TECHNISCHE UNIVERSITÄT MÜNCHEN

Fakultät für Informatik
Computer Aided Medical Procedures & Augmented Reality / I16

Contributions to Stereo Vision

Christian Unger

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr. D. Cremers
Prüfer der Dissertation:	1.	Priv.-Doz. Dr. S. Ilic
	2.	Prof. Dr. P. Sturm, INRIA Grenoble / Frankreich

Die Dissertation wurde am 06.03.2013 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 02.06.2013 angenommen.

Abstract

This thesis provides methods for binocular stereo matching and multi-view stereo reconstruction. Approaches that acquire a dense reconstruction from two or more views are an important subject of research in computer vision and are useful for a large variety of applications. Since the background of this thesis is automotive driver assistance, our work is motivated for efficient and accurate reconstruction techniques.

In particular, we present a new algorithm for the efficient computation of dense disparity maps in the context of local stereo matching. We also propose techniques to make the matching robust to inaccurately rectified image pairs. Our method maintains a high efficiency, even in challenging scenarios.

For accurate stereo vision, we propose a novel method that is based on simulated random walks. We introduce several systematic measures to explicitly address challenging problems like discontinuities, occlusions and slanted surfaces. Our proposal produces high grade disparity maps on difficult images and compares very well to the current state of the art.

Further, we address the accurate and, at the same time, efficient stereo reconstruction from multiple views. For this, we present a novel probabilistic multi-view stereo approach that fuses disparity maps of different input view pairs. Our implementation in the vehicle achieves real-time performance on an ordinary processor and computes robust and high-quality disparity maps.

Finally, we introduce various camera-based parking assistance functionalities including an automatic parking slot detection, a collision detector for the pivoting ranges of the doors, and novel image based rendering techniques for visualization. The influence of reconstruction quality on application reliability is showcased with extensive experiments.

Zusammenfassung

Diese Arbeit stellt Methoden für binokulares Stereo-Matching sowie für die Rekonstruktion von mehreren Bildern vor. Ansätze die eine 3D-Rekonstruktion von zwei oder mehreren Bildern erzeugen bilden eine wichtige Forschungsrichtung im Bereich der Bildverarbeitung und sind darüber hinaus für unterschiedlichste Anwendungen nützlich. Da der praktische Hintergrund dieser Doktorarbeit Fahrerassistenzsysteme sind liegt der Schwerpunkt auf effizienten und möglichst exakt arbeitenden Methoden.

Insbesondere präsentieren wir einen neuen Algorithmus für die effiziente Berechnung von dichten Disparitätskarten im Kontext der lokalen Stereoverarbeitung. Dabei untersuchen wir verschiedene Maßnahmen um die Korrespondenzbildung robust gegenüber Beleuchtungsänderungen zu machen und stellen Verallgemeinerungen vor um die Disparitätsbestimmung bei fehlerhaft rektifizierten Bildpaaren zu ermöglichen.

Für die akkurate stereoskopische Rekonstruktion führen wir eine neue Methode ein die auf simulierten Zufallsbewegungen basiert (random walks). Dabei stellen wir verschiedene systematische Maßnahmen vor um die Korrespondenzbildung gegen schwierige Probleme robust zu machen, wie etwa Diskontinuitäten, Verdeckungen und geneigte Oberflächen. Unser Vorschlag generiert Disparitätskarten von sehr hoher Güte und schneidet im Vergleich mit aktuellen Methoden sehr gut ab.

Ferner adressieren wir in dieser Arbeit die akkurate und zugleich effiziente 3D-Rekonstruktion von mehreren Ansichten. Dazu stellen wir einen neuen probabilistischen Ansatz vor, welcher Disparitätskarten von mehreren Eingangsbildpaaren fusioniert. Unsere Implementierung im Fahrzeug erreicht Echtzeitperformance auf Standard-Prozessoren und berechnet robuste, qualitativ hochwertige und zeitlich kohärente Disparitätskarten.

Schließlich detaillieren wir verschiedene kamerabasierte Parkassistenzfunktionen, wie beispielsweise eine automatische Parklückenvermessung, einen Kollisionswarner für die Schwenkbereiche der Türen, sowie innovative bildgestützte Visualisierungen. Der Einfluss der Rekonstruktionsqualität auf die Zuverlässigkeit der automotiven Kundenfunktionen wird mit einer Vielzahl an Experimenten dargestellt.

Acknowledgements

First of all I would like to thank Slobodan Ilic for the supervision of this thesis. Working hours until 1:00 AM at the chair or nightly discussions via Skype during holidays is surely anything but granted, especially considering that I am an external PhD student. This thesis would not have been possible without your great support.

I also feel highly obliged to Professor Nassir Navab for making this thesis possible, for always having the right advice at hand and for guiding me in my early computer vision days.

Not less I wish to express my gratitude to Eric Wahl who made this thesis at BMW possible, who always stood behind my work, and who spent an invaluable amount of encouragement to push the topic within the BMW Group.

I also would like to acknowledge the people from CAMP and BMW. I always appreciate the creative suggestions of Bertram Drost, the very open and honest criticism of Cedric Cagniart. Thanks also go to my fellow Basti Lieberknecht and my student Vladimir Haltakov. Further, I would like to thank Selim Benhimane, for supervising my thesis in the first year and for giving the right impulses. Moreover, I want to send credits to Peter Sturm for giving precious advices. I also enjoyed the company of Christian Schmidts who was sitting right (left) next to me at BMW and who shared many thoughts with me about computer vision and also other topics. At this opportunity Andreas Koschar must be mentioned for producing some (at times highly) enjoyful office hours and thanks to the BMW Group for being a great company and for building such excellent cars! My thanks also go to Mario Nagelstrasser for his support and to Michele Del Mondo for his very early efforts. In addition, I am really grateful to Professor Winfried Recknagel for highly interesting mathematical and philosophical discussions and for laying a highly important foundation for my research interests.

I also feel very indebted to my parents, who always stood behind me and supported me, and made all my studies as well as this thesis possible. Their understanding and simply the right words helped me a lot and this is truly invaluable. Moreover, I am very thankful to my sister Sabine, who is always absolutely supportive, regardless of the size of the problem. Finally, I would like to thank my friends who accomanied me during this intense and unforgettable time.

