reason to use rather large lenses is the shorter exposure time for larger apertures. Typical exposure times for pinhole cameras range from 5 seconds to hours or days. A larger *pinhole* represented by a lens reduces this exposure time to a few milliseconds. In automobile environments with fast scene motion this short exposure time is crucial, demanding high quality camera lenses to minimize distortion artifacts.

Camera Sensor In order to process images, the light rays originating from a three-dimensional scene point have to be converted into electrical signals. Technically, this is done by either CCD (charged coupled device) or CMOS (complementary metal oxide semiconductor) sensors. An overview of these two technologies is given in [86]. The single sensor elements are arranged on a grid and called picture elements (abbreviated as pixel or px). Common image sizes for vehicle applications are 640×480 px and 1024×512 px. Both, CCD and CMOS sensors, may have different color depth resolution, usually ranging from 8 bit to 16 bit. In this thesis, no conclusion about preferences will be drawn and no attention is given to the type of sensor used, unless otherwise stated. Figure 3.2 shows a typical camera used in industrial applications. The question remains how different cameras and their respective image planes are located in a (fixed) world coordinate system. This becomes even more important when several synchronized cameras view the same object and multiple view object reconstruction (for instance using stereo vision) needs to be performed.

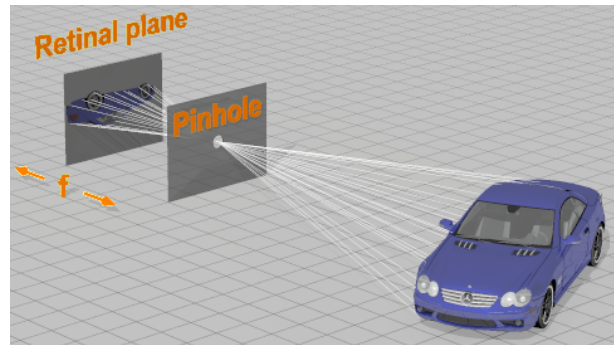


Figure courtesy of Clemens Rabe.

Figure 3.1: Principle of a pinhole camera. Light rays from an object pass through a small hole and form an image.



Figure 3.2: The image shows a typical camera used in industry.

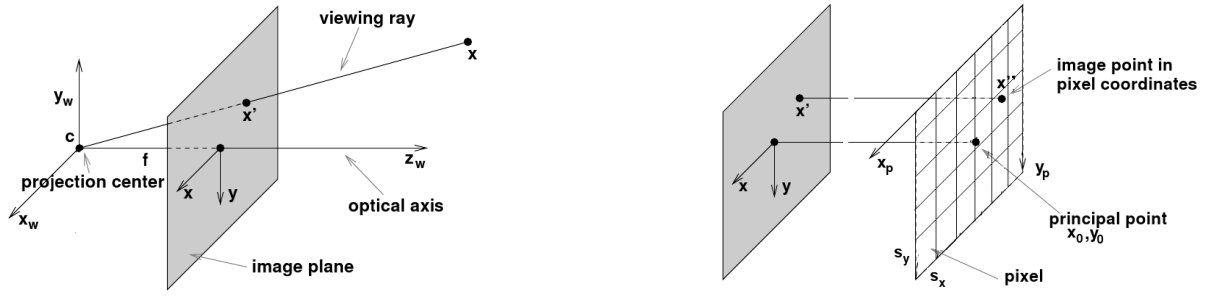


Figure 3.3: Coordinate systems for the pinhole camera model. The 3D world point \mathbf{X} is projected onto the image plane yielding \mathbf{x}' (left). The camera center is placed at the origin of the camera coordinate system. After projection the point is transformed into pixel coordinates (right).

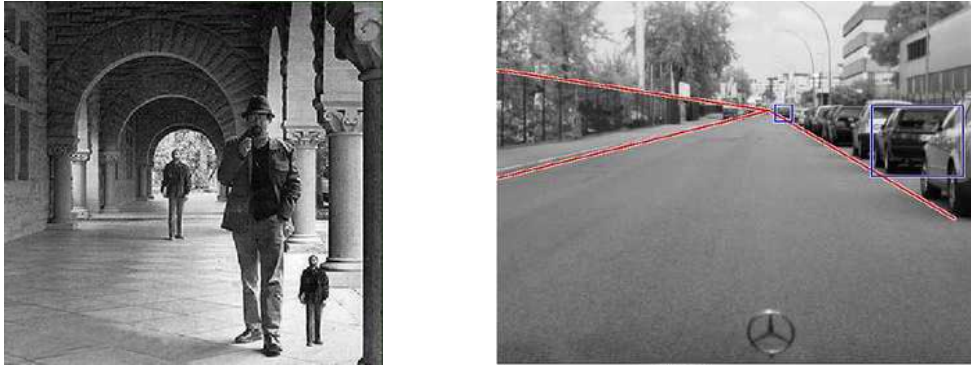


Figure 3.4: The man without the hat appears to be two different sizes, even though they are identical when measured in pixels. In 3D, the man without the hat is about 6m behind the man with the hat. This shows how much size is expected to change due to perspective projection [20]. The scale change can also be seen in a typical traffic scene [77]. Cars at different distances are different in size (blue rectangles) and become infinitely small at the vanishing point (where the red lines converge).

3.1.2 Coordinate Systems

The goal of this subsection is to derive the basic geometry of the projection of three dimensional points (3D points) onto the two dimensional image plane (yielding 2D points). Throughout this thesis a left-handed image coordinate system is used with the X -axis as the horizontal direction and the Y -axis as the vertical direction, pointing upwards. This means that the optical axis or gaze direction is the positive Z -axis, as illustrated in Figure 3.3. Unless otherwise stated, capital variables (e. g. X or P_4) denote 3D coordinates while lower case variables (e. g. \tilde{v} or y_k) denote image or 2D coordinates. A vector is denoted by bold letters in dark brown color as in \mathbf{X} and matrices are represented using dark blue, bold letters (e. g. \mathbf{M} or \mathbf{R}^{-1}).

Camera Projection Consider the projection of a point $\mathbf{X} = (X_W, Y_W, Z_W)$ in 3D space with the camera at the origin, onto the image plane $x \times y$. Using the theorem of similar triangles (see Figure 3.3) yields

$$\mathbf{x}' = \frac{f}{Z_W} \begin{pmatrix} X_W \\ -Y_W \end{pmatrix}. \quad (3.1)$$

Here, f is the distance of the image plane to the camera center, also called focal length. However, f does not refer to the focal length of the camera lens used and is a purely arithmetical value. The projection process is called perspective projection. Under a perspective projection, distant objects appear smaller than near objects as shown in Figure 3.4. In homogeneous coordinates

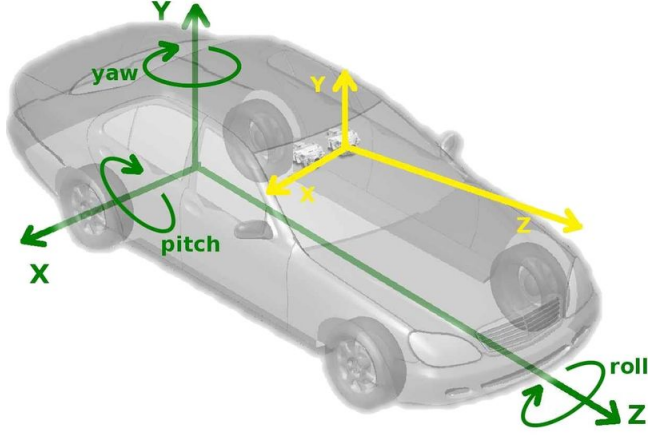


Figure 3.5: Camera setup in the test vehicle. The camera coordinate system (yellow) and the world coordinate system (green) are shown in the figure.

this projection rewrites to

$$\mathbf{x}' = \begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = \mathbf{cP}_W \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & -f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix}. \quad (3.2)$$

Once world points are projected onto the image plane, a transformation from metric coordinates into pixel coordinates has to be performed. With the pixel width s_x and pixel height s_y , respectively, and the intersection point (x_0, y_0) of the optical axis with the image coordinate system, the perspective transformation into homogeneous image coordinates becomes

$$\mathbf{x}'' = \begin{pmatrix} x'' \\ y'' \\ w'' \end{pmatrix} = \underbrace{\begin{bmatrix} \frac{f}{s_x} & 0 & x_0 & 0 \\ 0 & -\frac{f}{s_y} & y_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{K}} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix} = [\mathbf{K}|\mathbf{0}] \mathbf{X}. \quad (3.3)$$

The transformation matrix \mathbf{K} contains all (intrinsic) camera parameters and is called calibration matrix. Thus, if \mathbf{K} for a specific camera is known, the camera is called calibrated. Camera calibration, similar to determining the parameters for image rectification, is done prior to the system run-time; for instance using the toolbox described in [18]. Although small changes of the calibration parameters due to temperature drifts and vibration may occur, these effects are negligible in most settings and will not be considered in this thesis.

Rotation and Translation of the Camera Center If the (external) world coordinate system does not coincide with the camera coordinate system, a rotation and translation needs to be performed before projecting world points. In this thesis, the world coordinate system coincides with the vehicle coordinate system. Its origin lies on the ground plane and the z-axis points towards the vehicle's heading direction. The origin of the world coordinate system is located at the center of the rear axle of the vehicle.

The translation vector \mathbf{T} is the camera position in the world coordinate system. The angles of the camera coordinate system and world coordinate system are referred to as pitch, roll, and yaw angle, respectively. Figure 3.5 illustrates this arrangement. With the rotation matrix \mathbf{R} between camera and world coordinate system and the translation vector \mathbf{T} , the total world to image transformation in homogeneous coordinates becomes:

$$\mathbf{x}'' = \begin{pmatrix} x'' \\ y'' \\ w'' \end{pmatrix} = [\mathbf{K} \quad \mathbf{0}] \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{T} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix} = \mathbf{KR}[\mathbf{Id} | -\mathbf{T}] \begin{pmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{pmatrix}. \quad (3.4)$$

The projection matrix $\mathbf{P} = \mathbf{KR}[\mathbf{Id} | -\mathbf{T}]$ describes both, the relative orientation of the camera in the world coordinate system and the perspective transformation of world points onto the image plane. Because the perspective transformation reduces the three-dimensional space onto two dimensions, the imaging process clearly is not reversible. Only the viewing ray can be reconstructed (see Figure 3.3), which implies that no direct depth measurement is possible. However, if the same world point is observed in more than a single camera, a three dimensional reconstruction becomes possible by intersecting multiple viewing rays.

3.2 Fundamental Matrix Geometry

The geometry describing the geometric relation of two different camera systems is referred to as fundamental geometry. A system of two cameras which are fixed in relation to each other is called a stereo camera system. An essential distinction is made between conventional stereo systems (two different cameras) and motion stereo constellations (two different images of a moving camera). Both constellations can be transformed into a *standard stereo case* by virtually rotating the images, using a rectification step, such that the optical axes of the individual cameras are parallel. After rectification, the camera coordinate systems differ only by a translation component (see Figure 3.6). The geometric properties of these stereo camera systems are discussed in this section, replicating [24].

When two cameras view a 3D scene from different view points, there are a number of geometric relations between the 2D projections of a 3D point. These geometric relations lead to constraints between the two projected image points. Figure 3.7 depicts two cameras looking at a 3D point \mathbf{X} . \mathbf{O}_L and \mathbf{O}_R represent the origin of the two camera coordinate systems (also called focal points). The 2D points \mathbf{x}_L and \mathbf{x}_R are the projections of the point \mathbf{X} onto the two image planes.

Since the origins of the two cameras are distinct, each focal point projects onto a distinct point into the other camera's image plane, denoted by \mathbf{e}_L and \mathbf{e}_R and called epipoles. The two epipoles \mathbf{e}_L and \mathbf{e}_R and the camera origins \mathbf{O}_L and \mathbf{O}_R are located on a single connecting line.

The line between the focal point of the left camera, \mathbf{O}_L , and the world point \mathbf{X} is seen as a point in the left camera. The right camera, however, sees this line as a line in its image plane (and vice versa a point in the right camera corresponds to a line in the left image). This line, $\mathbf{l}_R = \mathbf{e}_R - \mathbf{x}_R$, is called the epipolar line. The plane which is spun by \mathbf{O}_L , \mathbf{O}_R , and \mathbf{X} is called the epipolar plane. The epipolar lines in turn are the intersection of the epipolar plane and the image planes.

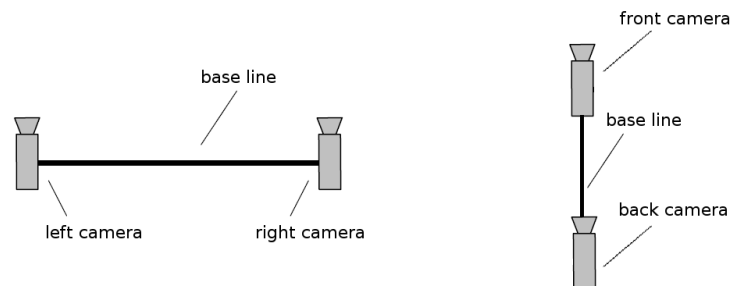


Figure 3.6: Conventional stereo (left) and motion stereo (right).

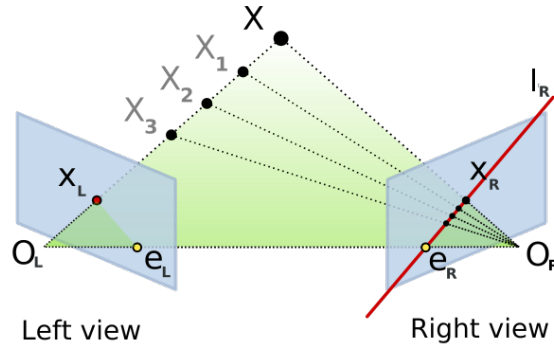


Figure 3.7: The epipolar geometry [24].

If the relative camera orientation, the camera calibration matrices, and the points \mathbf{x}_L and \mathbf{x}_R are known, their projection lines are also known. If the two image points correspond to the same 3D point \mathbf{X} , the projection lines must intersect precisely at \mathbf{X} . This means that \mathbf{X} can be calculated from the coordinates of the two image points, a process called triangulation.

If the projection point \mathbf{x}_L is known, then the epipolar line \mathbf{l}_R can be computed and the point \mathbf{X} projects into the right image, on a point \mathbf{x}_R which must lie on this particular epipolar line. This means that for each point observed in one image, the same point must be observed in the other image on a known epipolar line. This provides an epipolar constraint which corresponding image points must satisfy. Thus it is possible to test whether two points may correspond to the same 3D point (necessary but not sufficient test). Epipolar constraints can also be described by the essential matrix or the fundamental matrix between the two cameras. More specifically, if \mathbf{x}_L and \mathbf{x}_R are homogeneous image coordinates corresponding to the same world point \mathbf{X} , then the following *epipolar constraint* equation holds:

$$\mathbf{x}_R^\top \mathbf{F} \mathbf{x}_L = 0. \quad (3.5)$$

Here, the fundamental matrix \mathbf{F} is a 3×3 matrix. The line $\mathbf{l}_R = \mathbf{F} \mathbf{x}_L$ describes the epipolar line in the right image. That is, the line \mathbf{l}_R in the right image is the projected viewing ray of every world point projected into the pixel \mathbf{x}_L in the left image.

Being of rank two and determined only up to scale, the fundamental matrix can be estimated given at least seven point correspondences. The term *fundamental matrix* was formulated by Luong in his PhD thesis [93]. It is sometimes also referred to as the *bifocal tensor*. With the camera calibration matrices for the left and right camera, \mathbf{K}_R and \mathbf{K}_L , the rotation matrix \mathbf{R} , and translation vector \mathbf{T} between the two camera coordinate systems, the fundamental matrix can be decomposed into

$$\mathbf{F} = \mathbf{K}_R^\top \mathbf{R} [\mathbf{T}]_\times \mathbf{K}_L \quad (3.6)$$

where $[\mathbf{T}]_\times$ is the matrix representation of the cross product with \mathbf{T} . The middle part, $\mathbf{R} [\mathbf{T}]_\times$, is also called the essential matrix \mathbf{E} . The essential matrix has only five degrees of freedom: both the rotation matrix \mathbf{R} and the translation vector \mathbf{T} have three degrees of freedom, but there is an overall scale ambiguity – like the fundamental matrix, the essential matrix is a homogeneous quantity. The reduced number of degrees of freedom translates into extra constraints that are satisfied by an essential matrix when compared to the fundamental matrix. A more detailed investigation can be found in [107, 153].

For a fixed stereo camera system the relative orientation can be determined off-line. For a monocular camera however the relative orientation has to be determined on-line, either using inertial sensors or directly estimating the fundamental matrix from image correspondences. This process is described in the next section. If the fundamental matrix is known, it can then be decomposed into a rotation and translation matrix, keeping in mind that the translation parameters can only be determined up to scale due to the scale ambiguity of homogeneous matrices.

3.3 Robust Ego-Motion Estimation

In the field of driver assistance systems and computer vision, the term *ego motion* refers to the motion of the camera. If the camera is rigidly mounted to a vehicle, the motion of the vehicle and the motion of the camera are identical, apart from camera vibrations (but usually given in a different coordinate system). Ego motion estimation, or simply *motion estimation*, is a crucial part of environment perception. The correct understanding of scene dynamics depends on the correct estimation of the ego motion. This can be seen when looking at the introduction example of this chapter. If one has no information whether the image is taken from a moving or stationary train, it is not possible to estimate the travelling speed of the passing train. Three situations are possible: either the passing train seen through the window is moving and the foreground train is stationary, or both trains are moving, or only the train in the background is stationary. If the motion of the train from which the image is taken is exactly known, the ambiguity in this situation can be solved.

This small example reveals two very important observations. Firstly, only relative motion can be estimated from images without the knowledge of the ego motion. Secondly, the (absolute) ego motion can only be estimated based on stationary points. This second observation directly follows from the first: since stationary points do not move the relative motion to these stationary points is the absolute motion.

But how can the motion of a camera system be estimated from the images itself? The answer is given by the fundamental matrix geometry and Equations (3.5) and (3.6). Points in the two images are related according to the epipolar constraint Equation (3.5). The only unknown in the epipolar constraint equation is the fundamental matrix, which can be decomposed into the calibration matrix, the rotation, and a translation according to Equation (3.6). A common approach to ego motion estimation is to estimate the ego motion using as many image correspondences as possible, assuming that most of the image is stationary. The following two subsections present linear and non-linear criteria for the computation of the fundamental matrix, according to [92]. The challenge remains how to robustly estimate the motion parameters if some parts of the image are not stationary and how to handle outliers. A common approach is discussed in Section 3.3.3.

3.3.1 The Linear Criterion

Equation 3.5 is linear in the 9 unknown coefficients of the fundamental matrix \mathbf{F} . It is also homogeneous, hence, in general, the fundamental matrix can be computed from 8 point matches. If additionally the rank 2 constraint is employed (determinant has to equal zero), even 7 points are sufficient [69]. In practice, the epipolar constraint equations are not exactly fulfilled due to noise and inaccurate point correspondences. A direct solution has been proven to be very sensitive to noise. If more than 8 matches are given, a least-squares minimization is used to solve

$$\min_{\mathbf{F}} \sum_i \left(\mathbf{l}_i^\top \mathbf{F} \mathbf{r}_i \right)^2 \quad (3.7)$$

with $(\mathbf{l}_i, \mathbf{r}_i)$ being the correspondences from the left to the right image. However, in [92] the authors show, that “if the epipole is in the image, the epipolar geometry described by the fundamental matrix obtained from the linear criterion [Equation (3.7)] will be inaccurate.” This problem is particularly fatal because such constellations are common for cameras mounted in vehicles and monitoring the driving corridor. They also show, that the linear criterion “shifts epipoles towards the image center” due to the lack of normalization of the epipolar constraint equations. It is, thus, straight-forward to replace the linear criterion with non-linear criteria, which can be interpreted as distance measures.

3.3.2 The Non-Linear Criteria

The two most common non-linear criteria for the estimation of the fundamental matrix are

$$\min_{\mathbf{F}} \sum_i \left(\frac{1}{(\mathbf{F}\mathbf{r}_i)_1^2 + (\mathbf{F}\mathbf{r}_i)_2^2} + \frac{1}{(\mathbf{F}^\top \mathbf{l}_i)_1^2 + (\mathbf{F}^\top \mathbf{l}_i)_2^2} \right) (\mathbf{l}_i^\top \mathbf{F} \mathbf{r}_i)^2 \quad (3.8)$$

and

$$\min_{\mathbf{F}} \sum_i \left(\frac{1}{(\mathbf{F}\mathbf{r}_i)_1^2 + (\mathbf{F}\mathbf{r}_i)_2^2 + (\mathbf{F}^\top \mathbf{l}_i)_1^2 + (\mathbf{F}^\top \mathbf{l}_i)_2^2} \right) (\mathbf{l}_i^\top \mathbf{F} \mathbf{r}_i)^2. \quad (3.9)$$

In the formulas a subindex denotes the index of the vector component, in particular $(x, y, z)_2^\top = y$. Both formulas can be interpreted as distances from the epipolar lines or as weighting with variances. A detailed stability and convergence analysis is found in [92]. The authors show, that the non-linear computation techniques provide significant improvement in the accuracy of the fundamental matrix determination, even if noise is not important.

3.3.3 Re-weighted Least Squares and Parametrization

Unfortunately, the fundamental matrix geometry only holds when the two images are taken at the same time instance, or when the world is stationary (in particular assuming perfect correspondences). In a moving world, the world point \mathbf{X} in Figure 3.7 may have moved to any arbitrary position in between the two camera images taken. Hence, only stationary points contribute to the correct solution. A common approach is to assign low weights or even to reject correspondences with a large distance to the epipolar line and iteratively minimize the least squares sum until convergence (see [77]).

Another way to increase robustness is to limit the number of free parameters in the fundamental matrix. The most natural representation of the fundamental matrix takes into account that \mathbf{F} is only defined up to a scale factor (due to the homogeneous entities). Hence, fixing one of the coefficients to 1 would be desirable. Using the linear criterion allows to use a simple normalization, namely $\|\mathbf{F}\|$ [92].

The use of the essential matrix (the restricted version of the fundamental matrix) yields a parametrization, which has only five degrees of freedom (rotation and translation direction). Assume for now, that $N > 5$ point correspondences $(\mathbf{l}_i, \mathbf{r}_i)$ are given and the camera calibration matrix \mathbf{K} of a moving camera is known. Then, according to equations (3.5) and (3.6), the rotation and translation is the solution of

$$0 = \mathbf{r}_i^\top \mathbf{K}^\top \mathbf{R} [\mathbf{T}]_\times \mathbf{K} \mathbf{l}_i \quad \forall i.$$

Taking the derivative w.r.t. the individual rotation parameters and translation parameters using the linear or non-linear criteria yields the final solution. A common approach is to use lagged feedback by linearizing the objective function, using the first order Taylor approximation, and performing gradient descent until convergence. As starting point, the latest known motion or, if available, the ego motion from inertial sensors is used. Linearizations of the rotation matrix and translation vector can be found in [77]. Note that only five parameters can be estimated due to the scale ambiguity of homogeneous matrices. The approach used in [77] fixes the length of the translation vector to reduce the number of free parameters for the translation from three to two.

The remaining question is how point correspondences are found. This question will be answered in the next chapter.

Part I

Visual Kinesthetic Perception (Theory)

Optical Flow



Space is a still of time, while time is space in motion.

Christopher R. Hallpike

Contents

4.1	Approaches in Literature	24
4.1.1	Census Based Optical Flow	25
4.1.2	The Optical Flow Constraint	25
4.1.3	Total Variation Optical Flow	28
4.1.4	Other Optical Flow Approaches	30
4.2	A General Flow Refinement Framework	30
4.2.1	Data Term Optimization	30
4.2.2	Smoothness Term Evaluation	36
4.2.3	Implementation Details	40
4.3	Increasing Robustness to Illumination Changes	42
4.4	Experimental Results	46
4.4.1	Quantitative Evaluation	46
4.4.2	Results for Traffic Scenes	53
4.5	Ideas for Future Research	56

Whenever a camera records a scene for a given time, the resulting image stream is called an image sequence. The image sequence can be considered as a function $I(x, y, t)$ of the gray value at image pixel position $\mathbf{x} = (x, y)^\top$ and time t . If the camera, or an object, moves within the scene, this motion results in a time-dependent displacement of the gray values in the image sequence. The resulting two dimensional apparent motion field in the image domain is called the optical flow field. Figures 4.1 and 4.2 show the optical flow field for sample scenes.

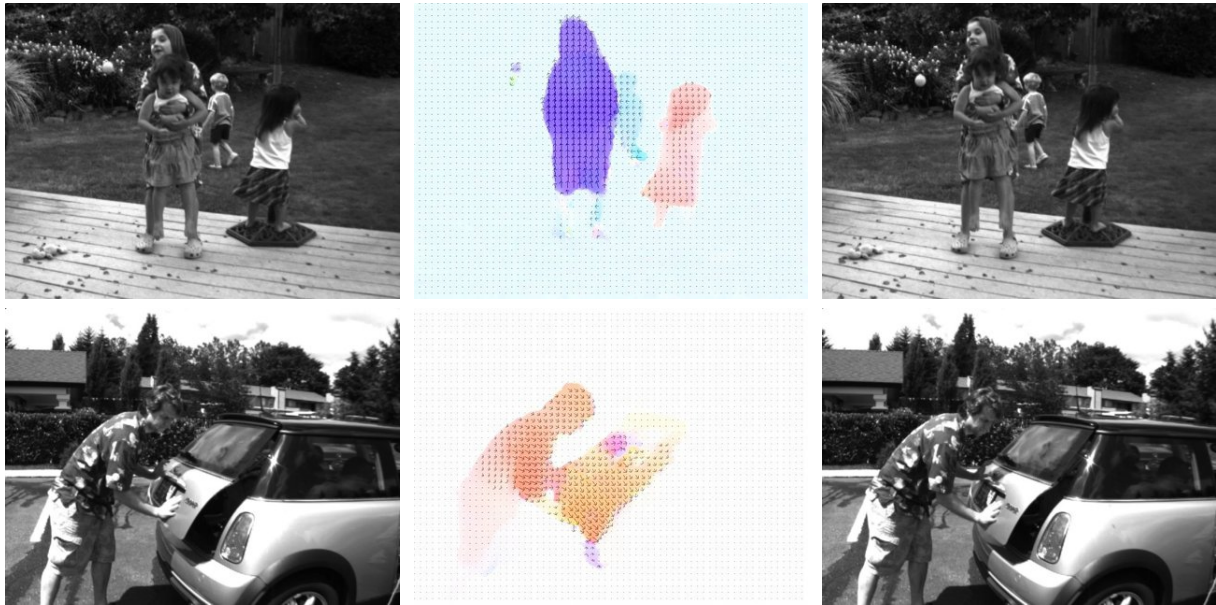


Figure 4.1: Optical flow for the *backyard* and *mini cooper* scene of the Middlebury optical flow benchmark. Optical flow captures the dynamics of a scene by estimating the motion of every pixel between two frames of an image sequence. The displacement of every pixel is shown as displacement vectors on top of the commonly used flow color scheme (see Figure 4.2).

The optical flow field describes the dynamics of a scene and is composed of the camera motion and the motion of objects itself. Unfortunately, not every object motion yields a change in gray values and not every change in gray values is generated by body motion. Illumination changes, shadows, and reflections (e.g. due to a moving light source) may yield gray value changes while the depicted object remains stationary. Another source of gray value changes is the inherent noise of the camera imager. Especially in bad illumination conditions (e.g. at night time) the number of photons, which are collected by a pixel, may vary over time.

If an untextured object moves within the image, image motion in terms of gray value change can only be seen at the object boundaries, where the gray value of the background differs from the gray value of the object itself. This again is only partially true due to the *aperture problem*. The aperture problem arises as a consequence of the ambiguity of one-dimensional motion viewed through an aperture. Consider Figure 4.3 for apparent motion through an aperture. The striped background is masked by an occluding bull's eye. If the stripes are moved upwards, the pattern of lines in the aperture shifts. If the stripes are moved leftwards, the pattern within the aperture shifts in the same way. Thus, our perceptual system is faced with a motion ambiguity which can only be solved if the motion of the boundary of the pattern is known [21].



Figure 4.2: Color coding of the flow vectors: Direction is coded by hue, length is coded by saturation. The example on the *right* shows the expanding flow field of a forward moving camera. Flow vectors above 20 px are saturated and appear in darker colors.

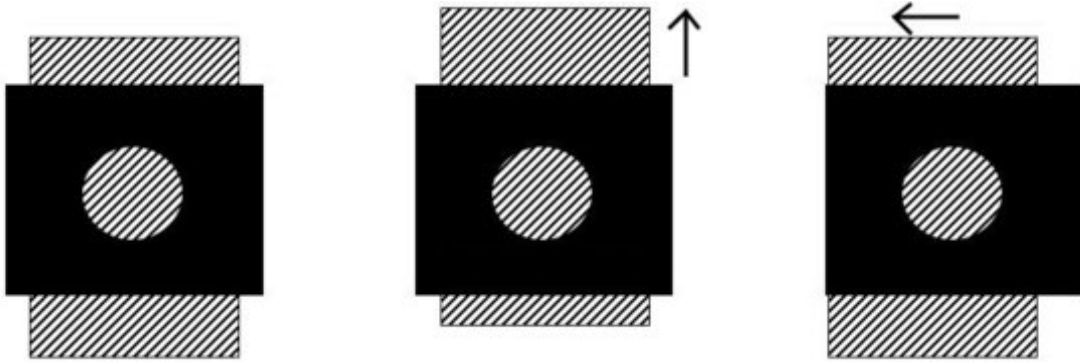


Figure 4.3: The aperture problem. The stripe pattern in the left image is shifted upwards or to the left. Both image motions yield the same visual motion within the bull’s eye (the aperture)[21].

Chapter overview: This chapter presents methods to estimate the optical flow field of two consecutive images of an image sequence. In Section 4.1 different approaches for the optical flow estimation from the literature are reviewed. A more detailed look is taken at total variation optical flow, yielding a dense (e.g. for every image pixel) and piecewise smooth flow field. It turns out, that the algorithmic design behind the mathematical notations consists of an iterative data term evaluation procedure with subsequent denoising steps.

Section 4.2 generalizes this algorithmic design, presenting a novel framework for optical flow estimation which I call Refinement Optical Flow. Within this general framework different data terms and smoothness terms are derived which are then used to estimate the optical flow field between two input images. Furthermore, this section provides implementation details for some chosen data terms and smoothness terms.

Section 4.4 concludes the chapter with experimental results. It shows results for the plain optical flow estimation and for optical flow with a fundamental matrix prior, favoring flow along the epipolar lines. The novel Refinement Optical Flow approach is carefully investigated for different data terms and smoothness filters. Experiments on the Middlebury optical flow benchmark prove the accuracy of the novel Refinement Optical Flow framework. The robustness and the ability to handle large displacements are shown on real images from traffic scene sequences.

4.1 Approaches in Literature

In this section, various approaches to optical flow in the literature are presented. Optical flow algorithms can be categorized into two general classes: pixel accurate optical flow algorithms and sub-pixel accurate optical flow algorithms.

Pixel accurate optical flow algorithms assign pixels of the input image to pixels of the output image. This is usually done by evaluating a pixel matching score based on gray values of a pixel’s neighborhood. Algorithms of this class can be understood as a combinatorial task, because the number of possible flow vectors or displacement vectors for a certain pixel is bounded by the image size. Due to the combinatorial structure, pixel accurate optical flow algorithms can be parallelized yielding real-time efficiency on dedicated hardware. They have the nice property of robustness due to the restricted solution space (only discrete integer pixel positions are considered and outliers are less likely), but at the same time suffer from accuracy limitations, i.e. the displacements are only pixel-discrete. In Subsection 4.1.1 a census based optical flow algorithm is reviewed as a representative of this class.

Subsection 4.1.2 introduces the optical flow constraint, which is employed by most sub-pixel accurate optical flow algorithms. It assumes, that the gray value image sequence has a continuous domain. The change of a pixel’s gray value between two time instances can then be computed evaluating the image gradient. The early works of Horn and Schunck [73] and Lucas and Kanade

[90] are reviewed. They both compute optical flow using the linearized version of the optical flow constraint.

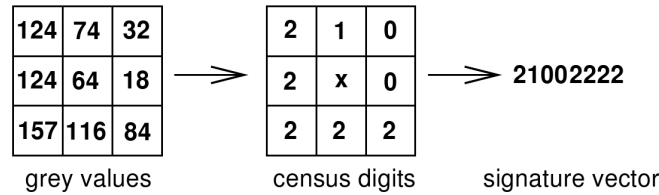
The method of Horn and Schunck is then extended by introducing robust norms into the elementary energy functional. This yields the total variation optical flow, which will be presented in Subsection 4.1.3. Total variation optical flow yields the currently best results on standard evaluation benchmarks. Subsection 4.1.4 names other optical flow approaches and concludes this section.

4.1.1 Census Based Optical Flow [124]

The census transformation, as applied in [77], compares the center pixel $\mathbf{x} = (x, y)^\top$ of an image patch to the other pixels \mathbf{x}' inside the patch:

$$\theta(I, \mathbf{x}, \mathbf{x}') = \begin{cases} 0 & \text{if } I(\mathbf{x}) - I(\mathbf{x}') > c \\ 1 & \text{if } |I(\mathbf{x}) - I(\mathbf{x}')| \leq c \\ 2 & \text{if } I(\mathbf{x}) - I(\mathbf{x}') < -c \end{cases},$$

with the gray value intensity $I(\mathbf{x})$ at pixel position \mathbf{x} . Typically, the threshold c is chosen between 12 and 16. The census digit θ measures the similarity between the gray values at pixel positions \mathbf{x} and \mathbf{x}' . Such representation is insensitive to a wide range of illumination changes. All census digits of the image patch are unrolled clockwise, building the signature vector:



The signature vector is used to search for corresponding point pairs. To this end, all signature vectors of the first image are stored in a hash-table together with their pixel position. Then, all signature vectors of the second image are compared to the hash-table entries. This gives a list of putative correspondences (hypotheses) for each signature. The list is empty if a signature in the second image does not exist in the first image. In the event of multiple entries, the list is reduced by applying photometric and geometric constraints (see [124] for further details). If there are still multiple entries, the shortest displacement vector wins.

Thanks to the indexing scheme, arbitrarily large displacements are allowed. Even if an image patch moves from the top left image corner to the bottom right corner it would be correctly matched. The method has been successfully implemented on parallel hardware, allowing for real-time computation. A disadvantage of the census method is its pixel-accuracy; the results suffer from discretization artifacts. Furthermore, low contrast information (gray value difference below the threshold c) is ignored.

4.1.2 The Optical Flow Constraint

Most sub-pixel accurate solutions to optical flow estimation are based on the (linearized) optical flow constraint (see optical flow evaluation in [36]). The optical flow constraint states that the gray value of a moving pixel stays constant over time, i. e.

$$I(x, y, t) = I(x + u, y + v, t + 1)$$

where $\mathbf{u} = (u, v)$ is the optical flow vector of a pixel $\mathbf{x} = (x, y)$ from time t to time $t + 1$.

Its linearized version (using the first order Taylor approximation) reads

$$I(x, y, t) \approx I(x, y, t+1) + \nabla I(x, y, t+1)^\top \begin{pmatrix} u \\ v \end{pmatrix}$$

$$0 = \underbrace{I(x, y, t+1) - I(x, y, t)}_{I_t(x, y, t+1)} + \nabla I(x, y, t+1)^\top \begin{pmatrix} u \\ v \end{pmatrix} .$$

From now on the partial derivatives of the image function are denoted as I_t , I_x , and I_y and also the inherent dependency on the image position (x, y) is dropped yielding the optical flow constraint equation

$$\text{OFC}(u, v) : \quad 0 = I_t + I_x u + I_y v . \quad (4.1)$$

The optical flow constraint has one inherent problem: it yields only one constraint to solve for two variables. It is well known that such an under-determined equation system yields an infinite number of solutions. For every fixed u a valid v can be found fulfilling the constraint.

Lucas-Kanade Method A common workaround to solve this ambiguity is to assume a constant (or affine) optical flow field in a small neighborhood of a pixel \mathbf{x} . Such a neighborhood \mathcal{N} typically consists of $n \times n$ pixels with n smaller than 15. The optical flow constraint is then evaluated with respect to all pixels within this neighborhood window \mathcal{N} . The optical flow constraint will usually not be perfectly fulfilled for all pixels as the assumption of equal flow vectors within the window is violated. Minimizing the sum of quadratic deviations yields the approach proposed by Lucas and Kanade in 1981 [90]:

$$\min_{u, v} \left\{ \sum_{\mathbf{x}' \in \mathcal{N}(\mathbf{x})} (I_t(\mathbf{x}') + I_x(\mathbf{x}')u + I_y(\mathbf{x}')v)^2 \right\} . \quad (4.2)$$

Extensions to this approach have been proposed; replacing the sum of squared errors with an absolute error measurement [35] or using Kalman filters to further gain robustness over time [113].

The computed flow vector is sub-pixel accurate. But due to the Taylor approximation, the method is only valid for small displacement vectors [35]. Larger displacements are found by embedding the method into a pyramid approach, solving for low frequency structures in low resolution images first and refining the search on higher resolved images (see Figure 4.4). While the maximum track length depends on image content, generally speaking, flow vectors with large displacements are less likely to be found than those within a few pixels displacement. The Lucas

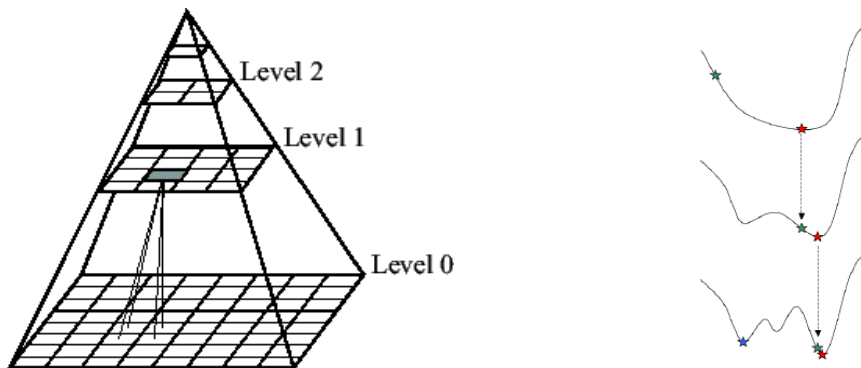


Figure 4.4: The original image corresponds to level 0 of the image pyramid. The upper pyramid levels are down-sampled versions of the image with lower resolution; hence, high frequency image details are filtered out (*left*). For pyramid approaches, the solution on an upper pyramid level is propagated onto the lower levels and refined using the high frequency details (*right*) [43, 101].

and Kanade method is currently beyond real-time if a flow vector is estimated for every pixel. Hence, Tomasi proposed to evaluate only those regions which yield a well-conditioned equation system when solving Equation (4.2) [129]. Currently, GPU based implementations are able to track up to 10,000 image points at frame rates of 25 Hz [148]. Such systems are used for example in driver assistance systems.

Variational Methods Another approach to cope with the under-determined optical flow constraint was proposed by Horn and Schunck [73] at the same time as the KLT approach first appeared (1981). The authors reverted to regularization, or smoothness, for the resulting flow field. Such smoothness was introduced by penalizing the derivative of the optical flow field, yielding an energy which was minimized by variational approaches:

$$\min_{u(\mathbf{x}), v(\mathbf{x})} \left\{ \int_{\Omega} (|\nabla u(\mathbf{x})|^2 + |\nabla v(\mathbf{x})|^2) d\Omega + \lambda \int_{\Omega} (I_t + I_x u(\mathbf{x}) + I_y v(\mathbf{x}))^2 d\Omega \right\}. \quad (4.3)$$

Here, $u(\mathbf{x}) \in \mathbb{R}$ and $v(\mathbf{x}) \in \mathbb{R}$ is the two-dimensional displacement vector for an image pixel $\mathbf{x} \in \mathbb{R}^2$; Ω is the image domain. The first term (regularization term) penalizes high variations in the optical flow field to obtain smooth displacement fields. The second term (data term) evaluates the optical flow constraint (4.1). The free parameter λ weights between the optical flow constraint and the regularization force. Variational optical flow approaches compute the optical flow field for all pixels within the image, hence they result in a dense optical flow field.

Being computationally more expensive, variational approaches only recently became quite popular as processor speed has increased, yielding real time computation of dense flow fields. In [48, 49], a highly efficient multi-grid approach on image pyramids is employed to obtain real-time performance. In [150] a duality-based method was proposed, which uses the parallel computing power of a GPU to gain real-time performance.

The original work of Horn and Schunck suffers from the fact that it does not allow for discontinuities in the optical flow field and does not handle outliers in the data term robustly. To date the work of Horn and Schunck has attracted 3900 citations, many of these dealing with applications of motion estimation in different scenarios, many suggesting alternative cost functionals, and many investigating alternative minimization strategies. Since discontinuities in the optical flow often appear in conjunction with high image gradients, several authors replace the homogeneous regularization in the Horn-Schunck model with an anisotropic diffusion approach [106, 141]. Others substitute the squared penalty functions in the Horn-Schunck model with more robust variants. Cohen [54] as well as Black and Anandan [40] apply estimators from robust statistics and obtain a robust and discontinuity preserving formulation for the optical flow energy. Aubert et al. [31] analyze energy functionals for optical flow incorporating an L^1 norm for the data fidelity term and a general class of discontinuity preserving regularization forces. Brox et al. [45] employ a differentiable approximation of the L^1 norm for both, the data and smoothness term and formulate a nested iteration scheme to compute the displacement field.

The integral of the L^1 norm of the gradient, i. e. $\int |\nabla f(x)| dx$, is also called the TV norm, an abbreviation for *total variation* norm. Essentially, the integral *counts* the amount of variation, or *fluctuation*, in the data. In contrast to the original quadratic L^2 -regularity suggested by Horn and Schunck, the L^1 -regularity is known to better preserve discontinuities [40, 45, 59, 106, 116]. Figure 4.5 illustrates the advantage of using the L^1 norm for denoising: “For fixed boundary conditions, all monotone functions, regardless if discontinuous or smooth, have the same total variation. This essential property enables the total variation regularization to reconstruct non-smooth signals. In other words, total variation has implicitly no bias against discontinuities” [109].

Zach et al. [150] proposed to solve the robust TV optical flow formulation by rewriting it into a convex dual form, allowing a separation of the data term and the smoothness term in an

$$\text{TV:} \quad \int |\nabla f(x)|^2 dx$$

$$\text{Quadratic:} \quad \int |\nabla f(x)| dx$$

Function	TV	Quadratic
$f_1(x)$	1.0	1.0
$f_2(x)$	1.0	0.11
$f_3(x)$	1.0	0.01

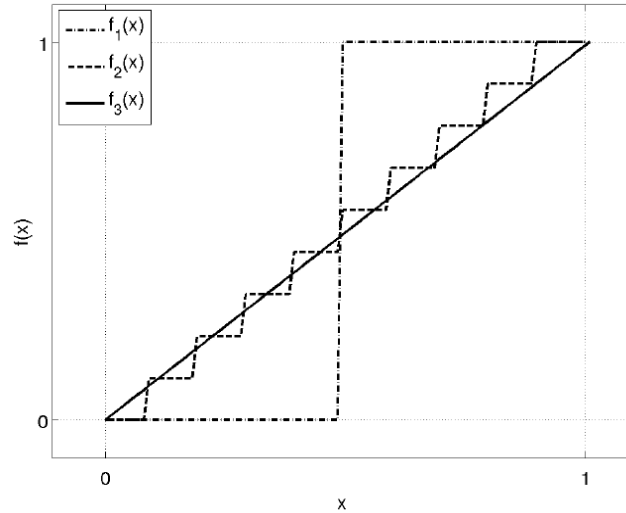


Figure 4.5: Total variation does not see any difference between these three functions (figure courtesy of Thomas Pock [109]).

iterative solving process. Let us replicate this approach in Section 4.1.3 and draw conclusions about the algorithmic design behind the mathematical notations. It will turn out, that basically two steps are performed independently, (1) data term evaluation and (2) flow field denoising.

4.1.3 Total Variation Optical Flow

Recall that the general Horn and Schunck energy (Equation (4.3)) for a two-dimensional flow field (u, v) (dropping the inherent dependency on \mathbf{x}) is given by

$$\int_{\Omega} \left(|\nabla u|^2 + |\nabla v|^2 + \lambda |I_t + I_x u + I_y v|^2 \right) d\Omega .$$

It has the disadvantage of not allowing for sharp edges in the smoothness term and does not handle outliers in the data term robustly due to the quadratic penalizers. Replacing the quadratic penalizers with robust versions (i. e. the absolute function) yields

$$\int_{\Omega} \left(|\nabla u| + |\nabla v| + \lambda |I_t + I_x u + I_y v| \right) d\Omega . \quad (4.4)$$

Although Equation (4.4) seems to be simple, it offers computational difficulties. The main reason is that both, the regularization term and the data term, are not continuously differentiable. One approach is to replace the absolute function $|x|$ with the differentiable approximation $\Psi_{\varepsilon}(x) = \sqrt{x^2 + \varepsilon^2}$, and to apply a numerical optimization technique on this slightly modified functional (e. g. [45, 52]). In [150] the authors propose an exact numerical scheme to solve Equation (4.4) by adopting the *Rudin-Osher-Fatemi* (ROF) energy [114] for total variation based image denoising to optical flow. Let us review this approach now. The ROF energy for image denoising is covered in more detail in Section 4.2.

In [150] the authors introduce auxiliary (so called dual) variables u' and v' and replace one instance of the original variables u and v in Equation (4.4) with the dual variables, yielding

$$\min_{u, v, u', v'} \left\{ \int_{\Omega} \left(|\nabla u| + |\nabla v| + \lambda |I_t + I_x u' + I_y v'| \right) d\Omega \right\} \text{ with } u = u' \text{ and } v = v' .$$

This step does not change the energy at all. The ingenious novelty is the following: the side conditions $u = u'$ and $v = v'$ are now relaxed (see [26] for more details on Lagrangian relax-

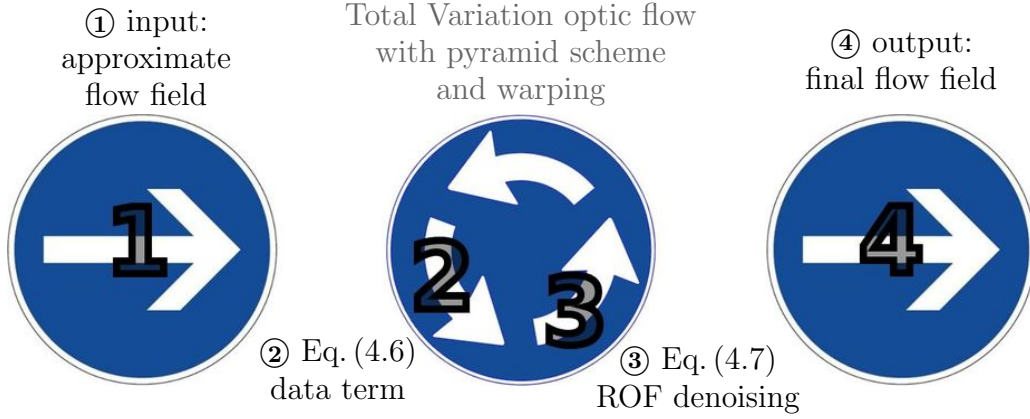


Figure 4.6: Basic algorithmic design of the total variation optical flow algorithm. Subsequent data term evaluation and denoising steps, embedded into a warping strategy, are applied to an approximate flow field (e. g. the zero flow field) to compute the resulting flow field.

ation) and embedded into the energy functional to be minimized. This yields the following *convex approximation* of the original total variation regularized optical flow problem

$$\min_{u,v,u',v'} \left\{ \int_{\Omega} \left(|\nabla u| + |\nabla v| + \frac{1}{2\theta}(u - u')^2 + \frac{1}{2\theta}(v - v')^2 + \lambda |I_t + I_x u' + I_y v'| \right) d\Omega \right\}, \quad (4.5)$$

where θ is a small constant, such that u is a close approximation of u' (and equivalent v is a close approximation of v'). The resulting energy is strictly convex, hence it has a unique minimum. This minimum is found by alternating steps updating either the dual variables, u' and v' , or the primal variables, u and v , in every iteration (see also Figure 4.6):

1. For u and v being fixed, solve

$$\min_{u',v'} \int_{\Omega} \left\{ \frac{1}{2\theta}(u - u')^2 + \frac{1}{2\theta}(v - v')^2 + \lambda |I_t + I_x u' + I_y v'| \right\} d\Omega. \quad (4.6)$$

Note, that this minimization problem can be independently evaluated for every pixel because no spatial derivatives of the flow field contribute to the energy. More specifically, the optical flow constraint data term, $\lambda |I_t + I_x u' + I_y v'|$, is minimized w. r. t. the dual flow variables u' and v' such that deviations from the primal variables are penalized quadratically. This demands a solution close to the given (approximate) primal flow field.