Contents

1. Introduction

$9, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, \ldots$ **?** LOTHAR COLLATZ

In computer vision, stereo methods reconstruct a scene from two or more images. Until today, such approaches have been investigated for decades and it is still an active area of research that is thereby bolstered by a huge number of works. Mainly two directions are substantiated by different practical applications which naturally impose contradictory objectives. While some approaches try to improve the reconstruction quality and usually have low restrictions regarding execution time, other methods are required to run in real-time to support reactive control systems like robotic navigation, automotive driver assistance or industrial quality inspection.

In this work both aspects are of eminent importance. On the one hand, reconstruction quality is important to avoid an incorrect behavior of the application utilizing the depth data. On the other hand, an online operation is only possible if the real-time constraint is met reproducibly. In this spirit, we present various algorithms that address the efficiency, or the accuracy, or even both efficiency and accuracy. In particular, we present an efficient binocular stereo matching method which is robust to the decalibration of stereo rigs. We also introduce an innovative method for accurate stereo vision based on random walks which produces high-grade disparity maps on difficult images. Last but not least, we formalize a novel probabilistic approach that achieves robust and high-quality disparity maps from multiple camera positions and operates in real-time on ordinary computers.

Although this thesis was developed in an automotive context, our contributions for the reconstruction from two or more images are very generic and can be used in many applications. Albeit real-time performance is also a matter of the current technological development level and the amount of image data to be processed, computational efficiency is in many cases a driving factor. But also the correctness of depth measurements, the robustness of the reconstruction algorithm with respect to environmental influences and the precision of individual depth measurements are highly important for many applications. In this work, we analyze the connection between reconstruction quality and application reliability with the implementation of several driver assistance functionalities, which leads to our practical motivation.

1.1. Practical Motivation

Camera-based driver assistance has received a lot of attention in the past decade. Huge efforts have been made to bring various cameras and a variety of useful customer functions into serial production vehicles. Prominent examples are a front camera for forward looking driver assistance (such as traffic sign recognition, lane departure warning or high beam automation) and a rear camera for a reverse display with guide lines. Lesser known are side looking cameras which are integrated either into the side mirrors or into the front bumper, and help the driver during parking maneuvers or to observe crossing traffic (see also Fig. 1.1). Especially for parking assistance, low speed maneuvering and autonomous driving tasks, depth information about the vehicle's lateral space is required.

While cameras slowly but progressively conquer also cheaper serial production vehicles at different mounting positions, there are also ultrasonic, RADAR or LIDAR sensors available. These have the advantages of actively measuring depth and of being more robust in difficult environmental situations, but are less desirable than cameras in other aspects. Due to mechanical and physical limitations these sensors strongly reduce details acquired from the world to a small set of depth measurements. While LIDAR is capable of measuring reflectivity, none of these alternative sensors may provide true color information. Thus, every model derived from these sensors is coarse and less rich, and therefore the number of possible applications is limited. Moreover, the geometric integration of these sensors is often complicated and the cost of them is still higher than that of simple passive cameras, which might be further reasons for the fact that only lateral ultrasonic sensors have been commercially used yet for parking slot detection.

Even though cheap, lateral cameras were, so far, only used to improve the driver's visual perception by simply displaying the camera images. The complexity of image processing algorithms and the limited hardware resources (*i.e.* only mobile CPUs in our case) may have hid their potential for driver assistance systems, since real-time performance is required for most of these applications. Another reason is surely the fact that monocular cameras do not directly deliver depth information. To estimate distances many monocular systems rely on object detection, combined with assumptions about object size or flatness of the ground [32]. Naturally, in complex real world situations, these assumptions are often violated and, in turn, may impose large errors. Further, these approaches are less generic, because the classes and orientations of detectable objects must be defined and trained beforehand.

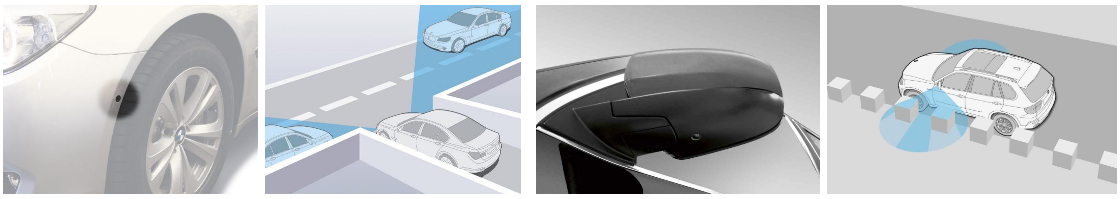
A more universal idea to compute depth information using monocular cameras is the principle of *motion-stereo*. However, the high computational overhead of most dense multi-view algorithms in combination with the limited processing power available on mobile vehicular platforms contradicts with strong real-time requirements. This situation made many researchers resort to feature-based approaches, but the availability of enough significant interest points cannot always be ensured in real world scenarios. Different to other works, we will establish dense correspondences and will make excessive use of this principle throughout the whole thesis.

To summarize, the main motivation for this work is to turn monocular cameras into robust depth sensors using dense stereo and multi-view stereo techniques.

1.2. Practical Challenges

The ultimate vision of this work is a camera-based parking assistant, which is able to densely reconstruct the environment in real-time in order to detect parking slots when passing by at every environmental condition. We broke this vision down into individual goals:

1. **Usage of serial production cameras**: Since this work tackles a practical problem, it is important that the used hardware is in step with actual vehicle equipment to enable a



Side-view camera at the front bumper

Top-view camera at the side mirror

Figure 1.1. Real-Time Motion-Stereo for automotive driver assistance: a camera on the vehicle observes the lateral space. If the vehicle moves, depth is inferred via motion-stereo.

later commercialization, but such cameras are often a compromise between cost and image quality. We resort to two particular classes of currently available side looking cameras which are illustrated in Fig. 1.1:

a) **Side-View cameras**: The mounting position, orientation and field of view (45-60 degrees) are very well suited for multi-view stereo algorithms.

b) **Top-View cameras**: In this case, the mounting position and orientation allow multi-view reconstruction. However, the large field of view (160-170 degrees) is a limiting factor, because it makes the search for correspondences difficult and it results in a larger reconstruction uncertainty.

2. **Real-time performance**: Our goal is that all computations are performed in real-time on commodity hardware. The main motivation is that lower required processing resources will, at least to some extent, lead to lower costs when implementing the algorithms on an automotive ECU. Another reason is that components which consume a lot of power are not realistic for vehicles. In our case, we use an ordinary mobile CPU without a GPU whose performance ranges between 18 and 41 GFLOPS[1]. As a comparison, the NVIDIA® GeForce® 8800 GTX GPU has a theoretical maximum (ignoring the memory bandwidth limitation that occurs in most practical implementations) of around 518 GFLOPS[2] and consumes much more power.

3. **Dense reconstruction without models**: The developed methods must produce dense 3D data of the environment to maximize the obstacle detection rate and the measurement accuracy. By not relying on monocular classification-based object detection cues like the trained classifiers of Viola and Jones [148], we avoid training and preserve genericity with respect to detectable objects.

4. **Robustness of obstacle detection**: We consider the detection of obstacles more important than the detection of free space. Having autonomous vehicle movements in mind, the detection of a false free space might have much worse consequences (for

[1] GFLOPS: Number of floating point operations per second times one billion; GFLOPS of some of the used CPUs (Source: Intel® Website): Core™2 Extreme Q9300 (quad core, 2.53GHz, 12MB L2, 1066MHz Bus): 40.48, Core™2 Duo E8200 (dual core, 2.66GHz, 6MB L2, 1333MHz Bus): 21.28, Core™2 Duo T7600 (dual core, 2.33GHz, 4MB L2, 667MHz Bus): 18.64

[2] Computed by multiplying 1.35GHz × 128 cores × 3 FLOPS (because some arithmetic operations can run in parallel)

example, a collision) than the false detection of an obstacle (for example, making the driver search longer for parking space).

5. **Robustness with respect to adverse vision conditions**: The determination of correspondences is complicated by several impacts that happen during the process of optical projection, ranging from weather influences to optical aberration within the lens, and camera internal control of the imager:

   a) **Difficult lighting situations**: Insufficient or artificial lighting is very challenging for computer vision approaches, but is common for our application. For example, incident sunlight leads to glare light effects and, for cameras that perform active exposure control, frequent exposure changes. Such glaring leads to suboptimal exposure of the imager and blooming. Furthermore, light which is scattered within the lens results in reduced contrast and lens flare patterns such as "starbursts" and circles. Our goal is to develop methods that work as robust as possible in these situations.

   b) **Dirty lenses**: In practice, the obstruction of the optical path can have a huge impact on image quality resulting in smoothing, reduced contrast and even "blind spots".

   c) **Active camera control**: The cameras we use operate with active control of exposure and white value to ensure an optimal image quality for functionalities that display camera images to the driver and to generally ensure optimal exposure in any situation. However, this may have negative effects on the performance of machine vision, especially when relating images of different time instances.

6. **Robustness with respect to extrinsic calibration**: Challenging are ground unevennesses and inaccuracies in estimating the vehicle position using odometry. In real world situations the ground is not perfectly flat and an accurate orientation (*e.g.* roll angle) of the vehicle is not available. Moreover, sensors on the vehicle that are used for position estimation are inaccurate (for example, odometry sensors make assumptions about the sizes of the wheels and their performance is even temperature dependent). Therefore, the developed methods must account for uneven ground and inaccurate camera positions provided by the vehicle, *i.e.* inaccurate epipolar geometry.

In short, we are in favour of dense real-time reconstruction methods that are able to operate with low-cost cameras and we put a very strong emphasis on robustness. These goals will be addressed throughout this work and will be reviewed in the end.

1.3. Contributions

In the course of this work, several algorithms for binocular and multi-view stereo as well as new driver assistance systems have been developed. Here is a short summary of the contributions that resulted from our continuous research on computer vision and camera-based driver assistance.

Efficient Reconstruction. First, we introduce a novel algorithm for binocular disparity computation which does not rely on an a priori choice of the maximum disparity [138]. This is mainly realized by iteratively performing minimization and propagation steps at every pixel. The proposed matching principle is more accurate and at the same time faster than a brute-force search and helps to achieve our real-time goal in the vehicle. It is particularly well suited for an application to our motion-stereo sequences, because usually there is a huge variation of the maximum disparity over time.

Second, we generalize our disparity computation algorithm so that inaccurately rectified image pairs can be processed efficiently and robustly [141]. This is highly relevant to long-term installations of stereo rigs in vehicles and to the application of stereo methods to images from moving monocular cameras. In these cases, pairs of images must usually be rectified very well to allow the application of dense stereo methods. The main idea is that additional scanlines are evaluated for possible correspondences, so that in this sense *epipolar deviations* are considered. In traditional matchers the required processing time increases linearly with the maximum epipolar deviation. Our experiments show that our proposal is much more robust than current art and that it is very efficient at the same time. For example, at some datasets our method is faster than traditional correlation without epipolar deviations. Moreover, we address radiometric variations of the image pairs using robust matching cost functions. The experiments demonstrate that our stereo proposal performs very well with difficult real world images. These generalizations particularly improve quality of the disparity maps in challenging motion-stereo scenarios.

It can also be shown that even very large displacements can be handled and that robust matching cost functions can be used in the disparity computation algorithm [140]. We show this with difficult real world image sequences in adverse vision conditions. In these cases, our correspondence algorithm performs very well, even when compared to recent methods like TV-L1, and achieves a 90 times speed up over traditional block matching.

Accurate Reconstruction. We introduce an innovative stereo method based on random walks for robust and accurate stereo vision. The central idea is to use simulations of random walks as matching primitives to achieve sharp object boundaries. We make the matching process systematically robust to challenging problems like discontinuities, occlusions and slanted surfaces. This is mainly achieved by using random walks as matching primitives because they, in some sense, perform a localized soft segmentation. Further, we introduce a few a priori surface orientations for cost aggregation in order to handle slanted surfaces and by using left-right random walk simulations we increase robustness in occluded regions. We explore the space of hypotheses contributed by random walks and build a voting space that serves to identify the most probable disparities and occluded pixels. Finally, we perform a propagation of confident disparities into inconsistent regions and use global optimization on a probability distribution over disparities to handle ambiguities. Extensive evaluations and top rankings at Middlebury show the versatility of our method on challenging images and demonstrate that these measures lead to very reliable and very accurate disparity maps.