2. For u' and v' being fixed, solve

$$\min_{u,v} \int_{\Omega} \left\{ |\nabla u| + |\nabla v| + \frac{1}{2\theta}(u - u')^2 + \frac{1}{2\theta}(v - v')^2 \right\} d\Omega. \quad (4.7)$$

This is the total variation based image denoising model of Rudin, Osher, and Fatemi [114]. The denoising will be presented in more detail within the general framework in Section 4.2. For now, note that a denoising of the dual variables, u' and v' , is computed.

Embedding Equations (4.6) and (4.7) into a pyramid warping scheme implies that the denoised solution vector (u, v) serves as a new approximate flow field (u, v) for the next iteration or next lower pyramid level. Iteratively solving the data term and subsequent denoising yields the final optical flow result.

This basic idea of data term solving and denoising is picked up in Section 4.2 and a generalized approach is presented which I call *Refinement Optical Flow*. In that section the solutions for the minimization problems (4.6) and (4.7) are also outlined. But beforehand, let us conclude the literature overview on optical flow approaches.

4.1.4 Other Optical Flow Approaches

Robust variational approaches have been shown to yield the most accurate results to the optical flow problem known in literature. But they cover only a subset of dense optical flow algorithms.

In [123], a two-step method is presented where first optical flow is computed in a local neighborhood and secondly a dense flow field is derived in a relaxation step. Other optical flow algorithms with outstanding performance are graph cuts (discrete optimization) [56] or the fusion of previously mentioned approaches into an optimized result [85]. The Middlebury optical flow evaluation homepage gives a nice summary of other state-of-the-art optical flow algorithms [36].

4.2 A General Flow Refinement Framework

In this section, a general framework for optical flow, based on repeatedly refining a flow field, is derived. This generalizes the idea of data term evaluation and ROF denoising from the total variation optical flow. An overview of the general make-up is given in Figure 4.7.

The general make-up consists of a pixel-wise data term optimization step ② and a global smoothness term evaluation step ③. Both steps are presented separately in Subsections 4.2.1 and 4.2.2. Subsection 4.2.3 summarizes the algorithm and provides implementation details.

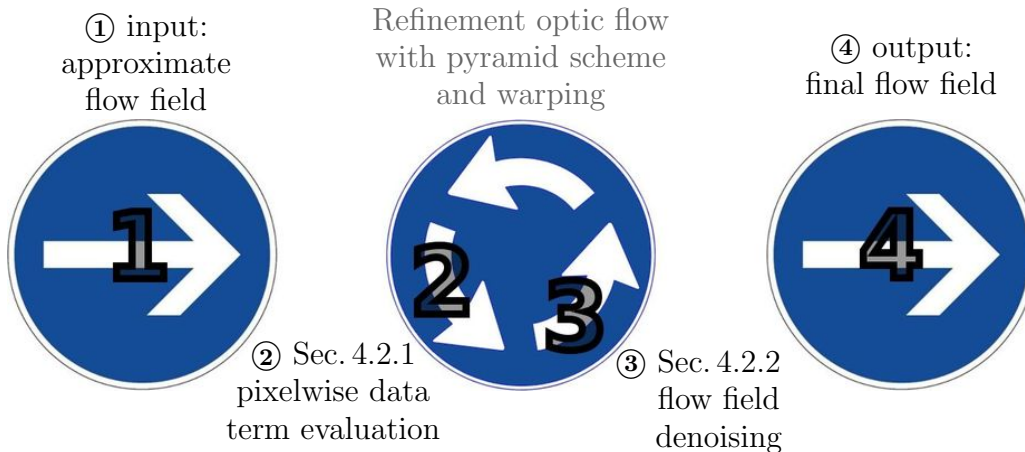


Figure 4.7: The general framework for Refinement Optical Flow. An approximate flow field is refined by iteratively evaluating the data term and subsequent smoothing, embedded in a pyramid warping scheme. The result is an optimized and refined optical flow field (compare with Figure 4.6).

4.2.1 Data Term Optimization

The data term represents the driving force for optical flow computation. Given an approximate n -dimensional flow field $\mathbf{u}' \in \mathbb{R}^n$ as prior, an optimal solution is computed for every pixel which fulfills best a single or multiple data terms. More specifically, a solution $\mathbf{u} = (u_1, \dots, u_n)^\top \in \mathbb{R}^n$ is obtained, which minimizes the weighted sum of the distance to the prior flow field and the individual data term violations $p(\cdot)$,

$$\mathbf{u} = \min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\| + \sum_{p \in \mathcal{P}} \omega \left(\lambda_p p(\mathbf{u}) \right) \right\}. \quad (4.8)$$

Here, \mathcal{P} is the set of all data terms and each data term is given an individual weight λ_p . ω is a function penalizing deviations from 0. In this subsection, the robust L^1 norm is used. A typical data term is the brightness constancy constraint, $p(\mathbf{u}) = I_1(\mathbf{x}) - I_2(\mathbf{x} + \mathbf{u})$. In fact, for a single data term this becomes the methodology employed in [150], from where I derived my ideas. See Appendix A for more details on different data terms.

The investigated data terms in this subsection employ linearized constraint violations (e. g. violations of the optical flow constraint (4.1)). The linearized version of a constraint $p(\mathbf{u})$ consists of a constant scalar value, denoted by a subscript zero p_0 , and a vector denoted by \mathbf{p} , i. e.

$$\lambda_p p(\mathbf{u}) \approx p_0 + \mathbf{p}^\top \mathbf{u} = p_0 + p_1 u_1 + \cdots + p_n u_n .$$

Note, that the weight λ_p is included in the data term entries p_i for convenience reasons. In summary, we have

$$\min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + \sum_{p \in \mathcal{P}} |\lambda_p p(\mathbf{u})| \right\} = \min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + \sum_{p \in \mathcal{P}} |p_0 + \mathbf{p}^\top \mathbf{u}| \right\} . \quad (4.9)$$

The remainder of this subsection deals with the optimization of Equation (4.9) and presents a simple example employing the optical flow constraint. Due to the absolute function in the robust data deviation term, Equation (4.9) is not differentiable at $p_0 + \mathbf{p}^\top \mathbf{u} = 0$. This is a pity, because the violations of the data terms should be as small as possible demanding the data terms to vanish and causing numerical instabilities. Two ways to solve this issue are: using an ε -approximation of the absolute norm [45] and quadratic optimization techniques.

The ε -approximation of the absolute norm yields a lagged iterative solution, while quadratic optimization yields an exact solution within a fixed number of thresholding steps. In the following subsection, the lagged iterative solution is presented. Then, quadratic optimization techniques are derived for a single, two, and multiple data terms.

(1) Approximating the Absolute Function

The absolute function $|x|$ is not differentiable in $x = 0$. An often used approximation (e. g. [45]) is to replace the absolute function by $\Psi(x) = \sqrt{x^2 + \epsilon^2}$ with a small constant $\epsilon > 0$. This function is convex and differentiable. Its derivative is given by $\Psi'(x) = \frac{x'}{\sqrt{x^2 + \epsilon^2}}$. Due to the appearance of x in the derivative, an iterative approach is commonly used in order to minimize the function w.r.t. x by gradient descent. Such a solution approach is also called *lagged feedback* because, usually, a few iterations are needed to get close to the global minimum.

Replacing the absolute functions in Equation (4.9) with the Ψ functions yields the following approximation for the optimization problem:

$$\min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + \sum_{p \in \mathcal{P}} \Psi \left((p_0 + \mathbf{p}^\top \mathbf{u})^2 \right) \right\} . \quad (4.10)$$

The objective function now is convex and the minimum is found by successive gradient descent steps. This is done by setting its derivative w.r.t \mathbf{u}^k , with k being the iteration index, equal zero. The starting point \mathbf{u}^0 is arbitrary because the objective function is convex. This yields

$$\mathbf{u}^k = \mathbf{u}' - \sum_{p \in \mathcal{P}} \left\{ \frac{1}{\sqrt{(p_0 + \mathbf{p}^\top \mathbf{u}^{k-1})^2 + \epsilon^2}} \mathbf{p} (p_0 + \mathbf{p}^\top \mathbf{u}^k) \right\} .$$

$$\mathbf{u}^k = \left(\mathbf{Id} + \sum_{p \in \mathcal{P}} \frac{1}{\sqrt{(p_0 + \mathbf{p}^\top \mathbf{u}^{k-1})^2 + \epsilon^2}} \mathbf{p} \mathbf{p}^\top \right)^{-1} \left(\mathbf{u}' - \sum_{p \in \mathcal{P}} \left\{ \frac{1}{\sqrt{(p_0 + \mathbf{p}^\top \mathbf{u}^{k-1})^2 + \epsilon^2}} p_0 \mathbf{p} \right\} \right) .$$

Due to the lagged feedback, in general a few iterations (re-linearizations) are needed to find a minimum which is close to the optimum.

(2) Quadratic Optimization

Another approach to solve the objective equation, Equation (4.9), is quadratic programming. In this subsection, the optimization problem involving the absolute function is transformed into a quadratic optimization problem with inequality constraints. The basics for optimization techniques based on quadratic optimization will now be presented. Finally, a fast and optimal thresholding scheme for solving the resulting quadratic optimization problem is presented.

In a first step, the absolute functions (the original data term $p \in \mathcal{P}$) are replaced by dual variables p' and inequality constraints (see Figure 4.8 for an example). The remaining minimization problem is to find the minimum of

$$\frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + \sum_{p \in \mathcal{P}} p' \quad (4.11)$$

with inequality constraints

$$\begin{aligned} \text{(i)} \quad & p_0 + \mathbf{p}^\top \mathbf{u} - p' \leq 0 \quad \forall p \in \mathcal{P} \text{ and} \\ \text{(ii)} \quad & -p_0 - \mathbf{p}^\top \mathbf{u} - p' \leq 0 \quad \forall p \in \mathcal{P}. \end{aligned} \quad (4.12)$$

Equation (4.11) with side conditions (4.12) yields the same minimum as Equation (4.9). The absolute function has been replaced yielding a differentiable objective function. This, however, is at the cost of one additional variable and two inequality constraints per data term. Note, that Equation (4.11) is a combination of convex functions, and hence is convex itself. Let us now use the Karush-Kuhn-Tucker theorem [76] for convex quadratic minimization problems under linear inequality constraints to find an optimal solution to Equation (4.11) and hence the original problem (4.9):

For a convex quadratic minimization problem $\min_x \{f(x)\}$, under N linear inequality constraints $g_i(x) \leq 0$ where $0 \leq i \leq N$, a global optimum of a solution x^* holds true if there exist constants μ_i such that the Karush-Kuhn-Tucker (KKT) conditions [76] are fulfilled:

$$\text{Stationarity: } \nabla f(x^*) + \sum_i \mu_i \nabla g_i(x^*) = 0.$$

$$\text{Primal feasibility: } g_i(x^*) \leq 0.$$

$$\text{Dual feasibility: } \mu_i \geq 0.$$

$$\text{Complementary slackness: } \mu_i g_i(x) = 0 \quad \forall i.$$

Solution schemes for a single data term, two data terms based on thresholding, and for the general case of multiple data terms are presented in the following.

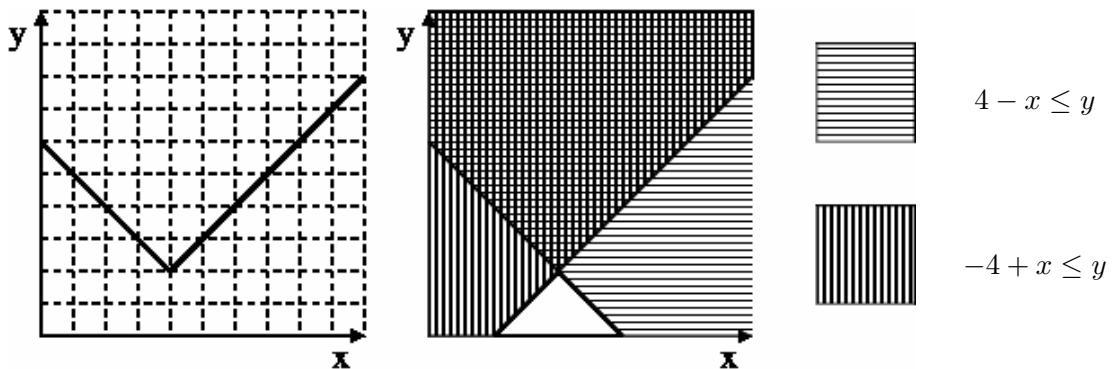


Figure 4.8: Example for finding $\min_x \{2 + |4 - x|\}$ (minimum $x^* = 4$). The $|\cdot|$ function is replaced by a dual variable y and inequality constraints (equations to right), yielding $\min_{x,y} \{2 + y\}$. The minimum is $(x^*, y^*) = (4, 2)$. Note that both problems yield the same minimum for the primal variable x^* .

Single Data Term

For a single data term, the task is to find the vector \mathbf{u}^* which solves the minimization problem

$$\mathbf{u}^* = \min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + |p_0 + \mathbf{p}^\top \mathbf{u}| \right\},$$

or its dual formulation

$$\mathbf{u}^* = \min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + p' \right\}$$

with linear side conditions

$$\begin{aligned} p_0 + \mathbf{p}^\top \mathbf{u} - p' &\leq 0 \\ \text{and } -p_0 - \mathbf{p}^\top \mathbf{u} - p' &\leq 0 \end{aligned} .$$

The solution computes as (first presented by [150]; see Appendix B.2 for the proof):

Thresholding Check	Solution
$p(\mathbf{u}') < -\mathbf{p}^\top \mathbf{p}$	$\mathbf{u}^* = \mathbf{u}' + \mathbf{p}$
$ p(\mathbf{u}') \leq \mathbf{p}^\top \mathbf{p}$	$\mathbf{u}^* = \mathbf{u}' - \frac{p(\mathbf{u}')}{\mathbf{p}^\top \mathbf{p}} \mathbf{p}$
$p(\mathbf{u}') > \mathbf{p}^\top \mathbf{p}$	$\mathbf{u}^* = \mathbf{u}' - \mathbf{p}$

Two Data Terms

In this subsection, an efficient solution scheme to minimize the objective function (4.9) or, equivalently, its dual quadratic optimization problem (4.11) with inequality constraints (4.12) for two data terms is presented. To this end, we have two data terms, $\lambda_1 p_1(\mathbf{u}) = a_0 + \mathbf{a}^\top \mathbf{u}$ and $\lambda_2 p_2(\mathbf{u}) = b_0 + \mathbf{b}^\top \mathbf{u}$, where a_0 and b_0 are constant and \mathbf{a} and \mathbf{b} represent the linear parts of the data terms. The minimization problem states

$$\min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + \underbrace{|a_0 + \mathbf{a}^\top \mathbf{u}|}_{\lambda_1 p_1(\mathbf{u})} + \underbrace{|b_0 + \mathbf{b}^\top \mathbf{u}|}_{\lambda_2 p_2(\mathbf{u})} \right\}. \quad (4.13)$$

Its dual formulation with dual variables a' and b' is

$$\min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + a' + b' \right\} \quad \text{such that} \quad \begin{aligned} (i) \quad & a_0 + \mathbf{a}^\top \mathbf{u} - a' \leq 0 \\ (ii) \quad & -a_0 - \mathbf{a}^\top \mathbf{u} - a' \leq 0 \\ (iii) \quad & b_0 + \mathbf{b}^\top \mathbf{u} - b' \leq 0 \\ (iv) \quad & -b_0 - \mathbf{b}^\top \mathbf{u} - b' \leq 0. \end{aligned}$$

The thresholding steps for solving 4.13 are given in Table 4.1; the derivation can be found in Appendix B.3.

Multiple Data Terms

If more than two data terms are given, a direct solution via thresholding becomes computationally expensive due to the high number of comparisons needed. Let the number of data terms be n . Then for every possible solution (each data term may be negative, positive or equal to zero yielding 3^n possible solutions), one has to perform n thresholding checks. Hence, the total number of comparisons is $n \cdot 3^n$; it increases exponentially. Of course, the search for the minimum can be stopped once a solution is found, yielding $n \cdot 3^n$ only in the worst case. With one

Thresholding Checks	Solution
$a_0 + \mathbf{a}^\top(\mathbf{u}' - \mathbf{a} - \mathbf{b}) \geq 0$ $b_0 + \mathbf{b}^\top(\mathbf{u}' - \mathbf{a} - \mathbf{b}) \geq 0$	$\mathbf{u}^* = \mathbf{u}' - \mathbf{a} - \mathbf{b}$
$a_0 + \mathbf{a}^\top(\mathbf{u}' - \mathbf{a} + \mathbf{b}) \geq 0$ $b_0 + \mathbf{b}^\top(\mathbf{u}' - \mathbf{a} + \mathbf{b}) \leq 0$	$\mathbf{u}^* = \mathbf{u}' - \mathbf{a} + \mathbf{b}$
$a_0 + \mathbf{a}^\top(\mathbf{u}' + \mathbf{a} - \mathbf{b}) \leq 0$ $b_0 + \mathbf{b}^\top(\mathbf{u}' + \mathbf{a} - \mathbf{b}) \geq 0$	$\mathbf{u}^* = \mathbf{u}' + \mathbf{a} - \mathbf{b}$
$a_0 + \mathbf{a}^\top(\mathbf{u}' + \mathbf{a} + \mathbf{b}) \leq 0$ $b_0 + \mathbf{b}^\top(\mathbf{u}' + \mathbf{a} + \mathbf{b}) \leq 0$	$\mathbf{u}^* = \mathbf{u}' + \mathbf{a} + \mathbf{b}$
$a_0 + \mathbf{a}^\top(\mathbf{u}' - \mathbf{a} - \frac{\mathbf{b}}{\mathbf{b}^\top\mathbf{b}}(b_0 + \mathbf{b}^\top\mathbf{u}' - \mathbf{b}^\top\mathbf{a})) \geq 0$ $\left \frac{1}{\mathbf{b}^\top\mathbf{b}}(b_0 + \mathbf{b}^\top\mathbf{u}' - \mathbf{b}^\top\mathbf{a}) \right \leq 1$	$\mathbf{u}^* = \mathbf{u}' - \mathbf{a} - \frac{\mathbf{b}}{\mathbf{b}^\top\mathbf{b}}(b_0 + \mathbf{b}^\top\mathbf{u}' - \mathbf{b}^\top\mathbf{a})$
$a_0 + \mathbf{a}^\top(\mathbf{u}' + \mathbf{a} - \frac{\mathbf{b}}{\mathbf{b}^\top\mathbf{b}}(b_0 + \mathbf{b}^\top\mathbf{u}' + \mathbf{b}^\top\mathbf{a})) \leq 0$ $\left \frac{1}{\mathbf{b}^\top\mathbf{b}}(b_0 + \mathbf{b}^\top\mathbf{u}' + \mathbf{b}^\top\mathbf{a}) \right \leq 1$	$\mathbf{u}^* = \mathbf{u}' + \mathbf{a} - \frac{\mathbf{b}}{\mathbf{b}^\top\mathbf{b}}(b_0 + \mathbf{b}^\top\mathbf{u}' + \mathbf{b}^\top\mathbf{a})$
$b_0 + \mathbf{b}^\top(\mathbf{u}' - \mathbf{b} - \frac{\mathbf{a}}{\mathbf{a}^\top\mathbf{a}}(a_0 + \mathbf{a}^\top\mathbf{u}' - \mathbf{a}^\top\mathbf{b})) \geq 0$ $\left \frac{1}{\mathbf{a}^\top\mathbf{a}}(a_0 + \mathbf{a}^\top\mathbf{u}' - \mathbf{a}^\top\mathbf{b}) \right \leq 1$	$\mathbf{u}^* = \mathbf{u}' - \mathbf{b} - \frac{\mathbf{a}}{\mathbf{a}^\top\mathbf{a}}(a_0 + \mathbf{a}^\top\mathbf{u}' - \mathbf{a}^\top\mathbf{b})$
$b_0 + \mathbf{b}^\top(\mathbf{u}' + \mathbf{b} - \frac{\mathbf{a}}{\mathbf{a}^\top\mathbf{a}}(a_0 + \mathbf{a}^\top\mathbf{u}' + \mathbf{a}^\top\mathbf{b})) \leq 0$ $\left \frac{1}{\mathbf{a}^\top\mathbf{a}}(a_0 + \mathbf{a}^\top\mathbf{u}' + \mathbf{a}^\top\mathbf{b}) \right \leq 1$	$\mathbf{u}^* = \mathbf{u}' + \mathbf{b} - \frac{\mathbf{a}}{\mathbf{a}^\top\mathbf{a}}(a_0 + \mathbf{a}^\top\mathbf{u}' + \mathbf{a}^\top\mathbf{b})$
If all checks fail	$\mathbf{u}^* = \begin{bmatrix} \mathbf{a}^\top \\ \mathbf{b}^\top \end{bmatrix}^{-1} \begin{bmatrix} -a_0 \\ -b_0 \end{bmatrix}$

Table 4.1: Thresholding checks for two data terms.

data term, three comparisons have to be performed. Two data terms lead to a worst case of 18 comparisons (see above table) and for three data terms up to 81 comparisons are necessary.

This leads to the question, whether other quadratic optimization techniques can be used to minimize the objective function. A large number of approximate algorithms are known which approximate the linear constraint functions with barrier or penalty functions [91]. An exact solution is given by the so-called *active set method* [55]. Here, the solution is found by gradient descent within an active subset of inequality conditions. However, the algorithmic complexity prohibits a very large number of data terms. Hence, approximating the absolute function (e. g. as in Equation (4.10)) or approximating the inequality constraints is the most practicable approach to handle multiple data terms efficiently.

Toy example: optical flow data term

This subsection presents a toy example, minimizing Equation (4.9) with one data term, the linearized optical flow constraint. The prior flow field is set to $(u'_1, u'_2) = (0, 0)$. Ergo, small flow vectors are favored, which is correct in this very example, where most of the flow vectors have a displacement below one pixel. For the simple case of a one-channel gray value image, the data term becomes

$$\lambda p(\mathbf{u}) = \lambda (I_t + I_x u_1 + I_y u_2) .$$

This yields the following thresholding table to solve for $u_{1,2}$ (see also [150] and Appendix B.2):

Thresholding Check	Solution
$I_t + I_x u'_1 + I_y u'_2 < -\lambda \ \nabla I\ $	$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} u'_1 \\ u'_2 \end{pmatrix} + \lambda \nabla I$
$ I_t + I_x u'_1 + I_y u'_2 \leq \lambda \ \nabla I\ $	$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} u'_1 \\ u'_2 \end{pmatrix} - \frac{I_t + I_x u'_1 + I_y u'_2}{\ \nabla I\ } \nabla I$
$I_t + I_x u'_1 + I_y u'_2 > \lambda \ \nabla I\ $	$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} u'_1 \\ u'_2 \end{pmatrix} - \lambda \nabla I$

As one would expect, if only the thresholding step is performed for a given approximate flow field, the resulting flow field usually is noisy. See Figure 4.9 for the result ($\lambda = 50$). The data term is nearly fulfilled for most part of the image while the optical flow field proves to be very noisy. Hence, it is straightforward to denoise and smooth the flow field in order to derive a more appealing solution. The next section presents methods for the second step of the general Refinement Optical Flow, the smoothness term evaluation.

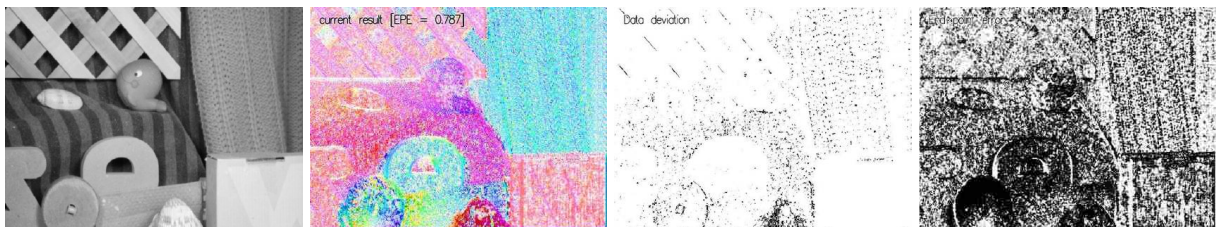


Figure 4.9: Optical flow for the *rubber whale* sequence of the Middlebury optical flow data set using a simple thresholding step. The computed flow field (second from left) is noisy and shows many outliers. The data term deviation (third from left) is small (black corresponds to 3.75% gray value error) while the optical flow end point error is fairly large (right image, black denotes 0.5px error).

4.2.2 Smoothness Term Evaluation

The first step in the general framework for Refinement Optical Flow consists of a data term evaluation step. The data term optimization (Equation 4.8) is independently performed for each pixel. An advantage of such an algorithm is that it can be sped up using multiple processors and parallel computing power, e. g. modern graphics processing units (GPUs).

The disadvantage is that the solution is local in the way that only the pixel itself contributes to the solution. As seen above (compare Figure 4.9), this leads to noisy flow fields, mainly due to three reasons; corrupted image data due to sensor noise, low entropy (information content) in the image data, and illumination artifacts.

Assuming that the noise is uncorrelated and has zero mean, the common approach to lower the noise level is a subsequent smoothing operation. However, smoothing the flow field may lead to a lack of sharp discontinuities that exist in the true flow field, especially at motion boundaries. Hence, discontinuity-preserving filters yielding a piecewise smooth flow field have to be employed.

The aperture problem (see Figure 4.3) can only be solved if information from a region's boundary is propagated into the interior. While smoothing can be performed locally, this does not wholly solve the aperture problem (unless the filter mask is chosen large enough). Only global techniques, propagating information across the whole image, promise improvements. The three main objectives for the smoothing step can be summarized as

- Discard outliers in the flow field due to corrupted image data (denoising).
- Preserve edges, i. e. do not smooth over flow edges.
- Propagate information into areas of low texture (smoothing).

Illumination artifacts are different in nature as they cannot be removed by denoising or smoothing. Section 4.3 will tackle the illumination problem and show how to increase robustness to illumination changes by preparing the input image accordingly. In this subsection, different smoothing (or denoising) filters are presented; the median filter, total variation denoising filters, an anisotropic and edge preserving diffusion filter, and a second order prior denoising. Quantitative results obtained using the different filters are discussed in Section 4.4. In general, a filter \mathcal{F} is applied to a given n -dimensional flow field \mathbf{u} , where the gray value image I may yield as prior; it returns the smoothed flow field $\hat{\mathbf{u}}$ such that

$$\hat{\mathbf{u}} = \mathcal{F}(\mathbf{u}, I) . \quad (4.14)$$

Median Filtering

Discarding outliers and preserving edges is a well-known characteristic of rank filters [98]. The median filter probably is the best-known rank filter. According to [19], the median of N numerical values has the following properties:

- The median of a list of N values is found by sorting the input array in increasing order, and taking the middle value.
- The median of a list of N values has the property that in the list there are as many greater as smaller values than this element.
- The median of a list of N values minimizes the sum of deviations over all list entries.

The last property makes the median filter especially suitable for denoising if noise characteristics are unknown. A median filter has also the nice property of converging to a periodic solution if recursively executed on an input image. This periodic (for most part of the image stationary)

image is called the root image of the median filter [146]. For recursively applying n median filter steps, denoted by a superscript, i. e.

$$\hat{\mathbf{u}} = \mathcal{MF}^n(\mathbf{u}) = \mathcal{MF}(\mathcal{MF}^{n-1}(\mathbf{u})) , \quad (4.15)$$

the root image property yields

$$\exists n > 0, i > 0 : \mathcal{MF}^{n+i}(\mathbf{u}) = \mathcal{MF}^n(\mathbf{u}) .$$

Recursively applying the median filter propagates information across the image and produces piecewise smooth flow fields. For the root image, the above definitions of the median filter are inherently fulfilled; outliers are replaced by local medians, such that deviations between neighboring pixels are minimized.

In the experiments, a fixed number of median filter iterations is used. Using the *Bubble-Sort* algorithm, median filtering can be implemented quite efficiently employing a fixed number of comparisons [19]. The window size used in the experiments is 3×3 .

TV- L^1 Denoising

The continuous counterpart to median filtering is total variation denoising with an L^1 data term, which provides an edge-preserving and smooth flow field. The classical TV denoising algorithm is described by Rudin, Osher, and Fatemi in [114]. Their algorithm seeks an equilibrium state (minimal energy) of an energy functional consisting of the TV norm of the data field, $|\nabla \hat{\mathbf{u}}|$, and a fidelity term of the data field $\hat{\mathbf{u}}$ to the noisy input data field \mathbf{u} :

$$\hat{\mathbf{u}} = \mathcal{TV}_{L^p}(\mathbf{u}) = \min_{\hat{\mathbf{u}}} \left\{ \int_{\Omega} \left(|\nabla \hat{\mathbf{u}}| + \frac{1}{2} \lambda |\hat{\mathbf{u}} - \mathbf{u}|^p \right) d\Omega \right\} . \quad (4.16)$$

Here, $\lambda \in \mathbb{R}$ is a scalar value controlling the fidelity of the solution to the input image (inversely proportional to the measure of denoising); $p = 2$ yields a quadratic penalizer and $p = 1$ linear penalizing. The resulting data field contains the *cartoon part* of the image [147]. The cartoon has a curve discontinuities, but elsewhere it is assumed to have small or null gradient, $|\nabla \hat{\mathbf{u}}| \approx 0$. In the original work [114], a quadratic data fidelity term ($p = 2$) was used. A closer approximation of the discrete median filter is an absolute data fidelity term [37]. The following solution scheme for total variation with an absolute data term is found in [109]:

Proposition 1 *The solution of*

$$\mathcal{TV}_{L^1}(\mathbf{u}) = \min_{\hat{\mathbf{u}}} \int_{\Omega} |\nabla \hat{\mathbf{u}}| + \frac{1}{2} \lambda |\hat{\mathbf{u}} - \mathbf{u}| d\Omega . \quad (4.17)$$

is given by

$$\hat{\mathbf{u}} = \mathbf{q} + \frac{1}{2} \lambda \mathbf{div} \mathbf{p} .$$

The dual variables $\mathbf{p} = [p_1, p_2]$ and \mathbf{q} are defined iteratively by computing

$$\tilde{\mathbf{p}}^{n+1} = \mathbf{p}^n + \frac{2\tau}{\lambda} \nabla \left(\mathbf{q} + \frac{1}{2} \lambda \mathbf{div} \mathbf{p} \right) , \text{ followed by } \mathbf{p}^{n+1} = \frac{\tilde{\mathbf{p}}^{n+1}}{\max \left\{ 1, |\tilde{\mathbf{p}}^{n+1}| \right\}}$$

and

$$q_{i,j}^{n+1} = \begin{cases} q_{i,j} - \frac{1}{2} \lambda \theta & \text{if } q_{i,j} - f_{i,j} > \frac{1}{2} \lambda \theta \\ q_{i,j} + \frac{1}{2} \lambda \theta & \text{if } q_{i,j} - f_{i,j} < -\frac{1}{2} \lambda \theta \\ f & \text{otherwise} \end{cases} .$$

where $\mathbf{p}^0 = \mathbf{0}$, $\mathbf{q}^0 = \mathbf{u}$, $\theta = 0.2$ and the time step $\tau \leq 1/4$.

The implementation of Proposition 1 uses backward differences to approximate $\mathbf{div} \mathbf{p}$ and forward differences for the numerical gradient computation in order to have mutually adjoint operators [50]. The discrete version of the forward difference gradient $(\nabla u)_{i,j} = ((\nabla u)_{i,j}^1, (\nabla u)_{i,j}^2)$ at pixel position (i, j) for a data field of width N and height M is defined as

$$(\nabla u)_{i,j}^1 = \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } i < N \\ 0 & \text{if } i = N \end{cases} \quad \text{and} \quad (\nabla u)_{i,j}^2 = \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } j < M \\ 0 & \text{if } j = M \end{cases} .$$

The discrete version of the backward differences divergence operator is

$$(\mathbf{div} \mathbf{p})_{i,j} = (\mathbf{div} [p^1, p^2])_{i,j} = \begin{cases} p_{i,j}^1 - p_{i-1,j}^1 & \text{if } 1 < i < N \\ p_{i,j}^1 & \text{if } i = 1 \\ -p_{i-1,j}^1 & \text{if } i = N \end{cases} + \begin{cases} p_{i,j}^2 - p_{i,j-1}^2 & \text{if } 1 < j < M \\ p_{i,j}^2 & \text{if } j = 1 \\ -p_{i,j-1}^2 & \text{if } j = M \end{cases} .$$

TV-L² Denoising

A solution scheme using the quadratic penalizers for Equation (4.16), the ROF model,

$$\mathcal{TV}_{L_2}(\mathbf{u}) = \min_{\hat{\mathbf{u}}} \int_{\Omega} |\nabla \hat{\mathbf{u}}| + \frac{1}{2} \lambda |\hat{\mathbf{u}} - \mathbf{u}|^2 \, d\Omega ,$$

was proposed in [51]:

Proposition 2 *The solution of*

$$\mathcal{TV}_{L_2}(\mathbf{u}) = \min_{\hat{\mathbf{u}}} \int_{\Omega} |\nabla \hat{\mathbf{u}}| + \frac{1}{2} \lambda (\mathbf{u} - \hat{\mathbf{u}})^2 \, d\Omega \quad (4.18)$$

is given by

$$\hat{\mathbf{u}} = \mathbf{u} + \frac{1}{\lambda} \mathbf{div} \mathbf{p} .$$

The dual variable $\mathbf{p} = [p_1, p_2]$ is defined iteratively with $\mathbf{p}^0 = \mathbf{0}$, the time step $\tau \leq 1/4$, and

$$\tilde{\mathbf{p}}^{n+1} = \mathbf{p} + \lambda \tau \left(\nabla \left(u + \frac{1}{\lambda} \mathbf{div} \mathbf{p}^n \right) \right) \quad \text{and} \quad \mathbf{p}^{n+1} = \frac{\tilde{\mathbf{p}}^{n+1}}{\max \{1, |\tilde{\mathbf{p}}^{n+1}|\}} . \quad (4.19)$$

Structure-Adaptive Smoothing

For many image sequences, discontinuities of the motion field tend to coincide with object boundaries and discontinuities of the brightness function. Although this is certainly not always true, the quantitative experiments on the optic flow benchmark [36] will demonstrate that the introduction of brightness-adaptive smoothness leads to improvements of optic flow estimates.

An elegant theoretical treatise of image-adaptive regularization of flow fields was presented in [142]. There, the authors introduce regularizers of the form

$$\Psi \left(\nabla v_1^\top D(\nabla I) \nabla v_1 \right) + \Psi \left(\nabla v_2^\top D(\nabla I) \nabla v_2 \right) , \quad (4.20)$$

corresponding to an inhomogeneous and potentially anisotropic regularization induced by a structure-dependent tensor $D(\nabla I)$. The central idea is that the smoothness of v along the two eigenvectors of D is weighted by the corresponding eigenvalues. In fact, anisotropic structure-dependent regularization was already proposed by Nagel in 1983 [105]. This is achieved by setting

$$D(\nabla I) = \frac{1}{|\nabla I|^2 + 2\lambda} \left(\nabla I^\perp \nabla I^{\top\perp} + \lambda^2 \text{Id} \right)$$

where Id denotes the unit matrix. This leads to an anisotropic smoothing of v along the level lines of the image intensity while preserving discontinuities across level lines. Lately, a fast numerical approximation scheme for anisotropic diffusion was proposed by Felsberg in [62]. Thanks to the author I was able to compare the algorithm with other denoising approaches in the experimental result section.

In order to include structure-adaptive smoothing into the total variation denoising, an inhomogeneous isotropic regularization is considered. Following [30], discontinuities of the motion field arising at locations of strong image gradients are favored setting $D(\nabla I) = g(|\nabla I|) \text{Id}$ with a strictly decreasing positive function

$$g(|\nabla I|) = \exp\left(-\alpha|\nabla I|^\beta\right). \quad (4.21)$$

Instead of solving Equation (4.18), this yields the following solution scheme [132]:

$$\mathcal{TV}_{L_2}(u, I) = \min_{\hat{\mathbf{u}}} \int_{\Omega} g(|\nabla I|) |\nabla \hat{\mathbf{u}}| + \frac{1}{2} \lambda (u - \hat{\mathbf{u}})^2 \, d\Omega. \quad (4.22)$$

This weighted total variation problem can be solved using Proposition 2 and replacing Equation (4.19) with

$$\mathbf{p}^{n+1} = \frac{\tilde{\mathbf{p}}^{n+1}}{\max\left\{1, \frac{|\tilde{\mathbf{p}}^{n+1}|}{g(|\nabla I|)}\right\}}.$$

Second Order Prior

Total variation techniques favor piecewise constant flow fields. This effect is not always desired, especially if the flow field is slightly affine, and causes a so-called *stair-casing effect* [131]. In [131], the authors propose to penalize the variation of the second derivative (for details I refer to [131]). This approach is also considered in the experiments in Section 4.4.

But before, the toy example for denoising will be continued and implementation details to the Refinement Optical Flow framework are given within the next section.

Toy example cont.: denoising the flow field

Let us continue the toy example from Section 4.2.1, Figure 4.9. The inherent problem is that the optical flow vectors after the data term evaluation are quite noisy; hence, the flow field needs to be denoised. In the toy example, the median filter is applied recursively to the flow field shown in Figure 4.9. The result can be seen in Figure 4.10. Clearly, the flow field is much smoother than simply minimizing the data term equations while edges are preserved (compare with Figure 4.9). The average end point error is twice as low as using a simple thresholding procedure. At the same time the data term deviation (gray value difference between current position and gray value at the end point of the flow vector) is larger for most of the image, as one would expect.

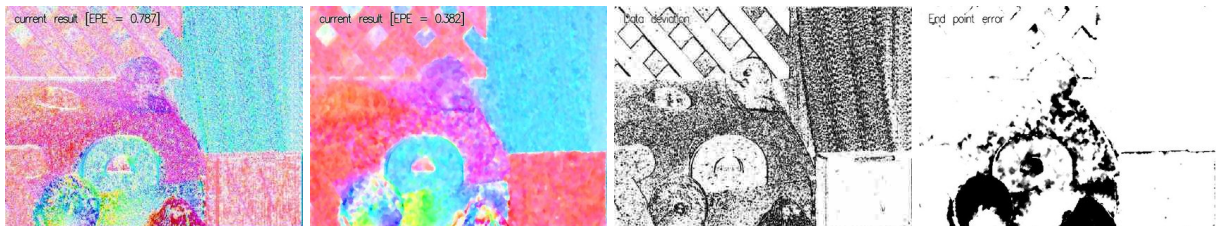


Figure 4.10: Results after convergence of median filtering to the root image. The leftmost image reviews flow result from simple thresholding around the initial zero flow field. The result after convergence of iterative median filtering yields a smoother flow field and smaller flow error (same color scale as in Figure 4.9). This is at the cost of data term deviations.

Notice, that the approach in the toy example uses simple, pixel-wise thresholding and pixel-wise median filtering and can be seen as a local approach. Nevertheless, due to the iterated median filtering message passing does take place, suggesting a touch of globality. Due to the median filter, the presented method is inherently robust against a wide range of outliers. Furthermore it benefits from the easy implementation, compared to the classical approaches of optical flow estimation (KLT [90] and Horn and Schunck [73]).

While small flow vectors are well approximated using this simple approach, large flow vectors, especially present in the lower part of the image, are still not precise. The next section reviews two common techniques around this problem: warping and image pyramids. Furthermore it provides implementation details for the involved numerical scheme. Evaluation results, comparing different data and smoothness terms as well as an evaluation of the Refinement Optical Flow framework on a optical flow benchmark, are found in Section 4.4.

4.2.3 Implementation Details

This section gives details on the implementation for the proposed Refinement Optical Flow framework. Insights on the implementation of the numerical scheme, the restriction and prolongation operators and the used warping scheme are given.

Pyramid Restriction and Prolongation

Recall, that large flow vectors in Figure 4.10, especially present in the lower part of the image, are still not precise. This is mainly due to the linearized flow constraint which does not allow for large flow vectors. Two common approaches are known to solve this problem; image pyramids and warping [45]. image pyramids use down sampled versions of the image to find larger flow vectors whereas warping linearizes the optical flow constraint using the derived solution as a new starting point. Therefore pyramid approaches inherently use warping to propagate the results between pyramid levels.

In this thesis, image pyramids with a scale factor of 2 are used. Propagating results from a lower pyramid level, say 640×480 px, to a higher pyramid level, say 320×240 px is called a restriction operation. Restriction has to be performed on the gray value input images in order to assemble the image pyramid. The restriction operator is a combination of a low pass 5×5 Gaussian filter and subsequent down-sampling [126]. That is, a Gaussian filter with $\sigma = 1$,

$$\frac{1}{16} \begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix} \times \frac{1}{16} [1 \ 4 \ 6 \ 4 \ 1] = \frac{1}{256} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix},$$

is applied to the image. Then odd rows and columns are removed from the image, yielding the lower resolved pyramid image (note, that such procedure does require the size of the input image to be a power of 2 times the size of the lowest resolved image).

For the opposite direction, flow vectors and dual variables have to be transformed from the higher pyramid level onto the lower pyramid level. This operation is called prolongation. The prolongation operator up-samples the image, that is, inserts odd zero rows and columns, and then applies the 5×5 Gaussian filter multiplied by 4 to it. Here, one must differentiate between up-sampling of the flow vectors \mathbf{u} , which have to be multiplied by a factor of 2, and up-sampling of the dual variable \mathbf{p} . The dual variable \mathbf{p} is not multiplied by a factor. Instead, Dirichlet boundary conditions are enforced by first setting the border of the dual variable to 0 and then up-sampling the dual variable.

Optical flow results from upper pyramid levels (e.g. 320×240 px) are a good approximation for the optical flow on lower pyramid levels (e.g. 640×480 px). Hence, they can be passed to the

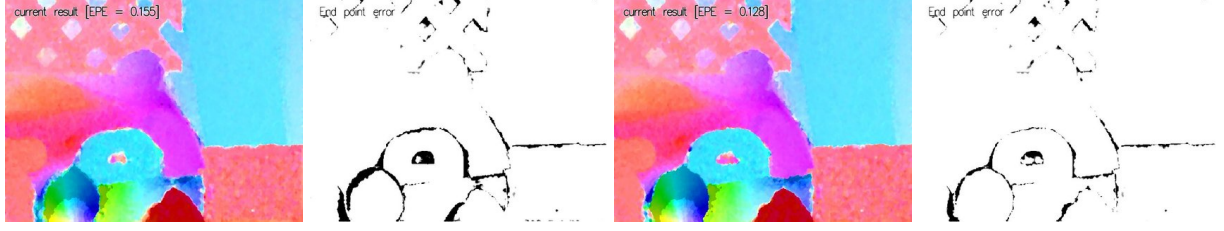


Figure 4.11: The two left images show the results of the median filtered optical flow on an image pyramid. The two right images have been achieved using additional warps on each pyramid level. Both approaches further improve the derived optical flow field from 0.38 px average end point error (as in Figure 4.10) to 0.155 px and 0.128 px average end point error, respectively.

Refinement Optical Flow data term evaluation step as the approximate solution \mathbf{u}' . However, in order to allow for the estimation of large flow vectors and not to get stuck in local minima of the gray value intensity function, the optical flow constraint has to be evaluated in the vicinity of the approximate solution. This process is called warping.

Re-sampling the Coefficients of the Optical Flow Data Term via Warping

To compute the image warping, the image I_1 is linearized using the first order Taylor approximation near $\mathbf{x} + \mathbf{u}_0$ (instead of solely x), where \mathbf{u}_0 is a given (approximate) optical flow map:

$$I_1(\mathbf{x} + \mathbf{u}) = I_1(\mathbf{x} + \mathbf{u}_0) + \nabla I_1(\mathbf{x} + \mathbf{u}_0) (\mathbf{u} - \mathbf{u}_0) .$$

The data fidelity term $\rho(\mathbf{u})$ now reads

$$\rho(\mathbf{u}) = \underbrace{I_1(\mathbf{x} + \mathbf{u}_0) - I_0(\mathbf{x}) - \nabla I_1(\mathbf{x} + \mathbf{u}_0) \mathbf{u}_0}_c + \nabla I_1(\mathbf{x} + \mathbf{u}_0) \mathbf{u}$$

where the left part, denoted by c , is independent of \mathbf{u} , and hence fixed. Commonly, bi-linear or bi-cubic look-up is used to calculate the intensity value $I_1(\mathbf{x} + \mathbf{u}_0)$ and the derivatives of I_1 . The derivatives on the input images are approximated using the five-point stencil $\frac{1}{12} \begin{bmatrix} -1 & 8 & 0 & -8 & 1 \end{bmatrix}$. If the bi-cubic look-up falls onto or outside the original image boundary, a value of 0 is returned for the derivative and the gray value.

Toy Example cont.: Pyramids and Warping

Figure 4.11 shows the progress in terms of optical flow end point error when applying firstly only a pyramid approach and secondly additional warps on each pyramid level. 5 pyramid levels and 10 warps on each pyramid level were used for this toy example. The decrease in end point error becomes visible as more sophisticated methods are used for the optical flow estimation.

Symmetric Gradients for the Data Term Evaluation

Assuming that \mathbf{u}_0 is a good approximation for \mathbf{u} , the optical flow constraint states that $I_0(\mathbf{x}) \approx I_1(\mathbf{x} + \mathbf{u}_0)$. Taking this further onto image derivatives, we obtain that $\nabla I_0(\mathbf{x})$ is a good approximation for $\nabla I_1(\mathbf{x} + \mathbf{u}_0)$. Note, that replacing $\nabla I_1(\mathbf{x} + \mathbf{u}_0)$ with $\nabla I_0(\mathbf{x})$ implies that no bi-cubic look-up for the image gradients has to be employed and the computation time can be sped up. However, it turns out that using blended versions of the derivatives larger flow vectors can be matched and hence even better results are achieved. Figure 4.12 reveals that the accuracy for blended versions of the derivatives, $\nabla I = (1 - \beta) \nabla I_1(\mathbf{x} + \mathbf{u}_0) + \beta \nabla I_0(\mathbf{x})$, increases (keeping all other parameters fix). Values for β around 0.5 show the best results in terms of optical flow accuracy. This can be explained by the fact that both images contribute to the gradient, increasing the amount of image information used.

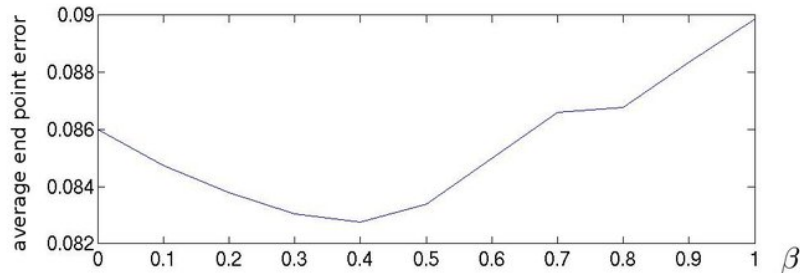


Figure 4.12: The plot shows the optical flow accuracy, measured as the average end point error for different β values for the blending of the gradients from image I_0 and I_1 of the *rubber whale* test sequence. The improvement using a blended version of the gradients for the optical flow computation is visible.

Numerical Scheme

The general numerical implementation for a specific setting of the Refinement Optical Flow framework is as follows. Beginning with the coarsest level, Equation (4.9) is solved at each level of the pyramid, the result (Equation 4.14) is smoothed, and the solution is propagated to the next finer level.

This solution is further used to compute the coefficients of the linearized data terms on the corresponding pyramid levels. Hence, a warping step for the input images takes place every time when the solution is propagated within the pyramid and furthermore additional warps are used on each level to achieve more accurate results. For implementation on modern graphic processing units (GPUs), bi-linear warping is essentially available at no additional cost, therefore the concept of outer iterations is only used on CPU implementations.

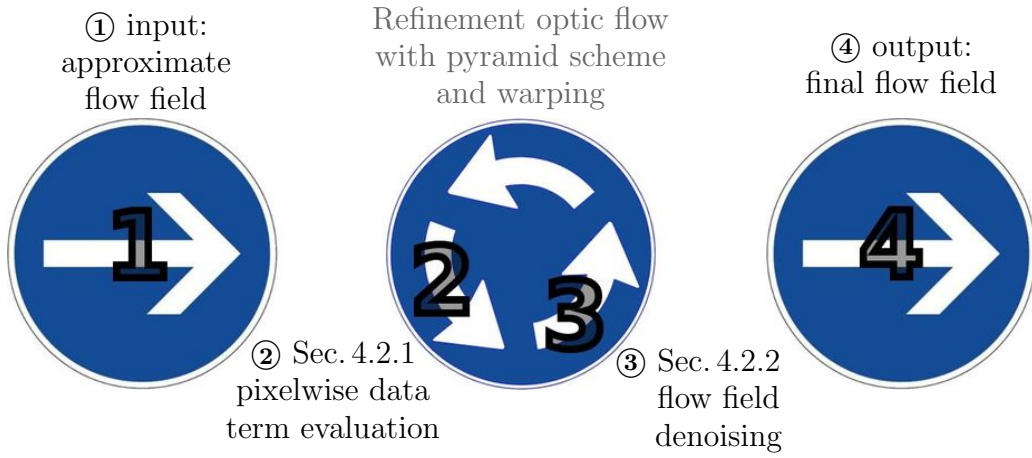
Avoiding poor local minima is not the only advantage of the coarse-to-fine approach. It turns out, that the filling-in process induced by the regularization (smoothness) occurring in texture-less region is substantially accelerated by a hierarchical scheme as well. The resulting numerical scheme is summarized in Algorithm 1.

The next section tackles the problem of illumination artifacts, which were not addressed in the Refinement Optical Flow framework. Using color images, a common way to tackle illumination artifacts is to change the color space representation (e. g. from RGB to HSV) and to work solely on the color itself instead of using illumination. The approach for gray value images needs to be somehow more subtle.

4.3 Increasing Robustness to Illumination Changes

The image data fidelity term states that the intensity values of $I_0(\mathbf{x})$ do not change during its motion to $I_1(\mathbf{x} + \mathbf{u}(\mathbf{x}))$. For many sequences this constraint is violated due to sensor noise, illumination changes, reflections, and shadows. Hence, real scenes generally show artifacts that violate the optical flow constraint. Figure 4.13 shows an example, where the ground truth flow is used to register two images from the Middlebury optical flow benchmark data base [36]. Although the two images are registered at the best using the ground truth flow, the intensity difference image between the source image and the registered target image reveals the violations of the optical flow constraint. Some of these regions, showing artifacts of shadow and shading reflections, are marked by blue circles in the intensity difference image.

A physical model of brightness changes was presented in [70], where brightness change and motion is estimated simultaneously; shading artifacts however have not been addressed. In [134] and [102] the authors used photometric invariants to cope with brightness changes, which requires color images. A common approach in literature to tackle illumination changes is to use image gradients with, or instead of, the plain image intensity values in the data term [45]. This implies that multiple data fidelity terms have to be used and images are differentiated twice, which is known to be noisy.



Input: Two intensity images I_0 and I_1

Output: Flow field \mathbf{u} from I_0 to I_1

Preprocess the input images;

for $L = 0$ to max_level **do**

Calculate restricted pyramid images ${}^L I_0$ and ${}^L I_1$;

end

Initialize ${}^L \mathbf{u} = 0$ and $L = max_level$;

while $L \geq 0$ **do**

for $W = 0$ to max_warps **do**

Re-sample coefficients of ρ using ${}^L I_0$, ${}^L I_1$, and ${}^L \mathbf{u}$; (Warping)

for $Out = 0$ to $max_outer_iterations$ **do**

Solve for ${}^L \mathbf{u}'$ via thresholding; (Section 4.2.1)

for $In = 0$ to $max_inner_iterations$ **do**

Smoothness term iteration on ${}^L \mathbf{u}'$; (Section 4.2.2)

end

end

end

if $L > 0$ **then**

Prolongate ${}^L \mathbf{u}$ to next pyramid level $L - 1$;

end

end

Algorithm 1: Numerical scheme of the *Refinement Optical Flow* algorithm. In the numerical scheme, a super-scripted L denotes the pyramid level.

An alternative is to employ a structure-texture decomposition similar to the approach used in [131] in order to remove the intensity value artifacts due to shading reflections and shadows. The basic idea behind this splitting technique is that an image can be regarded as a composition of a structural part, corresponding to the main large objects in the image, and a textural part, containing fine scale-details [32]. See Figure 4.14 for an example of such a structure-texture decomposition, also known as cartoon-texture decomposition. The expectation is, that shadows show up only in the structural part which includes the main large objects.

The structure-texture decomposition is accomplished using the total variation based image denoising model of Rudin, Osher and Fatemi [114]. For the intensity value image $I(\mathbf{x})$, the structural part is given by the solution of

$$\min_{I_S} \int_{\Omega} \left\{ |\nabla I_S| + \frac{1}{2\theta} (I_S - I)^2 \right\} d\mathbf{x}. \quad (4.23)$$

The textural part $I_T(\mathbf{x})$ is then computed as the difference between the original image and its denoised version, $I_T(\mathbf{x}) = I(\mathbf{x}) - I_S(\mathbf{x})$.

Figure 4.15 shows the intensity difference images between the source image and the registered target image using the ground truth flow as look-up for both, the original image and its decomposed parts. For most parts the artifacts due to shadow and shading reflections show up in the original image and the structural part. The intensity value difference in the textural part, which contains fine-scale details, is noisier than the intensity value difference in the structural part. These intensity value differences are mainly due to sensor noise and sampling artifacts while shadow and shading reflection artifacts have been almost completely removed. This is best visible in the area of the punched hole of the rotated D-shaped object.

This observation leads to the assumption that the computation of optical flow using the textural part of the image is not perturbed by shadow and shading reflection artifacts, which cover large image regions. To prove this assumption experimentally, a blended version of the textural part is used as input for the optical flow computation, $I_T(\alpha, \mathbf{x}) = I(\mathbf{x}) - \alpha I_S(\mathbf{x})$.

Figure 4.16 shows the accuracy for optical flow computation using a fixed parameter set and varying the blending factor α . The plot reveals that for larger values of α the accuracy of the optical flow is 50% better than using a small value for α . This confirms the assumption that removing large perturbations due to shadow and shading reflections yields better optical flow estimates. In the experiments, the image decomposition is computed as follows:

The original source and target images are scaled into the range $[-1, 1]$ before computing the structure part. Proposition 2 is then used to solve Eq. (4.23). A good choice of the parameters is $\alpha = 0.95$, $\theta = 0.125$ and 100 for the number re-projection iterations.

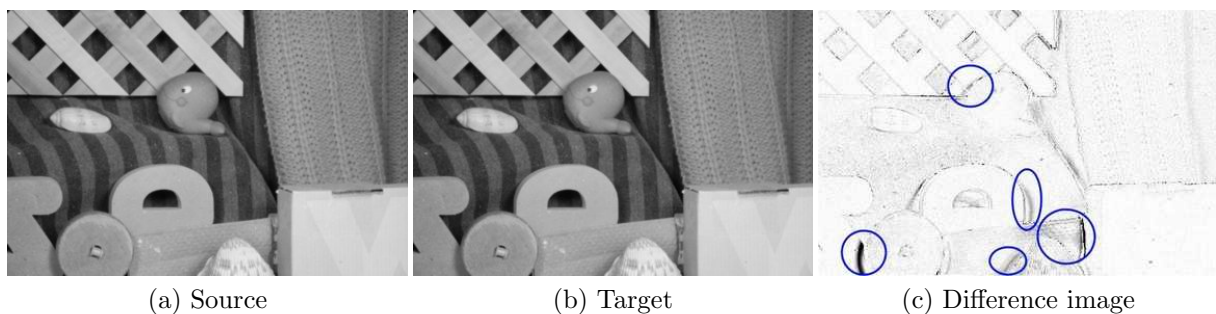


Figure 4.13: The source and target images of the *rubber-whale* sequence in the Middlebury optical flow benchmark have been registered using the ground truth optical flow. Still, intensity value differences are visible due to sensor noise, reflections, and shadows. The intensity difference image is encoded from white (no intensity value difference) to black (10% intensity value difference). Pixels which are visible in a single image due to occlusion are shown in white.

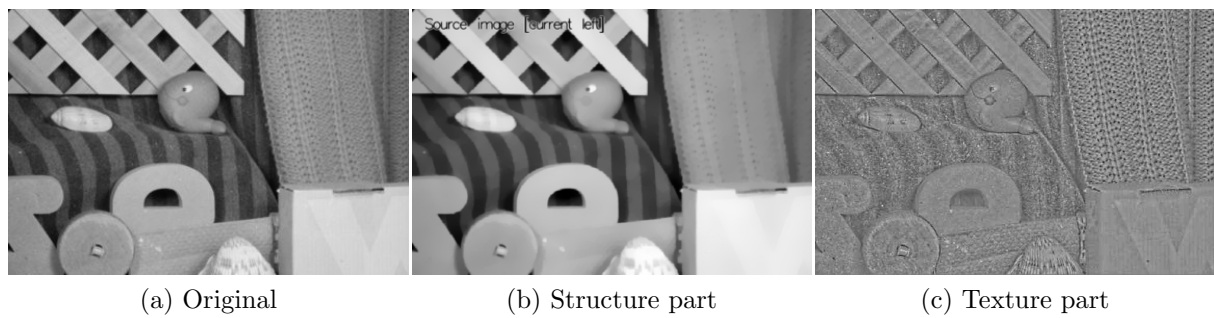


Figure 4.14: The original image is decomposed into a structural part, corresponding to the main large objects in the image, and a textural part, containing fine-scale details. All images are scaled into the same intensity value range after decomposition.

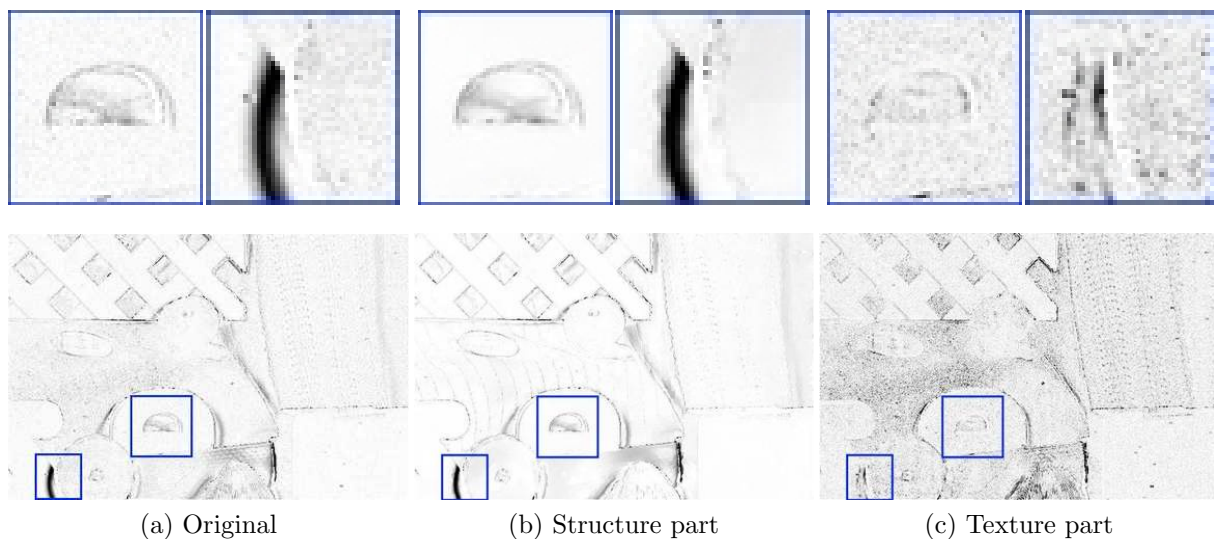


Figure 4.15: Intensity difference images between the source image and the registered target image using ground truth optical flow for the original image pairs and their structure-texture decomposed versions (intensity coding as in Figure 4.13). Note the presence of shading reflection and shadow artifacts in the original image and in the structure image.

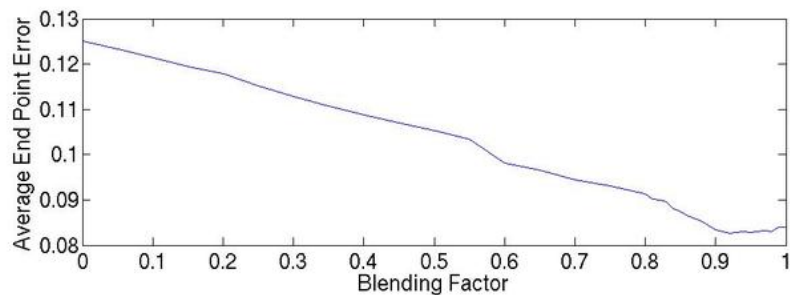


Figure 4.16: The plot shows the optical flow accuracy, measured as the average end point error, using different α values for the blending of the textural part of the image. The improvement using the textural part for the optical flow computation becomes visible.

4.4 Experimental Results

In this section, specific settings of the Refinement Optical Flow framework, derived in Section 4.2, are quantitatively evaluated based on the Middlebury optical flow benchmark [36]. The benchmark provides a training data set where the ground truth optical flow is known and an evaluation set used for a comparison against other algorithms in literature. A subset of the investigated data terms and smoothness terms in this section have been uploaded for this comparison evaluation.

In a second experimental part, the TV-L1-improved version [12] of the *Refinement Optical Flow* algorithm is evaluated on real scenes, taken from a moving vehicle. The results demonstrate the performance of the optical flow algorithm under different illumination conditions and under large image motion. For visualization of the flow vectors the color coding scheme shown in Figure 4.2 is used.

4.4.1 Quantitative Evaluation of the Refinement Optical Flow

Table 4.3 demonstrates the accuracy of the optical flow field for different settings of the Refinement Optical Flow framework on the training data set. The table shows the average end-point error between the ground truth flow vectors and the estimated flow vectors for the data sets in the training set.

The table has three sections, a *Performance* section where the focus is set on real-time optical flow, a *Smoothness Filters* section where the different denoising algorithms presented in Section 4.2.2 are systematically evaluated, and an *Accuracy* section which demonstrates the accuracy gain using the additional adaptive fundamental prior term (see Appendix A) and weighted total variation in the smoothing. The last row shows the average deviation from the epipolar lines, hence reflecting the amount of *dynamics* within the scene.

Performance. The performance section compares real-time capable implementations for optical flow. Both, the TV-L¹ optical flow algorithm (Equation (4.5)) and the image decomposition described in Section 4.3, employ the TV-L² denoising algorithm. This denoising step can be efficiently implemented on modern graphics cards, putting up with small accuracy losses: For parallel processing, the iterative TV-L² denoising is executed on sub-blocks of the image in parallel, where boundary artifacts may occur. Hence, high accuracy is exchanged versus run-time performance (see Table 4.2).

In all three algorithm settings, P, P-MF, and P-I_T-MF, the linearized optical flow constraints (4.1) is used as data term. In the plain version, algorithm P, 5 iterations of the TV-L² denoising are used to smooth the flow field in every warping step. The number of refinement warps on every pyramid level was set to 25. The parameter settings are $\lambda = 25$ and $\theta = 0.2$. Gray value look-up is bi-linear, as this can be done without additional costs on modern graphics cards. The image gradient is computed via central derivatives from the average of both input images.

	Algorithm	Processor	Avg. Accuracy	Run-time
Performance	P(GPU)	NVidia® GeForce® GTX 285	0.486	0.039 [sec]
	P	Intel® Core™2 Extreme 3.0GHz	0.468	0.720 [sec]
	P-MF(GPU)	NVidia® GeForce® GTX 285	0.416	0.055 [sec]
	P-MF	Intel® Core™2 Extreme 3.0GHz	0.375	0.915 [sec]
	P-I _T -MF(GPU)	NVidia® GeForce® GTX 285	0.408	0.061 [sec]
	P-I _T -MF	Intel® Core™2 Extreme 3.0GHz	0.361	1.288 [sec]

Table 4.2: Run-time comparison for the **Performance** section in Table 4.3. Using the parallel power of a GPU yields performance gain at the cost of accuracy loss. The run-time is measured on the *Grove3* test image (640×480 px).


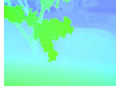






									
		Dimetrodon	Grove2	Grove3	Hydrangea	RubberWhale	Urban2	Urban3	Venus
Performance	P(GPU) [150]	0.259	0.189	0.757	0.258	0.218	0.652	1.069	0.482
	P [150]	0.236	0.190	0.803	0.240	0.302	0.598	0.897	0.486
	P-MF(GPU)	0.224	0.173	0.671	0.251	0.183	0.508	0.889	0.433
	P-MF	0.202	0.161	0.666	0.236	0.161	0.468	0.679	0.428
	P-I _T -MF(GPU)	0.186	0.200	0.743	0.186	0.118	0.487	1.026	0.314
	P-I _T -MF	0.171	0.191	0.730	0.173	0.109	0.390	0.812	0.311
Smoothness Filters	TV-L ²	0.196	0.188	0.690	0.153	0.094	0.354	0.831	0.279
	Felsberg	0.202	0.192	0.728	0.181	0.115	0.455	0.838	0.315
	2nd Order	0.201	0.179	0.676	0.171	0.099	0.384	0.911	0.290
	TV-L ¹ (0.25)	0.201	0.217	1.109	0.146	0.148	0.402	0.780	0.299
	TV-L ¹ (1.0)	0.238	0.218	0.73	0.178	0.117	0.547	1.058	0.352
	Median (2)	0.192	0.205	0.814	0.198	0.135	1.343	1.156	0.380
	Median (20)	0.162	0.237	0.720	0.219	0.148	1.045	1.058	0.363
	Median (200)	0.307	0.258	0.704	0.231	0.161	0.736	1.441	0.421
Accuracy	TV-L ¹ -imp. [12]	0.190	0.154	0.665	0.147	0.092	0.319	0.630	0.260
	F-TV-L ¹ [10]	0.284	0.152	0.649	0.484	0.170	0.288	0.487	0.256
	∇ I-TV-L ¹	0.195	0.152	0.580	0.152	0.084	0.322	0.595	0.260
	Adaptive [11]	0.195	0.146	0.555	0.152	0.084	0.296	0.457	0.250
	Adapt. comb. [11]	0.196	0.145	0.556	0.153	0.082	0.297	0.446	0.252
rel- $\rho_{\mathbf{F}}$		0.108	0.009	0.022	0.214	0.286	0.012	0.009	0.008

Table 4.3: Evaluation results on the Middlebury training data. The evaluation is grouped into a real-time *Performance* section, a *Smoothness Filter* comparison section, and an *Accuracy* section, which demonstrates the systematic accuracy gain of the flow field using the adaptive fundamental matrix prior and gradient driven smoothing. The table shows the average end point error of the estimated flow fields; apart from the last row where $\text{rel-}\rho_{\mathbf{F}} = \int_{\Omega} \rho_{\mathbf{F}}(\mathbf{u}, \mathbf{x}) / \|\mathbf{u}\| d^2\mathbf{x}$ is the average relative epipolar line distance. Parameters have been carefully chosen for algorithm comparison (see text for parameters and run-time).

The P-MF algorithm extends the basic algorithm by an additional Median filter step, hence 5 iterations of the $TV-L^2$, followed by a median filter step, are performed for each warp. The Median filter makes the whole scheme more robust against outliers. For this reason the influence of the data term, weighted by λ can be increased to $\lambda = 50$. All other parameters are kept fix.

In the third algorithm, P-T_I-MF, the textural part of the image is used, as described in Section 4.3. Again the increase of accuracy at the cost of a longer execution time can be seen in the quantitative evaluation. It is interesting to note that only the flow fields for the *real scenes* within the test set benefit from the image decomposition. The optical flow for the *rendered scenes*, Grove and Urban, is actually worse. This is not surprising as texture-extraction removes some structure information in the images; such a procedure is only beneficial if the images contain illumination artifacts. Because this is the fact for all natural scenes (which are for obvious reasons more interesting and challenging), in the remaining experiments the texture-structure decomposition is performed inherently.

Smoothness Filters. The smoothness filter section quantitatively evaluates different smoothness filters from Section 4.2.2. The parameter setting is carefully chosen to be fixed wherever possible, allowing one to compare the different smoothness techniques amongst each other. An essential difference to the performance section is the preparation of the input images. The two input images are decomposed into their structural and textural part as described in Section 4.3. Then, the texture input images are scaled into the range $[-1, 1]$, yielding the same gray value range for every input test image. Note, that consequently this scaling is performed before **and** after the decomposition of the images because the gray value range of the texture image may be different from the range of the original image. Obviously such procedure needs the minimum and maximum of an image and is not well suited for real-time implementation on a graphics card.

Throughout the experiments the parameter settings are $\lambda = 30$ (except for the median filter where λ is given in the brackets) and $\theta = 0.25$ ($TV-L^1$ also $\theta = 1$, see brackets). The number of refinement warps is kept fix at 35 and bi-cubic look-up is used for the interpolation method. The number of outer and inner iterations is the only varying parameter as some algorithms need more inner iterations than others. Gradients are computed using the five-point gradient mask with a blending factor of $\beta = 0.4$. Table 4.4 examines the advantages and disadvantages of the single denoising filters. In summary, $TV-L^2$ denoising does yield the best results while the Median filter shows tremendous robustness against outliers. A combination of these two techniques is investigated in more detail in the third set of results, the Accuracy section.

Accuracy. In this section the $TV-L^2$ denoising with subsequent Median filtering is carefully examined. More precisely, for all 35 warps in every outer iterations (5 total), one $TV-L^2$ step followed by one Median filter step is performed on the flow field. Parameter settings are as in the Smoothness Filter section, in particular $\lambda = 30$ and $\theta = 0.25$. These settings describe the $TV-L^1$ -improved algorithm.

Keeping all parameters fix and adding an additional fundamental matrix prior data term (see Appendix A) yields the F-TV- L^1 algorithm. The ∇I -TV- L^1 setting uses the structure-aware smoothness term. Last, *Adaptive* uses a combination of both, adaptive fundamental matrix prior and structure weighting. The *Adaptive combined* approach further combines the denoising of both optical flow variables (re-projecting using the length of the sum of all four respective dual variables).

Evidently the non-adaptive fundamental matrix prior increases accuracy in static scenes but worsens the results if the scene is dynamic. The structure-aware regularization does improve the optical flow accuracy on most test examples but only the combined adaptive approach yields top performing results. Run-times in the table on the right is given for optical flow estimation on the *Grove3* test image (640×480 px).

Algorithm	Run-time
TV- L^1 -imp.	3.46 [sec]
F-TV- L^1	7.13 [sec]
∇I -TV- L^1	5.23 [sec]
Adaptive	8.87 [sec]
Adapt. comb.	8.72 [sec]

	Algorithm	Avg. EPE	Run-time	Summary
Smoothness Filters	TV-L ²	0.348	2.17 [sec]	The TV-L ² is not only the fastest but also the most accurate smoothing technique out of all evaluated techniques. Although part of the run-time performance depends on the implementation, this fact is a major advantage of TV-L ² denoising. Furthermore, it is the simplest algorithm besides the (trivial) Median filter. For the algorithm the number of outer iterations is 5; in every outer iteration one inner iteration is performed. Disadvantages are the forming of regions with constant displacement and the negative effect of outliers in the data term when increasing λ .
	Felsberg	0.378	5.94 [sec]	The numerical approximation scheme for anisotropic diffusion described by Felsberg in [62] is well suited to denoise the optical flow field. It has the advantage that the iterative diffusion process does handle outliers robustly. However, parameter tuning is complicated and more investigation is necessary to formulate intelligent stopping criteria for the diffusion process. Here a fixed number of iterations was used: 1 outer and 5 inner iterations.
Smoothness Filters	2nd Order	0.364	61.5 [sec]	For the second order prior suggested in [131] 10 inner iterations are performed for each of the 5 outer iterations per warp. The large number of iterations yields poor run-time results. However, the resulting flow field is fairly accurate and smooth with some ringing artifacts at object boundaries. Main disadvantage is the inherent numerical instability; coupling the second order denoising with e.g. a Median filter leads to diverging effects and hence wrong flow fields.
Smoothness Filters	TV-L ¹ (0.25)	0.413	3.95 [sec]	Being relatively stable (changing λ has low effect on the results), the TV-L ¹ has an inherent advantage over other denoising methods. Parameter settings are: 5 outer iterations with one inner iteration each. Although the method is robust w. r. t. outliers it seems to over-smooth the results. This can be anticipated by increasing the influence of the data term, hence λ (with low effect) or by increasing θ and allowing a larger deviation of the data term from the smooth solution. Latter does yield better results for some test scenes but the overall performance is still far from the TV-L ² denoising filter. Furthermore computational time and memory consumption is larger due to the more complicated minimization process.
	TV-L ¹ (1.0)	0.429		
	Median (2)	0.553	3.21 [sec]	The Median filter is the simplest denoising method out of all in this evaluation. It is also the worst in terms of accuracy. The robustness however is tremendous. For different settings of λ the results do not change significantly. This is not surprising as Median filtering removes outliers independent of their distance to the current flow field. The run-time is slow because values within every 3×3 window need to be sorted prior to filtering. Median filtering does not result in interpolated values in order to exactly reconstruct inclines in the flow field but it does add robustness to a large range of λ . In the experiments one Median filter step and 5 outer iterations are used.
Median (20)	0.494			
Median (200)	0.533			

Table 4.4: Summary of the evaluation for different smoothness filters shown in Table 4.3. TV-L² denoising does yield the best results while the Median filter shows tremendous robustness against outliers. The given run-time is measured on the *Grove3* test image (640×480 px).

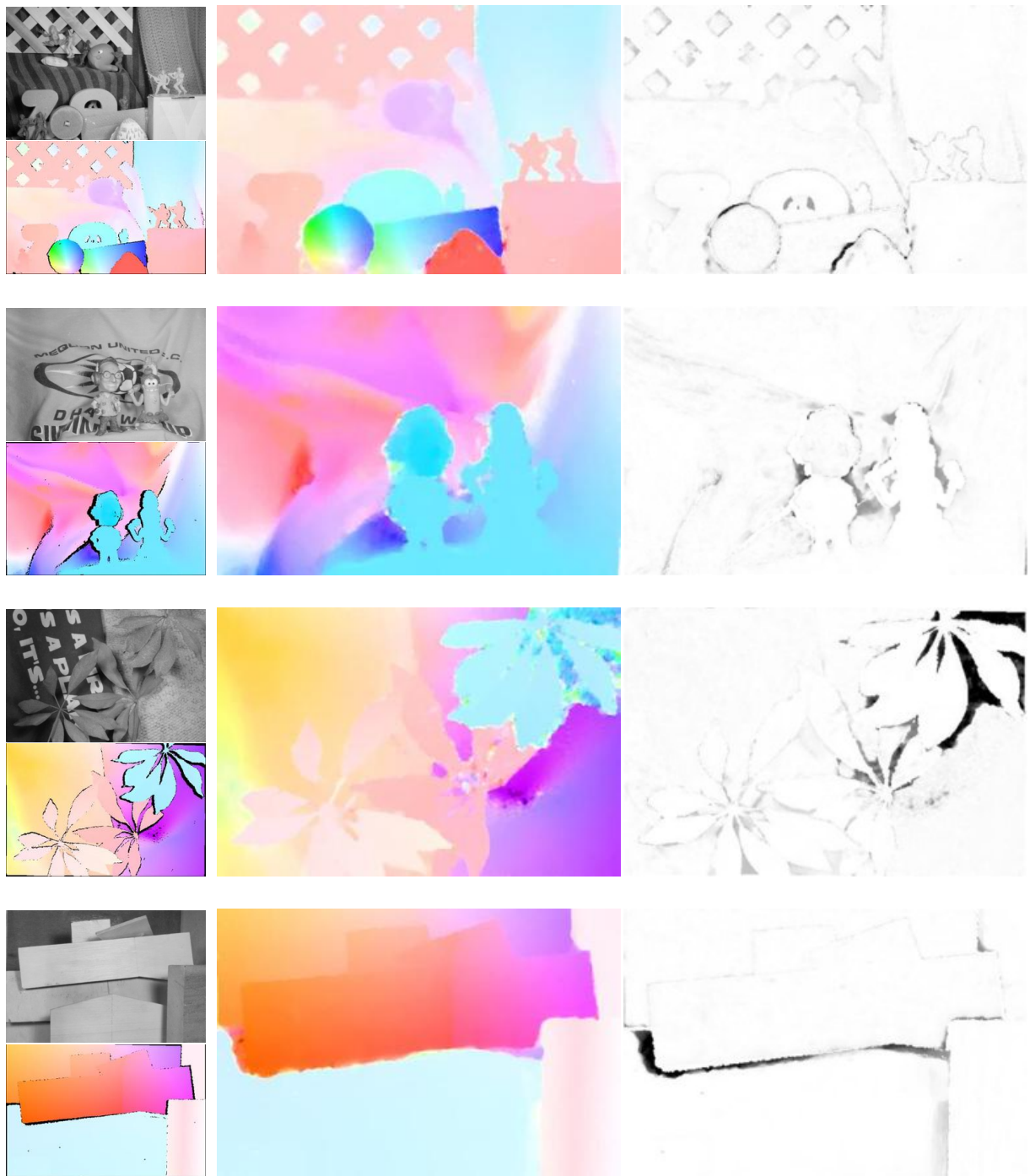


Figure 4.18: Optical flow results for the *army*, *mequon*, *schefflera*, and *wooden* sequence of the Middlebury flow benchmark. The left images show the first input image and the ground truth flow. The middle image shows the optical flow using the proposed algorithm. The right image shows the end point error of the flow vector, where black corresponds to large errors.

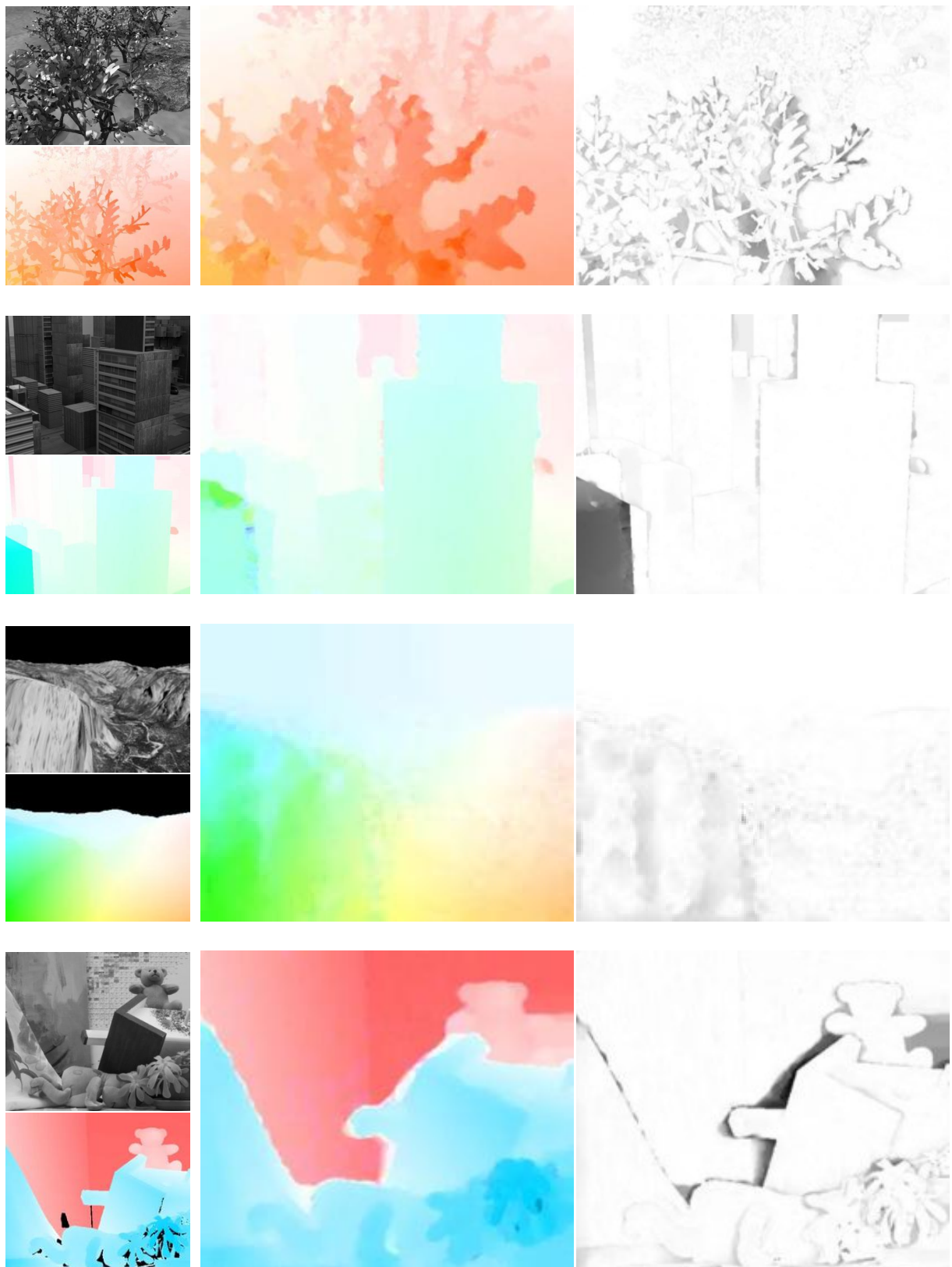


Figure 4.18: Optical flow results (cont.) for the *grove*, *urban*, *yosemite*, and *teddy* sequence of the Middlebury flow benchmark. The left images show the first input image and the ground truth flow. The middle image shows the optical flow using the proposed algorithm. The right image shows the end point error of the flow vector, where black corresponds to large errors.

4.4.2 Results for Traffic Scenes

The computation of optical flow is important to understand the dynamics of a scene. This section evaluates the optical flow estimation in different scenarios under different illumination conditions (night, day, shadow). The images are taken from a moving vehicle where the camera monitors the road ahead.

The first experiment in Figure 4.19 shows the optical flow computation on an image sequence with a person running from the right into the driving corridor. Due to illumination changes in the image (compare the sky region for example) and severe vignetting artifacts (images intensity decreases circular from the image middle), flow computation using plain gray value intensities fails. Using the proposed structure-texture decomposition, a valid flow estimation is still possible. See Figure 4.20 for the same two frames where optical flow is computed on the structure-texture decomposed images. Note the reflection of the moving person on the engine hood which is only visible in the structure-texture decomposed images. What is more important, artifacts due to vignetting and illumination change are not visible in the structure-texture decomposed images. This demonstrates the increase in robustness for the optical flow computation under illumination changes using the proposed decomposition of the input images.

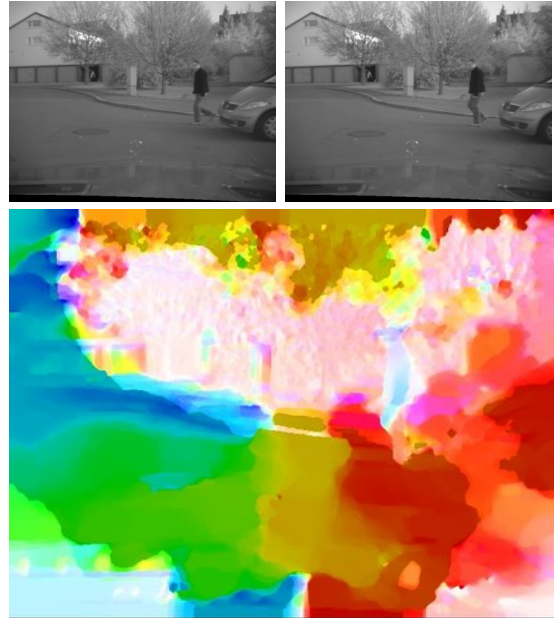


Figure 4.19: Optical flow computation with illumination changes. Due to illumination changes, the optical flow constraint in the two input images (*upper images*) is violated and flow computation using plain gray value intensities fails.

Figure 4.21 shows the same scene a few frames later using the additional fundamental matrix data term. Most of the image, except for the running person, is static and the expanding flow field should follow the epipolar rays. The results show that, except for the Mercedes star and the running person, this assumption holds. It is not in the scope of this chapter to segment moving objects, the results however are well suited to detect independently moving regions of the image. There are more constraints available than just the *distance to epipolar rays constraint* to detect moving objects (see [2]); segmentation approaches will be discussed in Chapter 6.



Figure 4.20: Optical flow computation with illumination changes. Using the structure-texture decomposed images, a valid flow estimation is now possible (compare with Figure 4.19).



Figure 4.21: Optical flow for a scene with a running person and a moving camera, installed in a vehicle. The distance to the epipolar rays encodes independently moving objects.

Another example to illustrate the robustness of the structure-texture decomposition is shown in Figure 4.22. Here a scene at night with reflections on the ground plane is used. In the intensity images the scene is very dark and not much structure is visible. The structure-texture decomposed images reveal much more about the scene. Note, that this information is also included in the intensity image but most structure in the original images is visible in the cloud region. The figure shows the optical flow using the decomposed images. Note the correct flow estimation of the street light on the left side.

The next examples in Figure 4.23 demonstrates the accurate optical flow computation for large displacements. It shows a scene with shadows on the road. Clearly, the structure-texture decomposed images reveal the structure on the road surface better than the original intensity images (the shadows are still noticeable because a blended version of structure and texture is used as presented in Section 4.3). Different scales for the optical flow color scheme are used to demonstrate the accuracy of the optical flow algorithm. Although nothing about epipolar geometry is used in the flow algorithm, the effect of expansion (and hence depth) corresponding to flow length becomes visible. Note, that optical flow for the reflection posts is correctly estimated even for flow length above 8px. Optical flow is correctly estimated for the road surface up to 30px. The shadows in the scene have no negative impact on the flow calculation. The engine hood acts like a mirror and optical flow on the engine hood is perturbed due to reflections. Although the optical flow for the engine hood is very much different from flow vectors on the road surface, this has no negative impact on the estimation of the optical flow for the road surface. Note the accurate flow discontinuity boundary along the engine hood and in the tree regions.

In Figure 4.24 the image is taken while driving below a bridge on a country road. Note, that the shadow edge of the bridge is visible in the original images but not in the decomposed images. The large flow vectors on the reflector post are correctly matched.

The figure also illustrates the limits of the presented Refinement Optical Flow approach. In the vicinity of the car optical flow is perturbed due to missing texture on the road surface. Due to the dependency on the linearized optical flow constraint, optical flow in texture-less regions and in regions with very large displacements is still not satisfactory, highlighting the need of further research.

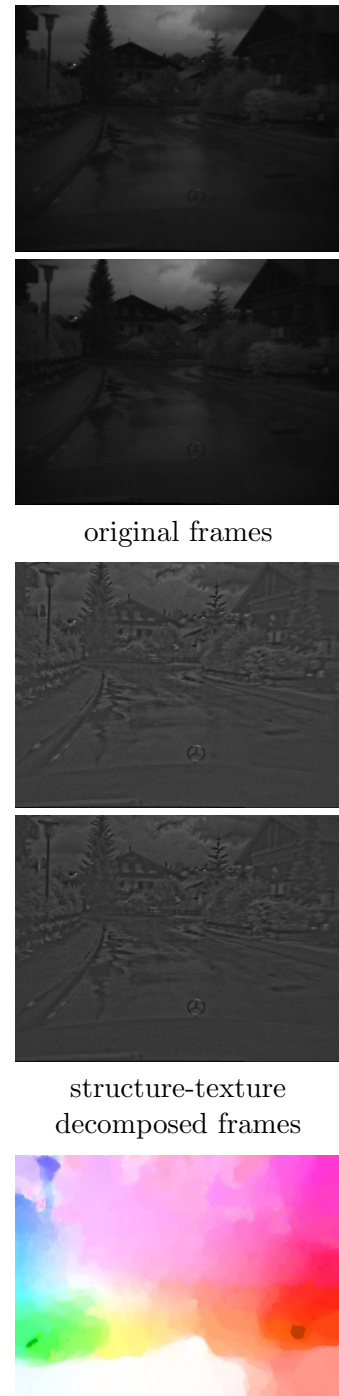


Figure 4.22: Optical flow result for a night scene.

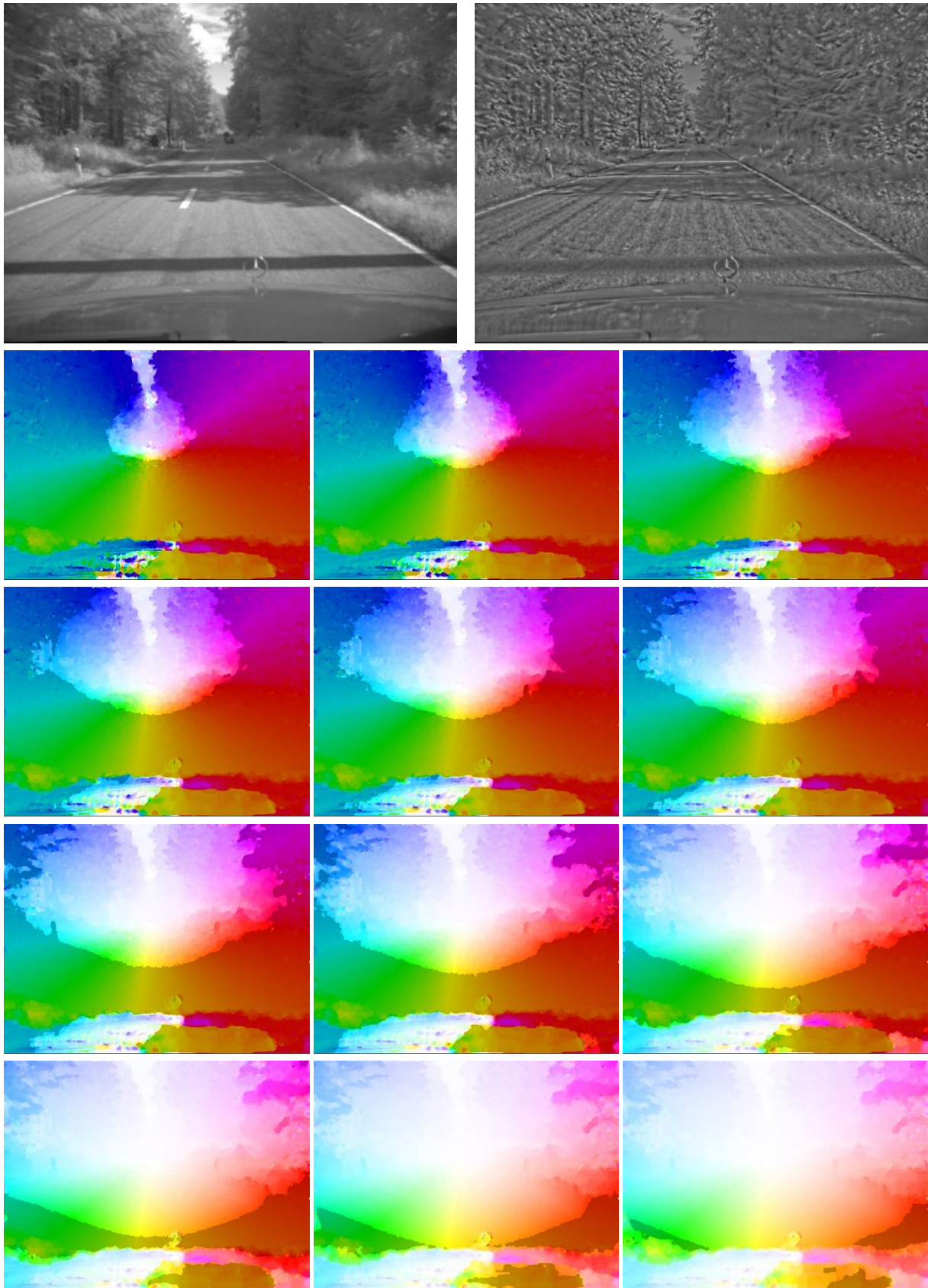


Figure 4.23: Optical flow field for the scene depicted in the upper left with the original and structure-texture image. The flow is saturated for flow vector length above 1, 2, 3, 4, 5, 6, 8, 10, 15, 20, 25, 30 pixels from left to right.