Efficient and Accurate Reconstruction. We present a novel real-time multi-view reconstruction method that probabilistically fuses disparity maps [143]. We use a given set of disparity maps computed between pairs of input images and project them to a reference view pair. We estimate the reference disparity map efficiently using a probability density function of disparities and employ projection uncertainties. Using a variety of challenging data sets we show that our proposal is able to recover very accurate disparity maps. Notably, our method excels in areas near discontinuities and performs much better than competing state of the art approaches. Further, if the epipolar geometry is constrained to the motion-stereo use case, real-time operation is possible. In practice, our fusion method is very important and addresses many of the aforementioned challenges. Namely, it works in real-time, it significantly increases the robustness in difficult lighting conditions and it avoids many false matches.

Applications. Throughout this thesis we developed an innovative, interactive parking assistance system in a real vehicle that uses dense depth data computed in real-time using our efficient stereo and fusion methods. The flexibility of our motion-stereo framework is showcased with different customer-oriented applications [153, 142]. This includes an automatic parking slot detection that achieves a very high accuracy, reliable object detection rates, high availability and robustness to environmental influences. Moreover, we implemented a collision warning application that automatically detects objects that are located in the pivoting ranges of the doors, such that occupants are warned before opening the doors. We also present a novel image-based rendering technique[3], called *Augmented Parking* that visualizes the environment around the host vehicle from a bird's eye view. The detected parking slots, the host vehicle and surrounding obstacles are displayed over an image of the ground plane. We provide a highly elaborate evaluation using a huge amount of very challenging video sequences that comprises over 700 parking slots and different environmental conditions. We compare to current state of the art parking applications and provide many results of the customer functionalities.

1.4. Publications

In the course of this thesis, the following articles have been published:

Christian Unger, Selim Benhimane, Eric Wahl, and Nassir Navab. **Efficient disparity computation without maximum disparity for real-time stereo vision.** In *British Machine Vision Conference (BMVC)*, London, September 2009.

Eric Wahl, **Christian Unger**, Armin Zeller, and Dirk Rossberg. **3D-Environment Modeling as an Enabler for Autonomous Vehicles.** *ATZ Automobiltechnische Zeitschrift*, Ausgabe 02/2010.

Christian Unger, Eric Wahl, and Slobodan Ilic. **Efficient stereo and optical flow with robust similarity measures.** In Rudolf Mester and Michael Felsberg, editors, *Pattern Recognition (DAGM)*, volume 6835 of *Lecture Notes in Computer Science*, pages 246–255. Springer Berlin Heidelberg, 2011.

[3] Image-Based Rendering (IBR): a principle to generate virtual views of a scene, for example [139].

Christian Unger, Eric Wahl, and Slobodan Ilic. **Efficient stereo matching for moving cameras and decalibrated rigs.** In *Intelligent Vehicles Symposium*, pages 417–422, 2011.

Christian Unger, Eric Wahl, and Slobodan Ilic. **Parking assistance using dense motion-stereo.** *Machine Vision and Applications*, pages 1–21, 2011.

Christian Unger, Eric Wahl, Peter Sturm, and Slobodan Ilic. **Stereo fusion from multiple viewpoints.** In *Pattern Recognition (DAGM)*, volume 7476 of *Lecture Notes in Computer Science*, pages 468–477. Springer Berlin Heidelberg, 2012.

1.5. Outline

CHAPTER 2 gives a quick overview over the methodological background of this thesis, namely the projective geometry of one, two and more views. CHAPTER 3 reviews the current state of the art in binocular stereo, multi-view reconstruction and automotive driver assistance. CHAPTER 4 introduces efficient stereo matching, being one important building block of our application, and is critically analyzed with other efficient matching methods. In CHAPTER 5 we present our novel approach for accurate stereo vision using random walks, which is able to achieve top rankings at the famous Middlebury benchmark. In CHAPTER 6 we develop our novel probabilistic stereo fusion approach, which is highly efficient and in many comparisons more accurate than competing techniques. In CHAPTER 7 we describe our customer oriented applications in detail and present an exhaustive experimental validation using a vehicle equipped with cameras. Finally this thesis is concluded by CHAPTER 8. Moreover, a short overview on the symbols used in this thesis is given in appendix A.

2. Background

> *Das Bild ist ein Modell der Wirklichkeit* [. . .] *Die Form der Abbildung ist die Möglichkeit, dass sich die Dinge so zu einander verhalten, wie die Elemente des Bildes* [. . .] *Nach dieser Auffassung gehört also zum Bilde auch noch die abbildende Beziehung, die es zum Bild macht* [. . .] *Die abbildende Beziehung besteht aus den Zuordnungen der Elemente des Bildes und der Sachen.* LUDWIG WITTGENSTEIN [160]

This chapter gives an overview on the theoretical background for monocular, binocular and multiple view vision systems. Basically, in our application the scene is observed from multiple cameras. We therefore introduce single view geometry consisting of images, projective geometry using a pinhole camera model and radial lens distortion in section 2.1. Based on these terms, we derive an approximation of the theoretical error in monocular object detection systems. In section 2.2 we present basics on two view geometry, including epipolar geometry, rectification and the definition of disparity. We also perform an error analysis for binocular object detection systems. Here, we give only a very brief insight of well known concepts and refer to [56] for a detailed presentation.

2.1. Single View Geometry

The main goal of this section is to roughly outline the process of image formation, namely the projection of a 3D scene onto a 2D image plane.