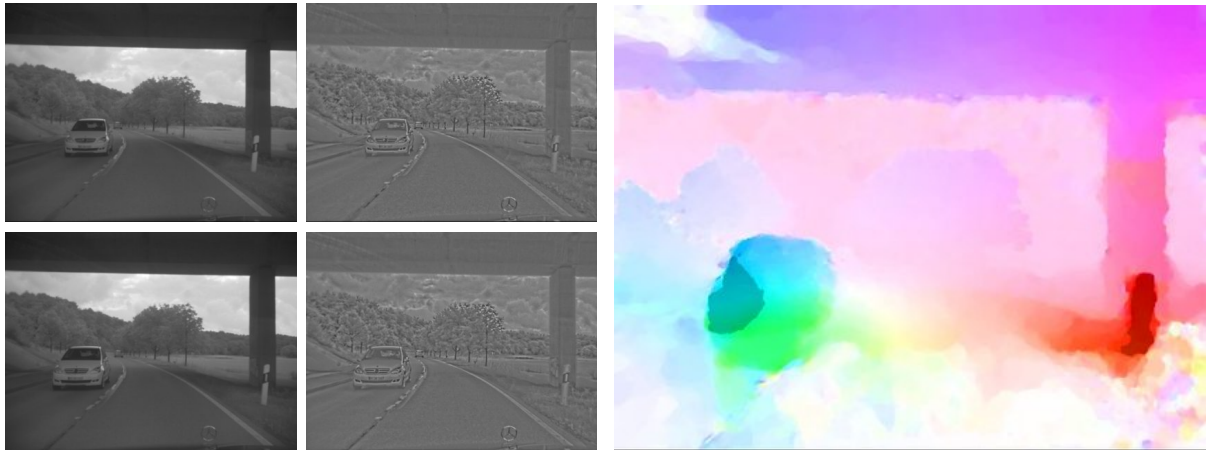


Figure 4.24: The scene shows the computation of optical flow with large displacement vectors. The original input images are shown on the *left*. The *middle* images are the blended structure-texture images. Flow vectors above 20px are color-saturated in the optical flow color image.

4.5 Ideas for Future Research

This chapter presented a novel framework for optical flow computation which I called *Refinement Optical Flow*. It is based on the idea of iterative data term evaluation and denoising to derive a highly accurate optical flow field.

In the data term step, the brightness constancy constraint and a fundamental matrix constraint were evaluated and a flow field in the vicinity of a given flow field was derived. Future research may include additional constraints or extensions of the proposed method, e. g. the use of color images. The fundamental matrix constraint can be used to segment the image into regions of different motions where a fundamental matrix may be estimated for each region independently (as in [130, 139]).

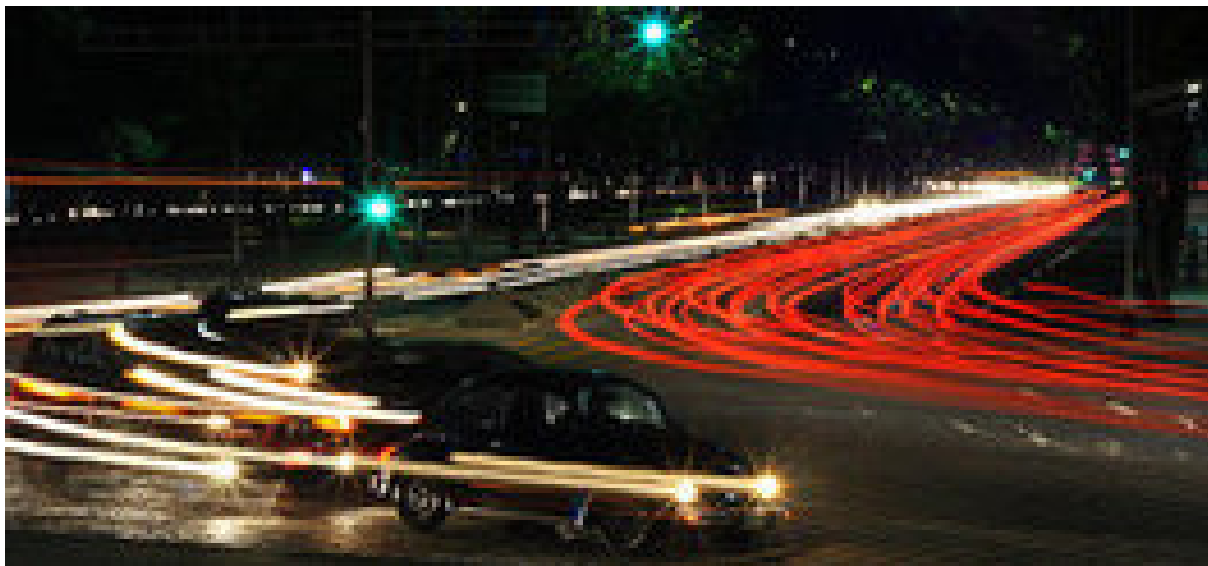
A promising idea which is pursued in literature lately is the use of high-level information to steer the optical flow field. The brightness constancy constraint has the inherent weakness that it only allows one to recover relatively small flow vectors. In [44] variational approaches and descriptor matching is combined for optical flow computation. Such descriptor matching methods do not suffer from local minima in the brightness constancy assumption. In Section 4.1.1 a Census based optical flow method was presented, which does yield optical flow via descriptor matching [124]. An interesting research idea is to include flow vectors from the Census based method in the data term evaluation step.

A promising investigation topic is statistical learning for optical flow regularization [127]. So far these have not been able to outperform the more naive approaches. Of course it is hard to say why this is the case, one reason may be that the challenge of learning “typical” flow patterns may not be feasible, given that different image structures, unknown object deformations, and camera motions may give rise to a multitude of motion patterns. Including such an approach as a data term in the presented Refinement Optical Flow framework and careful comparison of this method with other regularizers may provide answers.

The decomposition of the input image into texture and structure as presented in Section 4.3 is not the only method to derive a texture image. In [85] for example, optical flow is computed on the difference image between the original image and its Gaussian-filtered version. Also first experiments show, that the here presented TV- L^2 denoising outperforms other approaches, in general any smoothing operation can be used to generate a *texture image*. Hence more research needs to be done to prove this statement.

Last, occlusion artifacts and initialization with previously computed flow fields (on the same sequence) have not been addresses in this chapter. Both topics can substantially improve the quality of the optical flow estimation and provide areas for future research.

Scene Flow



The speed of light is too fast to be captured.

© under the creative commons license, Esparta, www.flickr.com

Contents

5.1	Introduction and Related Work	58
5.2	Formulation and Solving of the Constraint Equations	59
5.2.1	Stereo Computation	60
5.2.2	Scene Flow Motion Constraints	61
5.2.3	Solving the Scene Flow Equations	62
5.2.4	Visualizing the 3D Velocity Field	63
5.3	Comparison of Results for Different Stereo Inputs	63
5.4	Derivation of a Pixel-wise Accuracy Measure	66
5.5	Ideas for Future Research	75

This chapter presents an approach for estimating the three-dimensional velocity vector field that describes the motion of each visible scene point. This motion is the three-dimensional analogy to the optical flow presented in Chapter 4 and is called the *scene flow*. The main contribution is to decouple the image position (x , y , and disparity) and image velocity (optical flow and change in disparity) estimation steps, and to estimate dense image velocities using a variational approach. The decoupling strategy has two benefits: Firstly, the algorithm is independent from the disparity estimation technique, which can yield either sparse or dense correspondences, and secondly, it achieves frame rates of 5 fps on standard hardware.

Section 5.1 reviews related work in literature. The novel decoupled approach to scene flow estimation from two consecutive stereo image pairs is then presented in Section 5.2. In Sections 5.3 and 5.4, an evaluation approach is presented to compare scene flow algorithms on synthetic sequences and to derive accuracy estimates. The derived scene flow estimates will be used in Chapter 6 as input to segment moving objects in traffic scenes.

5.1 Introduction and Related Work

The two-dimensional image motion, or optical flow, can be directly estimated from an image sequence by means presented in Chapter 4. This two-dimensional motion is the three-dimensional scene motion, projected onto the image plane. During this projection process, one dimension is lost and cannot be recovered without additional constraints or information. Hence, one may say that motion computation using a single camera is not well constrained.

This is a different story, once a stereo camera system is available. The distance estimate from the triangulation of stereo correspondences provides the necessary additional information needed to reconstruct the three-dimensional scene motion. Then, ambiguities only arise

- ... if the camera motion is not known (in particular the camera is not stationary). Then the motion of the scene involves two primary parts; the motion of dynamic objects within the scene and the motion of the static background from the motion of the camera.
- ... around areas with missing structure in a local neighborhood (i. e. this leads to the aperture problem, see Figure 4.3).

The ambiguity between motion induced by camera motion and dynamic objects can only be solved if the ego-motion of the camera is known (e.g. using methods from Chapter 3). A common way to deal with missing structure and to achieve dense estimates is, similarly to the two-dimensional optical flow estimation presented in Chapter 4, the use of variational approaches that incorporate a smoothing function.

In this chapter the term *dense scene flow* refers to the three-dimensional image velocity field for points that can be seen by both cameras (i. e. omitting occlusion). This defines scene flow as velocity and position estimates in image coordinates (pixels). The scene flow is translated into real-world coordinates to become the motion of objects in the scene, the *world flow*.

See Figure 5.1 for an example, where the motion of a preceding vehicle becomes visible in the world flow. The motion of objects within the scene is one of the most important features to extract in image sequences from a dynamic environment. Humans perform this using visual kinesthesia. This encompasses both, the perception of movement of objects in the scene and also the observers own movement.

Related Work Scene flow estimation can be achieved by estimating all parameters in a combined approach: the optical flow in the left and right images (using consecutive frames) and enforcing the stereo disparity constraints in the two stereo image image pairs. Besides optical flow estimation this involves an additional disparity¹ estimation problem, as well as the task to estimate the change of disparity over time.

¹The disparity is also needed to calculate the absolute 3D position, from the perspective of the camera.

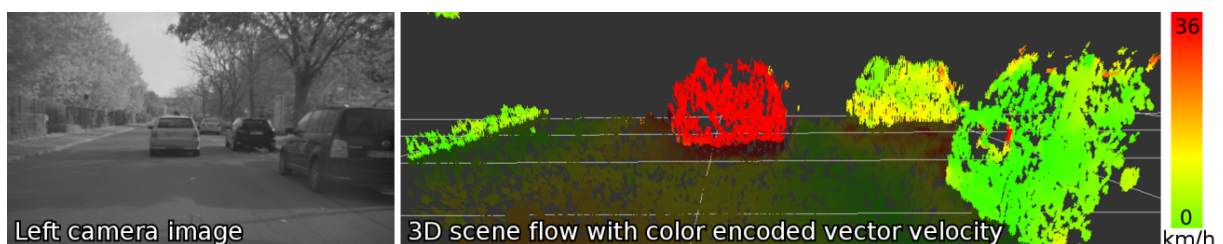


Figure 5.1: Scene flow example. Despite similar distance from the viewer, the moving car (red) can be clearly distinguished from the parked vehicles (green).

The work in [108] introduced scene flow as a joint motion and disparity estimation method (*coupled approach*). The succeeding works in [74, 103, 152] presented energy minimization frameworks including regularization constraints to provide dense scene flow. Other dense scene flow algorithms have been presented in multiple camera set-ups [110, 137]. However, these allow for non-consistent flow fields in single image pairs.

None of the above approaches run in real-time, giving best performances in the scale of minutes. Real-time scene flow algorithms, such as those presented in [64, 112], provide only sparse results both for the disparity and the velocity estimates. The work in [75] presents a probabilistic scene flow algorithm with computation times in the range of seconds, but yielding only integer pixel-accurate results.

Contributions In this chapter, the motion estimation is *decoupled* from the position estimation, while still maintaining a soft disparity constraint. The decoupling of depth (disparity) and motion (optical flow and disparity change) estimation might look unfavorable at a first glance; but it has at least two important advantages:

Firstly, the challenges in motion estimation and disparity estimation are quite different. With disparity estimation, thanks to the epipolar constraint, only a scalar field needs to be estimated. This enables the use of global optimization methods, such as dynamic programming or graph-cuts, to establish point correspondences. Optical flow estimation, on the other hand, requires the estimation of a vector field, which rules out such global optimization strategies. Additionally, motion vectors are usually smaller in magnitude than disparities. With optical flow, occlusion handling is less important than the sub-pixel accuracy provided by variational methods.

Splitting scene flow computation into the estimation sub-problems, disparity and optical flow with disparity change, allows to choose the optimal technique for each task.

At this point it is worth noting that, although the problems of disparity estimation and motion estimation are separated, the here presented method still involves a coupling of these two tasks, as the optical flow is enforced to be consistent with the computed disparities.

Secondly, the two sub-problems can be solved more efficiently than the joint problem. This allows for real-time computation of scene flow, with a frame rate of 5 fps on QVGA images (320×240 pixel, assuming the disparity map is given).

The splitting approach to scene flow is about 500 times faster compared to recent techniques for joint scene flow computation.

Nevertheless, an accuracy that is at least as good as the joint estimation method is achieved on test sequences. Furthermore, in combination with a dense disparity map the scene flow field and its corresponding world flow field are dense.

This chapter is organized as follows. The formulation and solving of scene flow from two consecutive stereo image pairs is presented in Section 5.2. In Section 5.3 a comparison of different stereo methods as input for the scene flow algorithm is presented. Section 5.4 evaluates the quality of the scene flow and derives individual standard deviations for every pixel in the image employing a goodness-of-fit quality measure. Section 5.5 summarizes the results and provides an outlook on future work.

5.2 Formulation and Solving of the Constraint Equations

This section derives the formulation of the scene flow algorithm. It identifies how the decoupling of position and motion is put to use effectively, while still maintaining the stereo disparity constraint.

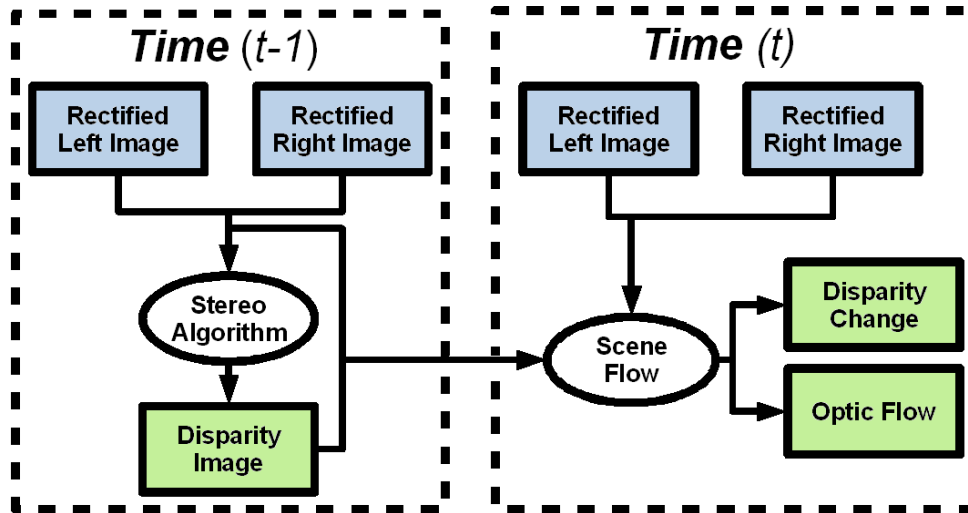


Figure 5.2: Outline of the scene flow algorithm. The input images (*blue*) are processed by the stereo and scene flow algorithms; the disparity and flow data (*green*) represent the resulting scene flow information.

Figure 5.2 outlines the approach. As seen from this figure, the stereo image pair is needed for both, the previous and current time frame. The derived approach also requires the disparity map in the previous time frame. This information is passed to the scene flow algorithm for processing, resulting in the optical flow and in the change in disparity between the image pairs.

5.2.1 Stereo Computation

The presented scene flow algorithm requires a pre-computed disparity map. Current state-of-the-art algorithms (e.g. [72]) require normal stereo epipolar geometry, such that pixel rows y for the left and right images coincide (rectified stereo images). The goal of the stereo correspondence algorithm is to estimate the disparity d from the left to the right image for every non-occluded pixel. This is accomplished by local methods (using a small matching window from the left image to the right image) or global methods (incorporating global energy minimization). The disparity information can then be used to reconstruct the 3D scene.

The scene flow algorithm presented in this chapter has the flexibility to be able to use any disparity map as input. Dense or sparse algorithms are handled effectively due to the variational nature of the approach. To demonstrate this, different disparity estimation techniques are used in this chapter and evaluated in Section 5.3.

Hierarchical correlation algorithms, yielding sparse sub-pixel accuracy disparity maps, are commonly employed due to their real-time capability. A typical implementation described in [63] is used, which allows for disparity computation at about 100 Hz. In addition, an implementation based on the algorithm described in [124] is used. It computes sparse pixel-discrete disparity maps and is available in hardware without extra computational cost.

Using globally consistent energy minimization techniques, it becomes possible to compute dense disparity maps, which yield a disparity value for every non-occluded pixel. The Semi-Global Matching algorithm (SGM) is used here [72]. The algorithm is implemented on dedicated hardware and runs at 25 Hz on images with a resolution of 640×480 pixels.

The remaining part of this section presents the data terms make-up of the scene flow energy functional with solution strategies. Section 5.3 demonstrates results with the aforementioned sparse and dense stereo algorithms.

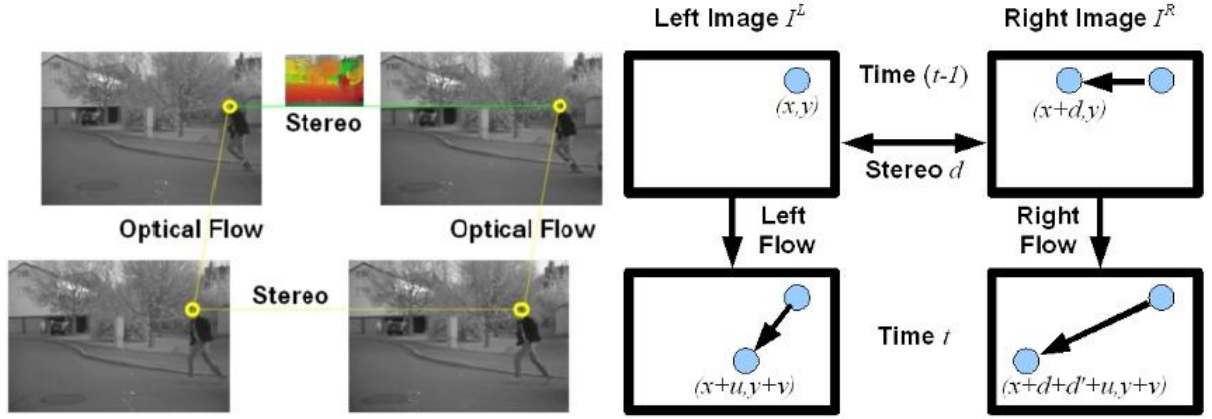


Figure 5.3: Motion and disparity constraints employed in the scene flow algorithm. The *left* images show two stereo image pairs with a corresponding point visible in every single image. The schematic images on the *right* illustrate the mathematical relations of corresponding image points.

5.2.2 Scene Flow Motion Constraints

If two stereo image pairs at times $t-1$ and t are given, then the two-dimensional optical flow for the left images can be estimated with approaches described in Chapter 4. The resulting flow vector field encodes for every image pixel (x, y) the temporal change of image position, (u, v) , from time $t-1$ to time t . For the three-dimensional scene flow computation also the temporal change d' of the disparity d is essential. The three-dimensional velocity field can only be reconstructed, when both the image data (x, y, d) and its temporal change (u, v, d') are known. d is estimated using an arbitrary stereo algorithm, see Section 5.2.1. The disparity change and the two-dimensional optical flow field have to be estimated from the stereo image pairs.

See Figure 5.3 for the basic outline of this approach. To estimate the disparity change, the optical flow constraint is extended to the right images. Let $I(x, y, t)^L$ be the intensity value of the left image and $I(x, y, t)^R$ be the intensity of the right image, at pixel position (x, y) and time t . Due to rectification, the optical flow in the left image and in the right image will have the same y component. Hence the difference is found only in the x component of the flow vectors. This results in the following constraints for the left and right images:

$$I(x, y, t-1)^L = I(x+u, y+v, t)^L \quad \text{and} \quad (5.1)$$

$$I(x+d, y, t-1)^R = I(x+d+\underbrace{d'+u}_{u^R}, y+v, t)^R. \quad (5.2)$$

If the optical flow field for the left and right images is estimated independently, the disparity change can be calculated as the difference $u^R - u$. However, to estimate the disparity change more accurate, consistency of the left and right image at time t is enforced. More precisely, the gray values of corresponding pixels in the stereo image pair at time t should be equal. This yields the third constraint,

$$I(x+u, y+v, t)^L = I(x+d+d'+u, y+v, t)^R. \quad (5.3)$$

Figure 5.4 shows a summary of the above equations with the resulting calculated world flow from the scene flow estimates. Rearranging the above equations results in:

$$\begin{aligned} E_{LF} &:= I(x+u, y+v, t)^L - I(x, y, t-1)^L \stackrel{!}{=} 0, \\ E_{RF} &:= I(x+d+d'+u, y+v, t)^R - I(x+d, y, t-1)^R \stackrel{!}{=} 0, \\ E_{DF} &:= I(x+d+d'+u, y+v, t)^R - I(x+u, y+v, t)^L \stackrel{!}{=} 0. \end{aligned} \quad (5.4)$$

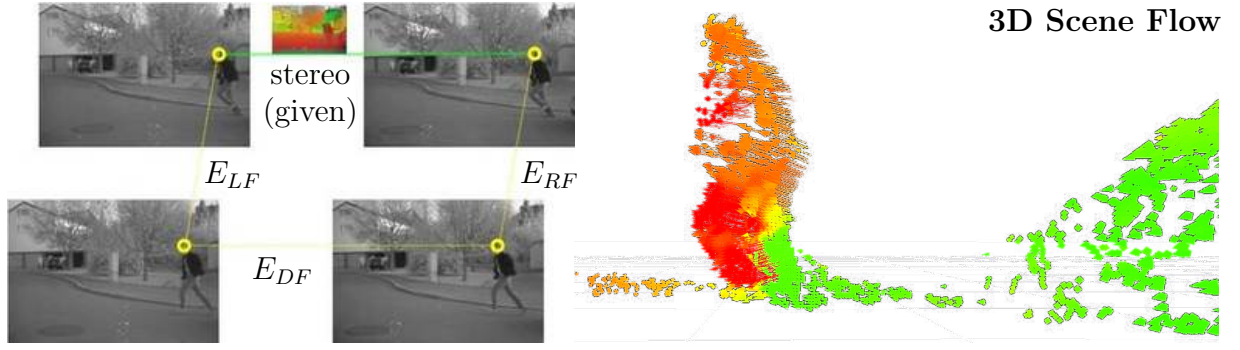


Figure 5.4: The scene flow equations from Equation (5.4) are summarized in the left schema. The right image shows the corresponding world flow result with color encoding velocity (where the color fades from green to red as in stationary to moving). Note the correct motion estimation of the feet as the person is running forward.

Occlusion handling is an important aspect in scene flow estimation. It comes along with increased computational costs, but clearly improves results. Regarding the influence of occlusion handling on disparity estimation and velocity estimation, the magnitude of the disparity is generally much larger than the magnitude of the optical flow and the disparity change, which is only a few pixels. Accordingly, occlusion handling is much more decisive for disparity estimation than for motion estimation.

From a practical point of view, the effort in occlusion handling should be set in comparison to the gained accuracy in the scene flow result. Hence, occlusion handling is not explicitly modeled for the scene flow. Occluded areas identified by the stereo estimation algorithm (using left-right consistency checks) and areas with no disparity information are simply discarded. More precisely, for pixels with no valid disparity value, Equations 5.2 and 5.3 will not be evaluated. This procedure implicitly enables the use of sparse disparity maps.

5.2.3 Solving the Scene Flow Equations

The constraints (or gray value constancy assumptions) in Equation (5.4) are used as data terms to solve the scene flow problem. Similarly to the optical flow case, the linearized versions of the constraint equations yield three data terms,

$$\begin{aligned}
 p_{LF} &:= I_t^L(x, y) + I_x^L(x, y)u + I_y^L(x, y)v \\
 p_{RF} &:= \text{occ}(x, y) \left(I_t^R(x_d, y) + I_x^L(x_d, y)(u + d') + I_y^L(x_d, y)v \right) \\
 p_{DF} &:= \text{occ}(x, y) \left((I^R(x_d, y) - I^L(x, y)) + \right. \\
 &\quad \left. (I_x(x_d, y)^R - I_x(x, y)^L)(u + d') + (I_y(x_d, y)^R - I_y(x, y)^L)v \right), \tag{5.5}
 \end{aligned}$$

where I_t , I_x , and I_y denote the partial derivatives of the image function (compare with Equation (4.1)). $x_d = x + d$ is the horizontal image position in the right image corresponding to a pixel at position x in the left image; the occlusion flag occ returns 0 if there is no disparity known at (x, y) (due to occlusion or sparse stereo method), or 1 otherwise. If a pixel is occluded, only the first data term, p_{LF} is evaluated. Otherwise all three data terms contribute to the solution.

The above data terms can be used to compute the scene flow using the *Refinement Optical Flow* framework presented in Section 4.2 using the three dimensional flow vector $(u, v, d')^\top$. Alternatively, Appendix C presents how the the Brox et al. approach [45] can be adapted to the scene flow case; embedding the above formulas into a variational framework to be solved by fix point iterations.

5.2.4 Visualizing the 3D Velocity Field

To visualize the scene flow, the corresponding 3D translation vector is computed. The corresponding equations are derived from the inverse of Equation 3.3, using the disparity and disparity change to compute the distance of points:

$$X_{t-1} = (x - x_0) \frac{b}{d}, \quad Y_{t-1} = (y - y_0) \frac{b}{d}, \quad Z_{t-1} = \frac{f_x b}{d}$$

and

$$X_t = (x + u - x_0) \frac{b}{d + d'}, \quad Y_t = (y + v - y_0) \frac{b}{d + d'}, \quad Z_t = \frac{f_x b}{d + d'}$$

For simplicity the assumption $f_y = f_x$ was used. This yields for the translation vector

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \begin{pmatrix} X_t - X_{t-1} \\ Y_t - Y_{t-1} \\ Z_t - Z_{t-1} \end{pmatrix} = b \begin{pmatrix} \frac{x-x_0}{d} - \frac{x+u-x_0}{d+d'} \\ \frac{y-y_0}{d} - \frac{y+v-y_0}{d+d'} \\ \frac{f_x}{d} - \frac{f_x}{d+d'} \end{pmatrix} \quad (5.6)$$

Obviously this yields the translation vector in the camera coordinate system at time $t-1$. In order to detect moving objects and to calculate the absolute motion, the motion of the camera has to be compensated. For follow-on calculations (e.g. speed, accuracy of world flow, detection of moving objects, segmentation of objects, integration, etc.) besides the flow vectors the accuracy of the flow vectors is needed. Section 5.4 evaluates the scene flow and derives such accuracy estimates for the computed flow field.

5.3 Comparison of Results for Different Stereo Inputs

This section presents a summary of the original studies in [14] of the scene flow algorithm. Different stereo methods for the disparity input are compared, together with the full (coupled) scene flow estimation approach presented in [74]. To assess the quality of the scene flow algorithm, it was tested on synthetic sequences, where the ground truth is known.

The first ground truth experiment is the *rotating sphere*² sequence from [74] depicted in Figure 5.5. In this sequence the spotty sphere rotates around its y -axis to the left, while the two hemispheres of the sphere rotate in opposing vertical directions. The resolution is 512×512 pixels. The scene flow method was tested together with four different stereo algorithms as input for the disparity estimates (see Section 5.2.1):

- Semi-Global Matching (SGM [72]),
- SGM with hole filling³.
- correlation pyramid stereo [63], and
- an integer accurate census-based stereo algorithm [124].

The ground truth disparity was also used for comparison, i. e., using the ground truth disparity as the input disparity for the algorithm. For each stereo algorithm, the absolute angular error (AAE) and the root mean square (RMS) error were calculated as evaluation measurements

²I thank Huguet and Devernay for providing their *sphere scene*.

³SGM with hole filling computes a disparity map using the SGM algorithm [72]. Due to occlusions and low texture a disparity cannot be assigned to every pixel within the image. To achieve a dense disparity map, holes are filled by replicating the disparity from the left or right border pixel, favoring background objects; hence replicating the smaller disparity value (in common stereo set-ups the pixel to the left).

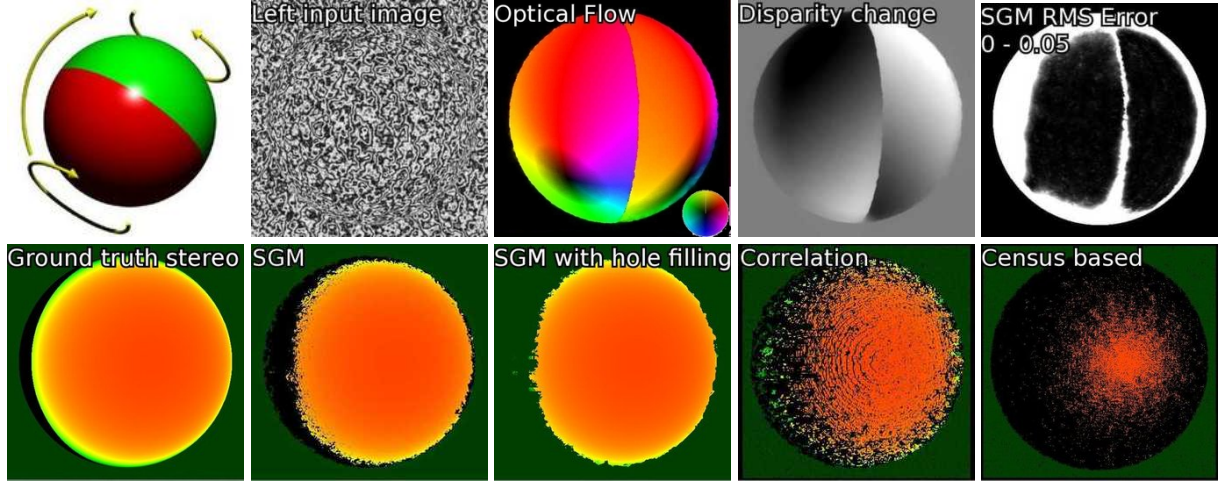


Figure 5.5: Ground truth test on the *rotating sphere* sequence. Quantitative results are shown in Table 5.1. **Top:** The left image shows the movement of the sphere. Color encodes the direction of the optical flow (key in bottom right), intensity its magnitude. Disparity change is encoded from black (increasing) to white (decreasing). Bright parts of the RMS figure indicate high $RMS_{u,v,d'}$ error values of the computed scene flow using the SGM stereo method. **Bottom:** disparity images are color encoded green to orange (low to high). Black areas indicate missing disparity estimates or occluded areas.

where a superscript $*$ denotes the ground truth solution and n is the number of pixels:

$$AAE_{u,v} = \frac{1}{n} \sum_{\Omega} \arctan \left(\frac{uv^* - u^*v}{uu^* + vv^*} \right) \text{ as used in [74] and}$$

$$\underbrace{RMS_d = \sqrt{\frac{1}{n} \sum_{\Omega} \text{occ} \|(d)^\top - (d^*)^\top\|^2}}_{\text{disparity evaluation}}, \quad \underbrace{RMS_{u,v} = \sqrt{\frac{1}{n} \sum_{\Omega} \|(u,v)^\top - (u^*,v^*)^\top\|^2}}_{\text{optical flow evaluation}},$$

$$\underbrace{RMS_{u,v,d'} = \sqrt{\frac{1}{n} \sum_{\Omega} \|(u,v,d')^\top - (u^*,v^*,d'^*)^\top\|^2}}_{\text{scene flow evaluation}}.$$

The errors were calculated using two different image domains Ω : firstly, calculating statistics over all non-occluded areas, and secondly calculating over the whole input image. As in [74], pixels from the stationary background were not included in the statistics. The resulting summary

Stereo Algorithm	RMS _d (density)	Without occluded areas			With occluded areas		
		RMS _{u,v}	RMS _{u,v,d'}	AAE _{u,v}	RMS _{u,v}	RMS _{u,v,d'}	AAE _{u,v}
Ground truth	0 (100%)	0.31	0.56	0.91	0.65	2.40	1.40
SGM [72]	2.9 (87%)	0.34	0.63	1.04	0.66	2.45	1.50
Correlation [63]	2.6 (43%)	0.33	0.73	1.02	0.65	2.50	1.52
Fill-SGM	10.9 (100%)	0.45	0.76	1.99	0.77	2.55	2.76
Hug.-Dev. [74]	3.8 (100%)	0.37	0.83	1.24	0.69	2.51	1.75
Census based [124]	7.8 (16%)	0.32	1.14	1.01	0.65	2.68	1.43

Table 5.1: Root mean square (pixels) and average angular error (degrees) for the scene flow of the *rotating sphere* sequence. Various stereo algorithms are used as input for the scene flow estimation. Ranking is done according to $RMS_{u,v,d'}$ error in non-occluded areas.

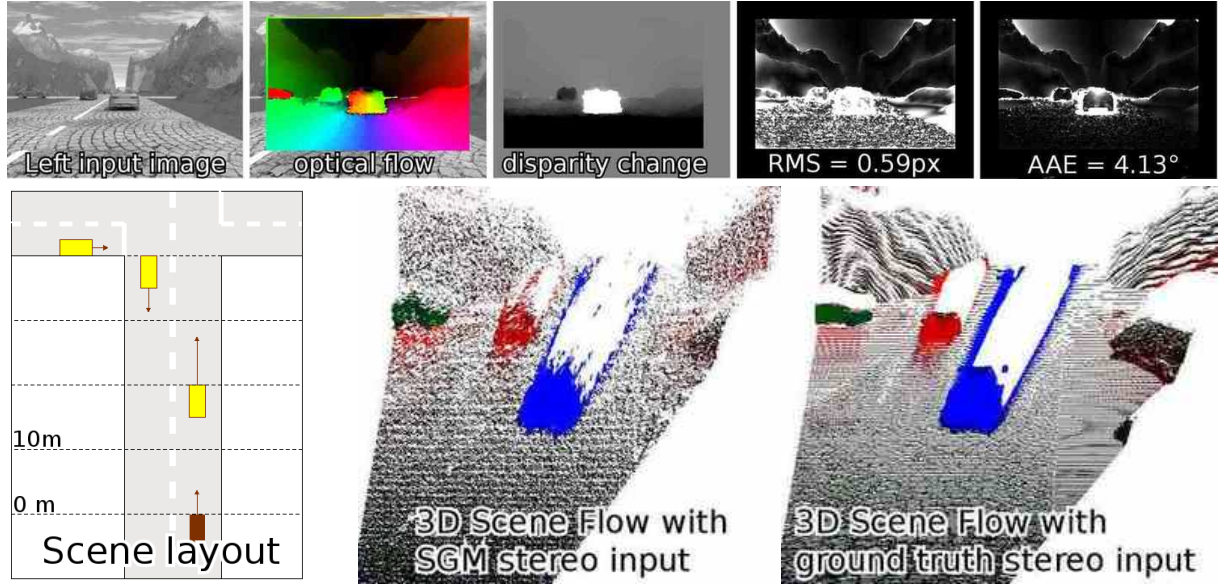


Figure 5.6: Povray-rendered traffic scene (Frame 11). **Top:** Color encodes direction (border = direction key) and intensity the magnitude of the optical flow vectors. Brighter areas in the error images denote larger errors. For comparison, running the code from [74] generates an RMS error of 0.91px and AAE of 6.83°. **Bottom right:** 3D views of the scene flow vectors. Color encodes their direction and brightness their magnitude (black = stationary). The results from the scene are clipped at a distance of 100m. Accurate results are obtained even at greater distances.

can be seen in Table 5.1.

The presented method achieves lower errors than the baseline fully combined variational method by Huguet and Devernay, even using sparse correlation stereo. Particularly, the RMS error of the scene flow is much smaller, while still being considerably faster. This is explained by the higher flexibility in choosing the disparity estimation method (e.g. the accurate SGM method can be used for disparity estimation).

The table also shows that SGM with hole filling yields inferior disparity results than the other stereo methods. This is due to false disparity measurements in the occluded areas. It is better to feed the non-occluded measurements of SGM to the variational framework, which yields dense scene flow estimates as well, but with higher accuracy. SGM was chosen as the best method and is used in the remainder of this chapter; it is also available on dedicated hardware without any extra computational cost.

The second ground truth experiment uses a Povray-rendered traffic scene, which is available on-line publicly for comparison [136]. The scene layout is shown in Figure 5.6. The two error measurements used are the $\text{RMS}_{u,v,d'}$ error and the 3D angular error defined by:

$$\text{AAE}_{3D} = \frac{1}{n} \sum_{\Omega} \arccos \left(\frac{uu^* + vv^* + d'd^* + 1}{\sqrt{(u^2 + v^2 + d'^2 + 1)((u^*)^2 + (v^*)^2 + (d'^*)^2 + 1)}} \right)$$

where again a superscript * defines a ground truth value. Results are shown in Figure 5.6. They compare favorably to the results obtained when running the code from [74]. The average $\text{RMS}_{u,v,d'}$ error for the whole sequence (sub-region as in Figure 5.6) was 0.64px and the 3D angular error was 3.0° (see [14] for further results on this scene).

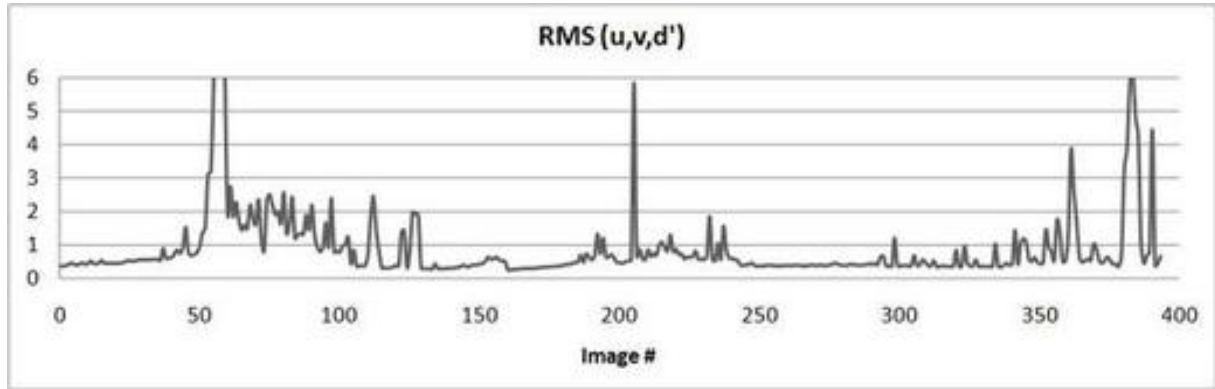


Figure 5.7: RMS error evaluation over the entire sequence.

5.4 Derivation of a Pixel-wise Accuracy Measure

This section analyzes the error distribution of the scene flow variables. A complex driving scene, involving hills, trees, and realistic physics has been generated using Povray. It consists of 400 sequential stereo image pairs. For each image pair at time t , the following measures are calculated:

- $\text{RMS}_{u,v,d'}$ from the section above.
- Error at each pixel, i. e., difference between estimate and ground truth.
- Mean and variance of the error for u, v, d' , and d .

An example picture is shown in Figure 5.8. This figure reveals that the major errors are at object boundaries, and the errors in d' are the lowest in magnitude. Another single frame example is shown in Figure 5.9, where the optical flow is perturbed due to optical flow occlusion.

The evaluation results for the entire image sequence (all 400 frames) can be seen in the graphs in Figures 5.7, 5.10, and 5.11. From these graphs, one can see which frames are causing problems for the algorithms.

It becomes visible, that the errors and standard deviation for the scene flow variables u , v , and d' (Figure 5.10) are of similar shape, yet different magnitudes. This is expected as they are solved using the variational energy minimization and smoothed in the one framework. On the other hand, the disparity error graph (Figure 5.11) has a much different shape, with variances increasing in different frames.

The figures also show an error histogram for one specific frame of the sequence. It becomes visible, that the error distribution has one salient peak around zero (no error), and long tails to both sides. All four error histograms show these criteria. This leads to the assumption, that these errors follow a certain error distribution. The next subsection investigates this distribution in more detail and derives conclusions about quality criteria and accuracy of the scene flow.

Estimation of Individual Standard Deviations for Each Pixel

The disparity d and the scene flow variables d' , u , and v are estimated by joint minimizing of a data and smoothness term. This implies that the solution is not exact as in without errors; hence some parts of the image show certain inaccuracies as seen in Figures 5.8 and 5.9. It can be seen that regions containing object boundaries tend to be less accurate than others.

For follow-on calculations (e. g. speed, accuracy of world flow, detection of moving objects, segmentation of objects, integration, etc.) variances for each computed flow vector are needed. It seems reasonable to assign different variances to different regions of the image because, as

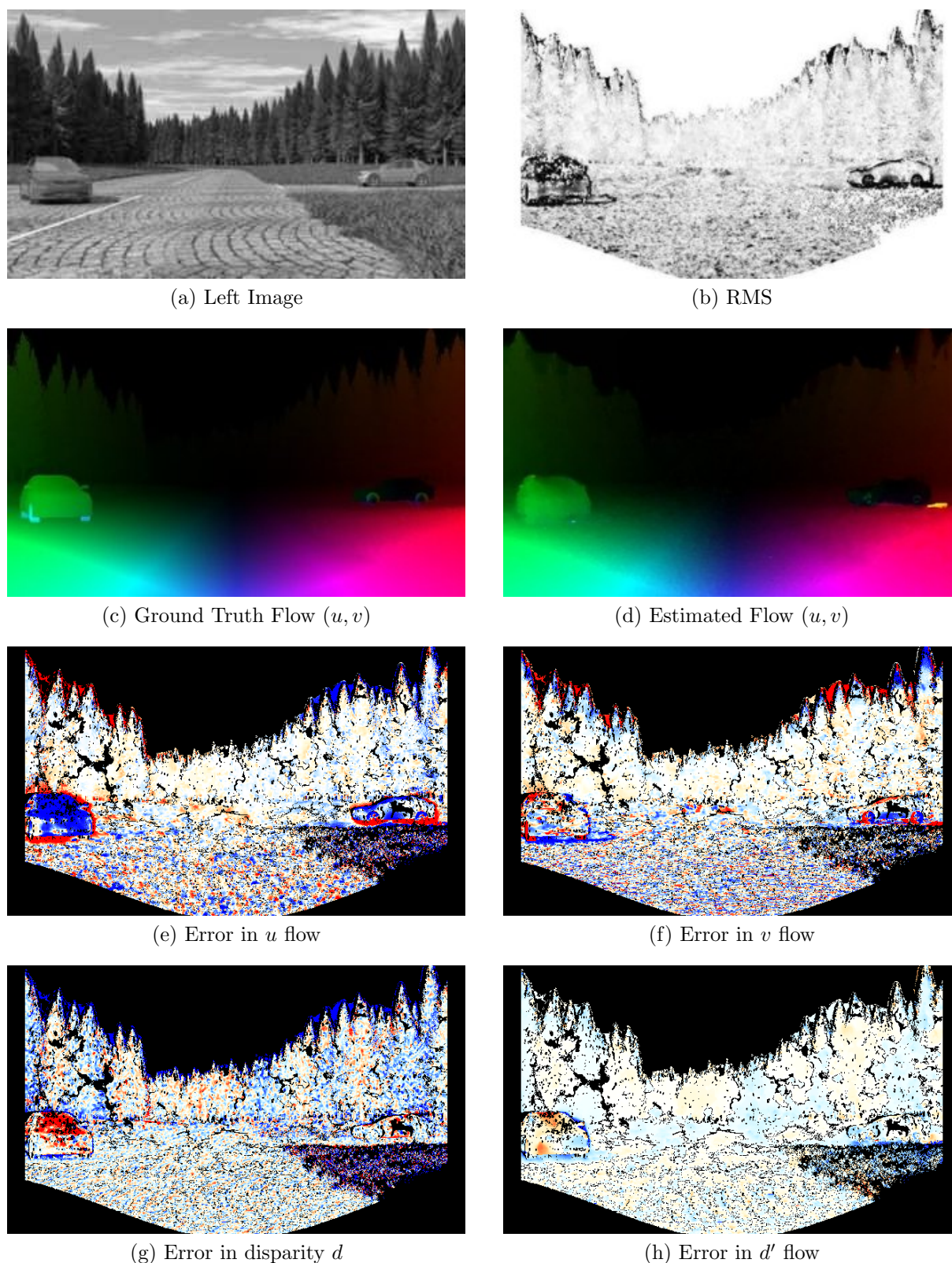


Figure 5.8: Frame 215 in the Povray-rendered traffic scene. The RMS is encoded in intensity, white to black as low to high RMS error (saturated at 1 px error); occluded points are shown as zero values. Flow color is encoded as in Figure 5.5. Error images are encoded in color and intensity (saturated at 1 px error), where red denotes negative error (ground truth value larger than estimate), blue encodes positive error (ground truth smaller than estimate), and black denotes occluded pixels.

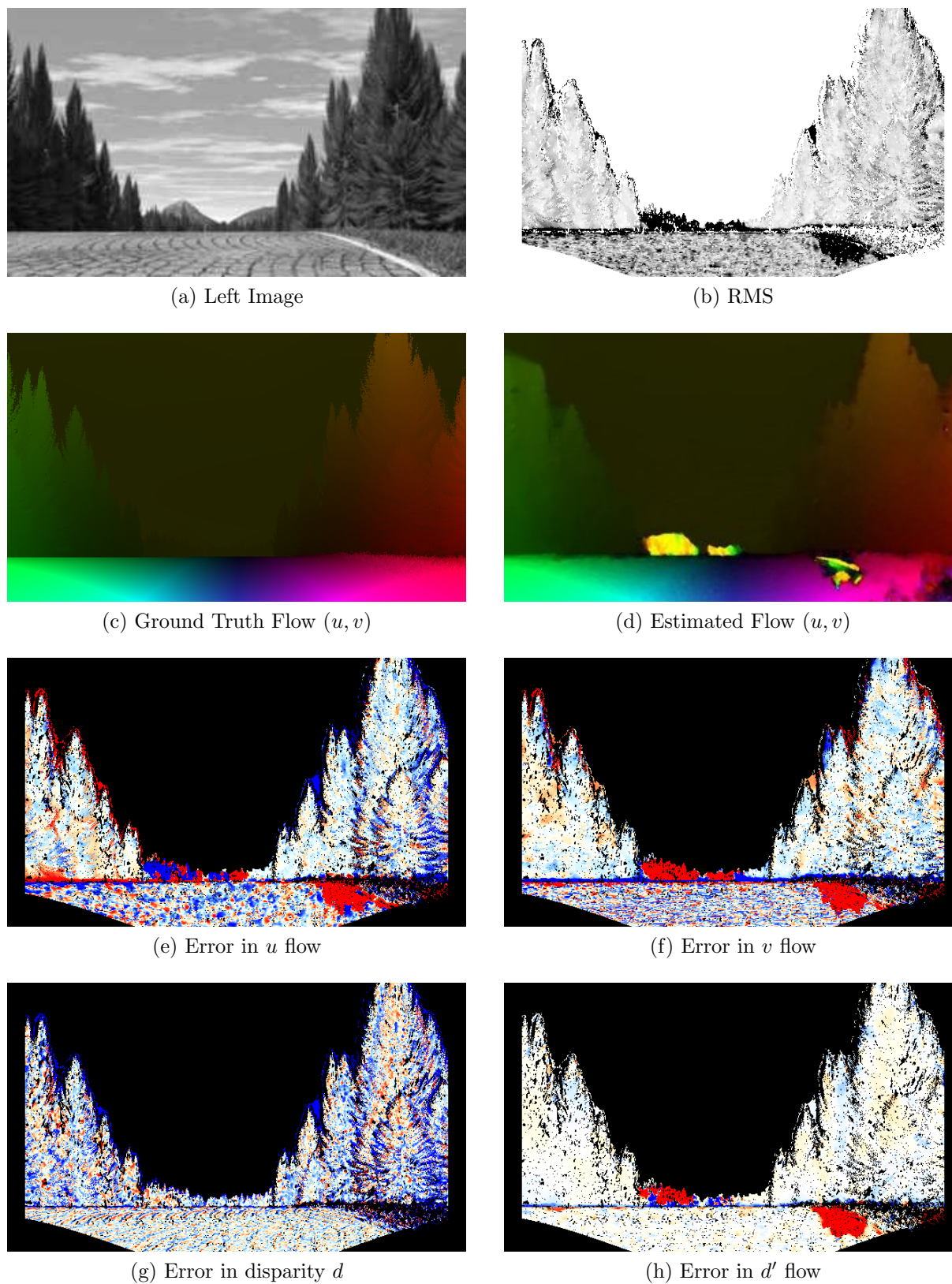


Figure 5.9: Frame 58 in the Povray-rendered traffic scene. Encoding as Figure 5.8.

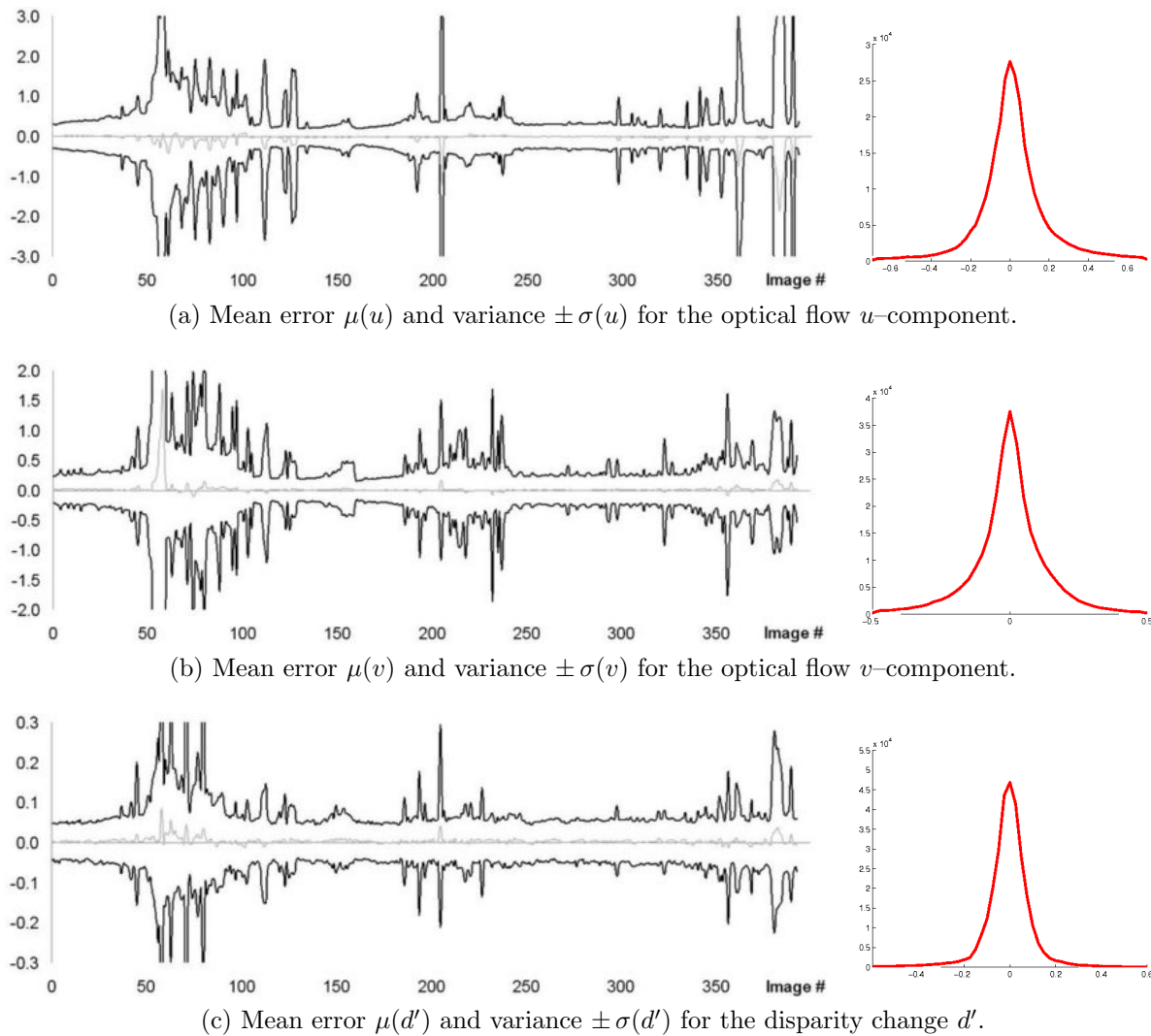


Figure 5.10: The *left graphs* show the results for the mean error (*light colored line* of the scene flow components, u , v , and d'). The bounding dark lines are one standard deviation from the mean. The *right graphs* show the error histogram for the scene flow between frames 122 and 123.

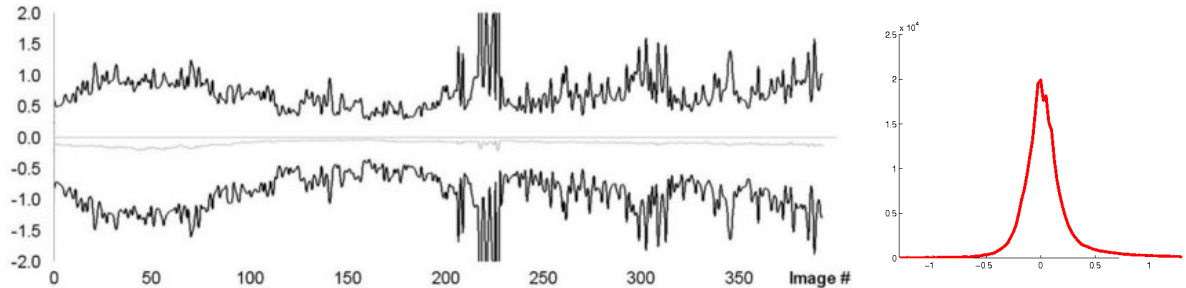


Figure 5.11: The *left graph* shows the results for the mean error (*light colored line*, $\mu(d)$) of the disparity d . The bounding dark lines are one standard deviation from the mean (i. e. $\pm\sigma(d)$). The *right graph* shows the error histogram for frame 122.

discussed above, errors are not spatially equally distributed in the image. This subsection presents methods to estimate a quality measure for the variance of the scene flow estimates for every pixel. Using this quality measure, individual standard deviations for each pixel are derived, assuming a standard distribution of the error.

A Quality Measure for the Disparity

The core Semi-Global Matching stereo method [72] estimates only pixel-accurate disparity maps. One option to obtain sub-pixel accuracy is parabolic fitting of the obtained cost [72]. Here, sub-pixel accuracy is achieved by a subsequent local fit of a symmetric first-order function [119]. Let \tilde{d} be the disparity estimate of the core SGM method for a certain pixel in the left image.

The SGM method in [72] is formulated as an energy minimization problem. Hence, changing the disparity by ± 1 yields an increase in costs (yielding an increased energy). The minimum, however, may be located in between pixels, motivating a subsequent sub-pixel estimation step. The basic idea of this step is illustrated in Figure 5.12. The costs for the three disparity assumptions $\tilde{d}-1$ px, \tilde{d} px, and $\tilde{d}+1$ px are taken and a symmetric first order function is fitted to the costs. This fit is unique and yields a specific sub-pixel minimum, located at the minimum of the function. Note, that this might not be the minimum of the underlying (continuous) energy but is a close approximation, evaluating the energy only at pixel position.

The slope of this fitting function serves as a quality measure for the goodness-of-fit. If the slope is low, the disparity estimate is not accurate in the sense that the cost function evaluated at other disparity positions is very similar. On the other hand, if the slope is large, the sub-pixel position of the disparity is expected to be quite accurate as deviation from this position increases the costs of the solution by a large amount. Hence, the larger the slope, the better is the expected quality of the disparity estimate. Note that the costs mentioned here are accumulated costs that also incorporate smoothness terms.

Based on this observation an uncertainty measure is derived for the expected accuracy (variance) of the disparity estimate:

$$U_D(x, y) = \frac{1}{\Delta y}, \quad (5.7)$$

where Δy is the larger of the two relative cost differences. This measure can be calculated for every pixel within the image, where a disparity is estimated. It can now be used to estimate the scale of the expected error distribution of this pixel.

If the ground truth disparity is known, the disparity error can be calculated for every pixel of the image. Figure 5.13 displays a resulting error image and its quality measure; a low quality value and a large error is displayed in black while a good quality or low error is displayed in white. One cannot expect that the images are exactly equal; if that would be the case, this would yield a way to estimate the exact ground truth disparity. Note, that the uncertainty values are unit-less; the error image is encoded such that errors above 1 px are black. Pixels with zero disparity and occluded pixels are displayed in white (e.g. the sky region).

To assess the quality of the presented uncertainty measure, Figure 5.14a depicts a 3D plot where errors for different uncertainty measures are accumulated over all 400 frames of the sequence. From this accumulated data one can calculate a curve of the variance over the uncertainty measure (see Figure 5.14b). As expected, the variance increases as the uncertainty measure increases. Furthermore, the figure shows that the disparity error for low uncertainty measures is approximately Laplacian distributed.

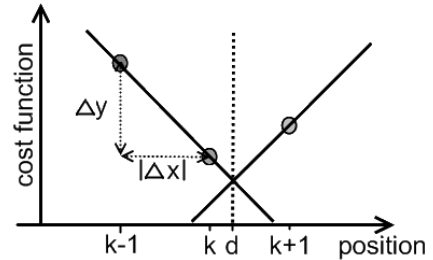


Figure 5.12: The slope of the disparity cost function serves as a quality measure for the disparity estimate.

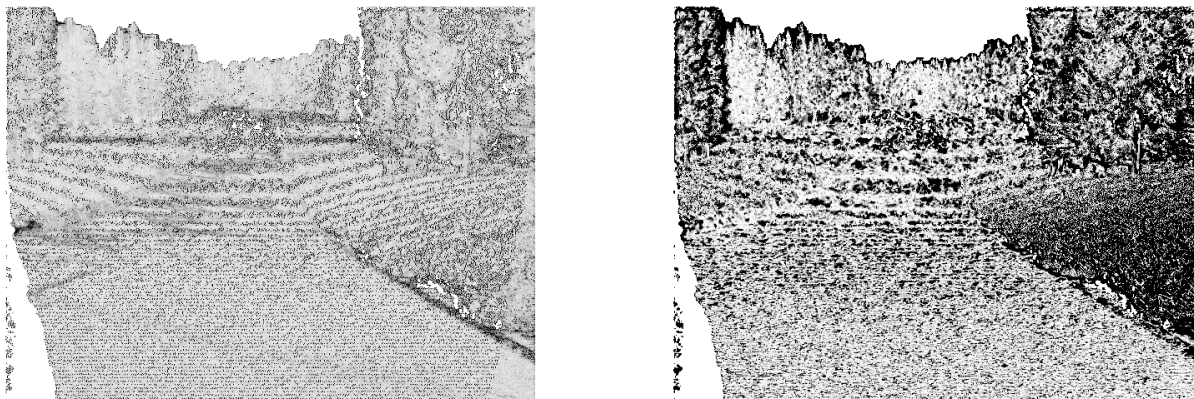
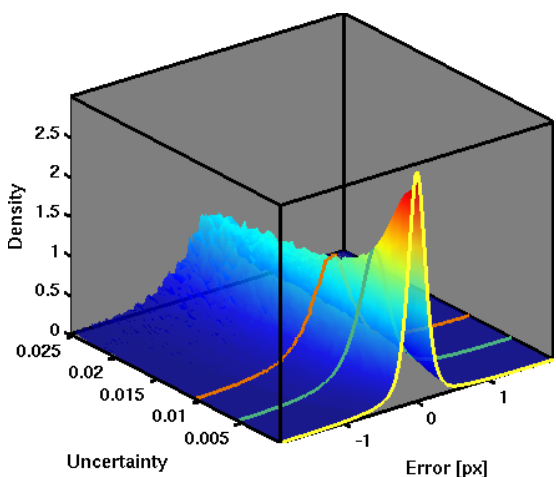
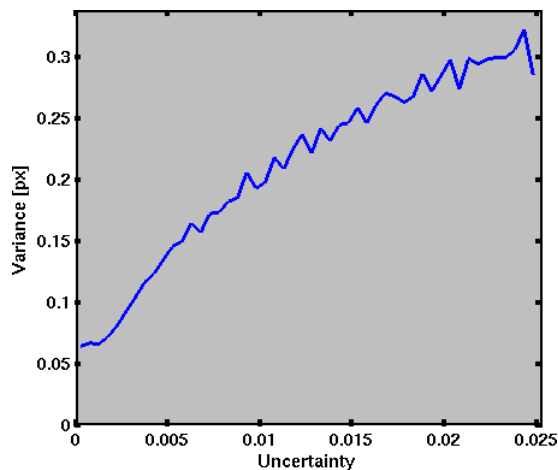


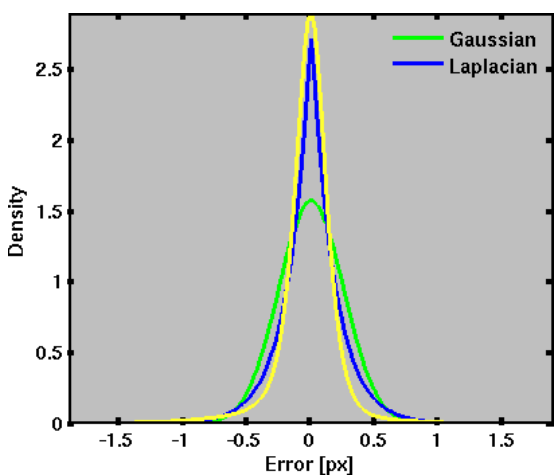
Figure 5.13: The images show the disparity uncertainty measure (*left*) and the disparity estimate error (*right*, compared to ground truth) for frame 123 of the evaluation sequence. Saturated pixels correspond to large uncertainty values and large scene flow estimate errors.



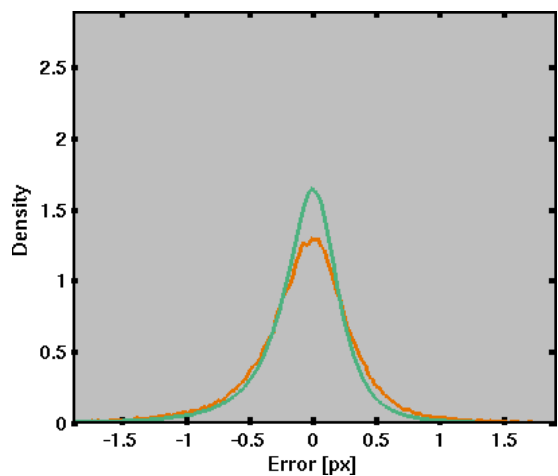
(a) 3D plot of the density in the disparity error vs. uncertainty measure domain. The density is normalized, s. t. the integral for a specific uncertainty measure over the error is one.



(b) Variance vs. uncertainty measure for the disparity error distribution (slice of (a)).



(c) Disparity error distribution (yellow, compare with (a)) and fitted Laplace and Gaussian distribution with the same variance.



(d) Disparity error distribution for larger uncertainty levels as in (b); compare with (a).

Figure 5.14: Quality analysis of the uncertainty measure over all 400 frames of the ground truth sequence. The figure shows that as the uncertainty measure increases, the variance of the error increases, too.

A Quality Measure for the Scene Flow

Similar to the one-dimensional disparity estimate, a quality measure has to be derived for the three-dimensional scene flow estimate. Due to the higher dimensionality, a simple fit of a linear symmetric function and evaluation of its slope is not adequate. More elaborate quality measures for optical flow have been presented and evaluated in [47]. The authors propose to directly evaluate the energy functional for the variational optical flow computation. The same idea can be carried forward to the scene flow case.

Recall the energy functional for the scene flow case. Minimizing the three constraint equations from Equation 5.4 jointly in a variational approach and enforcing smoothness of the flow field u , v , and the disparity change d' , yields the following energy functional:

$$E = \int \left\{ |E_{LF}| + \text{occ} \left(|E_{RF}| + |E_{DF}| \right) \right\} + \lambda \int \left\{ |\nabla u| + |\nabla v| + |\nabla d'| \right\}.$$

As before, the *occ*-function returns 1, if a disparity value is estimated for a pixel and 0 otherwise. The minimum of the above functional yields the resulting scene flow estimates. Evaluating the above sum within the integral for each pixel yields a goodness-of-fit, or uncertainty value, for the scene flow variables. If the constraint equations are fulfilled and additionally the solution is smooth, the pixel-wise evaluation,

$$U_{SF}(x, y) = \left\{ |E_{LF}| + \text{occ} \left(|E_{RF}| + |E_{DF}| \right) \right\} + \lambda \left\{ |\nabla u| + |\nabla v| + |\nabla d'| \right\},$$

returns a low uncertainty value. If on the other hand, constraint equations are not fulfilled, or at discontinuities in the scene flow field, the pixel-wise evaluation yields a large uncertainty value. Note that as in the stereo disparity case, this value is unit-less.

Figure 5.15 displays the pixel-wise uncertainty image and the RMS scene flow error for every pixel. A large uncertainty value and a large error are displayed in black while a low uncertainty or low errors are displayed in white; the error image is encoded such that errors above 1 px are black. Again, one cannot expect that both images are exactly equal; if that would be the case, this would yield a way to estimate the exact ground truth scene flow. The quality of the uncertainty measure is investigated in more detail over the whole test sequence as done before for the disparity error. Figure 5.16 shows the result exemplary for the error in u .

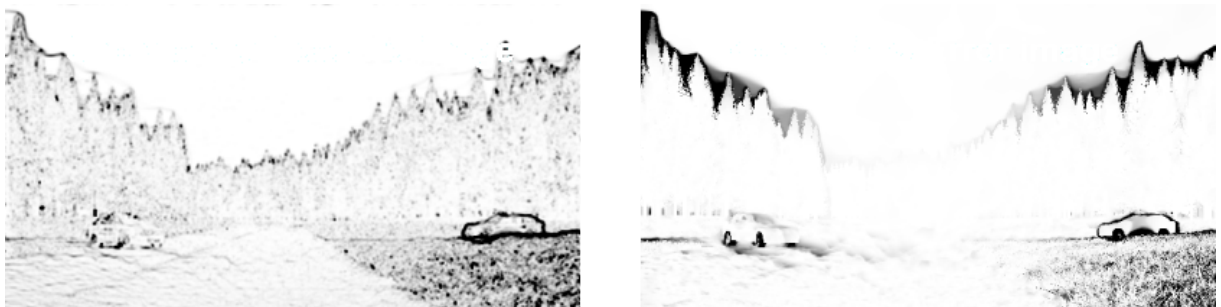
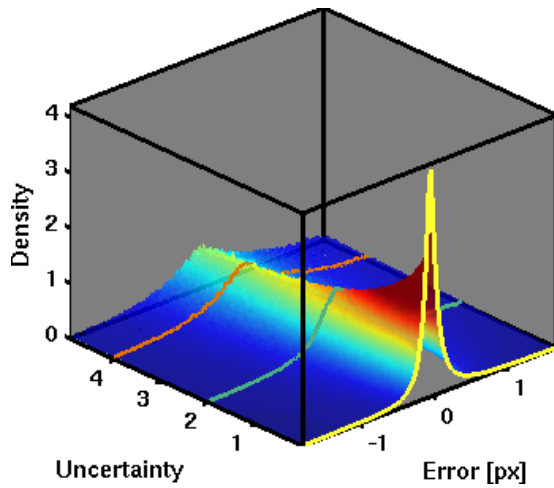
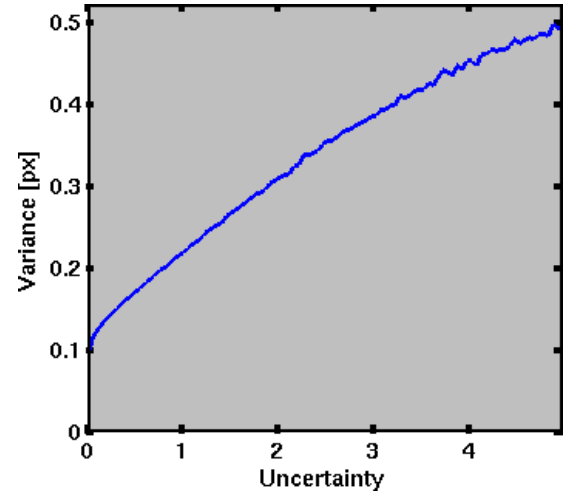


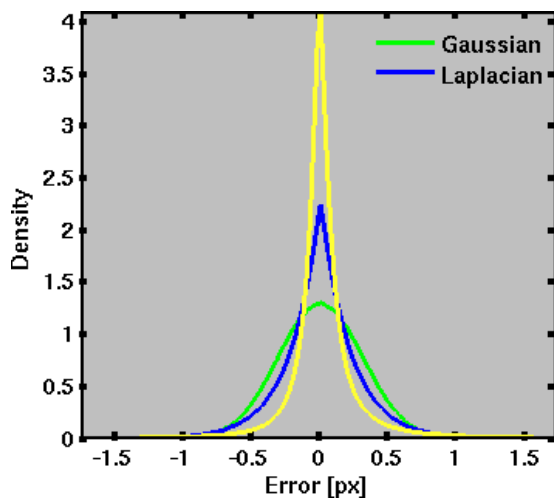
Figure 5.15: The images show the scene flow uncertainty measure (*left*) and the scene flow RMS error (*right*) for frame 205 of the evaluation sequence. Saturated pixels correspond to large uncertainty values and large disparity estimate errors.



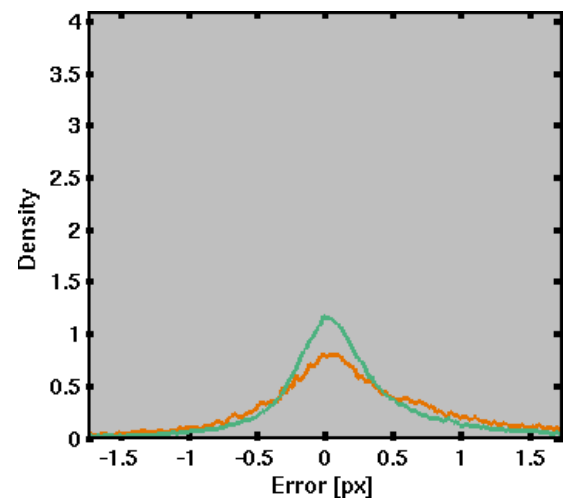
(a) 3D plot of the density in the scene flow u component error vs. uncertainty measure domain. The density is normalized, s. t. the integral for a specific uncertainty measure over the error is one.



(b) Variance vs. uncertainty measure for the scene flow u component error distribution (slice of (a)).



(c) Scene flow u component error distribution (yellow, compare with (a)) and fitted Laplace and Gaussian distribution with the same variance.



(d) Scene flow u component error distribution for larger uncertainty levels as in (b); compare with (a).

Figure 5.16: Quality analysis of the scene flow uncertainty measure over all 400 frames of the ground truth sequence. The figure shows that as the uncertainty measure increases, the variance of the error increases, too. The plots show only the scene flow u component; the v -component and d' -component yield similar results.

Estimating Individual Standard Deviations from the Uncertainty Measures

The last two subsections presented quality measures for the disparity and the three scene flow variables for every pixel in the image. A careful analysis of the quality measure demonstrated that these quality measures are correlated to the variance of the estimate errors. More precisely, the variances of the estimated scene flow values can be approximated by a linear function,

$$\sigma(a(x, y)) = \sigma_{0,a} + \gamma_a \cdot U_a(x, y),$$

where a is the estimated scene flow variable (u , v , d , or d' respectively), $\sigma_{0,a}$ and γ_a are constants, and $U_a(x, y)$ is either $U_D(x, y)$ or $U_{SF}(x, y)$, the uncertainty measure for the scene flow variable a at image position (x, y) . The parameters $\sigma_{0,a}$ and γ_a are found by fitting a line into the uncertainty-variance plot for the individual scene flow variable.

Such procedure ignores that the computed variances for a given uncertainty measure clearly depend on the amount of outliers in the errors; the more outliers, the larger is the variance. This can be seen in Figure 5.16c, where the shape of the error distribution seems to be close to a Laplacian distribution. However, the fit of a Laplacian with the same variance as the error distribution seems to be too large – due to the heavy tails of the error distribution the scale parameter of the Laplacian distribution is over-estimated. This boils down to the question how much influence should be given to large errors, in particular to outliers.

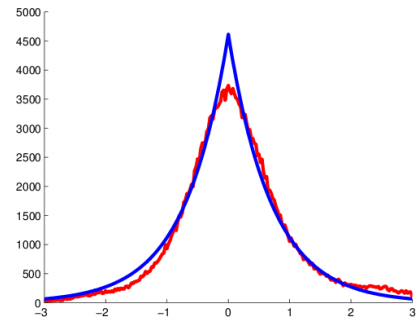
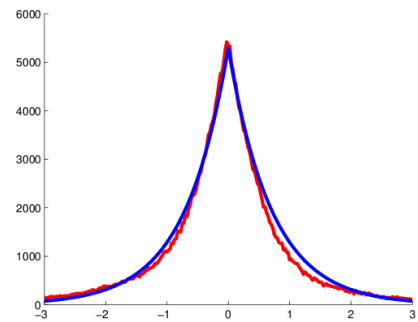
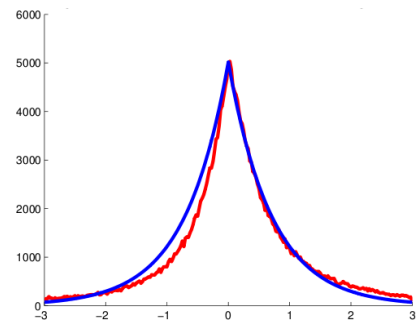
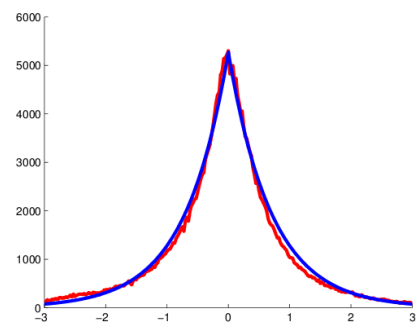
The proportion of large to small errors should not influence the error distribution, thus I propose to choose the parameters $\sigma_{0,a}$ and γ_a in such a way, that the resulting weighted distribution,

$$e'_a = e_a / \sigma(a(x, y)),$$

is standard Laplacian distributed (mean 0 and scale 1) on a three-sigma interval. The standard Laplacian distribution is given by

$$\frac{1}{2} \exp(-|x|). \quad (5.8)$$

The plots on the right show exemplary for one frame of the sequence, how the scene flow errors, weighted with their corresponding variances derived from the uncertainty measures, are approximately standard Laplacian distributed; apart from the tails which are thicker for the observed distribution due to outliers.

(a) Disparity d (b) Disparity change d' (c) Scene Flow u (d) Scene Flow v

The plots show the standard Laplacian distribution (mean 0 and scale 1) in *blue* and the true error distribution, weighted by individual variances derived from the uncertainty measures, in *red* (frame 123 of the ground truth sequence). Both curves are similar, confirming that the uncertainty measures are a valid approximation of the variances.

5.5 Ideas for Future Research

This chapter presented a variational framework for dense scene flow estimation, which is based on the decoupling of the disparity estimation from the motion estimation, while enforcing consistent disparity at all time instances. Such an approach has two main advantages: Firstly, one can choose optimal methods for estimating both, the disparity and the velocity. Secondly, for the first time, dense scene flow results close to real-time were obtained. In particular, I presented an approach employing occlusion handling and the semi-global-matching disparity algorithm [72], currently ranked in the top-10 on the Middlebury stereo evaluation, combined with dense, sub-pixel accurate motion estimation, currently ranked first place on the Middlebury motion evaluation (see Chapter 4). The latter method was extended to include the change of disparity and the two-dimensional optical flow. Both, the semi-global disparity estimation method as well as the variational motion estimation, are available on dedicated hardware in real-time (note, that common CPU implementations do need more than one second each).

In a second part, I carefully investigated the error distribution of the estimated scene flow variables, the disparity d , the disparity change d' , and the optical flow (u, v) in the left image. With this investigation I presented an approach to derive, for every pixel, individual variances for the estimated scene flow variables, based on quality measures proposed in literature. This allows subsequent applications, such as motion integration approaches or motion detection algorithms, to not only use the scene flow variables themselves but also accuracy of the estimates.

Nevertheless, there are a few open questions which provide prospects for future work. This includes the detection and handling of outliers in the scene flow estimation process. Such outliers may result from false stereo disparities, from inconsistent image data (due to non-modelled occlusion), shadow regions where the motion of the background and the motion of the shadow interfere, or bad illumination conditions such as rain. While detecting outliers is one research topic, robustness of the scene flow algorithm against a wide range of artifacts is another interesting field of research.

Another research topic is the use of scene-relevant information for the scene flow. In driver assistance, the common assumption is made that objects are limited by their *up-down* motion, mathematically the translation in Y -direction. This restrains the three-dimensional motion to the two-dimensional X - Z plane. The consideration of this additional constraint in the scene flow estimation process will yield more accurate motion results.

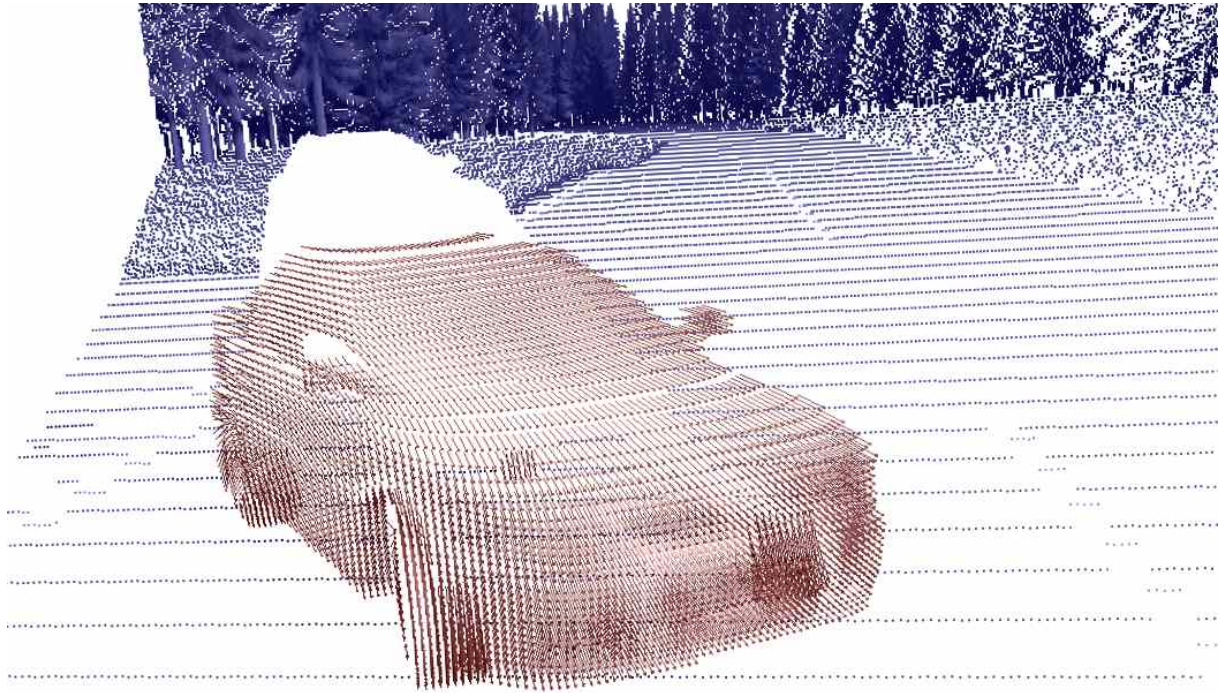
Another approach to boost accuracy is integration over time. In this chapter, only two consecutive images were considered to calculate the scene flow. Similar to the work presented in [64, 112], one would expect that using a Kalman filter to track the position and motion of every image point improves the accuracy of dense scene flow estimates.

As a last research topic, I would like to mention feedback loops. A feedback loop is used for instance in [145], where segmentation and optical flow are estimated in a single framework. The authors propose to segment the optical flow field into regions of similar motion and gray values, yielding accurate motion boundaries and precise optical flow fields for the single segments. The next chapter derives a segmentation technique based solely on the scene flow result to detect and segment independently moving objects. How this segmentation can be used to, in turn, improve the scene flow result, is an interesting idea for future research.

Part II

Visual Kinesthetic Cognition (Application)

Flow Cut - Moving Object Segmentation



Life is pointless without geometry.

Jens Klappstein

Contents

6.1 Segmentation Algorithm	79
6.1.1 Energy Functional	79
6.1.2 Graph Mapping	80
6.2 Deriving the Motion Metrics	81
6.2.1 Monocular Motion Analysis	81
6.2.2 Stereo Motion Analysis	84
6.3 Experimental Results and Discussion	88
6.3.1 Robust Segmentation	89
6.3.2 Comparing Monocular and Binocular Segmentation	89
6.4 Ideas for Future Research	90

This chapter presents the detection and segmentation of moving traffic participants. Both steps are based on image motion analysis using either optical flow, presented in Chapter 4, or scene flow, found in Chapter 5.

Most hazards in traffic situations result from moving traffic participants. Hence, detecting moving objects is a crucial step for many driver assistance systems. I call this special part of machine visual kinesthesia *flow cut*.

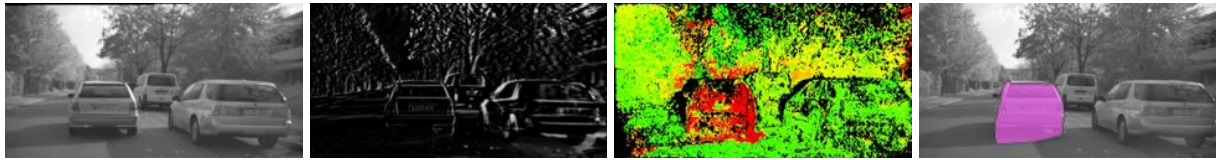


Figure 6.1: From left to right: input image, difference image between two consecutive frames, motion likelihood, and segmentation result. With the motion likelihood derived from scene flow, the segmentation of the moving object becomes possible although the camera itself is moving.

Related Work Classically, moving objects are separated from the stationary background by *change detection* (e.g. [128]). But if the camera is moving in a dynamic scene, motion fields become rather complex. Thus, the classic change detection approach is not suitable as can be seen in Fig. 6.1 (second from left). The goal in this chapter is to derive a segmentation of moving objects for the more general dynamic setting. The motion of the camera itself is not constrained nor are assumptions implied on the structure of the scene, such as rigid body motion.

Rigid objects constrain the motion onto sub-spaces which yield efficient means to segment dynamic scenes using two views [139]. Another approach is used in [46], where the segmentation process is solved efficiently by incorporating a shape prior. If nothing about object appearance is known, the segmentation clearly becomes more challenging.

High-dynamic scenes with a variety of different conceivable motion patterns are especially challenging and reach the limits of many state-of-the-art motion segmentation approaches (e.g. [57, 82]). The segmentation algorithm derived in this chapter can handle such scene dynamics implicitly with multiple independently moving object and additional camera motion. Although the camera motion is not constrained, it is assumed that the camera motion is (approximately) known (e.g. computed with means presented in Chapter 3).

In [151] the authors use dense optical flow fields over multiple frames and estimate the camera motion and the segmentation of a moving object by bundle adjustment. The necessity of rather long input sequences however limits its practicability; furthermore, the moving object has to cover a large part of the image in order to detect its motion. The closest work related to the approach presented in this chapter is the work presented in [4]. It presents a monocular and a binocular approach to moving object detection and segmentation in high-dynamic situations using sparsely tracked features over multiple frames. Here, the focus is set on moving object detection (instead of tracking) and the minimal number of two consecutive stereo pairs is used.

Chapter Overview Fig. 6.2 illustrates the segmentation pipeline. The segmentation is performed in the image of a reference frame (left frame at time t) employing the graph cut segmentation algorithm [41]. The motion cues used are derived from dense scene flow. To my knowledge, the direct use of dense scene flow estimates for the detection and segmentation of moving objects is novel.

In Section 6.1 the core graph cut segmentation algorithm is presented. It minimizes an energy consisting of a motion likelihood for every pixel and a length term, favoring segmentation boundaries along intensity gradients.

The employed motion likelihoods are derived from dense scene flow in Section 6.2. In the monocular setting, only the optical flow component of the scene flow is used. Compensating for the camera motion is a prerequisite step to detecting moving objects. Additionally, one has to deal with inaccuracies in the estimates. I will explain how the variances of the flow vectors, derived from reliability measures in Chapter 5, are used to derive the motion likelihoods.

In Section 6.3 the monocular method and the binocular method for the segmentation of independently moving objects in different scenarios are compared. It will be systematically shown, that the consideration of inaccuracies when computing the motion likelihoods for every pixel yields increased robustness for the segmentation. Furthermore, the limits of the monocular

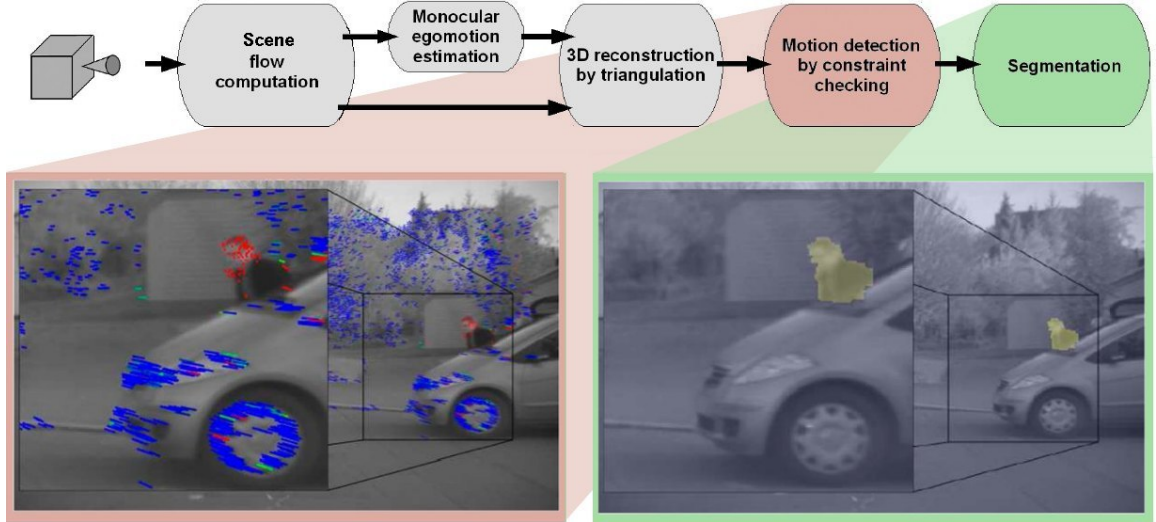


Figure 6.2: Work flow for the segmentation of independently moving objects. The lower two images show the main two steps: the *motion likelihood* result where red denotes independent object motion and the *segmentation* result. In the images sparse features are used for better visualization.

and binocular segmentation methods are demonstrated and ideas to overcome these limitations are given.

6.1 Segmentation Algorithm

6.1.1 Energy Functional

The segmentation of the reference frame into parts representing moving and stationary objects can be expressed by a binary labelling of the pixels,

$$\mathcal{L}(\mathbf{x}) = \begin{cases} 1 & \text{if the pixel } \mathbf{x} \text{ is part of a moving object} \\ 0 & \text{otherwise.} \end{cases} \quad (6.1)$$

The goal is now to determine an optimal assignment of each pixel to *moving* or *not moving*. There are two competing constraints: Firstly, a point should be labelled *moving* if it has a high motion likelihood ξ_{motion} derived from the scene flow information and vice versa. Secondly, points should favor a labelling which matches that of their neighbors. Both constraints enter a joint energy of the form

$$E(\mathcal{L}) = E_{\text{data}}(\mathcal{L}) + \lambda E_{\text{reg}}(\mathcal{L}), \quad (6.2)$$

where λ weighs the influence of the regularization force. The data term is given by

$$E_{\text{data}} = - \sum_{\Omega} \left\{ \mathcal{L}(\mathbf{x}) \xi_{\text{motion}}(\mathbf{x}) + (1 - \mathcal{L}(\mathbf{x})) \xi_{\text{static}}(\mathbf{x}) \right\} \quad (6.3)$$

on the image plane Ω , where ξ_{static} is a fixed prior likelihood of a point to be static. The regularity term favors labellings of neighboring pixels to be identical. This regularity is imposed more strongly for pixels with similar brightness, because it is assumed that neighboring pixels of similar brightness are more likely to represent the same object:

$$E_{\text{reg}} = \sum_{\Omega} \left\{ \sum_{\hat{\mathbf{x}} \in \mathcal{N}_4(\mathbf{x})} g(I(\mathbf{x}) - I(\hat{\mathbf{x}})) |\mathcal{L}(\hat{\mathbf{x}}) - \mathcal{L}(\mathbf{x})| \right\}, \quad (6.4)$$

where \mathcal{N}_4 is the 4 neighborhood (upper, lower, left, right) of a pixel and $g(\cdot)$ is a positive, monotonically decreasing function of the brightness difference between neighboring pixels. Here $g(z) = \frac{1}{z+\alpha}$ with a positive constant α is used.

6.1.2 Graph Mapping

Summarizing the above equations yields

$$E(\mathcal{L}) = \sum_{\Omega} \left\{ -\mathcal{L}(\mathbf{x}) \xi_{\text{motion}}(\mathbf{x}) - (1 - \mathcal{L}(\mathbf{x})) \xi_{\text{static}}(\mathbf{x}) + \lambda \sum_{\hat{\mathbf{x}} \in \mathcal{N}_4(\mathbf{x})} \frac{|\mathcal{L}(\hat{\mathbf{x}}) - \mathcal{L}(\mathbf{x})|}{|I(\mathbf{x}) - I(\hat{\mathbf{x}})| + \alpha} \right\}. \quad (6.5)$$

Due to the combinatorial nature, finding the minimum of this energy is equivalent to finding the s - t -separating cut with minimum costs of a particular graph $\mathcal{G}(v, s, t, e)$, consisting of nodes $v(\mathbf{x})$ for every pixel \mathbf{x} in the reference image and two distinct nodes: the source node s and the target node t [83]. The figure on the right illustrates this mapping. The edges e in this graph connect each node with the source, target, and its \mathcal{N}_4 neighbors. The individual edge costs are defined as follows:

edge	edge cost
source link: $s \rightarrow v(\mathbf{x})$	$-\xi_{\text{motion}}(\mathbf{x})$
target link: $v(\mathbf{x}) \rightarrow t$	$-\xi_{\text{static}}(\mathbf{x})$
\mathcal{N}_4 neighborhood: $v(\hat{\mathbf{x}}) \leftrightarrow v(\mathbf{x})$	$\lambda \frac{1}{ I(\mathbf{x}) - I(\hat{\mathbf{x}}) + \alpha}$

Table 6.1: Flow cut edge costs.

The cost of a cut in the graph is computed by summing up the costs of the cut (also *removed*) edges. Removing the edges of an s - t -separating cut from the graph yields a graph where every node v is connected to exactly one terminal node: either to the source s or to the target t .

If we define nodes that are connected to the source as static and those connected to the target as moving, it is easy to see that the cost of an s - t -separating cut is equal to the energy in Equation (6.5) with the corresponding labelling, and vice versa. Thus, the minimum s - t -separating cut yields the labeling that minimizes Equation (6.5). The minimum cut is found using the graph cut algorithm from [41].

Clearly, the result depends on the costs of the edges, especially on the regularization parameter λ . If λ is low, the segmentation only contains single pixels whereas a high value of λ results in only one small segment (or no segment at all) because removing edges connected to the source or the target becomes less costly than removing those edges connecting image pixels. Both situations can be seen in Figure 6.4.

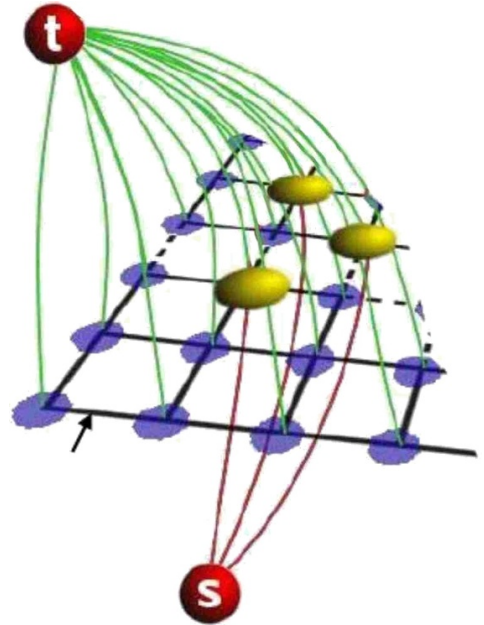


Figure 6.3: Illustration of the graph mapping. Red connections illustrate graph edges from the source node s to the nodes, green connections illustrate graph edges from nodes to the target node t . Note, that the ξ_{motion} likelihood may be sparse due to occlusion. In the illustration only pixels with yellow spheres contribute to this motion likelihood. Black connections (indicated by the arrow) illustrate edges between neighboring pixels.

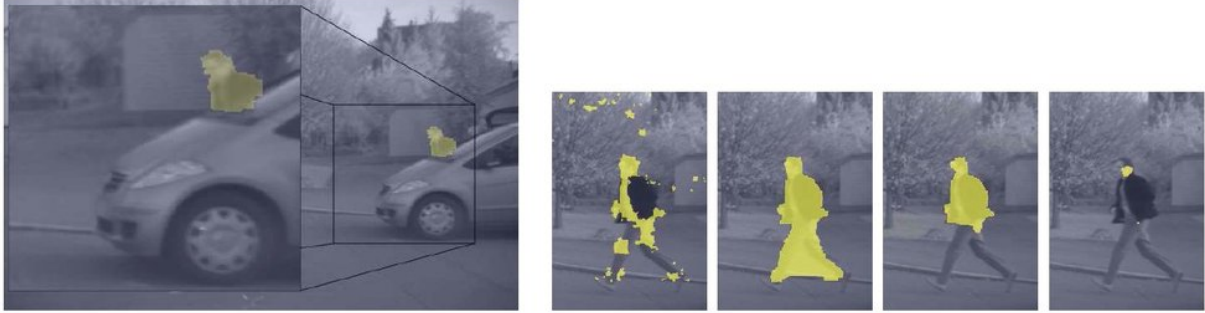


Figure 6.4: The images on the left show the segmentation for a moving pedestrian appearing behind a stationary vehicle. Outliers are rejected and the segmentation border is accurate. The four images on the right show the influence of the edge costs on the segmentation result (later in the sequence). While small edge costs result in many small segments with only a few pixels (left), high edge costs result in very few small regions (such that the number of cut edges is minimized, right). From left to right: $\lambda = \{1.5, 50, 500, 1000\}$.