2.1.1. Images

In our work, an image $\mathcal{I}$ is a function that maps a spatial two dimensional location to a grayscale or color value:

$$\mathcal{I} \,:\, \Omega^2 \longrightarrow \mathbb{R}^d \tag{2.1}$$

The value d is, for example, 3 for RGB color images. In practice, when using digital cameras, color values and spatial locations are discrete values, for example $\Omega \subset \mathbb{N}_0$. Consequently, given a pixel location $\mathbf{x}$, the color value is $\mathcal{I}(\mathbf{x})$.

2.1.2. Camera Model

We use central projection cameras, following the basic pinhole model as shown in Fig. 2.1. When using homogenous coordinates, the relationship between 3D locations $\mathbf{X}$ and 2D images $\mathbf{x}$ can simply be written as a matrix multiplication:

$$\mathbf{x} = \mathbf{P}\mathbf{X} \tag{2.2}$$

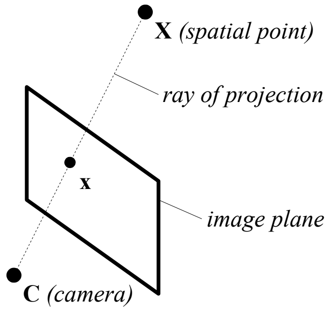


Figure 2.1. The pinhole camera model: the camera with optical center $\mathbf{C}$ projects the spatial point $\mathbf{X}$ on the image plane at position $\mathbf{x}$.

The projection matrix $\mathbf{P}$ is constructed using intrinsic (matrix $\mathbf{K}$) and extrinsic parameters (rotation matrix $\mathbf{R}$ and the optical camera center represented by the inhomogeneous vector $\mathbf{C}$):

$$\mathbf{P} = \mathbf{KR}\big(\mathbf{I}\,|\,-\mathbf{C}\big) \tag{2.3}$$

For finite projective cameras, $\mathbf{K}$ is considered of the form

$$\mathbf{K} = \begin{pmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{pmatrix} \tag{2.4}$$

where α_x and α_y is the focal length in pixels, s is a skew parameter (which is usually zero for most normal cameras) and (x_0, y_0) is the principal point in pixel coordinates. For the focal length α_x (and α_y accordingly), we have the following relationship:

$$\alpha_x = f \cdot \frac{1}{s_x} \tag{2.5}$$

where s_x is the metric pixel size (*e.g.* $6\,\mu m$) and f is the metric focal length.

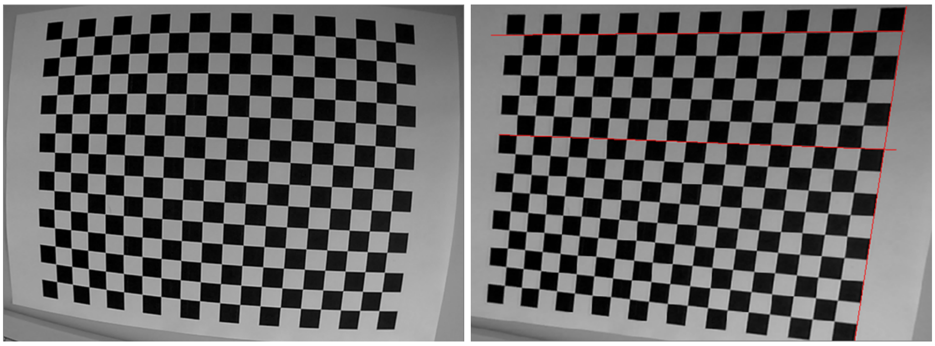
In practice, cameras are built by combining an imager with a lens. In these cases it is useful to compute the focal length from the imager resolution and the field of view of the lens:

$$\alpha_x = \frac{R_x}{2\tan\big(\frac{1}{2}F_x\big)} \tag{2.6}$$

where R_x is the horizontal resolution and F_x is the horizontal field of view.

2.1.3. Radial Lens Distortion

Especially lenses with a wide field of view have a form of optical aberration as result where straight lines in a scene do not remain straight in the projected image (see Fig. 2.2 for example images). For example, fisheye lenses utilize such distortions to project an infinite scene plane onto a finite region of the image plane. For many computer vision algorithms it is important to correct these effects. One way of doing this is by modeling it mathematically and by computing a compensating inverse transformation [30, 23, 76, 176].



(a) Distorted image

(b) Undistorted image

Figure 2.2. Example of Lens Distortion

In the Brown-Conrady distortion model [30, 23], radial and tangential (decentering) distortion is modeled using a polynomial:

$$x_u = x_d + (x_d - x_0) \cdot \sum_{i=1}^{n} \kappa_i r^{2i} + \tau_1 \left(r^2 + 2(x_d - x_0)^2\right) + 2\tau_2 (x_d - x_0)(y_d - y_0) \tag{2.7}$$

$$y_u = y_d + (y_d - y_0) \cdot \sum_{i=1}^{n} \kappa_i r^{2i} + \tau_2 \left(r^2 + 2(y_d - y_0)^2\right) + 2\tau_1 (x_d - x_0)(y_d - y_0) \tag{2.8}$$

$$r = \sqrt{(x_d - x_0)^2 + (y_d - y_0)^2} \tag{2.9}$$

where (x_u, y_u) is the undistorted point, (x_d, y_d) is the distorted pixel location, (x_0, y_0) is the principal point (*i.e.* the center of distortion), $(\kappa_i)_{1 \le i \le n}$ are the radial distortion coefficients and τ_1, τ_2 are the tangential distortion coefficients. In practice, n is often set to 2, since many lenses can be well approximated by a quadratic barrel distortion.

2.1.4. Homographies

A *homography* is an invertible projective 2D transformation and can be represented using a matrix in homogenous space: $\mathbf{x}' = \mathbf{H}\mathbf{x}$. In general, a projective 2D transformation has 8 degrees of freedom and may be used to transform the image plane. For example, a camera may be rotated virtually by $\mathbf{H} = \mathbf{K}\mathbf{R}'^{-1}\mathbf{R}^{-1}\mathbf{K}^{-1}$, because $\mathbf{H}\mathbf{P} = \mathbf{K}\mathbf{R}' \left(\mathbf{I}\,|{-\mathbf{C}}\right)$. In essence, this fact is used later for rectification.