Speed up techniques for flow vector segmentation can be achieved using the Multi-Resolution Graph Cut presented in Appendix D. In the next Section, I will derive the $\xi_{\text{motion}}(\mathbf{x})$ likelihoods from the scene flow estimates.

6.2 Deriving the Motion Metrics

This section derives motion constraints to detect independently moving objects (IMOs) in image sequences. The key idea to detect moving objects is to evaluate the hypothesis

“the object is not stationary.”

This is done by virtually reconstructing the three-dimensional translation vector for every flow vector within the image. The length of the reconstructed three-dimensional translation vector is then evaluated resulting in a motion likelihood for the corresponding flow vector. This reconstruction process needs both, the flow vector and the motion of the camera between the two time instances when the images were taken (recall Figure 6.2). More technically speaking one has to distinguish between motion caused by the ego-vehicle and motion caused by dynamic objects in the scene. The motion of the ego-vehicle greatly complicates the problem of motion detection because simple background subtraction of successive images yields no feasible result.

Analogous to the presentation of the monocular optical flow in Chapter 4 and the stereoscopic scene flow in Chapter 5, this section derives motion constraints for both cases. In the monocular case, the distance of world points to the camera is not known in the general setting (non-static scene) and hence a full three-dimensional reconstruction is not possible. However, due to the fundamental matrix geometry certain motion constraints can be derived. Subsection 6.2.1 presents this in more detail.

In the stereo case image points can be triangulated and the full three-dimensional translation vector can be reconstructed. Subsection 6.2.2 derives a likelihood that this translation vector does not vanish when subtracting the ego motion (ergo that the translation vector does belong to a moving object).

6.2.1 Monocular Motion Analysis

There is a fundamental weakness of monocular three-dimensional reconstruction when compared to stereo methods: moving points cannot be correctly reconstructed by monocular vision. This is due to the camera and unknown object movement between the two sequential images. Hence, optical flow vectors are triangulated, assuming that every point belongs to a static object. Such

triangulation is only possible, if the displacement vector itself does not violate the fundamental matrix constraint (see Chapter 3). Needless to say, that every track violating the fundamental matrix constraint belongs to a moving object and the distance to the fundamental rays directly serves as a motion likelihood.

Using this approach, flow vectors need to be projected onto the epipolar lines in order to triangulate these flow vectors. However, even if flow vectors are aligned with the epipolar lines, they may belong to moving objects. This is due to the fact that the triangulated point may be located behind one of the two camera positions or below the ground surface (see Figure 6.5). Certainly such constellations are only virtually possible, assuming that the point is stationary. In reality such constellations are prohibited by the law of physics. Hence, such points must be located on moving objects.

In summary, a point is detected as moving if its 3D reconstruction is identified as erroneous. To this end, one checks whether the reconstructed 3D point violates the constraints of a static 3D point, which are the following (note: these constraints are necessary, not sufficient):

Epipolar Constraint:

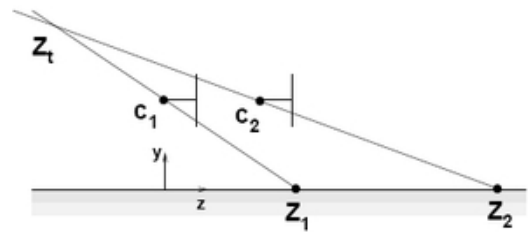
This constraint expresses that viewing rays of a static 3D point in the two cameras (lines joining the projection centres and the 3D point) must meet. A moving 3D point in general induces skew viewing rays violating this constraint.

Positive Depth Constraint:

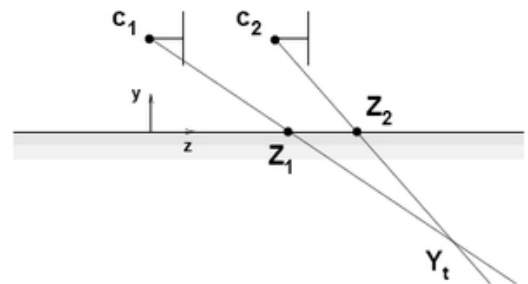
The fact that all points seen by a camera must lie in front of it is known as the positive depth constraint. It is also called *cheirality constraint* and illustrated in Figure 6.5. If viewing rays intersect behind the camera, then the actual 3D point must be moving.

Positive Height Constraint:

All 3D points must lie above the road plane. This principle is usually true for traffic scenes and illustrated in Figure 6.5. If viewing rays intersect underneath the road, then the actual 3D point must be moving. This constraint requires additional knowledge about the normal vector of and the camera distance to the road surface.



Positive Depth Constraint



Positive Height Constraint

Figure 6.5: Side view of erroneous triangulations. The camera moves from c_1 to c_2 . A point on the road surface is being tracked from Z_1 to Z_2 . Figure courtesy of J. Klappstein.

Evaluating the constraints presented above results in a likelihood whether the flow vector is located on a moving object. This likelihood then serves as input for the segmentation step. If a third camera view is available, the trifocal constraint yields an additional observation: a triangulated 3D point utilizing the first two views must triangulate to the same 3D point when the third view comes into consideration [79]. According to [79] there are no further constraints in the monocular case. The next subsections present the use of the above mentioned constraints in literature and how the motion metric is obtained.

Monocular Motion Constraints in Literature

Existing motion detection schemes exploit a subset of the above constraints either directly or indirectly. A popular scheme is the *angle criterion* [53, 144] which uses the direction of optical

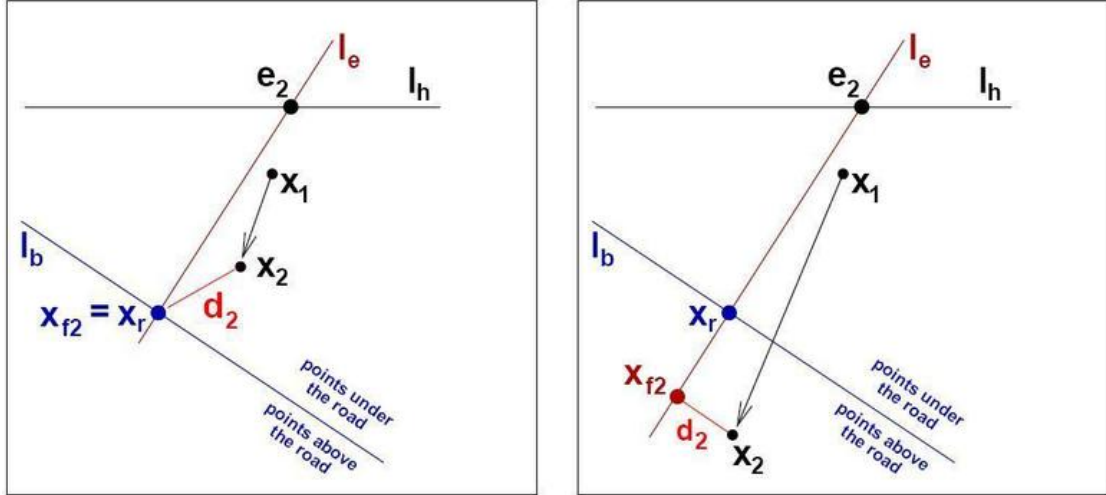


Figure 6.6: Two-view error evaluating the positive height constraint. The correspondence $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ violates the epipolar constraint. Additionally, in the *left* diagram the positive height constraint is violated. The two-view error d_2 measures the distance of \mathbf{x}_2 to the closest point fulfilling all constraints.

Figure courtesy of Jens Klappstein.

flow vectors. When moving purely translational in a scene, all flow vectors are parallel to the corresponding epipolar lines and point away from the epipole (focus of expansion). This holds true for the entire static scene. Hence, if a measured optical flow vector deviates from this expected flow direction (i. e. if the angle between the measured and the expected direction is not zero), the corresponding 3D point is moving. This angle criterion indirectly exploits the epipolar and the positive depth constraint.

Another popular scheme is the *planar motion parallax*. It is defined as the deviation of the measured optical flow from the expected flow on the road plane. The difference between a flow vector and the assumed flow vector for the same image position on the road surface is called parallax vector. For correspondences violating the positive height constraint, the parallax vector points towards the epipole because the measured flow is shorter than expected. [34, 65] evaluate the planar motion parallax.

A scheme exploiting the *trifocal constraint* is presented in [68]. It not only detects moving points but also clusters them. However, the computational burden is high.

The scheme used in this section is taken from [78]. In this section, it is now reproduced how the epipolar constraint, the positive depth constraint, and the positive height constraint are evaluated quantitatively. In the work flow diagram (Figure 6.2), reconstruction and detection are shown as two separate steps. However, the actual algorithm avoids the explicit reconstruction in favor of reduced computational complexity. This results in a motion metric measuring to which extent the constraints are violated. It is correlated to the likelihood that the point is moving (i. e. higher values indicate a higher probability).

The algorithm input is a point correspondence $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$. Additionally, the fundamental matrix \mathbf{F} and the rotation matrix \mathbf{R} between the two views, as well as the camera calibration matrix \mathbf{K} , are needed. The principle is illustrated in Figure 6.6 in the image domain.

The point in the first image, \mathbf{x}_1 , defines the epipolar line \mathbf{l}_e in the second image going through the epipole of the second image, e_2 . In a first step, the horizon line is computed via

$$\mathbf{l}_h = \mathbf{K}\mathbf{R}_R \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \times \mathbf{K}\mathbf{R}_R \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

where \mathbf{R}_R is the rotation matrix between the vehicle coordinate system and the camera coordinate system (see Figure 3.5). For points \mathbf{x}_1 above this horizon line, the positive depth constraint is evaluated. For points below the horizon line, the positive height constraint is evaluated. With either the infinite homography $\mathbf{H}_\infty = \mathbf{K}\mathbf{R}\mathbf{K}^{-1}$ or the road homography transformation \mathbf{H}_R between the two camera views (see [78]) a border point \mathbf{x}_r on the epipolar line l_e is computed as

$$\mathbf{x}_r = \begin{cases} \mathbf{H}_\infty \mathbf{x}_1 & \text{if } \mathbf{x}_1^\top \mathbf{l}_h \geq 0 \\ \mathbf{H}_R \mathbf{x}_1 & \text{otherwise.} \end{cases}$$

The border line \mathbf{l}_b is perpendicular to the epipolar line \mathbf{l}_e and intersects the calculated border point \mathbf{x}_r . It computes as

$$\mathbf{l}_b = \begin{bmatrix} 0 & (\mathbf{x}_r)_3 & 0 \\ -(\mathbf{x}_r)_3 & 0 & 0 \\ (\mathbf{x}_r)_2 & -(\mathbf{x}_r)_1 & 0 \end{bmatrix} \mathbf{F} \mathbf{x}_1.$$

The point \mathbf{x}_{f2} fulfilling all constraints depends on the location of \mathbf{x}_2 . If \mathbf{x}_2 lies on the same side of \mathbf{l}_b as the epipole \mathbf{e}_2 , then \mathbf{x}_{f2} is equal to the border point because every point on the epipolar line which is closer to \mathbf{x}_2 violates either the positive depth or the positive height constraint. On the other hand, if the epipole and \mathbf{x}_2 are located on different sides of the border line, the point fulfilling all three monocular two-view constraints is the projection of \mathbf{x}_2 onto the epipolar line. Due to the use of homogeneous entities, this computes as

$$\mathbf{x}_{f2} = \begin{cases} \mathbf{x}_r & \text{if } \mathbf{x}_2^\top \mathbf{l}_b \cdot \mathbf{e}_2^\top \mathbf{l}_b > 0 \\ \mathbf{d} \times \mathbf{x}_2 \times \mathbf{l}_e & \text{otherwise,} \end{cases}$$

with $d = ((\mathbf{l}_e)_1, (\mathbf{l}_e)_2, 0)^\top$. The resulting two-view reconstruction error, or motion likelihood, is the Euclidean distance

$$d_2 = \text{dist}(\mathbf{x}_2, \mathbf{x}_{f2}).$$

It is even more elaborate to take the Mahalanobis distance of the two image points allowing a weighting by accuracies. Assuming a covariance matrix Σ_x for the flow vector from \mathbf{x}_1 to \mathbf{x}_2 , this yields the monocular motion likelihood

$$\xi_{\text{motion}}(\mathbf{x}) = \sqrt{\{\mathbf{x}_2 \Sigma_x^{-1} \mathbf{x}_{f2}\}}. \quad (6.6)$$

The larger this distance value is, the more likely the flow vector belongs to a moving object. However, not every motion can be detected. The degenerate case where, object motion and camera motion is parallel, positive depth and height is fulfilled, and flow vectors lie exactly on the epipolar rays, is not possible to detect. This is due to the fact that a triangulation of a point is only virtually possible using monocular cameras. With a stereo camera such degenerate cases are solved. The next subsection derives motion constraints for such stereo camera systems.

6.2.2 Stereo Motion Analysis

In this section, the stereo image pairs of two consecutive camera images are known. With these two stereo image pairs, the scene flow which was derived in Section 5 can be computed. For every image pixel (x, y) this scene flow information yields an optical flow vector (u, v) , representing the change in position within the image over time and a disparity change d' , encoding the change in disparity d .

If these four values, $d, d', u,$ and v are determined for an image pixel (x, y) , its world position can be reconstructed for each of the two consecutive time stamps. Computing the difference of these two world points yields the translation vector; and hence the velocity as the length of this translation vector. A point is transformed from the image coordinates (x, y, d) into world coordinates (X, Y, Z) according to $X = (x - x_0) \frac{b}{d}$, $Y = (y - y_0) \frac{b}{d}$, and $Z = \frac{f b}{d}$, where b is the

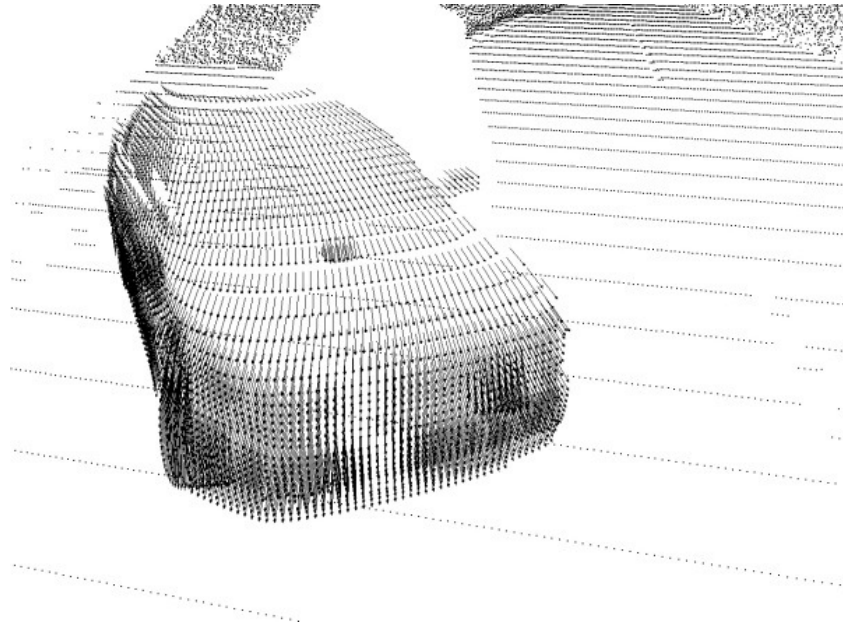


Figure 6.7: Reconstructed three dimensional translation vectors for every pixel of an image with ground truth values for the image flow estimates d , d' , u , and v .

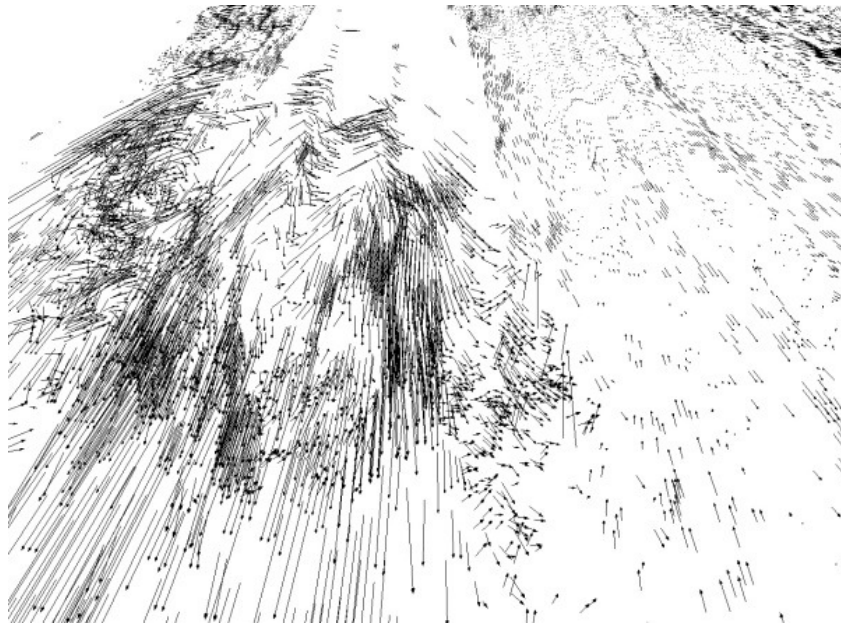


Figure 6.8: Reconstructed three dimensional translation vectors for every pixel of an image with estimated values for the image flow estimates d , d' , u , and v . The vehicle is about 50m away from the camera; compare with Figure 6.7, where ground truth values are used.

basis length of the stereo camera system, f the focal length of the camera, and (x_0, y_0) its principal point. Transforming the points (x, y, d) and $(x + u, y + v, d + d')$ into world coordinates and compensating the camera rotation \mathbf{R} and movement \mathbf{T} yields the three-dimensional residual translation (or motion) vector \mathbf{M} with

$$\mathbf{M} = \frac{b}{d} \mathbf{R} \begin{bmatrix} x - x_0 \\ y - y_0 \\ f \end{bmatrix} - \frac{b}{d + d'} \begin{bmatrix} x + u - x_0 \\ y + v - y_0 \\ f \end{bmatrix} + \mathbf{T} \quad (6.7)$$

The (absolute) translation vectors for every image point of a rendered scene with known camera motion (exact values are also known for d , d' , u , and v) are displayed in Figure 6.7. The translation vectors point into the direction of movement for every reconstructed scene flow vector. Points which are located on the road surface and background are stationary; hence their vector length is zero. In contrast, points which are located on the vehicle are moving and represented by translation vectors.

The length of a vector seems to be a valid measure for the amount of movement of a world point. Calculating the length of the reconstructed translation vector directly yields a motion metric for the corresponding scene flow vector. Unfortunately scene flow computation derived in Chapter 5 is an estimation procedure and the values d , d' , u , and v are not error-free. The estimated scene flow variables are noisy as has been shown in the investigation in Section 5.4.

Taking the estimated scene flow values for the exact same image frame as in Figure 6.7 and displaying the translation vectors yields a much less significant three-dimensional velocity reconstruction. This can be seen in Figure 6.8. Due to the noise in the input data for the three dimensional reconstruction, a feasible conclusion about the amount of motion cannot be drawn from the length of the translation vector. The errors in the input data (here scene flow or optical flow) are propagated into the resulting translation vector which yields to perturbed translation vectors. Thus, one has to take into account the variances of the scene flow estimates.

Scene Flow Motion Metric

Using error propagation, the Mahalanobis length of the translation vector is computed. Essentially, this incorporates the variances of the disparity and the scene flow estimates, and the camera ego motion parameters. Here, the variances of the camera rotation are assumed to be negligible. Such procedure is possible because the estimation of the fundamental matrix from the complete optical flow field does yield vanishing variances for the rotational parts. However, fixed variances in the camera translation are used because the speed information from the velocity sensor of the ego-vehicle is rather inaccurate. With the variances σ_u^2 , σ_v^2 , σ_p^2 , and σ_d^2 for the scene flow and $\sigma_{\mathbf{T}}^2$ for the ego motion this yields the Mahalanobis distance [94, 99]:

$$\xi_{\text{motion}}(\mathbf{x}) = Q = \sqrt{\mathbf{M}^\top \left(\mathbf{J}^\top \text{diag}(\sigma_u^2, \sigma_v^2, \sigma_p^2, \sigma_d^2, \sigma_{\mathbf{T}}^2) \mathbf{J} \right)^{-1} \mathbf{M}}, \quad (6.8)$$

where \mathbf{J} is the Jacobian of Equation (6.7) with respect to the scene flow estimates and the ego motion. This formula is a weighted dot product of the translation vector. The weighting by the inverse covariance matrix yields a normalization of the translation vector where every dimension contributes equally to the result, a new random variable Q .

Assuming a Gaussian error distribution, the squared Mahalanobis distance Q of the translation vector is χ^2 distributed and outliers are found by thresholding this distribution, using the assumed quantiles of the χ^2 distribution. For example, the 95% quantile of a distribution with three degrees of freedom is 7.81, the 99% quantile lies at 11.34. Hence if the Mahalanobis distance is above 11.34, a point is moving with a probability of more than 99%.

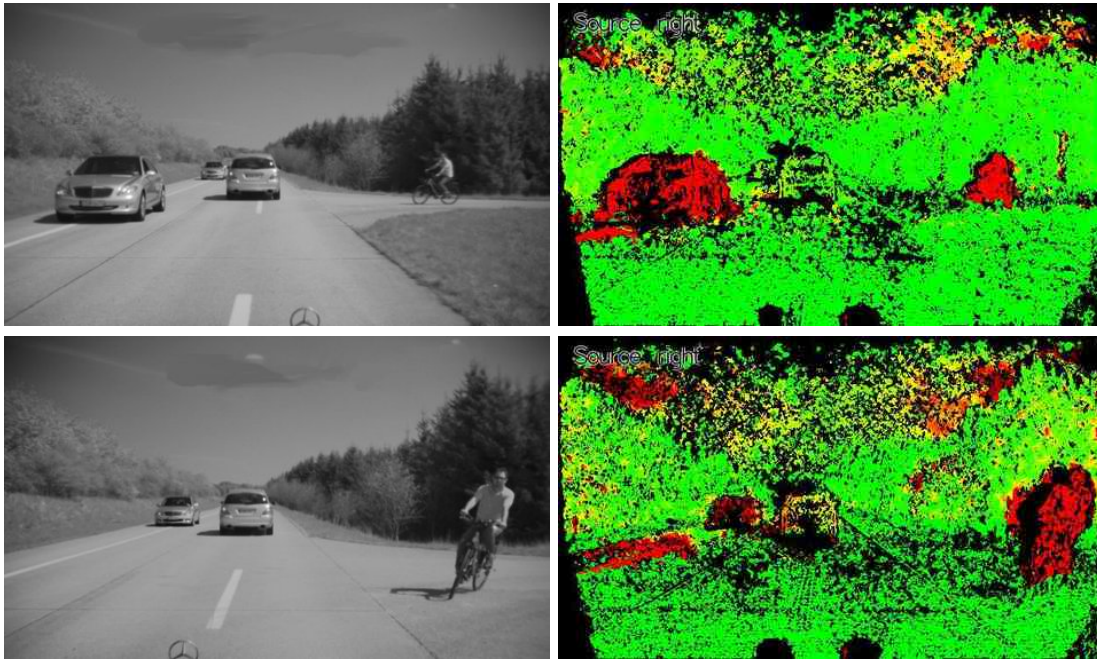
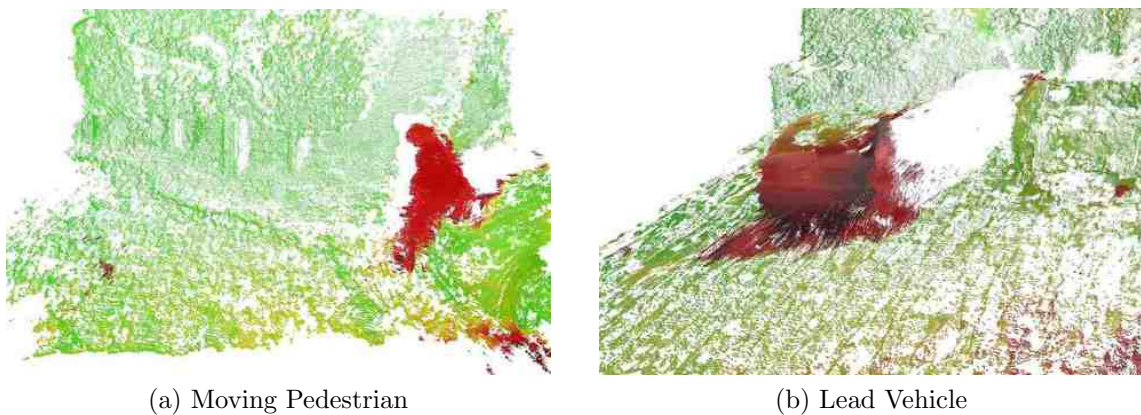


Figure 6.9: Two examples of the motion metric defined in Section 6.2.2. Left images show the original image, right images show the motion metric results, green \Leftrightarrow red represents low \Leftrightarrow high likelihood that point is moving.

If the underlying distribution is Laplacian, the quantiles are different. Here the 95% quantile lies at 8.92 and the 99% quantile is found at 15.92. Note, that the distribution of the scene flow values in Chapter 5 was found to be Laplacian.

Figure 6.9 demonstrates results using this metric. Note that this metric computes a value at every scene point, which is not occluded. Another two examples of results obtained using this metric can be seen in Figure 6.10. In the figures, it is easy to identify what parts of the scene are static, and which parts are moving. The movement metric Q only identifies the likelihood of a point being stationary, it does not provide any speed estimates. The remaining part of this section derives such velocity estimates.



(a) Moving Pedestrian

(b) Lead Vehicle

Figure 6.10: Results using the Mahalanobis distance metric Q . 6.10a shows a pedestrian running from behind a vehicle. 6.10b shows a lead vehicle driving forward. Colour encoding is Q , i. e. the hypothesis that the point is moving, green \leftrightarrow red \equiv low \leftrightarrow high.

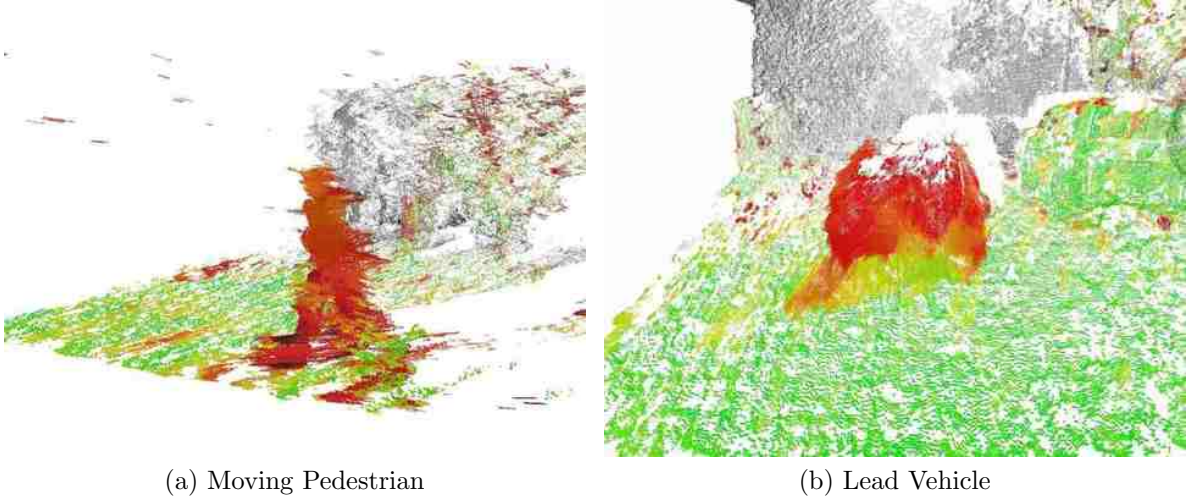


Figure 6.11: Results using speed S and its standard deviation σ_S . 6.11a shows the pedestrian running with a speed of 3.75 m/s. 6.11b shows a lead vehicle driving forward with a speed of 12.5 m/s. Colour encoding is S , green \leftrightarrow red \equiv stationary \leftrightarrow moving. σ_S is encoded using saturation, points in the distance are therefore grey or black.

Speed Metric

The Q metric omitted any information about velocity. This subsection derives information about speed and the associated certainty of the speed estimate. To estimate the speed S , the L^1 -norm (length) of the displacement vector is calculated:

$$S = \|\mathbf{M}\| . \quad (6.9)$$

The problem is that points at large distances are always estimated as moving. This is because small disparity changes yield large displacements in 3D. If inaccuracies are present in the length computation one can still not derive accurate speed information. One way around the problem is to give a lenient variance of the speed measurement σ_S^2 . Such a measurement is given by the *spectral norm* of the covariance matrix. This involves computing the eigenvalues of the squared matrix, then taking the square root of the maximum eigenvalue,

$$\sigma_S^2 = \|\Sigma_{\mathbf{M}}\| = \sqrt{\lambda_{\max}(\Sigma_{\mathbf{M}}^T \Sigma_{\mathbf{M}})} . \quad (6.10)$$

Using this we now have a speed S and associated variance σ_S^2 . Using these metrics leads to the examples in Figure 6.11. In this figure, it is easy to identify the speed of moving targets, and also how confident we are of the speed measurement. The pedestrian in Figure 6.11a had a displacement of 15 cm with a frame rate of 25 Hz, i.e. 3.75 m/s. The vehicle in 6.11b had a displacement of 50cm, i.e. 12.5 m/s. In both examples moving objects and the associated variance (or one may say certainty) of their speed measure can be identified.

6.3 Experimental Results and Discussion

In this section I present results which demonstrate the accurate segmentation of moving objects using scene flow. In the first part, it is shown that the presented reliability measures greatly improve the segmentation results when compared to a fixed variance for the disparity and scene flow variables. In the second part, the segmentation results using the monocular and binocular motion segmentation approaches are compared.

6.3.1 Robust Segmentation

Figure 6.12 illustrates the importance of using the reliability measures to derive individual variances for the scene flow variables. If the propagation of uncertainties is not used at all, the segmentation of moving objects is not possible (top row). Using the same variance for every image pixel the segmentation is more meaningful, but still outliers are present in both, the motion likelihoods and the segmentation results (middle row). Only when the reliability measures are used to derive individual variances for the pixels the segmentation is accurate and not influenced by outliers (bottom row).

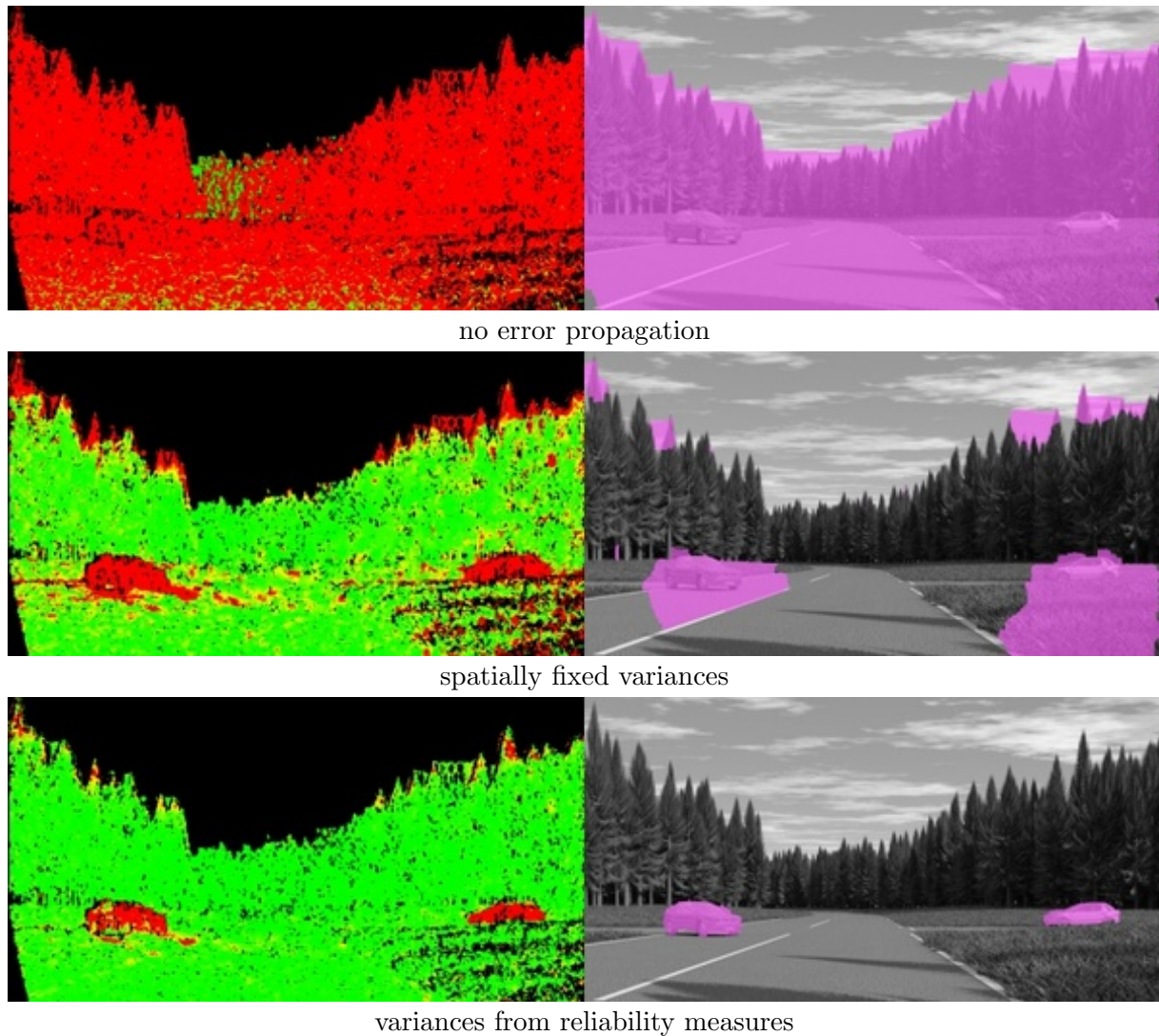


Figure 6.12: Results for different error propagation methods. The *left* images show the motion likelihoods and the *right* images the segmentation results.

6.3.2 Comparing Monocular and Binocular Segmentation

A binocular camera system will always outperform a monocular system, simply because more information is available. However, in many situations a monocular system is able to detect independent motion and segment the moving objects in the scene. In this section we demonstrate the segmentation of independently moving objects using a monocular and a binocular camera system and discuss the results.

In a monocular setting, motion which is aligned with the epipolar lines cannot be detected without prior knowledge about the scene. Amongst other motion patterns, this includes objects

moving parallel to the camera motion. For a camera moving in depth this includes all (directly) preceding objects and (directly) approaching objects. The *PrecedingCar* and *HillSide* sequences in Figure 6.13 show such constellations.

Using the ground plane assumption in the monocular setting (no virtually triangulated point is allowed to be located below the road surface) facilitates the detection of preceding objects. This can be seen in the *PrecedingCar* experiment, where the lower parts of the car become visible. If compared to the stereo settings, which does not use any information about scene structure, the motion likelihood for the lower part of the preceding car is more discriminative. However, if parts of the scene are truly located below the ground plane, as the landscape at the right in the *HillSide* experiment, these will always be detected as moving, too. Additionally, this does not help to detect approaching objects. Both situations are solved using a binocular camera.

If objects do not move parallel to the camera motion, they are essentially detectable in the monocular setting (*Bushes* and *Running* sequences in Figure 6.14). However, the motion likelihood using a binocular system is more discriminative. This is due to the fact that the three-dimensional position of an image point is known from the stereo disparity. Thus, the complete viewing ray for a pixel does not need to be tested for apparent motion in the images, as in the monocular setting. In the unconstrained setting (not considering the ground plane assumption), the stereo motion likelihood therefore is more restrictive than the monocular motion likelihood. Note, that non-rigid objects (as in the *Running* sequence in Figure 6.14) are detected as well as rigid objects and do not limit the detection and segmentation at any stage.

6.4 Ideas for Future Research

Building on the approach to scene flow estimation presented in Chapter 5, I proposed in this chapter an energy minimization method to detect and segment independently moving objects filmed in two synchronised video cameras installed in a driving car. The central idea is to assign, to each pixel in the image plane, a motion likelihood which specifies whether based on 3D structure and motion, the point is likely to be part of an independently moving object. Subsequently, these local likelihoods are fused in an MRF framework and a globally optimal spatially coherent labelling is computed using the min-cut max-flow duality. In challenging real world scenarios where traditional background subtraction techniques would not work (because the whole image content is moving), this approach enables to accurately localize independently moving objects. The results of the presented algorithm could directly be employed for automatic driver assistance.

Further research should focus on feedback loops in the whole motion estimation and segmentation process. Certainly, if motion boundaries and the segmentation of moving rigid objects are known, this provides additional cues for the motion estimation step. The work in [57] estimates piecewise parametric motion fields and a meaningful motion segmentation of the image in a joint approach. A similar approach could segment moving objects in the input images.

Another possibility for a feedback loop is the segmentation content itself for the segmentation stage. In the presented approach every motion, as long as it is not induced by the stationary scene, is segmented. Multiple objects may end up in the same segment and noise may influence segmentation boundaries. An individual object usually moves into the same direction and covers only a small disparity range. Such information could directly be used to iteratively improve the segmentation boundary and to separate individual objects.

Certainly, there are numerous other areas of research, such as the speeding up of the segmentation process (see also Appendix D) or using multiple frames of the sequence and propagating segmentation boundaries over time. This chapter presented a first application of scene flow; the next chapter will sketch concepts for three more applications and hopefully inspire the reader to come up with many more ideas for future research.

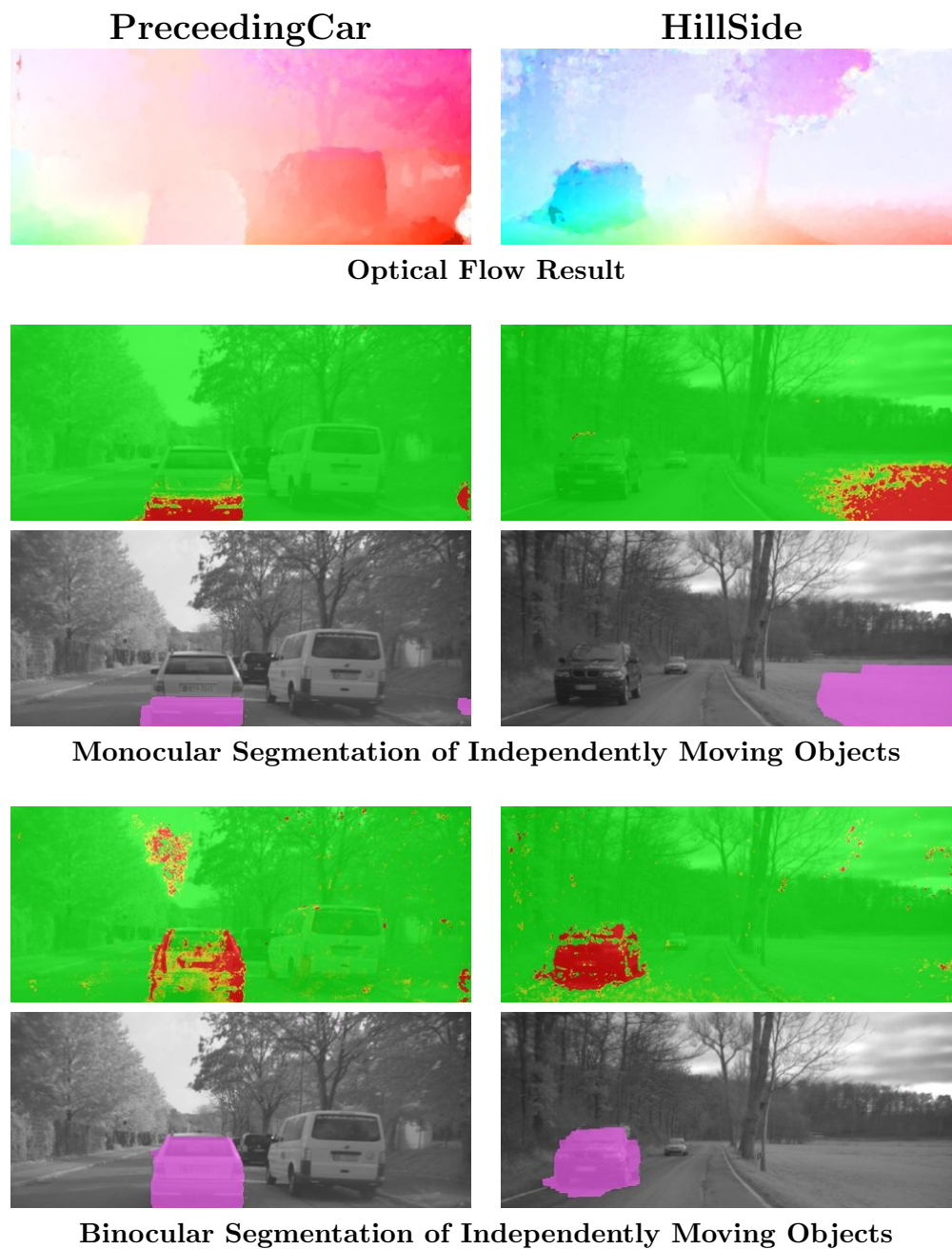


Figure 6.13: The figure shows the energy images and the segmentation results for objects moving parallel to the camera movement. This movement cannot be detected monocularly without additional constraints, such as a planar ground assumption. Moreover if this assumption is violated, this yields errors (as in the *HillSide* sequence). In a stereo setting prior knowledge is not needed to solve the segmentation task in these two scenes.

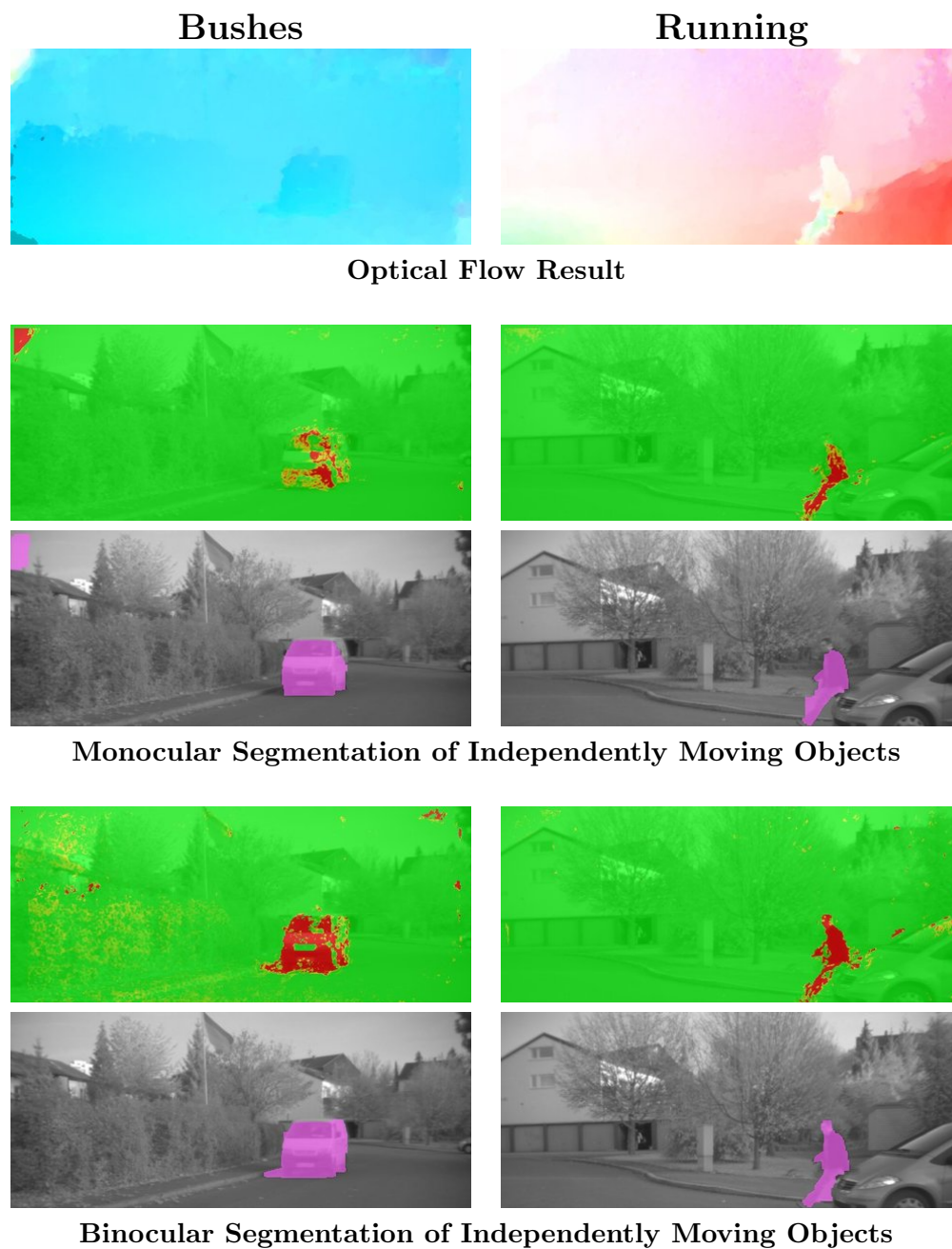


Figure 6.14: The figure shows the energy images and the segmentation results for objects which move not parallel to the camera motion. In such constellations a monocular as well as a binocular segmentation approach is successful. However, one can see in the energy images and in the more accurate segmentation results (the head of the person in the *Running* sequence) that stereo is more discriminative. Note, that the also non-rigid independently moving objects are segmented.

Research Ideas using Visual Kinesthesia



The future is in your hands.

© under the creative commons license, MichaelMarlatt, www.flickr.com

Contents

7.1	Extending Warp Cut	94
7.2	Motion from Motion	96
7.3	Dynamic Free Space	97

In this chapter, I discuss ideas for future research, building on the results obtained in this thesis. The precise estimation of optical flow and scene flow allows for novel approaches to many applications and problems arising in driver assistance. Here, concepts for three such applications in driver assistance are sketched:

1. **Object Segmentation** from scene flow vectors is only one way to detect moving objects. Especially if additional sensors are available (such as radar or lidar), information such as relative speed and distance to objects can be directly employed. The detection of stationary and moving objects then becomes applicable solely using the gray values and image warping. A joint approach, employing motion vectors and gray values in the segmentation process is within the scope of future research.
2. **Object Motion Estimation** is another research topic. In Chapter 3 methods were presented to estimate the motion of the camera from stationary points. The same principle can be used to estimate the relative motion between the camera and another (segmented) moving object. Then, the motion of the object can be inferred from the motion difference.
3. **Dynamic Free Space** has greatly been ignored in computer vision; it is important to note, that relative motion is a major issue though when using a radar sensor. In robotics, depth maps are commonly established from depth correspondences only. This might be motivated by history as laser and ultra-sound devices were used before stereo cameras became more popular. Using a camera system, tracking becomes possible and I am convinced that motion cues will one day be at least as important as depth cues. The idea of a dynamic free-space representation is outlined as another application of scene flow.

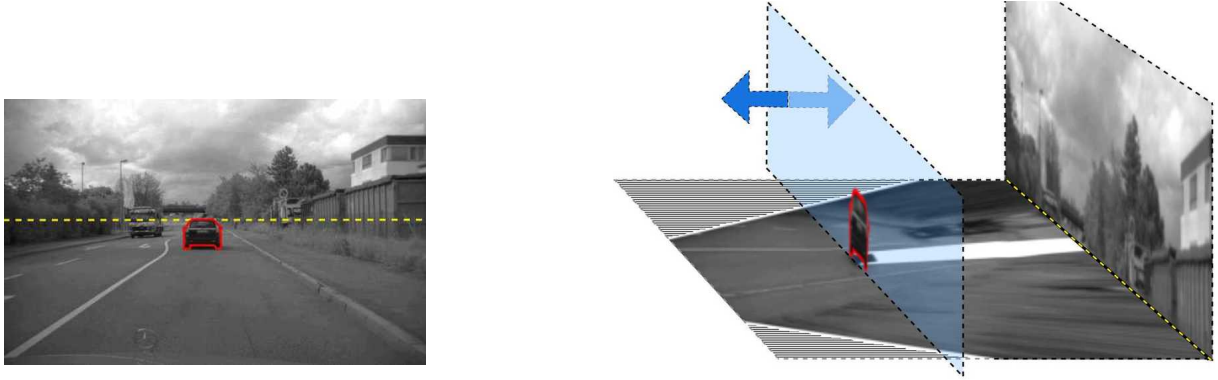


Figure 7.1: The *Warp Cut Segmentation* algorithm models the world with three planes: the ground plane and the plane at infinity, stitched together at the horizon, and the obstacle plane. Segmentation of the object is then performed on the brightness differences between the original first frame and the backward-motion-warped second frame for the respective planes (see Fig. 7.2). The approach is essentially limited because (additional) objects which violate the three-plane-assumption affect the results. The direct use of optical flow vectors may overcome some of these difficulties and yield an increase in robustness.

7.1 Extending Warp Cut

Autonomous collision avoidance in vehicles requires an accurate separation of obstacles from the background, particularly near the focus of expansion. In [15], I presented a technique for fast segmentation of stationary obstacles from video, recorded by a single camera that is installed in a moving vehicle. In [7], this approach was generalized for moving objects, where the distance and relative speed of the object was known from radar measurements.

The basic idea is illustrated in Figure 7.1. The input image is divided into three motion segments consisting of the ground plane, the background, and the obstacle. Due to the given scenario the following assumptions were imposed:

1. The street is approximately planar. Hence, the image motion in this area is described by a homography H_s . The homography can be approximated from the known camera motion and the camera parameters.
2. Visible object points on distant obstacles have approximately the same depth. Applying the weak perspective camera model, the motion field in the obstacle region is affine, which can be expressed by another homography H_o .
3. Finally, the background region, i.e., the region above the horizon can be approximated as a plane at infinity, which leads to a third homography H_b .

Consequently, there are three regions, each with a different motion model between frames. This constrained scenario allows for good initial estimates of the motion models for each segment, which are iteratively refined during segmentation. The separation of the obstacle region from the other two regions is done by the sought segmentation of the obstacle.

The street and background regions are separated a-priori by a horizontal line $y = y_{hor}$ that can be derived analytically from the camera parameters, which leaves us with a binary partitioning problem. This enables accurate obstacle segmentation without prior knowledge about the size, the shape, or the base point of obstacles.

Similar to the flow cut approach in Chapter 6, graph cut segmentation was used to segment the obstacle within the image, where the binary labeling $L_t(\mathbf{x})$ of each pixel $\mathbf{x} = (x, y)^\top$ is

$$L_t(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \text{ depicts the obstacle} \\ 0 & \text{otherwise.} \end{cases} \quad (7.1)$$

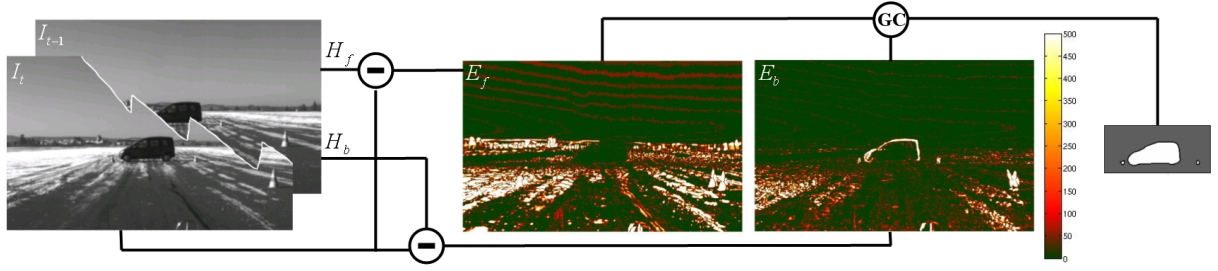


Figure 7.2: **Motion-compensated difference images.** From left to right: original gray value input images and difference images for foreground (E_f) and background (E_b) based on the squared difference between the motion-compensated image I_{t-1} and image I_t after the last iteration. The camera translation is 1.9 m. H_f and H_b denote foreground and background motion.

Segmentation by grouping similar gray values is not suitable in the presented context because the gray value of obstacles is not fixed and may be similar to the gray value of the street. Thus, the labeling is based on motion information. The classical approach to segmentation minimizes a joint energy on the labeling of the form

$$E(L_t) = E_{Data}(L_t) + \alpha E_{Smooth}(L_t) . \quad (7.2)$$

I will now review the data term in more detail: The key idea of distinguishing between obstacles and background is to penalize the difference between the current frame and the motion-compensated (*warped*) previous frame. Separate motion predictions are computed for the obstacle region (H_o) and the non-obstacle regions (H_b for background and H_s for the street region). Notice, that for the presented application this is much more sensible than the approaches described in [58, 128] as it allows one to drop the assumption of a static camera. The motion-compensated images are composed as follows (recall, that y_{hor} is the horizon in the image):

$$\text{Background:} \quad I_{0,t-1}^{mc}(\mathbf{x}) = \begin{cases} I_{t-1}(H_b(\mathbf{x})) & y < y_{hor} \\ I_{t-1}(H_s(\mathbf{x})) & y \geq y_{hor} \end{cases} , \quad (7.3)$$

$$\text{Object:} \quad I_{1,t-1}^{mc}(\mathbf{x}) = I_{t-1}(H_o(\mathbf{x})) . \quad (7.4)$$

Values between grid points are determined by bi-linear interpolation. Figure 7.2 shows the motion-compensated difference images for an example situation. The data term evaluates the consistency between the warped previous image and the current image. It consists of the sum over the squared differences between both images (recall, that $L_t(\mathbf{x}) \in \{0, 1\}$ denotes the labelling):

$$E_{Data}(L_t) = \sum_{\mathbf{x} \in \Omega} \left(I_t(\mathbf{x}) - I_{L_t(\mathbf{x}), t-1}^{mc}(\mathbf{x}) \right)^2 . \quad (7.5)$$

Such an approach is essentially limited, because (additional) objects which violate the three-plane-assumption affect the results. The direct use of optical flow vectors, combining the warp cut approach and the flow cut approach, may take away some of these limitations. On the other hand, gray value information can help to detect inconsistencies in the segmentation result derived from optical flow vectors (as presented in Chapter 6). Furthermore, the warp cut approach can be extended to stereo vision and combined with the flow cut approach employing scene flow vectors. Both ideas are expected to yield an increase in robustness and accuracy.

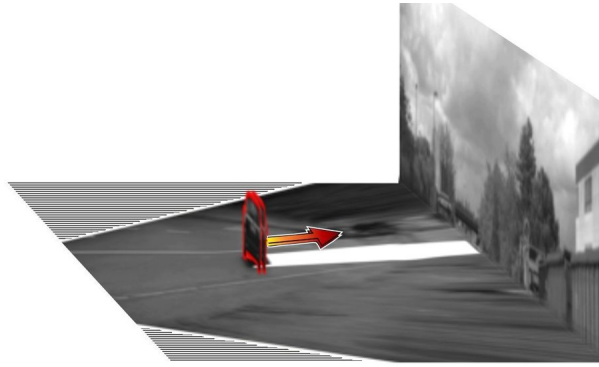


Figure 7.3: Motion from Motion. Once an object is segmented in the image, its relative motion can be estimated from the scene flow vectors. Compensating the camera motion directly yields the full motion state of the observed object from the two stereo image pairs.

7.2 Motion from Motion

The motion of other traffic participants is an important information for driver assistance systems. In [39] an algorithm was presented to estimate the position, the velocity, the acceleration, and the yaw rate of an oncoming vehicle with an extended Kalman filter. The filter uses sparsely tracked features and implements the dynamic motion model of a vehicle. Such an approach needs a low number of frames before it converges to the correct solution, depending on the accuracy of the initialization, as can be seen in Figure 7.4.

One idea for further research is to directly estimate the position, velocity, and yaw rate of previously segmented objects from dense scene flow (see Figure 7.3 for an illustration) and to use the obtained results in the initialization for the Kalman filter. Such approaches are currently being investigated by my colleague Alexander Barth in his PhD thesis. The estimation of acceleration is not possible from two frames, which shows the limits of scene flow. Extending the scene flow to a six-frame approach (three stereo image pairs) and to model the acceleration of objects is in the scope of future research.

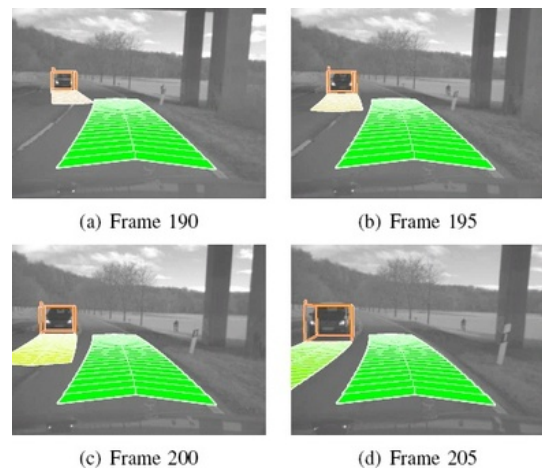


Figure 7.4: “Where will the oncoming vehicle be in the next second?” [39]. Selected frames of a real world scene with one oncoming vehicle captured in a left curve. It can be seen how the Kalman filter successively corrects the erroneous initial assumption of the driving path. Using dense scene flow estimates, the convergence of the Kalman filter may speed up.

Certainly, to some extent the estimation of position and motion is also possible using monocular vision. Then however, there remains a scale ambiguity in the relative translation of the oncoming vehicle and the camera. The prospects and limits of a monocular system for motion estimation of an oncoming vehicle have not been fully explored yet.

In his “Two-view multibody structure from motion” paper [138], René Vidal proposed a method to automatically estimate the fundamental matrix for every moving object within the scene, which in turn can be used for the segmentation process. In the original work, a stationary monocular camera and feature correspondences were used. An interesting research field is the generalization to binocular vision, the use of dense flow fields, and the combination of the proposed approach with a Kalman filter based tracking.



Figure 7.5: Estimation of the Free Space Band. The free space (middle image, green) is the instantaneous space free of any obstacles. By using an object height segmentation, the free space band is derived (right image). This free space band contains all important objects for autonomous decision making.

7.3 Dynamic Free Space

The vision of autonomous driving rises and falls with the ability to steer an unmanned vehicle. Taking a look into the animal kingdom helps us to get an insight on locomotion details. The condition precedent to successful locomotion is to distinguish between free space and clogged space. Snails, for instance, mainly use their sense of touch to investigate the surrounding area and possible obstacles. Almost everybody might once have touched a snail's head to see it withdraw its feelers or tentacles, a movement which is amazingly fast for an animal as slow as a snail. Such a procedure is obviously not appropriate in autonomous driving with high speeds and the sense of touch is in good hands with crash researchers. On the other hand most animals, and also humans, mainly rely on the sense of sight for obstacle and free space detection. Flies are able to navigate through unknown environments as long as no mirror or glass pane restricts the free space. Until now, no technology is able to imitate this ability. Therefore research in this area is unavoidable to preserve the dream of autonomous driving [38].

I propose two novel concepts for free space estimation: firstly, an optimal free space segmentation taking into consideration object height (see Figure 7.5), and secondly, the estimation of free space dynamics over time. The height segmentation of the free space is essential to employ free space dynamics because it identifies the object regions for which optical flow (or scene flow) information needs to be evaluated.

Obstacle Height. Free space is defined as the available space to maneuver a vehicle so as to avoid collision with objects. Any object limits the free space. Free space is commonly computed by thresholding the height of 3D measurements and accumulating measurements in occupancy grids (see Figure 7.6). The free space boundary is defined by the distance to the first object (occupied cell) in every spatial direction. A drawback of this method is the undefined handling of obstacles which are staggered in depth. Such objects lead to two opposing hypotheses for the free space boundary.

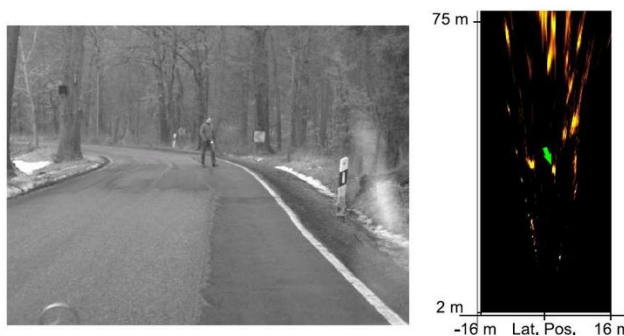


Figure 7.6: Depth map created from stereo measurements for the scene depicted on the left (the arrow points out the position of the person). Clearly, the existence of more than one obstacle in a viewing direction leads to opposing hypotheses for the free space boundary.

As a second drawback, obstacle height is not directly modelled. Therefore, objects have to be investigated in a second step to derive their height and size. I will present a concept to derive consistent free space with two obstacles staggered in depth and how to integrate free

space dynamics. This can be regarded as an extension of my previous work presented in [8] and is intended to trigger further research in this area.

The basic idea to estimate the free space and the height of the closest objects is to traverse every image column in the left image of a stereo image pair bottom-up. Every pixel up to the horizon serves as a hypothesis for the boundary change between free space and obstacle (*base-pixel*). For every base-pixel all pixels above, in the same image column, become potential *top-pixel* defining the boundary between obstacle and background. Figure 7.7 demonstrates the basic principle. In [33] this principle is picked up and the authors propose to represent the 3D world with what they name “stixel”. Note, that boundaries are defined in image coordinates, which defines an upper bound on the total number of possible boundaries. This is important for globally optimal solution-finding.

A concept for a global solution is to define cost functions for deviations from the assumption free space (C_{FS}), the obstacle (C_O) and the background (C_{BG}), depending on the base-pixel b and the top-pixel t . The resulting cost function for each column consists of the three sums, $C_{FS}(b) + C_O(b, t) + C_{BG}(b, t)$. The optimal segmentation for a single column is the minimum of this cost function for all potential values $y_{max} \geq b > t \geq 0$. In combination with a smoothness term for the free space boundary, the global minimum can be found using dynamic programming.

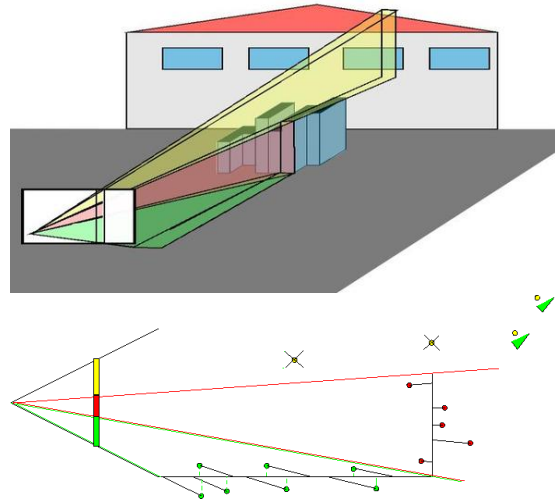


Figure 7.7: 3D view (top) and side view (bottom) of the basic principle to estimate free space and object height for a single image column. Measurements on the ground plane (green), the object (red), and in the background are counted and a fitting score is derived.

Dynamic Free Space

Common free space representations capture the instantaneous scene structure. But if objects are not stationary, the free space changes over time and is not fix. The concept of free space dynamics has already been addressed in [135], where the dynamics of preceding objects on interstate highways were considered in the integration of the free space over time.

A more naturally free space representation would be the use of *time-to-contact* (*ttc*) values. Time-to-contact measurements are derived from the relative speed between the camera (ego-vehicle) and the observed objects. The time-to-contact is essential for obstacle avoidance and autonomous driving: in order to determine the space *free of collisions*, the time-to-contact values are of higher importance than distances, as this allows one to calculate the time and place of a potential collision. In order to construct such time-to-contact-maps, the relative motion between the camera and objects needs to be known, which involves tracking of pixels (note that a radar sensor allows for direct velocity estimates and no tracking is necessary). Hence, monocular optical flow measurements are sufficient to derive such *ttc*-maps. To my knowledge this fact has not been explored in computer vision. Neither has the use of scene flow for dynamic depth maps been exploited.

The challenge with dynamic free space is, that a one-dimensional free space description (essentially only the distance to objects) becomes a two-dimensional velocity space because objects move on the X - Z ground plane. Thus, a one-dimensional *ttc*-only-representation might not be sufficient. Future research will have to find an optimal representation for such free space dynamics.

Epilogue



When you absolutely don't know what to do anymore, it is time to panic.

John van der Wiel

© under the creative commons license, EricMagnuson, www.flickr.com

The process of understanding visual kinesthesia has a lot in common with a game of chess. A game of chess is divided into three phases: the opening, the middle game and the end game. In the middle of the last century it was proclaimed that *building perceiving machines would take about a decade*. Scientists were almost certain that they had solved most of the difficulties with motion perception and that they were situated in the end game. But the process of motion perception turned out to be far more advanced. Every move towards *building perceiving machines* seems to reveal that there are even more complicated tasks to solve. Instead of gaining a clear understanding of the matter of motion analysis, one has to deal with a far more complex task than was ever imagined. Scientists found themselves still situated in the opening...

As with a chess game, every move needs to be well thought out in order to reach a successful outcome. Without losing focus on the goal of understanding the three-dimensional motion of the scene, motion analysis in image sequences was thoroughly studied in this thesis. A novel framework for optical flow estimation in image sequences, consisting of an iterative data term evaluation and a vector field smoothing process was acquired. This provided profound insights into the process of motion analysis and is currently the most accurate approach for the estimation of apparent motion in image sequences. The two-dimensional image motion was extended towards three-dimensional scene flow and presented as a real-time capable approach to dense scene flow estimation. These state-of-the-art techniques allow for a wide range of prospects for future research and will enable numerous computer vision applications; possibilities and opportunities for what some believe is the most complex phase, the middle game.

One application was presented in more detail: the segmentation of independently-moving objects in image sequences. Much as this is sometimes considered high-level computer vision, it is only a small step towards visual kinesthesia and building perceiving machines. Many more steps have to follow and it is up to others to make the next move.

This thesis presented a possible opening to the art of motion analysis. Understanding the relationship between objects and figures, predicting what is going to happen, and analyzing potential hazards are the main parts of ongoing research. After reading this thesis, the hope is that the reader is informed and inspired to proceed with novel ideas on how to achieve the dream of building perceiving machines.

Part III
Appendix

Data Terms for Refinement Optical Flow

Optical Flow Constraint Data Term

For the optical flow constraint (4.1) the solution space is two-dimensional, hence $\mathbf{u} = (u_1, u_2)^\top \in \mathbb{R}^2$. The optical flow constraint itself computes as $p(\mathbf{u}) = I_t + \nabla \mathbf{I}^\top \mathbf{u}$, hence $p_0 = \lambda I_t$ and $\mathbf{p} = \lambda \nabla \mathbf{I}$. The minimization problem then becomes

$$\min_{u_1, u_2} \left\{ \frac{1}{2} (u_1 - u'_1)^2 + \frac{1}{2} (u_2 - u'_2)^2 + \underbrace{\lambda |I_t + \nabla \mathbf{I}^\top \mathbf{u}|}_{|p_0 + \mathbf{p}^\top \mathbf{u}|} \right\}, \quad (\text{A.1})$$

where $\mathbf{u}' = (u'_1, u'_2)^\top$ is a given approximate solution. A closer look reveals, that Equation (A.1) is equivalent to the minimization problem in Equation (4.6) (up to a scale factor).

Adaptive Fundamental Matrix Constraint

Within the spectrum of conceivable optic flow patterns in Computer Vision, flow patterns that correspond to 3D rigid body motion play a central role. This is not surprising, since they invariably arise for static scenes filmed by a moving camera or for objects moving rigidly. It is well known that in the case of rigid body motion the two-dimensional optic flow estimation problem is reduced to a one-dimensional search along the epipolar lines which can actually be solved quite efficiently. The challenge is, that the epipolar lines are usually unknown and need to be estimated from established point correspondences themselves. This bootstrapping problem is usually solved iteratively and has one major drawback: If the scene is not stationary, both the epipolar lines and the optical flow suffer from the biasing prior inflicted by the other.

Variational optical flow techniques with prior knowledge of the fundamental matrix geometry were presented using hard constraints [121] and soft constraints [133], the latter estimating simultaneously the optical flow and the fundamental matrix. The algebraic distance to the epipolar rays was used in both approaches (Equation (3.7)).

In the following, an adaptive regularization will be proposed which favors rigid body motion only if this is supported by the image data. In particular, an adaptive weighting $\gamma(v)$ aims at engaging the rigid body constraint based on the amount of independent motion found within the scene. It is given by

$$\gamma(\mathbf{u}) = \begin{cases} \lambda_{\mathbf{F}} & \text{if } \int_{\Omega} \rho_{\mathbf{F}}(\mathbf{u}, \mathbf{x}) / \|\mathbf{u}\| d^2 \mathbf{x} < \delta_{\mathbf{F}} \\ 0 & \text{otherwise.} \end{cases} \quad (\text{A.2})$$

In the formula, $\rho_{\mathbf{F}}(\mathbf{u}, \mathbf{x})$ is the symmetric distance of the flow vector to the two epipolar lines,

$$\tilde{\mathbf{x}} = \mathbf{F}^\top \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{u}} = \mathbf{F} \begin{bmatrix} \mathbf{x} + \mathbf{u} \\ 1 \end{bmatrix}.$$

With the 3×3 fundamental matrix \mathbf{F} it is defined as

$$\rho_{\mathbf{F}}(\mathbf{u}, \mathbf{x}) = \frac{1}{\tilde{x}_1^2 + \tilde{x}_2^2 + \tilde{u}_1^2 + \tilde{u}_2^2} \left| c + \tilde{x}_1 u_1 + \tilde{x}_2 u_2 \right|, \quad (\text{A.3})$$

where a sub-index i of a vector denotes its i -th component and the constant c is computed as

$$c = \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}^\top \mathbf{F} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}. \quad (\text{A.4})$$

This formulation is symmetric w. r. t. the two input images and normalized in the sense that it does not depend on the scale factor used to compute \mathbf{F} [92].

To this end, first an optic flow field is computed and a fundamental matrix is estimated by minimizing the non-linear criterion

$$\min_{\mathbf{F}} \left\{ \sum \frac{1}{\tilde{x}_1^2 + \tilde{x}_2^2 + \tilde{u}_1^2 + \tilde{u}_2^2} \left(\begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}^\top \tilde{\mathbf{u}} \right)^2 \right\}. \quad (\text{A.5})$$

Then, the estimated fundamental matrix itself is used to drive the optical flow towards the epipolar lines.

The crucial part is how to weight this data term in order to maintain robustness in dynamic scenes and increase accuracy in static scenes. This is where the adaptive weighting $\gamma(v)$, which analyzes the amount of independent motion, becomes important. Essentially, a violation of the epipolar constraint denotes other moving objects in the scene while the counter-hypothesis does not generally hold (e. g. the motion of objects might coincide with the epipolar lines). However, simply computing the average symmetric distance to the epipolar lines, $\int_{\Omega} \rho_{\mathbf{F}}(\mathbf{u}, \mathbf{x}) d^2 \mathbf{x}$, does not yield useful results as flow vectors in a dynamic scene might be relatively small in magnitude, yielding only small errors although the complete scene is dynamic. Here the relative symmetric distance, weighted by the inverse length of the computed optical flow is used. Such measure yields a rather robust estimate of the relative motion contained in the scene depicted by the two images.

In summary, the adaptive rigid body regularization has the following effect:

- If the average relative deviation $\int_{\Omega} \rho_{\mathbf{F}}(\mathbf{u}, \mathbf{x}) / \|\mathbf{u}\| d^2 \mathbf{x}$ is above a predefined threshold $\delta_{\mathbf{F}}$, the fundamental matrix regularization will be switched off so it does not bias the estimation of motion in dynamic scenes.
- If on the other hand the relative deviation is smaller than $\delta_{\mathbf{F}}$, then the fundamental matrix regularization is imposed so as to favor the estimation of optic flow fields that are consistent with rigid body motion. Note that even in this case the fundamental matrix constraint is *not* enforced to be exactly fulfilled. Instead, deviations of the optic flow from rigid body motion are allowed wherever this is supported by the image data.

The resulting linearized data term becomes:

$$p(\mathbf{u}) = \gamma(\mathbf{u}) \rho_{\mathbf{F}}(\mathbf{u}, \mathbf{x}) = \underbrace{\gamma(\mathbf{u}) \frac{1}{\tilde{x}_1^2 + \tilde{x}_2^2 + \tilde{u}_1^2 + \tilde{u}_2^2}}_{\lambda_p} \left| \underbrace{c}_{p_0} + \underbrace{\begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{bmatrix}^\top}_{\mathbf{p}^\top} \mathbf{u} \right|, \quad (\text{A.6})$$

Note that the constant p_0 and the scalar value λ_p involve the optical flow vector. Both values are calculated using flow vectors from the last iteration (hence lagged feedback is used here).

Quadratic Optimization via Thresholding

Equation (4.9) is a combination of convex functions, and hence is convex itself. If Equation (4.9) was differentiable, the minimum could be found by setting its derivative, with respect to \mathbf{u} , equal to zero. However, the absolute function $|x|$ is degenerate at $|x| = 0$ and not differentiable. More sophisticated quadratic optimization techniques have to be used to find the true minimum of Equation (4.9).

In this chapter the Karush-Kuhn-Tucker (KKT) conditions [76] are used to derive the thresholding steps for a single data term and two data terms in Equation (4.9). For completeness, let us first reproduce the KKT conditions.

B.1 Karush-Kuhn-Tucker (KKT) Conditions

For a convex quadratic minimization problem $\min_x \{f(x)\}$, under N linear inequality constraints $g_i(x) \leq 0$ where $0 \leq i \leq N$, a global optimum of a solution x^* holds true if there exist constants μ_i such that the Karush-Kuhn-Tucker (KKT) conditions [76] are fulfilled:

$$\text{Stationarity: } \nabla f(x^*) + \sum_i \mu_i \nabla g_i(x^*) = 0.$$

$$\text{Primal feasibility: } g_i(x^*) \leq 0.$$

$$\text{Dual feasibility: } \mu_i \geq 0.$$

$$\text{Complementary slackness: } \mu_i g_i(x) = 0 \quad \forall i.$$

B.2 Single Data Term [150]

For a single data term, the task is to find the vector \mathbf{u}^* which solves the minimization problem

$$\mathbf{u}^* = \min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + |p_0 + \mathbf{p}^\top \mathbf{u}| \right\}, \quad (\text{B.1})$$

or its dual formulation

$$\mathbf{u}^* = \min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + p' \right\} \quad (\text{B.2})$$

with linear side conditions

$$p_0 + \mathbf{p}^\top \mathbf{u} - p' \leq 0 \quad (\text{B.3a})$$

$$\text{and } -p_0 - \mathbf{p}^\top \mathbf{u} - p' \leq 0. \quad (\text{B.3b})$$

Taking the above Equations into the KKT conditions for the global optimum of the dual formulation yields

Stationarity:

$$\begin{pmatrix} \mathbf{u}^* - \mathbf{u}' \\ 1 \end{pmatrix} + \mu_1 \begin{pmatrix} \mathbf{p} \\ -1 \end{pmatrix} + \mu_2 \begin{pmatrix} -\mathbf{p} \\ -1 \end{pmatrix} = \mathbf{0}. \quad (\text{B.4})$$

Primal feasibility:

$$\left\{ \begin{array}{l} p_0 + \mathbf{p}^\top \mathbf{u}^* - p' \leq 0 \\ -p_0 - \mathbf{p}^\top \mathbf{u}^* - p' \leq 0 \end{array} \right\}.$$

Dual feasibility:

$$\left\{ \begin{array}{l} \mu_1 \geq 0 \\ \mu_2 \geq 0 \end{array} \right\} .$$

Complementary slackness:

$$\left\{ \begin{array}{l} \mu_1 (p_0 + \mathbf{p}^\top \mathbf{u}^* - p') = 0 \\ \mu_2 (-p_0 - \mathbf{p}^\top \mathbf{u}^* - p') = 0 \end{array} \right\} .$$

The minimum can be found by analyzing the three possible cases for the single data term for the primal formulation; $p(\mathbf{u}^*) < 0$, $p(\mathbf{u}^*) = 0$, and $p(\mathbf{u}^*) > 0$. The KKT conditions yield a simple and efficient thresholding scheme to ensure the global optimum of the solution.

In the first case ($p(\mathbf{u}^*) < 0$), the minimum of Equation (B.1) is given by taking the derivative w. r. t. \mathbf{u} yielding $\mathbf{u}^* = \mathbf{u}' + \mathbf{p}$. This implies for the dual variable

$$p' = |p(\mathbf{u}^*)| = - \left(p_0 + \mathbf{p}^\top (\mathbf{u}' + \mathbf{p}) \right) = -p(u') - \mathbf{p}^\top \mathbf{p} .$$

Due to construction, side condition (B.3b) is binding (its left side is zero), directly yielding primal feasibility for Equation (B.3b). The primal feasibility for Equation (B.3a) is fulfilled iff

$$\begin{aligned} & p_0 + \mathbf{p}^\top \mathbf{u}^* - p' \leq 0 . \\ \Leftrightarrow & p_0 + \mathbf{p}^\top (\mathbf{u}' + \mathbf{p}) - \left(-p(u') - \mathbf{p}^\top \mathbf{p} \right) \leq 0 . \\ \Leftrightarrow & \underbrace{p_0 + \mathbf{p}^\top \mathbf{u}' + \mathbf{p}^\top \mathbf{p}}_{p(u')} + p(u') + \mathbf{p}^\top \mathbf{p} \leq 0 . \\ \Leftrightarrow & 2 \left(p(u') + \mathbf{p}^\top \mathbf{p} \right) \leq 0 . \end{aligned}$$

If this equation holds, the following holds also; setting $\mu_1 = 1$ and $\mu_2 = 0$ directly yields dual feasibility, complementary slackness, and stationarity of the solution \mathbf{u}^* . Hence, if the primal feasibility check returns true, the optimum for the point is found. This can be checked by evaluating $p(u') < -\mathbf{p}^\top \mathbf{p}$.

Equivalently, if $p(\mathbf{u}^*) > 0$, $\mathbf{u}^* = \mathbf{u}' - \mathbf{p}$ is the global minimum, iff the primal feasibility check for Equation (B.3b) holds, yielding the thresholding check

$$2 \left(p(u') - \mathbf{p}^\top \mathbf{p} \right) \leq 0 .$$

This can be evaluated by checking for $p(u') > \mathbf{p}^\top \mathbf{p}$.

If the data term vanishes (i.e. $p(\mathbf{u}^*) = p' = 0$), both side conditions are binding. The minimum of $\min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 \right\}$ under the side condition $p_0 + \mathbf{p}^\top \mathbf{u} = 0$ yields the solution

$$\mathbf{u}^* = \mathbf{u}' - \frac{p(u')}{\mathbf{p}^\top \mathbf{p}} \mathbf{p} .$$

Primal feasibility and complementary slackness is directly fulfilled for both side conditions, because they are both binding. A final thresholding check has to ensure dual feasibility for μ_1

and μ_2 fulfilling the stationarity constraint. Plugging \mathbf{u}^* into Equation (B.4) yields

$$0 = \begin{pmatrix} \mathbf{u}^* - \mathbf{u}' \\ 1 \end{pmatrix} + \mu_1 \begin{pmatrix} \mathbf{p} \\ -1 \end{pmatrix} + \mu_2 \begin{pmatrix} -\mathbf{p} \\ -1 \end{pmatrix} \quad (\text{B.5})$$

$$0 = \begin{pmatrix} \mathbf{u}' - \frac{p(\mathbf{u}')}{\mathbf{p}^\top \mathbf{p}} \mathbf{p} - \mathbf{u}' \\ 1 \end{pmatrix} + \mu_1 \begin{pmatrix} \mathbf{p} \\ -1 \end{pmatrix} + \mu_2 \begin{pmatrix} -\mathbf{p} \\ -1 \end{pmatrix} \quad (\text{B.6})$$

$$\begin{pmatrix} \frac{p(\mathbf{u}')}{\mathbf{p}^\top \mathbf{p}} \mathbf{p}^\top \mathbf{p} \\ 1 \end{pmatrix} = \mu_1 \begin{pmatrix} \mathbf{p}^\top \mathbf{p} \\ 1 \end{pmatrix} + \mu_2 \begin{pmatrix} -\mathbf{p}^\top \mathbf{p} \\ 1 \end{pmatrix} \quad (\text{B.7})$$

$$\begin{pmatrix} \frac{p(\mathbf{u}')}{\mathbf{p}^\top \mathbf{p}} \\ 1 \end{pmatrix} = \mu_1 \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \mu_2 \begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad (\text{B.8})$$

The solution is given by $\mu_1 = \frac{1}{2} \left(1 + \frac{p(\mathbf{u}')}{\mathbf{p}^\top \mathbf{p}}\right)$ and $\mu_2 = \frac{1}{2} \left(1 - \frac{p(\mathbf{u}')}{\mathbf{p}^\top \mathbf{p}}\right)$ and dual feasibility holds if $|p(\mathbf{u}')| \leq \mathbf{p}^\top \mathbf{p}$.

Analyzing the three cases $p(\mathbf{u}^*) < 0$, $p(\mathbf{u}^*) = 0$, and $p(\mathbf{u}^*) > 0$ directly yields three distinct and complete thresholding steps. This yields a very efficient algorithm for evaluating the minimum of Equation B.1, first presented by [150]. Summarized, the solution computes as

Assume $p(\mathbf{u}^*)$	Thresholding Check	Solution $\mathbf{u}^* =$
$p(\mathbf{u}^*) < 0$	$p(\mathbf{u}') < -\mathbf{p}^\top \mathbf{p}$	$\mathbf{u}' + \mathbf{p}$
$p(\mathbf{u}^*) = 0$	$ p(\mathbf{u}') \leq \mathbf{p}^\top \mathbf{p}$	$\mathbf{u}' - \frac{p(\mathbf{u}')}{\mathbf{p}^\top \mathbf{p}} \mathbf{p}$
$p(\mathbf{u}^*) > 0$	$p(\mathbf{u}') > \mathbf{p}^\top \mathbf{p}$	$\mathbf{u}' - \mathbf{p}$

Table B.1: Computation of the minimum \mathbf{u}^* for Equation (B.1).

B.3 Two Data Terms

In this section an efficient solution scheme to minimize the objective function (4.9) or, equivalently, its dual quadratic optimization problem (4.11) with inequality constraints (4.12) for two data terms is proposed and verified. To this end, we have two data terms $\lambda_1 p_1(\mathbf{u}) = a_0 + \mathbf{a}^\top \mathbf{u}$ and $\lambda_2 p_2(\mathbf{u}) = b_0 + \mathbf{b}^\top \mathbf{u}$ where a_0 and b_0 are constant and \mathbf{a} and \mathbf{b} represent the linear parts of the data terms. The minimization problem states

$$\min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + \underbrace{|a_0 + \mathbf{a}^\top \mathbf{u}|}_{\lambda_1 p_1(\mathbf{u})} + \underbrace{|b_0 + \mathbf{b}^\top \mathbf{u}|}_{\lambda_2 p_2(\mathbf{u})} \right\}. \quad (\text{B.9})$$

Its dual formulation with dual variables a' and b' is

$$\min_{\mathbf{u}} \left\{ \frac{1}{2} \|\mathbf{u} - \mathbf{u}'\|^2 + a' + b' \right\}$$

such that

$$\begin{aligned} (i) \quad & a_0 + \mathbf{a}^\top \mathbf{u} - a' \leq 0 \\ (ii) \quad & -a_0 - \mathbf{a}^\top \mathbf{u} - a' \leq 0 \\ (iii) \quad & b_0 + \mathbf{b}^\top \mathbf{u} - b' \leq 0 \\ (iv) \quad & -b_0 - \mathbf{b}^\top \mathbf{u} - b' \leq 0. \end{aligned} \quad (\text{B.10})$$

Differentiating this and taking it into the KKT conditions for the global optimum of the

dual formulation yields additional dual variables $\mu_{(i)}$, $\mu_{(ii)}$, $\mu_{(iii)}$, and $\mu_{(iv)}$ and the following constraints:

Stationarity:

$$\begin{pmatrix} \mathbf{u}^* - \mathbf{u}' \\ 1 \\ 1 \end{pmatrix} + \mu_{(i)} \begin{pmatrix} \mathbf{a} \\ -1 \\ 0 \end{pmatrix} + \mu_{(ii)} \begin{pmatrix} -\mathbf{a} \\ -1 \\ 0 \end{pmatrix} + \mu_{(iii)} \begin{pmatrix} \mathbf{b} \\ 0 \\ -1 \end{pmatrix} + \mu_{(iv)} \begin{pmatrix} -\mathbf{b} \\ 0 \\ -1 \end{pmatrix} = \mathbf{0}.$$

Primal feasibility:

$$\left\{ \begin{array}{l} (i) \quad a_0 + \mathbf{a}^\top \mathbf{u}^* - a' \leq 0 \\ (ii) \quad -a_0 - \mathbf{a}^\top \mathbf{u}^* - a' \leq 0 \\ (iii) \quad b_0 + \mathbf{b}^\top \mathbf{u}^* - b' \leq 0 \\ (iv) \quad -b_0 - \mathbf{b}^\top \mathbf{u}^* - b' \leq 0 \end{array} \right\}.$$

Dual feasibility:

$$\left\{ \begin{array}{l} \mu_{(i)} \geq 0 \\ \mu_{(ii)} \geq 0 \\ \mu_{(iii)} \geq 0 \\ \mu_{(iv)} \geq 0 \end{array} \right\}.$$

Complementary slackness:

$$\left\{ \begin{array}{l} \mu_{(i)} (a_0 + \mathbf{a}^\top \mathbf{u}^* - a') = 0 \\ \mu_{(ii)} (-a_0 - \mathbf{a}^\top \mathbf{u}^* - a') = 0 \\ \mu_{(iii)} (b_0 + \mathbf{b}^\top \mathbf{u}^* - b') = 0 \\ \mu_{(iv)} (-b_0 - \mathbf{b}^\top \mathbf{u}^* - b') = 0 \end{array} \right\}.$$

Looking at all possible combinations of the data terms $|p_i(\mathbf{u}^*)|$, namely $p_i(\mathbf{u}^*) \leq 0$, $p_i(\mathbf{u}^*) \geq 0$, and $p_i(\mathbf{u}^*) = 0$, directly yields a thresholding scheme to minimize the objective function. If both $p_1(\mathbf{u}^*)$ and $p_2(\mathbf{u}^*)$ are strictly positive or negative (i. e. $p_i(\mathbf{u}^*) \neq 0$), the optimal solution \mathbf{u}^* is found iff the thresholding checks in Table B.2 apply.

Assume $p_{1,2}(\mathbf{u}^*)$	Thresholding Checks	Solution $\mathbf{u}^* =$
$p_1 \geq 0$ $p_2 \geq 0$	$a_0 + \mathbf{a}^\top (\mathbf{u}' - \mathbf{a} - \mathbf{b}) \geq 0$ $b_0 + \mathbf{b}^\top (\mathbf{u}' - \mathbf{a} - \mathbf{b}) \geq 0$	$\mathbf{u}' - \mathbf{a} - \mathbf{b}$
$p_1 \geq 0$ $p_2 \leq 0$	$a_0 + \mathbf{a}^\top (\mathbf{u}' - \mathbf{a} + \mathbf{b}) \geq 0$ $b_0 + \mathbf{b}^\top (\mathbf{u}' - \mathbf{a} + \mathbf{b}) \leq 0$	$\mathbf{u}' - \mathbf{a} + \mathbf{b}$
$p_1 \leq 0$ $p_2 \geq 0$	$a_0 + \mathbf{a}^\top (\mathbf{u}' + \mathbf{a} - \mathbf{b}) \leq 0$ $b_0 + \mathbf{b}^\top (\mathbf{u}' + \mathbf{a} - \mathbf{b}) \geq 0$	$\mathbf{u}' + \mathbf{a} - \mathbf{b}$
$p_1 \leq 0$ $p_2 \leq 0$	$a_0 + \mathbf{a}^\top (\mathbf{u}' + \mathbf{a} + \mathbf{b}) \leq 0$ $b_0 + \mathbf{b}^\top (\mathbf{u}' + \mathbf{a} + \mathbf{b}) \leq 0$	$\mathbf{u}' + \mathbf{a} + \mathbf{b}$

Table B.2: Minimum \mathbf{u}^* if $p_1(\mathbf{u}^*) \neq 0$ and $p_2(\mathbf{u}^*) \neq 0$

Proof: For $p_1(\mathbf{u}^*) \neq 0$ and $p_2(\mathbf{u}^*) \neq 0$ the solution can be found by setting the derivative of the unconstrained objective function equal to zero. Let us prove using the KKT conditions, that this yields a global minimum iff the above thresholding steps succeed. Due to construction, exactly two inequality constraints are binding (the left side is 0). We set the μ_i for these constraints to 1 and the μ_i for the other two constraints to 0. This implies that the point is stationary because the solution is constructed to yield a derivative of zero. It directly follows, that complementary slackness and dual feasibility hold. The thresholding check ensures primal feasibility of the solution and hence, iff the thresholding check succeeds, a global minimal solution is found.

Vanishing data terms If (at least) one of the data terms is binding, the solution space is restricted to yield either $p_1(\mathbf{u}^*) = 0$ or $p_2(\mathbf{u}^*) = 0$. In this subsection thresholding checks to verify the necessary and sufficient conditions of a global minimum for the local solution in these restricted cases are derived. For the following analysis, let us assume that the first data term at the global minimum vanishes, i.e. $p_1(\mathbf{u}^*) = 0$. The case $p_2(\mathbf{u}^*)$ is equivalent (simply exchange the two data terms) and not handled explicitly.

For a vanishing $p_1(\mathbf{u}^*)$, three cases need to be examined: $p_2(\mathbf{u}^*) < 0$, $p_2(\mathbf{u}^*) = 0$, and $p_2(\mathbf{u}^*) > 0$. The case $p_2(\mathbf{u}^*) = 0$ is left out in the analysis. If all other cases do not yield a global minimum, it directly follows that both data terms must vanish; the unknown parameter vector \mathbf{u}^* can then be calculated from the two data term equations.

For now we stick with $p_1(\mathbf{u}^*) = 0$ and assume either $p_2(\mathbf{u}^*) > 0$ or $p_2(\mathbf{u}^*) < 0$. The (possible) global minimum \mathbf{u}^* is computed by setting the derivative of the objective function (B.9) equal to zero using the assumptions made on the $p_i(\mathbf{u}^*)$:

Assume $p_1(\mathbf{u}^*) = 0$	Solution (checks follow)
$p_2(\mathbf{u}^*) \geq 0$	$\mathbf{u}^* = \mathbf{u}' - \mathbf{b} - \frac{\mathbf{a}}{\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' - \mathbf{a}^\top \mathbf{b})$
$p_2(\mathbf{u}^*) \leq 0$	$\mathbf{u}^* = \mathbf{u}' + \mathbf{b} - \frac{\mathbf{a}}{\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' + \mathbf{a}^\top \mathbf{b})$

Table B.3: Minimum \mathbf{u}^* if $p_1(\mathbf{u}^*) = 0$ and $p_2(\mathbf{u}^*) \neq 0$.

Again, the KKT conditions have to be checked to verify a global optimum. Due to construction we have $a' = 0$, and the first two inequality constraints are binding (hence primal feasible). Out of the remaining two inequality constraints, one is primal feasible due to construction as $p_2(\mathbf{u}^*) \leq 0$ or $p_2(\mathbf{u}^*) \geq 0$ directly yield one inequality constraint that is binding. The last constraint is checked by the thresholding step in Table B.4.

First thresholding check if $p_1(\mathbf{u}^*) = 0$	
for $p_2(\mathbf{u}^*) \geq 0$ check	$b_0 + \mathbf{b}^\top (\mathbf{u}' - \mathbf{b} - \frac{\mathbf{a}}{\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' - \mathbf{a}^\top \mathbf{b})) \geq 0,$
for $p_2(\mathbf{u}^*) \leq 0$ check	$b_0 + \mathbf{b}^\top (\mathbf{u}' + \mathbf{b} - \frac{\mathbf{a}}{\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' + \mathbf{a}^\top \mathbf{b})) \leq 0 .$

Table B.4: Check for primal feasibility.

The complementary slackness condition states that the μ_i corresponding to the only non-binding inequality constraint has to be zero. This leaves us with three more μ_i , which have to be positive to fulfill the dual feasibility condition.

Example: Let us assume, $p_2(\mathbf{u}^*) \geq 0$. It follows that the inequality constraint (iii) is binding and the inequality constraint (iv) in Equation (B.10) is fulfilled. This directly implies that $\mu_{(iv)}$ has to be zero (as stated above).

Using the stationarity condition, an equation system to solve for the remaining μ_i is derived:

$$\begin{bmatrix} \mathbf{a} & -\mathbf{a} & \mathbf{b} \\ -1 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \mu_{(i)} \\ \mu_{(ii)} \\ \mu_{(iii)} \end{bmatrix} = \begin{bmatrix} \mathbf{u} - \mathbf{u}' \\ -1 \\ -1 \end{bmatrix}$$

From this it follows that $\mu_{(iii)} = 1$ and the remaining system of equations yields a unique solution

for $\mu_{(i)}$ and $\mu_{(ii)}$. This can be seen when plugging back in the possible solutions for \mathbf{u}^* , yielding:

$$\begin{aligned} \mathbf{a}(\mu_{(i)} - \mu_{(ii)}) &= \frac{\mathbf{a}}{\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' - \mathbf{a}^\top \mathbf{b}) \\ \mu_{(i)} + \mu_{(ii)} &= 1 \end{aligned}$$

The solution is given by

$$\mu_{(i)} = \frac{1}{2\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' - \mathbf{a}^\top \mathbf{b}) + \frac{1}{2} \tag{B.11}$$

$$\mu_{(ii)} = \frac{-1}{2\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' - \mathbf{a}^\top \mathbf{b}) + \frac{1}{2}. \tag{B.12}$$

The KKT dual feasibility constraint states that both, $\mu_{(i)}$ and $\mu_{(ii)}$ have to be positive. This can be checked very efficiently:

Second thresholding check if $p_1(\mathbf{u}^*) = 0$	
check	$\left \frac{1}{\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' - \mathbf{a}^\top \mathbf{b}) \right \leq 1$ if $p_2(\mathbf{u}^*) \geq 0$
check	$\left \frac{1}{\mathbf{a}^\top \mathbf{a}} (a_0 + \mathbf{a}^\top \mathbf{u}' + \mathbf{a}^\top \mathbf{b}) \right \leq 1$ if $p_2(\mathbf{u}^*) \leq 0$

Table B.5: Check for dual feasibility.