2.1.5. Monocular Range Error Analysis

In automotive development, often the question about bounds on errors is raised. In the following, we analyze the error of a monocular vision system similar to [126, 32] that estimates distances to objects using the following assumptions:

- Objects are located on the ground plane (*e.g.* vehicles or pedestrians) and are detected based on the appearance (*e.g.* by shape).

- The distance is estimated by assuming a planar ground plane where the slope can be estimated from lane markings [87, 126].

Based on the previous pinhole camera model (assuming that $\mathbf{R} = \mathbf{I}$ and $\mathbf{C} = (0, -H, 0)^T$), the y-coordinate of a point $\mathbf{X} = (X, 0, Z, 1)^T$ on the road at distance Z in front of the vehicle is given by $y = y_0 + \frac{\alpha_y H}{Z}$, where H is the distance of the optical center to the ground plane. This implies that from a correctly estimated contact point (between the vehicle and the road), the distance can be computed (without loss of generality we set $y_0 = 0$):

$$Z = \frac{fH}{y} \tag{2.10}$$

In other words, the distance to the object is estimated from the distance of the contact point to the horizon line. In practice, both estimations (object detection and computation of the horizon line) have associated a specific error $\Delta y = \Delta s + \Delta h$ (Δs for the estimation of the contact point based on the segmentation of the object and Δh for the estimation of the horizon line). The absolute error of the distance estimation ΔZ is then given by:

$$\Delta Z = \tilde{Z} - Z = \frac{\alpha_y H}{y \pm \Delta y} - Z = \frac{\alpha_y H}{\frac{\alpha_y H}{Z} \pm \Delta y} - Z = \frac{\mp \Delta y \cdot Z^2}{\alpha_y H \pm \Delta y \cdot Z} \tag{2.11}$$

$$\Longrightarrow \Delta Z = \frac{(\Delta s + \Delta h) \cdot Z^2}{\alpha_y H - (\Delta s + \Delta h) \cdot Z} \tag{2.12}$$

2.2. Two View Geometry

A result of the projection is the loss of the third dimension. Binocular stereo techniques aim at recovering the 3D structure from correspondences between two images. In this section we closely look at the geometric properties of correspondences between two views, namely the epipolar geometry.

2.2.1. Epipolar Geometry

Correspondences across two views are constrained: given a point $\mathbf{x}$ in the first view, the corresponding point $\mathbf{x}'$ in the second view lies on a line which is called the *epipolar line* $\mathbf{l}'$. The epipolar line can be constructed geometrically by intersecting the epipolar plane (a plane that contains both camera centers and $\mathbf{x}$) with the image plane of the second camera. In practice, this is a very important result, because the search for correspondences need not cover the whole two dimensional image plane but can be restricted to the line $\mathbf{l}'$.

Please note that lines and points are dual entities in projective space, both representing a one-dimensional set of points: all points $\mathbf{x}$ that lie on a line $\mathbf{l}$ can be determined by solving $\mathbf{x}^T\mathbf{l} = 0$. Further, the algebraic cross product is used to compute a line joining two points or to determine the point of intersection of two lines: $\mathbf{l} = \mathbf{x}_1 \times \mathbf{x}_2$ and $\mathbf{x} = \mathbf{l}_1 \times \mathbf{l}_2$.

In general, all possible epipolar lines $\mathbf{l}'$ intersect in exactly one point, the *epipole*, which is the image of the other optical center. Geometrically, an epipole may be constructed by intersecting the *baseline* with the image plane (the baseline is the line that joins the two camera centers and is exactly the line which is contained by all epipolar planes), because all back-projected rays meet in the optical center. Please see Fig. 2.3 for an illustration on epipolar geometry.

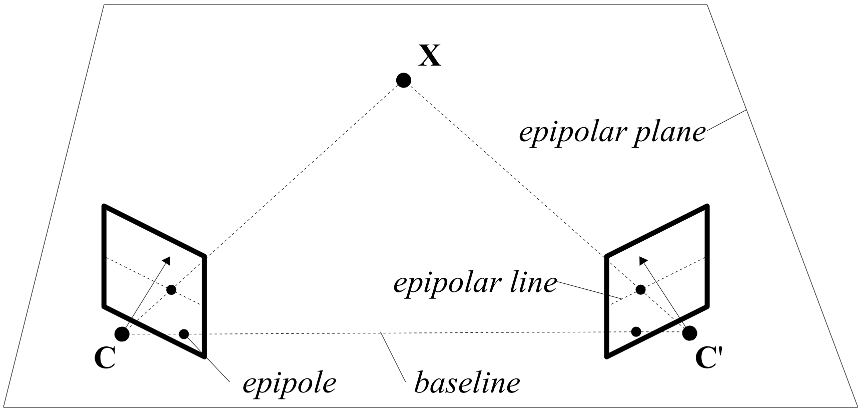


Figure 2.3. The epipolar geometry: the connecting line between two camera centers is called the baseline. This line is incident with the epipoles, which are the images of the other camera centers. Given a scene point $\mathbf{X}$ the epipolar plane is spanned together with the two camera centers $\mathbf{C}$ and $\mathbf{C}'$. The epipolar line in the right camera is the projection of the line joining $\mathbf{X}$ and $\mathbf{C}$. Hence, it can be computed by intersecting the image plane of the right camera with the epipolar plane.

2.2.2. The Fundamental Matrix

These geometric properties can be captured by the so called *fundamental matrix* $\mathbf{F}$. It maps from a point in the first image to the corresponding epipolar line in the second image:

$$\mathbf{F}(\mathbf{x}) \;:\; \mathbf{x} \;\mapsto\; \mathbf{l}' \tag{2.13}$$

One intuitive way to derive a closed form of the fundamental matrix was developed by Xu and Zhang [163]: the back-projected ray $\mathbf{X}(\lambda)$ in 3-space of point $\mathbf{x}$ may be obtained using the pseudo-inverse $\mathbf{P}^+$ of the projection matrix $\mathbf{P}$:

$$\mathbf{X}(\lambda) = \mathbf{P}^+\mathbf{x} + \lambda\mathbf{C} \tag{2.14}$$

where $\mathbf{C} = \mathbf{X}(\infty)$ is the optical center of $\mathbf{P}$. The epipolar line $\mathbf{l}'$ of $\mathbf{x}$ is then computed by joining the epipole $\mathbf{e}' = \mathbf{P}'\mathbf{C}$ with an arbitrary point on that line:

$$\mathbf{l}' = \left(\mathbf{P}'\mathbf{C}\right) \times \left(\mathbf{P}'\mathbf{X}(\lambda)\right) \tag{2.15}$$

Here, it is interesting that $(\mathbf{P}'\mathbf{C}) \times (\mathbf{P}'\mathbf{C}) = 0$, and directly leads to

$$= \left[\mathbf{e}'\right]_\times \left(\mathbf{P}'\mathbf{P}^+\right)\mathbf{x} \tag{2.16}$$

In this representation we use the so called *skew symmetric matrix* $[\mathbf{x}]_\times$ of a vector $\mathbf{x} = (x_1, x_2, x_3)^T$ and is defined as follows:

$$[\mathbf{x}]_\times = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix} \tag{2.17}$$

The fundamental matrix can therefore be found as:

$$\mathbf{F} = \left[\mathbf{e}'\right]_\times \mathbf{P}'\mathbf{P}^+ \tag{2.18}$$

The epipolar constraint. The fact that the epipolar line and the corresponding point are incident is known as the *epipolar constraint* and can be formulated algebraically:

$$\mathbf{x}'^T\mathbf{F}\mathbf{x} = 0 \tag{2.19}$$

Retrieval of Camera Matrices. With (2.18) the fundamental matrix may be obtained from known cameras. But also the opposite direction (*i.e.* the computation of camera matrices) is useful, for example, when the motion of the camera is unknown and the fundamental matrix has been computed from image correspondences by constructing a linear system of equations using (2.19). In this case, we may choose without loss of generality the first camera as $\mathbf{P} = (\mathbf{I}\,|\,\mathbf{0})$. For the second camera $\mathbf{P}'$, we need to first determine the epipole $\mathbf{e}'$ as the vector that spans the left null-space of $\mathbf{F}$ (this follows from $0 = (\mathbf{e}'^T\mathbf{F})\mathbf{x}$ for all $\mathbf{x}$ with $\mathbf{e}' \neq \mathbf{0}$). From that, there is no simple geometric intuition for the construction of $\mathbf{P}'$. However, with $\mathbf{x} = \mathbf{P}\mathbf{X}$ and $\mathbf{x}' = \mathbf{P}'\mathbf{X}$, it can be shown that $\mathbf{P}'$ should be constructed in a way such that $\mathbf{X}^T\mathbf{P}'^T\mathbf{F}\mathbf{P}\mathbf{X} = 0$ holds for all $\mathbf{X}$. From this observation, it is simple to verify that the suggestion of [94] is a good choice:

$$\mathbf{P} = \big(\mathbf{I}\,|\,\mathbf{0}\big) \qquad\qquad \mathbf{P}' = \big([\mathbf{e}']_\times\mathbf{F}\,|\,\mathbf{e}'\big) \tag{2.20}$$

2.2.3. Rectification

In general, an epipolar line may have any slope in image space which is hindering for efficient stereo implementations. From a practical point of view, the epipolar line must be computed for every pixel location of the first image when searching for correspondences. This involves a vector-matrix multiplication and a non-trivial iteration through discrete pixel locations of the second image along the epipolar line. From the motivation that a canonical epipolar geometry is most preferable, the idea is to transform both images in a way such that epipolar lines become horizontal and are matched up between views (this means that corresponding points have the same y-coordinate). From this follows that disparities measure the displacement in x-direction. This is the case when the image planes of both cameras are coplanar and when the baseline is parallel to the x-axes.

The basic idea behind rectification is simply to map the epipole to infinity using a homography [56]. Since this constrains only two degrees of freedom, the homographies are usually constructed in a way such that the image distortion is as small as possible. The standard approach is to create a homography $\mathbf{H}'$ for the first image which maps the epipole to the point at infinity $(1, 0, 0)^T$. Then, for the second image, a matched transformation $\mathbf{H}$ is computed based on $\mathbf{H}'$ so that epipolar lines are matched up. Further, $\mathbf{H}$ may be chosen to minimize the horizontal displacements between to two images. Another possibility is to explicitly use points on the plane at infinity, by ensuring that these points have a zero disparity.

After rectification the two cameras have the same orientation, same intrinsic parameters and their optical axes are orthogonal to the baseline:

$$\mathbf{P} = \mathbf{K}\mathbf{R}\big(\mathbf{I}\,|\,-\mathbf{C}\big) \qquad\qquad \mathbf{P}' = \mathbf{K}\mathbf{R}\big(\mathbf{I}\,|\,-\mathbf{C}'\big) \tag{2.21}$$

and

$$\mathbf{R}\big(\mathbf{C}' - \mathbf{C}\big) \times (1, 0, 0)^T = 0 \tag{2.22}$$

2.2.4. Disparity

The rectified camera setup consists of a left and a right camera, and the coordinates of corresponding image points $\mathbf{x}_L \leftrightarrow \mathbf{x}_R$ differ only in the x-coordinate. The difference of the two values is called the *disparity* and is stored per pixel in the *disparity map*: $\mathcal{D}(x_L, y_L) = x_L - x_R$.

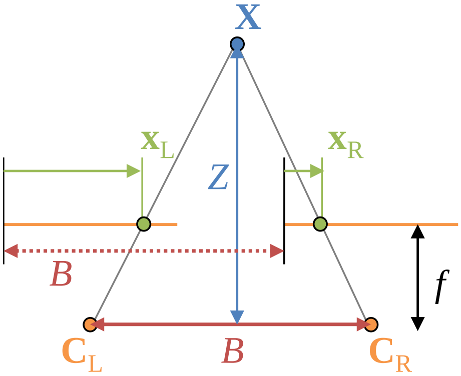


Figure 2.4. The relationship between depth and disparity: a 3D point $\mathbf{X}$ with depth Z is observed from a left and a right camera with baseline B, focal length f and optical centers $\mathbf{C}_L$ and $\mathbf{C}_R$. The projections of $\mathbf{X}$ are given by $\mathbf{x}_L$ and $\mathbf{x}_R$ for the left and right camera. The height and basis of the triangle $\triangle(\mathbf{X}\,\mathbf{C}_L\,\mathbf{C}_R)$ are directly given by Z and B, they have to be computed for $\triangle(\mathbf{X}\,\mathbf{x}_L\,\mathbf{x}_R)$ as $Z - f$ and $(B + x_R) - x_L$ accordingly. The intercept theorem together with $d = x_L - x_R$ leads to Eq. (2.23).