Iff all KKT conditions hold, \mathbf{u}^* yields a global minimum of the objective function. Following the arguments above, the check for $p_2(\mathbf{u}^*) = 0$ is straight-forward. The presented thresholding scheme yields a very efficient and exact total variation scheme for the optical flow with two (linear) data terms.

Variational Image Flow Framework

Image flow estimates according to the constraints formulated in Section 5.2 can be computed in a variational framework by minimising an energy functional consisting of a data term (E_D , derived from the constraints) and a smoothness term, E_S , that enforces smooth and dense image flow. By integrating over the image domain Ω we obtain:

$$E_{SF} = \int_{\Omega} \left(E_D + E_S \right) dx dy .$$

Using the constraints from Equation (5.4) in Section 5.2 yields the following data term:

$$E_D = \Psi \left(E_{LF}^2 \right) + c(x, y) \Psi \left(E_{RF}^2 \right) + c(x, y) \Psi \left(E_{DF}^2 \right) ,$$

where $\Psi(s^2) = \sqrt{s^2 + \varepsilon}$ ($\varepsilon = 0.0001$) denotes a robust function that compensates for outliers [45] and the function $c(x, y)$ returns 0 if there is no disparity known at (x, y) (due to occlusion or sparse stereo method), or 1 otherwise.

The smoothness term penalizes the local deviations in the image flow components and employs the same robust function as the data term in order to deal with existing discontinuities in the velocity field:

$$E_S = \lambda \Psi \left(|\nabla u|^2 \right) + \lambda \Psi \left(|\nabla v|^2 \right) + \gamma \Psi \left(|\nabla d|^2 \right) \text{ with } \nabla = (\partial/\partial x, \partial/\partial y)^T .$$

The parameters λ and γ regulate the importance of the smoothness constraint, weighting for optic flow and disparity change respectively. Interestingly, due to the fill-in effect of the above regularization, the proposed variational formulation provides dense image flow estimates $[u, v, d]^T$, even if the disparity d is non-dense.

Minimization of the Energy

For minimising the above energy, its Euler-Lagrange equations are computed:

$$\begin{aligned} \Psi'(E_{LF}^2) E_{LF} I_x^L + c \Psi'(E_{RF}^2) E_{RF} I_x^R + c \Psi'(E_{DF}^2) E_{DF} I_x^R - \lambda \operatorname{div} (\nabla u \Psi'(E_S)) &= 0 \\ \Psi'(E_{LF}^2) E_{LF} I_y^L + c \Psi'(E_{RF}^2) E_{RF} I_y^R + c \Psi'(E_{DF}^2) E_{DF} I_y^R - \lambda \operatorname{div} (\nabla v \Psi'(E_S)) &= 0 \\ c \Psi'(E_{RF}^2) E_{RF} I_x^R + c \Psi'(E_{DF}^2) E_{DF} I_x^R - \gamma \operatorname{div} (\nabla d \Psi'(E_S)) &= 0 \end{aligned}$$

where $\Psi'(s^2)$ denotes the derivative of Ψ with respect to s^2 . Partial derivatives of $I(x+u, y+v, t)^L$ and $I(x+d+d'+u, y+v, t)^R$ are denoted by subscripts x and y . For simplicity, $c = c(x, y)$.

These equations are non-linear in the unknowns, so we stick to the strategy of two nested fixed point iteration loops as suggested in [45]. This comes down to a warping scheme as also employed in [100]. The basic idea is to have an outer fixed point iteration loop that contains the linearization of the E_{LF} , E_{RF} , and E_{DF} . In each iteration, an increment of the unknowns is estimated and the second image is then warped according to the new estimate. The warping is combined with a coarse-to-fine strategy, where the equations are evaluated on down-sampled images that are successively refined with the number of iterations. For real-time estimates of the image flow variables, 4 iterations with 2 outer fixed point iterations at each scale are used.

The image intensity functions for the left and right images are linearized according to the following equations, where k denotes the iteration index. The iterations are started with $[u^0, v^0, d^0]^\top = (0, 0, 0)^\top$ for all $[x, y]^\top$:

$$I(x + u^k + \delta u^k, y + v^k + \delta v^k, t)^L \approx I(x + u^k, y + v^k, t)^L + \delta u^k I_x^L + \delta v^k I_y^L$$

$$I(x + d + d^k + \delta d^k + u^k + \delta u^k, y + v^k + \delta v^k, t)^R \approx I(x + d + d^k + u^k, y + v^k, t)^R + \delta u^k I_{(x+d)}^R + \delta d^k I_{(x+d)}^R + \delta v^k I_y^R.$$

From these expressions the linearized versions of E_{LF} , E_{RF} , and E_{DF} are derived.

The remaining non-linearity in the Euler-Lagrange equations is due to the robust function. In the inner fixed point iteration loop the Ψ' expressions are kept constant and are recomputed after each iteration. This finally leads to the following linear equations:

$$\begin{aligned} &\Psi'((E_{LF}^{k+1})^2)(E_{LF}^k + I_x^{L,k} \delta u^k + I_y^{L,k} \delta v^k) I_x^{L,k} \\ &\quad + c \Psi'((E_{RF}^{k+1})^2)(E_{RF}^k + I_x^{R,k} (\delta u^k + \delta d^k) + I_y^{R,k} \delta v^k) I_x^{R,k} \\ &\quad - \lambda \operatorname{div} \left(E_S'^{k+1} \nabla(u^k + \delta u^k) \right) = 0 \\ &\Psi'((E_{LF}^{k+1})^2)(E_{LF}^k + I_x^{L,k} \delta u^k + I_y^{L,k} \delta v^k) I_y^{L,k} \\ &\quad + c \Psi'((E_{RF}^{k+1})^2)(E_{RF}^k + I_x^{R,k} (\delta u^k + \delta d^k) + I_y^{R,k} \delta v^k) I_y^{R,k} \\ &\quad - \lambda \operatorname{div} \left(E_S'^{k+1} \nabla(v^k + \delta v^k) \right) = 0 \\ &c \Psi'((E_{RF}^{k+1})^2)(E_{RF}^k + I_x^{R,k} (\delta u^k + \delta d^k) + I_y^{R,k} \delta v^k) I_x^{R,k} \\ &\quad + c \Psi'((E_{DF}^{k+1})^2)(E_{DF}^k + I_x^{R,k} \delta d^k) I_x^{R,k} \\ &\quad - \gamma \operatorname{div} \left(E_S'^{k+1} \nabla(d^k + \delta d^k) \right) = 0 \end{aligned}$$

with

$$E_S'^{k+1} = \lambda \Psi' \left(|\nabla u^{k+1}|^2 \right) + \lambda \Psi' \left(|\nabla v^{k+1}|^2 \right) + \gamma \Psi' \left(|\nabla d^{k+1}|^2 \right).$$

We omitted the iteration index of the inner fixed point iteration loop to keep the notation uncluttered. Expressions with the iteration index $k + 1$ are computed using the current increments $\delta u^k, \delta v^k, \delta d^k$. We see that some terms from the original Euler-Lagrange equations have vanished. This is due to the use of $I(x + d, y, t)^r = I(x, y, t)^l$ from the linearized third constraint, Equation (5.3).

After discretization, the corresponding linear system is solved via successive over-relaxation. It is worth noting that, for efficiency reasons, it is advantageous to update the Ψ' after a few iterations of SOR. The shares of computation time taken by the different operations are shown in the pie graph below.

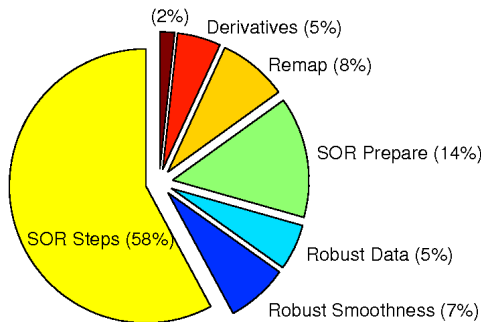


Image Size (pixels)	SOR Steps per Warp	Warps per Level	Total Time
640×480	45	2	975 ms
512×512	20	3	756 ms
320×240	45	2	205 ms

Break down of computational time for our algorithm (3.0GHz Intel®Core™2). The pie graph shows the time distribution for the 640×480 images. The real-time applicability of the algorithm for image sizes of (320×240) is indicated in the table.

Space-Time Multi-Resolution Cut

Applying real-time segmentation is a major issue when processing every frame of image sequences. This chapter reviews a modification of the well known graph-cut algorithm to improve speed for discrete segmentation (see also [2]). The *Space-Time Multi-Resolution Cut* algorithm yields real-time segmentation, using graph-cut, by performing a single cut on an image with regions of different resolutions; combining space-time pyramids and narrow bands. The fast computation time allows one to use information contained in every image frame of a VGA input image stream at 20 Hz, on a standard PC.

Section D.1 gives an overview on literature on speeding up the graph-cut algorithm. The novel approach is then described in Section D.2. It yields similar results to standard graph-cut, but with improved speed. Obtained results on traffic scene sequences and a comparison with the algorithms described in the literature conclude this chapter.



Figure D.1: The images show the flow field of image features in a traffic scene and a segmented moving object. The image on the right shows the multi-resolution graph structure obtained using the approach in this paper for real-time segmentation.

D.1 Literature Overview

Graph-cut for image segmentation was first introduced by [67] for image restoration. Its impact on the computer vision community rapidly increased after the publication of [41], where an algorithm especially designed for the segmentation of image domains was proposed. The algorithm boosted average segmentation times from several minutes to a few seconds for usual image domains, where every pixel yields one node in the resulting graph. However, most research on graph-cut focuses on single images and not on real-time image streams, where computation time becomes crucial. In computer vision for driver assistance frame rates of up to 25 frames per second have to be processed, demanding accordingly fast segmentations. See Chapter 6 for a more detailed description of the graph-cut algorithm and the assignment of nodes and edges to a graph. This section reviews techniques to speed up the graph-cut computation time.

Banded Cut on Image Pyramids

The simplest way to decrease computational time is using images of lower resolution, thus reducing the number of nodes and edges. However, this is at the expense of a reduced segmentation accuracy, especially along the boundaries. In [88] the authors propose the use of image pyramids and refine the boundaries on lower pyramid images as illustrated in Figure D.2. This approach has one problem; when image structures are small, they might not be detected on lower pyramid levels as they are smoothed out. One possibility to account for these changes was proposed in [120]. An additional Laplace pyramid is built, which represents the lost information from

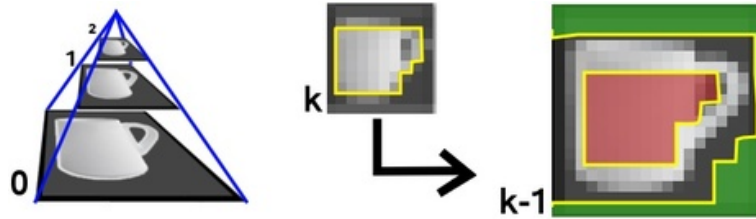


Figure D.2: The figure shows an example of an image pyramid (left). A full graph-cut is performed on the highest pyramid level and the result is iteratively propagated downwards, then refined in a band around the result of the upper pyramid level.

the image down-sampling process in the high frequency spectrum. If the values in the Laplace image exceed a certain threshold, additional pixels are introduced into the graph. The drawback of pyramid approaches is the inherent iterative process, where consecutive graph-cut steps on lower pyramid levels have to wait for the preceding step to finish. This causes the undesired effect of the graph-cut being carried out several times on the same input image, once for every pyramid resolution.

Temporal Banded Cut

Recall that we are interested in segmentations on image sequences. If the frame rate is high, this implies that segmentation boundaries in consecutive images are similar. Instead of predicting the segmentation boundary from the highest to lowest pyramid level, the segmentation of the current image is used as a hint for the cut of the next image in the sequence, and refined in a band around the segmentation border. This is done by [95] and [104]. The major drawback of this method is that essentially only tracking is performed, therefore new objects that appear are only detected if they are within the band width of the boundary for existing objects. In the papers, a human initialization step is required. Here, instead of a temporal prediction of a band, a temporal prediction on a whole pyramid is performed.

Temporal Prediction on Pyramids

Instead of propagating segmentation boundaries from a higher pyramid level onto a lower pyramid level of the same image, this propagation is done in the subsequent image of an image sequence (see Figure D.3). Using the assumption that the motion of the segment in the image is roughly known, this combines the detection in the pyramid scheme and the temporal prediction (tracking) of segmentation boundaries. Another advantage of this method is that the graph-cut algorithms for the single pyramid levels can be executed in parallel, resulting in an improved computational time when using parallel processing computers.

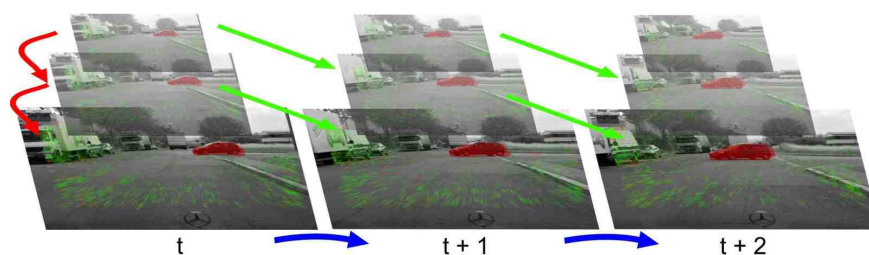


Figure D.3: The figure shows the temporal prediction on pyramids. Instead of propagating results down (red), or from one time instance to another on the same pyramid level (blue), a combined approach propagates results between frames and pyramid levels to combine both: real-time detection and real-time tracking.

Other techniques to speed up the segmentation algorithm have been presented by [81]. The authors use the search tree of the old segmentation as a hint for the current segmentation. In [42], an approximation of the energy is proposed which yields faster convergence. Both approaches are applied on a graph without changing its topology. In the following section, a novel approach is presented related to temporal prediction on pyramids, but performing a single cut on an image with different resolutions. This is related to surface approximations, where primitives may consist of different scales, such that a good approximation is achieved with the constraint, to use a minimal number of primitives (e.g. 3D surface modeling).

D.2 Multi-Resolution Graph-Cut

If parallel processing is not possible, the temporal prediction on pyramids yields no computation time gain compared to the plain vanilla pyramid implementation of the graph-cut algorithm. A closer look at the temporal prediction also reveals that situations occur, where the same segmentation boundary is found on every image level, such that a prediction onto the next time instance creates similar problems as using a full pyramid.

The idea to solve this issue is to take the highest pyramid image (lowest resolution) and iteratively refine the pixels into 4 sub pixels if they are within the segmentation border of the lower pyramid level (Figure D.4 shows the result and a schematic illustration). The resulting image consists of different resolutions, representing different pyramid layers; thus different segmentation accuracies. A high resolution is obtained at segmentation boundaries (inhomogeneous image regions) and a low resolution at homogeneous image regions. This idea is now transferred onto a graph representation. The advantage is obvious, as only a single cut has to be performed on the multi-resolution image, respectively graph.

The mapping of the multi-resolution image onto the graph is straightforward. In the first frame t , a coarse grid is created. Every grid section is a node, every node has an edge from the source and one to the sink. Pixels within the same grid are combined by adding their weights from the source and to the sink. The remaining edges, for each node, are created from the \mathcal{N}_4 boundary grid nodes (identified by any red pixel touching a blue grid line in Figure D.4). The weights of all pixel edges connecting two grid cells are summed to obtain a single edge weight. The graph-cut is performed on the coarse grid, providing a rough segmentation.

In the next time frame ($t + 1$), the grid sections on the boundaries of the segmentations are split into 4 sections. The old node (grid section) and corresponding edges are removed from the graph and the new nodes are added. Finally, the new edges are added to the graph. See Figure D.4 for a schematic illustration of this process. This process is then refined again in the next frame ($t + 2$). This will happen iteratively over time until the segmentation boundary is refined to single pixel detail.

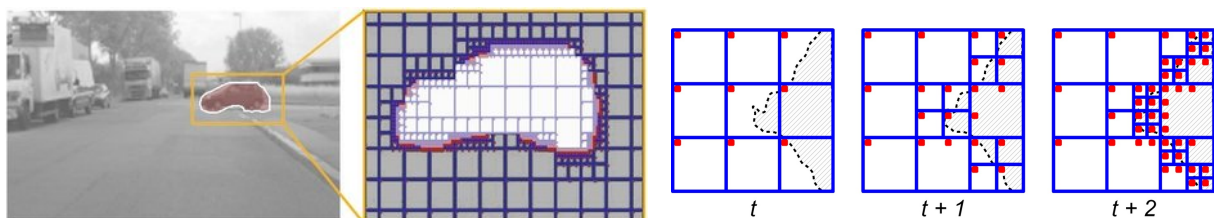
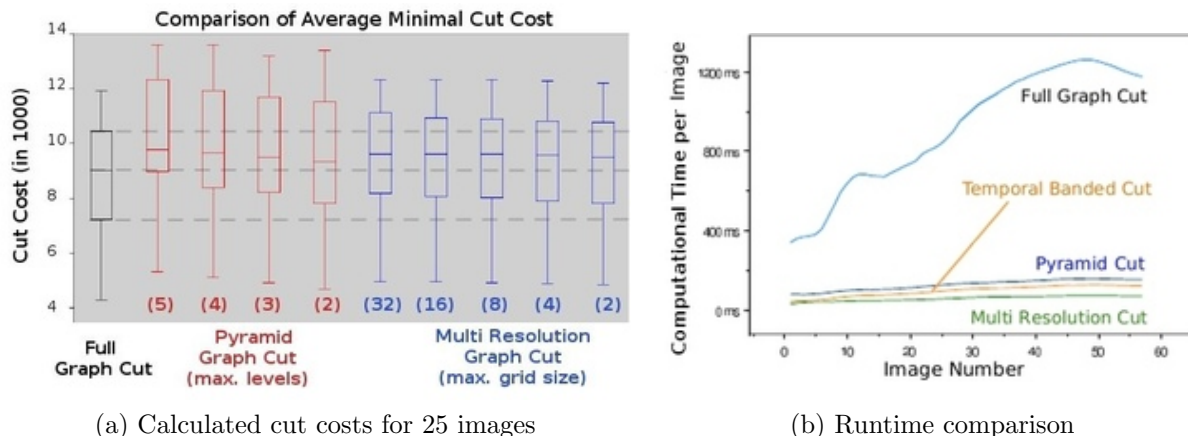


Figure D.4: The left image shows an example with a segmentation overlay. The graph on the right demonstrates the iterative segmentation approach of the multi-resolution graph-cut. The dashed line shows the edge of the true segmentation. Blue grid lines identify graph sections (nodes). Red dots touching a blue grid line indicate edges.



(a) Calculated cut costs for 25 images

(b) Runtime comparison

Figure D.5: The minimum cut cost of the multi-resolution method increases slightly with larger grid size and is close to the cost of the full graph-cut algorithm. Note: Do not mix up the cut costs, which deals with the energy minimum, and the computational cost.

As the segmentation is performed on image sequences, the implicit movement within the images over time has to be dealt with. The movement of the segmentation area can be estimated using the tracked features used for the motion constraints. In this case, the segmentation boundary in frame $(t + 1)$ is estimated from frame t using motion data (e. g. optical flow). The segmentation in frame $(t + 1)$ is performed as above, using the estimated boundary position. The graph-cut is performed over the entire image, still allowing new features to be detected.

Another issue that needs to be dealt with when using this approach is when there are refined areas in the image, that are not located on a segmentation boundary of the subsequent frame. When this happens, the process is reversed over the involved homogeneous grid cells, reverting to a lower resolution.

Figure D.5 shows a comparison of the presented algorithm in terms of computation time and minimal cut costs; identifying a minor loss in minimization. See Figure D.6 for a qualitative comparison on a sample image. In addition, the pyramid version and the temporal prediction techniques, other techniques used to speed up graph-cut computation time, are included in the runtime comparison. As the temporal banded cut is used for tracking and not detection, the resulting boundary from the multi-resolution method was provided as an initialization in every frame. The sequence used had increased movement near the end, this explains the increase in computational time, especially for the original graph-cut.

The multi-resolution method performed best on our test sequence. The results confirm that run times under 50 ms are achieved, such that real time segmentation becomes possible. The speed improvements using the multi-resolution approach, compared to the original graph-cut, lies between a factor of 10 and 20.



Figure D.6: Segmentation using the multi-resolution method in comparison to the full graph-cut method. The left images show the multi-resolution graph and the corresponding segmentation overlay. The right image shows the results using full graph-cut, yielding similar results.

Bibliography

Author publications

- [1] J. Klappstein, T. Vaudrey, C. Rabe, A. Wedel, and R. Klette. Moving object segmentation using optical flow and depth information. In *Proc. Pacific-Rim Symposium on Image and Video Technology*, pages 611–623, Tokyo, Japan, January 2009.
- [2] T. Vaudrey, D. Gruber, A. Wedel, and J. Klappstein. Space-time multi-resolution banded graph-cut for fast segmentation. In *Pattern Recognition (Proc. DAGM)*, pages 203–213, Munich, Germany, June 2008.
- [3] T. Vaudrey, A. Wedel, and R. Klette. A methodology for evaluating illumination artifact removal for corresponding images. Submitted to *Proc. International Conference on Computer Analysis of Images and Patterns*, Münster, Germany, September 2009.
- [4] T. Vaudrey, A. Wedel, C. Rabe, J. Klappstein, and R. Klette. Evaluation of moving object segmentation comparing 6D-Vision and monocular motion constraints. In *Online-Proc. Image and Vision Computing New Zealand*, Christchurch, New Zealand, November 2008.
- [5] A. Wedel. Detektion stationärer Hindernisse in monokularen Bildsequenzen. Master’s thesis, Computer Vision Group, University of Bonn, Germany, April 2006.
- [6] A. Wedel, H. Badino, C. Rabe, U. Franke, and D. Cremers. B-Spline modeling of road surfaces with an application to free space estimation. Accepted at *Special Issue of the IEEE Transactions on Intelligent Traffic Systems*, 2009.
- [7] A. Wedel and U. Franke. Monocular video serves RADAR-based emergency braking. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 93–98, Istanbul, Turkey, June 2007.
- [8] A. Wedel, U. Franke, H. Badino, and D. Cremers. B-Spline modeling of road surfaces for freespace estimation. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 828–833, Eindhoven, Netherlands, June 2008.
- [9] A. Wedel, U. Franke, J. Klappstein, T. Brox, and D. Cremers. Realtime depth estimation and obstacle detection from monocular video. In *Pattern Recognition (Proc. DAGM)*, pages 475–484, Berlin, Germany, September 2006.
- [10] A. Wedel, T. Pock, J. Braun, U. Franke, and D. Cremers. Duality TV-L1 flow with fundamental matrix prior. In *Online-Proc. Image and Vision Computing New Zealand*, Christchurch, New Zealand, November 2008.
- [11] A. Wedel, T. Pock, and D. Cremers. Structure- and motion-adaptive regularization for high accuracy optic flow. Submitted to *Proc. International Conference on Computer Vision*, Kyoto, Japan, August 2009.
- [12] A. Wedel, T. Pock, C. Zach, D. Cremers, and H. Bischof. An improved algorithm for TV-L1 optical flow. To appear in *Workshop on Statistical and Geometrical Approaches to Visual Motion Analysis*, Schloss Dagstuhl, Germany, September 2008.
- [13] A. Wedel, C. Rabe, A. Meissner, U. Franke, and D. Cremers. Detection and segmentation of independently moving objects from dense scene flow. Submitted to *Proc. International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*, Bonn, Germany, August 2009.
- [14] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. Franke, and D. Cremers. Efficient dense scene flow from sparse or dense stereo data. In *Proc. European Conference on Computer Vision*, pages 739–751, Marseille, France, October 2008.
- [15] A. Wedel, T. Schoenemann, T. Brox, and D. Cremers. WarpCut - Fast obstacle segmentation in monocular video. In *Pattern Recognition (Proc. DAGM)*, pages 264–273, Heidelberg, Germany, September 2007.
- [16] A. Wedel, T. Vaudrey, A. Meissner, C. Rabe, T. Brox, U. Franke, and D. Cremers. Decoupling motion and position of image flow with an evaluation approach to scene flow. To appear in *Workshop on Statistical and Geometrical Approaches to Visual Motion Analysis*, Schloss Dagstuhl, Germany, September 2008.

Web References

- [17] Bosch. Innovationen von Bosch — Night Vision. [Online; accessed 17-December-2008] http://www.bosch.de/innovationen_kfz/content/language1/html/nightvision.htm.
- [18] J. Y. Bouguet. Camera calibration toolbox for matlab, 1998. [Online; accessed 17-December-2008] <http://www.vision.caltech.edu/bouguetj>.
- [19] N. Devillard. Fast median search: an ANSI C implementation, July 1998. [Online; accessed 17-December-2008] <http://ndevilla.free.fr/median/median/index.html>.
- [20] D. Fleet and A. Hertzmann. Lecture notes on camera models, 2005. [Online; accessed 17-December-2008] <http://www.cs.utoronto.ca/~jepson/csc2503/readings/Camera.pdf>.
- [21] K. Nakahama. The aperture problem in apparent motion. [Online; accessed 25-April-2009] http://pages.slc.edu/~ebj/sight_mind/motion/Nakayama/aperture_problem.html.
- [22] D. Ottoson. Presentation speech: The nobel prize in physiology or medicine 1981, 1981. [Online; accessed 25-February-2009] http://nobelprize.org/nobel_prizes/medicine/laureates/1981/presentation-speech.html.
- [23] P. Sharke. Smart cars. [Online; accessed 25-April-2009] <http://www.memagazine.org/contents/current/features/smartcar/smartcar.html>.
- [24] Wikipedia. Epipolar Geometry — Wikipedia, the free encyclopedia, 2008. [Online; accessed 25-April-2009] http://en.wikipedia.org/wiki/Lagrangian_relaxation.
- [25] Wikipedia. Flip book — Wikipedia, the free encyclopedia, 2009. [Online; accessed 25-April-2009] http://en.wikipedia.org/wiki/Flip_book.
- [26] Wikipedia. Lagrangian Relaxation — Wikipedia, the free encyclopedia, 2009. [Online; accessed 25-April-2009] http://en.wikipedia.org/wiki/Lagrangian_relaxation.
- [27] Wikipedia. Phenakistoscope — Wikipedia, the free encyclopedia, 2009. [Online; accessed 25-April-2009] <http://en.wikipedia.org/wiki/Phenakistoscope>.
- [28] Wiktionary. Kinesthesia — Wiktionary, the free dictionary, 2008. [Online; accessed 25-April-2009] <http://en.wiktionary.org/wiki/kinesthesia>.
- [29] J. Yellott. A chronological history of vision research: 1600-1960, 2009. [Online; accessed 05-April-2009] http://www.cogsci.uci.edu/vision/yellott_dates.html.

Other References

- [30] L. Alvarez, J. Esclarín, M. Lefébure, and J. Sánchez. A PDE model for computing the optical flow. In *Proc. XVI Congreso de Ecuaciones Diferenciales y Aplicaciones*, pages 1349–1356, Gran Canaria, Spain, September 1999.
- [31] G. Aubert, R. Deriche, and P. Kornprobst. Computing optical flow via variational techniques. *SIAM Journal on Applied Mathematics*, 60(1):156–182, 1999.
- [32] J.-F. Aujol, G. Gilboa, T. Chan, and S. Osher. Structure-texture image decomposition—modeling, algorithms, and parameter selection. *International Journal of Computer Vision*, 67(1):111–136, 2006.
- [33] H. Badino, D. Pfeiffer, and U. Franke. The stixel world – a compact medium level representation of the 3d-world. Submitted to *Pattern Recognition (Proc. DAGM)*, Jena, Germany, September 2009.
- [34] D. Baehring, S. Simon, W. Niehsen, and C. Stiller. Detection of close cut-in and overtaking vehicles for driver assistance based on planar parallax. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 290–295, Las Vegas, USA, June 2005.
- [35] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, March 2004.
- [36] S. Baker, S. Roth, D. Scharstein, M. Black, J. Lewis, and R. Szeliski. A database and evaluation methodology for optical flow. In *Online-Proc. International Conference on Computer Vision*, Rio de Janeiro, Brazil, October 2007.
- [37] M. Barni, V. Cappellini, and A. Mecocci. Fast vector median filter based on Euclidean norm approximation. *Signal Processing Letters*, 1(6):92–94, June 2004.
- [38] G. Barrows, J. Chahl, and M. Srinivasan. Biomimetic visual sensing and flight control. *The Aeronautical Journal*, 107(1069):159–168, 2003.
- [39] A. Barth and U. Franke. Where will the oncoming vehicle be the next second? In *Proc. IEEE Intelligent Vehicles Symposium*, pages 1068–1073, Eindhoven, Netherlands, June 2008.
- [40] M. Black and P. Anandan. A framework for the robust estimation of optical flow. In *Proc. International Conference on Computer Vision*, pages 231–236, Nice, France, October 1993.
- [41] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.
- [42] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(11):1222–1239, 2001.
- [43] T. Brox. *From Pixels to Regions: Partial Differential Equations in Image Analysis*. PhD thesis, Faculty of Mathematics and Computer Science, Saarland University, Germany, April 2005.
- [44] T. Brox, C. Bregler, and J. Malik. Large displacement optical flow. In *To appear in Proc. International Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, June 2009.
- [45] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *Proc. European Conference on Computer Vision*, pages 25–36, Prague, Czech Republic, May 2004.
- [46] T. Brox, B. Rosenhahn, D. Cremers, and H.-P. Seidel. High accuracy optical flow serves 3-D pose tracking: exploiting contour and flow based constraints. In *Proc. European Conference on Computer Vision*, pages 98–111, Graz, Austria, May 2006.
- [47] A. Bruhn and J. Weickert. A confidence measure for variational optic flow methods. *Geometric Properties for Incomplete Data*, pages 283–298, March 2006.
- [48] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Variational optic flow computation in real-time. *IEEE Transactions on Image Processing*, 14(5):608–615, May 2005.
- [49] A. Bruhn, J. Weickert, T. Kohlberger, , and C. Schnörr. A multigrid platform for real-time motion computation with discontinuity-preserving variational methods. *International Journal of Computer Vision*, 70(3):257–277, 2006.

- [50] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1):89–97, 2004.
- [51] A. Chambolle. Total variation minimization and a class of binary MRF models. In *Proc. International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 136–152, St. Augustine, FL, USA, October 2005.
- [52] T. F. Chan, G. H. Golub, and P. Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM Journal on Applied Mathematics*, 20(10):1964–1977, 1999.
- [53] M. Clauss, P. Bayerl, and H. Neumann. Segmentation of independently moving objects using a maximum-likelihood principle. In *Proc. Autonome Mobile Systeme*, pages 81–87, Stuttgart, Germany, December 2005.
- [54] I. Cohen. Nonlinear variational method for optical flow computation. In *Scandinavian Conf. on Image Analysis*, pages 523–523, 1993.
- [55] T. Coleman and L. Hulbert. A direct active set algorithm for large sparse quadratic programs with simple bounds. *Mathematical Programming: Series A and B*, 45(3):373–406, 1989.
- [56] T. Cooke. Two applications of graph-cuts to image processing. In *Digital Image Computing: Techniques and Applications (DICTA)*, pages 498–504, Canberra, Australia, December 2008.
- [57] D. Cremers and S. Soatto. Motion competition: A variational framework for piecewise parametric motion segmentation. *International Journal of Computer Vision*, 62(3):249–265, May 2005.
- [58] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov. Bilayer segmentation of live video. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 53–60, New York, NY, USA, June 2006.
- [59] R. Deriche, P. Kornprobst, and G. Aubert. Optical flow estimation while preserving its discontinuities: A variational approach. In *Proc. Asian Conference on Computer Vision*, pages 290–295, Singapore, December 1995.
- [60] E. Eriksson. Movement parallax and distance perception in the grasshopper (*Phaulacridium Vitatum* (Sjostedt)): Short communications. *Journal of Experimental Biology*, 86(1):337–340, 1980.
- [61] O. Faugeras. In memory of Hugh Christopher Longuet-Higgins. *Proc. International Conference on Computer Vision*, 60(1):1–2, 2004.
- [62] M. Felsberg. On the relation between anisotropic diffusion and iterated adaptive filtering. In *Pattern Recognition (Proc. DAGM)*, pages 436–445, Munich, Germany, June 2008.
- [63] U. Franke and A. Joos. Real-time stereo vision for urban traffic scene understanding. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 273–278, Dearborn, October 2000.
- [64] U. Franke, C. Rabe, H. Badino, and S. Gehrig. 6D-Vision: Fusion of stereo and motion for robust environment perception. In *Pattern Recognition (Proc. DAGM)*, pages 216–223, Vienna, Austria, August 2005.
- [65] A. Giachetti, M. Campani, V. Torre, and C. CRS. The use of optical flow for road navigation. *IEEE Transactions on Robotics and Automation*, 14(1):34–48, 1998.
- [66] J. Gibson. *The perception of the visual world*. Houghton Mifflin, 1950.
- [67] Greig. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2):271–279, 1989.
- [68] R. Hartley and R. Vidal. The multibody trifocal tensor: Motion segmentation from 3 perspective views. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 769–775, Washington, DC, USA, June 2004.
- [69] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.
- [70] H. Haussecker and D. Fleet. Estimating optical flow with physical models of brightness variation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):661–673, 2001.
- [71] J. Heitwerth, R. Kern, and M. Egelhaaf. On the segregation of object and background by a neuron most sensitive to small-field motion. In *Göttingen Meeting of the German Neuroscience Society*, 2007.

- [72] H. Hirschmüller. Stereo vision in structured environments by consistent semi-global matching. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 2386–2393, New York, NY, USA, June 2006.
- [73] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1–3):185–203, 1981.
- [74] F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *Online-Proc. International Conference on Computer Vision*, Rio de Janeiro, Brazil, October 2007.
- [75] M. Isard and J. MacCormick. Dense motion and disparity estimation via loopy belief propagation. In *Proc. Asian Conference on Computer Vision*, pages 32–41, Hyderabad, India, January 2006.
- [76] W. Karush. *Minima of Functions of Several Variables with Inequalities as Side Constraints*. PhD thesis, Dept. of Mathematics, University of Chicago, 1939.
- [77] J. Klappstein. *Optical-Flow Based Detection of Moving Objects in Traffic Scenes*. PhD thesis, University of Heidelberg, Heidelberg, Germany, 2008.
- [78] J. Klappstein, F. Stein, and U. Franke. Monocular motion detection using spatial constraints in a unified manner. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 261–267, Tokyo, Japan, June 2006.
- [79] J. Klappstein, F. Stein, and U. Franke. Applying Kalman filtering to road homography estimation. In *Workshop on Planning, Perception and Navigation for Intelligent Vehicles in conjunction with IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, April 2007.
- [80] R. Klette and G. J. Tee. Understanding human motion: A historic review. Technical Report CITR-TR-192, Multimedia Imaging, University of Auckland, Auckland, NZ, January 2007.
- [81] P. Kohli and P. H. S. Torr. Efficiently solving dynamic Markov random fields using graph cuts. In *Proc. International Conference on Computer Vision*, pages 922–929, Washington, DC, USA, October 2005.
- [82] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother. Bi-layer segmentation of binocular stereo video. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 1186–1192, June 2005.
- [83] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? In *Proc. European Conference on Computer Vision*, pages 65–81, Copenhagen, Denmark, May 2002.
- [84] W. G. Kropatsch. History of computer vision – a personal perspective, May 2008.
- [85] V. Lempitsky, S. Roth, and C. Rother. Fusionflow: Discrete-continuous optimization for optical flow estimation. In *Online-Proc. International Conference on Computer Vision and Pattern Recognition*, Anchorage, USA, June 2008.
- [86] D. Litwiller. CCD vs. CMOS: Maturing technologies, maturing markets. *Photonics Spectra*, pages 54–58, August 2005.
- [87] P. Loftus. A car with its own eyes. *The Wall Street Journal*, September 2002.
- [88] H. Lombaert, Y. Sun, L. Grady, and C. Xu. A multilevel banded graph cuts method for fast image segmentation. In *Proc. International Conference on Computer Vision*, pages 259–265, Washington, DC, USA, October 2005.
- [89] H. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. *Proc. of the Royal Society of London. Series B, Biological Sciences (1934-1990)*, 208(1173):385–397, 1980.
- [90] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the 7th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679, Vancouver, British Columbia, Canada, April 1981.
- [91] D. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 1984.
- [92] Q. Luong and O. Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17(1):43–75, 1996.
- [93] Q.-T. Luong and O. D. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of Computer Vision*, 22(3):261–289, 1997.

- [94] P. C. Mahalanobis. On the generalised distance in statistics. In *Proc. of the National Institute of Science of India 12*, pages 49–55, India, 1936.
- [95] J. Malcolm, Y. Rathi, and A. Tannenbaum. Multi-object tracking through clutter using graph cuts. In *Online-Proc. International Conference on Computer Vision*, Rio de Janeiro, Brazil, October 2007.
- [96] H. Maldonado, M. Benko, and M. Isern. Study of the role of binocular vision in mantids to estimate long distances, using the deimatic reaction as experimental situation. *Comparative Physiology A*, 68(1):72–83, March 1970.
- [97] D. Marr. *Vision: A computational investigation into the human representation and processing of visual information*. W.H.Freeman & Co Ltd, 1982.
- [98] A. Masoomzadeh-Fard and A. Venetsanopoulos. An efficient vector ranking filter for colour image restoration. In *Proc. Canadian Conference on Electrical and Computer Engineering*, pages 1025–1028, Vancouver, BC, Canada, September 1993.
- [99] G. J. McLachlan. Mahalanobis distance. *Resonance-Journal of science education*, 4(6):20–26, June 1999.
- [100] E. Mémin and P. Pérez. Dense estimation and object-based segmentation of the optical flow with robust techniques. *IEEE Transactions on Image Processing*, 7(5):703–719, May 1998.
- [101] E. Mémin and P. Pérez. A multigrid approach for hierarchical motion estimation. In *Proc. International Conference on Computer Vision*, pages 933–938, Bombay, India, January 1998.
- [102] Y. Mileva, A. Bruhn, and J. Weickert. Illumination-robust variational optical flow with photometric invariants. *Pattern Recognition (Proc. DAGM)*, pages 152–162, September 2007.
- [103] D. Min and K. Sohn. Edge-preserving simultaneous joint motion-disparity estimation. In *Proc. International Conference on Pattern Recognition*, pages 74–77, Hong Kong, China, August 2006.
- [104] J. Mooser, S. You, and U. Neumann. Real-time object tracking for augmented reality combining graph cuts and optical flow. In *Int. Symposium on Mixed and Augmented Reality*, pages 145–152, Nara, Japan, November 2007.
- [105] H. H. Nagel. Constraints for the estimation of displacement vector fields from image sequences. In *Proc. Eighth Int. Conf. Artif. Intell.*, pages 945–951, Karlsruhe, Germany, 1983.
- [106] H.-H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(5):565–593, 1986.
- [107] D. Nistér and H. Stewénius. A minimal solution to the generalized 3-point pose problem. *Journal of Mathematical Imaging and Vision*, 1(27):560–567, 2006.
- [108] I. Patras, N. Alvertos, and G. Tziritas. Joint disparity and motion field estimation in stereoscopic image sequences. In *Proc. International Conference on Pattern Recognition*, pages 359–363, Vienna, Austria, August 1996.
- [109] T. Pock. *Fast Total Variation for Computer Vision*. PhD thesis, Institute for Computer Graphics and Vision, University of Graz, Graz, Austria, 2008.
- [110] J.-P. Pons, R. Keriven, and O. Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision*, 72(2):179–193, 2007.
- [111] M. Poteser and K. Kral. Visual distance discrimination between stationary targets in praying mantis: an index of the use of motion parallax. *Journal of Experimental Biology*, 198(1):2127–2127, 1995.
- [112] C. Rabe, U. Franke, and S. Gehrig. Fast detection of moving objects in complex scenarios. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 398–403, Istanbul, Turkey, June 2007.
- [113] C. Rabe, C. Volmer, and U. Franke. Kalman filter based detection of obstacles and lane boundary. *Autonome Mobile Systeme*, 19(1):51–58, 2005.
- [114] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1-4):259–268, 1992.

- [115] K. Sand-Jensen. How to write consistently boring scientific literature. *Oikos*, 116(5):723–727, May 2007.
- [116] C. Schnörr. Segmentation of visual motion by minimizing convex non-quadratic functionals. In *12th Int. Conf. on Pattern Recognition*, pages 661–663, Jerusalem, Israel, October 1994.
- [117] T. Schoenemann. *Combinatorial Solutions for Shape Optimization in Computer Vision*. PhD thesis, Computer Vision Group, University of Bonn, Bonn, Germany, 2009.
- [118] M. Schubert, T. Prokop, F. Brocke, and W. Berger. Visual kinesthesia and locomotion in parkinson’s disease. *Movement Disorders*, 20(1):141–150, 2005.
- [119] M. Shimizu and M. Okutomi. Precise sub-pixel estimation on area-based matching. In *Proc. International Conference on Computer Vision*, pages 90–97, Vancouver, Canada, July 2001.
- [120] A. Sinop and L. Grady. Accurate banded graph cut segmentation of thin structures using Laplacian pyramids. In *Proc. Medical Image Computing and Computer-Assisted Intervention*, pages 896–903, Copenhagen, Denmark, October 2006.
- [121] N. Slesareva, A. Bruhn, and J. Weickert. Optic flow goes stereo: A variational method for estimating discontinuity-preserving dense disparity maps. In *Pattern Recognition (Proc. DAGM)*, pages 33–40, Vienna, Austria, August 2005.
- [122] G. Sperling and M. S. Landy. Kinetic depth effect and identification of shape. *Journal of Experimental Psychology*, 15(4):826–840, 1989.
- [123] H. Spies, N. Kirchgesner, H. Scharr, and B. Jähne. Dense structure estimation via regularised optical flow. In *Proc. Vision, Modeling, and Visualization*, pages 57–64, Saarbrücken, Germany, November 2000.
- [124] F. Stein. Efficient computation of optical flow using the Census transform. In *Pattern Recognition (Proc. DAGM)*, pages 79–86, Tübingen, Germany, August 2004.
- [125] S. S. Stevens, H. E. Pashler, and S. Yantis. *Stevens’ Handbook of Experimental Psychology, Sensation and Perception*. John Wiley and Sons, 2004.
- [126] E. Stewart. *Intel Integrated Performance Primitives: How to Optimize Software Applications Using Intel IPP*. Intel Press, 2004.
- [127] D. Sun, S. Roth, J. P. Lewis, and M. J. Black. Learning optical flow. In *Proc. European Conference on Computer Vision*, pages 83–91, Marseille, France, October 2008.
- [128] J. Sun, W. Zhang, X. Tang, and H. Shum. Background cut. In *Proc. European Conference on Computer Vision*, pages 628–641, Graz, Austria, May 2006.
- [129] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, April 1991.
- [130] P. H. S. Torr and A. Zisserman. Concerning Bayesian motion segmentation, model averaging, matching and the trifocal tensor. In *Proc. European Conference on Computer Vision*, pages 511–528, Freiburg, Germany, June 1998.
- [131] W. Trobin, T. Pock, D. Cremers, and H. Bischof. An unbiased second-order prior for high-accuracy motion estimation. In *Pattern Recognition (Proc. DAGM)*, pages 396–405, Munich, Germany, June 2008.
- [132] M. Unger, T. Pock, and H. Bischof. Continuous globally optimal image segmentation with local constraints. In *Computer Vision Winter Workshop*, February 2008.
- [133] L. Valgaerts, A. Bruhn, and J. Weickert. A variational approach for the joint recovery of the optical flow and the fundamental matrix. In *Pattern Recognition (Proc. DAGM)*, pages 314–324, Munich, Germany, June 2008.
- [134] J. van de Weijer and T. Gevers. Robust optical flow from photometric invariants. In *Int. Conf. on Image Processing*, pages 1835–1838, Singapore, October 2004.
- [135] T. Vaudrey, H. Badino, and S. Gehrig. Integrating disparity images by incorporating disparity rate. In *Proc. Workshop on Robot Vision*, pages 29–42, Auckland, New Zealand, February 2008.
- [136] T. Vaudrey, C. Rabe, R. Klette, and J. Milburn. Differences between stereo and motion behaviour on synthetic and real-world stereo sequences. In *Online-Proc. Image and Vision Computing New Zealand*, Christchurch, NZ, November 2008.

- [137] S. Vedula, S. Baker, P. Rander, and R. C. T. Kanade. Three-dimensional scene flow. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):475 – 480, March 2005.
- [138] R. Vidal, Y. Ma, S. Soatto, and S. Sastry. Two-view multibody structure from motion. *International Journal of Computer Vision*, 68(1):7–25, 2006.
- [139] R. Vidal and S. Sastry. Optimal segmentation of dynamic scenes from two perspective views. In *Proc. International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 281–285, Madison, WI, USA, June 2003.
- [140] A. Watson and A. Ahumada, Jr. Model of human visual-motion sensing. *Journal of the Optical Society of America A*, 2(2):322–341, 1985.
- [141] J. Weickert and T. Brox. Diffusion and regularization of vector- and matrix-valued images. In *Inverse Problems, Image Analysis and Medical Imaging. Contemporary Mathematics*, volume 313, pages 251–268. AMS, 2002.
- [142] J. Weickert and C. Schnörr. A theoretical framework for convex regularizers in PDE-based computation of image motion. *International Journal of Computer Vision*, 45(3):245–264, 2001.
- [143] C. Wheatstone. On some remarkable, and hitherto unobserved, phenomena of binocular vision. *Philosophical Transactions of the Royal Society of London Series B: Biological Sciences*, 128(1):371–394, 1838.
- [144] F. Woelk and R. Koch. Fast monocular Bayesian detection of independently moving objects by a moving observer. In *Pattern Recognition (Proc. DAGM)*, pages 27–35, Tübingen, Germany, August 2004.
- [145] L. Xu, J. Chen, and J. Jia. A segmentation based variational model for accurate optical flow estimation. In *Proc. European Conference on Computer Vision*, pages 671–684, Marseille, France, October 2008.
- [146] Z. Yang and M. D. Fox. Speckle reduction and structure enhancement by multichannel median boosted anisotropic diffusion. *EURASIP Journal on Applied Signal Processing*, pages 2492–2502, 2004.
- [147] W. Yin, D. Goldfarb, and S. Osher. Image cartoon-texture decomposition and feature selection using the total variation regularized L^1 functional. In *Variational, Geometric, and Level Set Methods in Computer Vision*, pages 73–80, Beijing, China, October 2005.
- [148] C. Zach, D. Gallup, and J. Frahm. Fast gain-adaptive KLT tracking on the GPU. In *Online-Proc. International Conference on Computer Vision and Pattern Recognition*, Anchorage, AK, June 2008.
- [149] C. Zach, A. Irschara, and H. Bischof. What can missing correspondences tell us about 3d structure and motion? In *Online-Proc. International Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, June 2008.
- [150] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. In *Pattern Recognition (Proc. DAGM)*, pages 214–223, Heidelberg, Germany, September 2007.
- [151] G. Zhang, J. Jia, W. Xiong, T. Wong, P. Heng, and H. Bao. Moving object extraction with a hand-held camera. In *Online-Proc. International Conference on Computer Vision*, Rio de Janeiro, Brazil, October 2006.
- [152] Y. Zhang and C. Kambhamettu. On 3d scene flow and structure estimation. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 778–785, Kauai Marriott, Hawaii, December 2001.
- [153] Z. Zhang. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 27(2):161–195, 1998.