In general, the disparity is inversely related to the depth of the point. This can be intuitively derived using Fig. 2.4: a 3D point $\mathbf{X}$ with depth Z is observed from a left and a right camera with baseline B, focal length f and optical centers $\mathbf{C}_L$ and $\mathbf{C}_R$. The projections of $\mathbf{X}$ are given by $\mathbf{x}_L$ and $\mathbf{x}_R$ for the left and right camera. It can be immediately noticed that the triangles $\triangle_Z = \triangle(\mathbf{X}\,\mathbf{C}_L\,\mathbf{C}_R)$ and $\triangle_d = \triangle(\mathbf{X}\,\mathbf{x}_L\,\mathbf{x}_R)$ are congruent. From the fact that the baseline and the image planes are parallel, the intercept theorem may be applied: while the height and basis of $\triangle_Z$ are directly given by Z and B, they have to be computed for $\triangle_d$ as $Z - f$ and $(B + x_R) - x_L$ accordingly. So, with $d = x_L - x_R$ we get:

$$\frac{B}{Z} = \frac{B - d}{Z - f} \quad \Longrightarrow \quad d = \frac{fB}{Z} \tag{2.23}$$

Obviously, close scene points have a larger disparity than far ones and points on the plane at infinity have a disparity of zero.

2.2.5. Triangulation

The approach presented in the previous section may also be used to compute 3D locations from two view correspondences. Simple ways for the reconstruction are linear methods, where basically the algebraic property $\mathbf{x} \times (\mathbf{PX}) = \mathbf{0}$ is exploited to solve for the components of $\mathbf{X}$ using the singular value decomposition (SVD). For uncalibrated cameras and to rather minimize the geometric image error, minimization methods should be used [56].

In the calibrated rectified binocular case however, the reconstruction can be obtained more efficiently from the disparity values. We first determine calibrated depth values from disparities using:

$$Z = \frac{||\mathbf{KR}(\mathbf{C}' - \mathbf{C})||_2}{d} \tag{2.24}$$

Then, the inhomogeneous 3D location $\tilde{\mathbf{X}}$ is given by:

$$\tilde{\mathbf{X}} = Z\mathbf{R}^T\mathbf{K}^{-1}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} + \mathbf{C} \tag{2.25}$$

2.2.6. Binocular Range Error Analysis

We also perform an error analysis for the binocular position estimation. For the stereo rig we assume $\mathbf{R} = \mathbf{I}$, $\mathbf{C} = (0, -H, 0)^T$ and $\mathbf{C}' = (B, -H, 0)^T$ (which amounts to a baseline of B). In the binocular case, the distance is computed from disparity values. In practice, the disparity estimation has associated a specific error Δd, which depends on the stereo algorithm and the sub-pixel technology. So we let for the range error:

$$\Delta Z = \tilde{Z} - Z = \frac{fB}{d \pm \Delta d} - Z = \frac{fB}{\frac{fB}{Z} \pm \Delta d} - Z = \frac{\mp \Delta d Z^2}{fB \pm \Delta d Z} \tag{2.26}$$

$$\Longrightarrow \Delta Z = \frac{\Delta d Z^2}{fB - \Delta d Z} \tag{2.27}$$

2.2.7. Binocular Height Error Analysis

For specific applications, that base on a three dimensional reconstruction of the scene, the error of the height estimation is important. For example, one might be interested in reconstructing the ground plane in front of the vehicle using a forward looking stereo rig to dynamically adjust the height of the suspension, or to regulate traction control. For a moving vehicle, the range-error translates into a temporal uncertainty, but the error in Y-direction is critical for the applications. Using the assumptions of the previous section we get:

$$y = \frac{Y}{Z} f_y \tag{2.28}$$

So we let:

$$\Delta Y = \tilde{Y} - Y = \frac{y\tilde{Z}}{f_y} - \frac{yZ}{f_y} = \frac{y}{f_y}\Delta Z = \frac{Y}{Z} \cdot \frac{f_y}{f_y} \cdot \frac{Z^2 \Delta d}{f_x B - \Delta d Z} \tag{2.29}$$

$$\Longrightarrow \Delta Y = \frac{HZ\Delta d}{f_x B - \Delta d Z} \approx \frac{HZ\Delta d}{f_x B} \tag{2.30}$$

Here, it is interesting to note that $\Delta Y \propto H \cdot Z$ (lower mounting position is beneficial and only linearly depending on object distance).

2.2.8. Theoretical Error Discussion

The comparison of monocular and binocular range errors (ΔZ_M and ΔZ_B accordingly) is quite interesting due to the more or less surprising insight that both errors are in practice of the same magnitude. For an easier comparison, we repeat the results from (2.12) and (2.27):

$$\Delta Z_M = \frac{(\Delta s + \Delta h) \cdot Z^2}{f_y H - (\Delta s + \Delta h) \cdot Z} \qquad\qquad \Delta Z_B = \frac{\Delta d Z^2}{f_x B - \Delta d Z} \tag{2.31}$$

These equations already suggest a very similar structure and we will now further analyze these two terms based on practical considerations. First of all, we can assume that $f_x = f_y$. One major difference lies in the mounting height H and the baseline B: while both values should be as large as possible, the value H is usually around $1.20\,m$ for passenger cars and feasible values for B lie between $10 - 20\,cm$ (so in essence $H \approx 6B$).

The errors from image processing, namely Δs (for the contact point to the ground, which depends on the object segmentation), Δh (for the horizon line, which depends on pitch angle and the flatness of the road geometry, and may be determined from lane markings) and Δd (for the disparity estimation, which depends on the stereo algorithm, sub-pixel technology and also on image content), are more difficult to evaluate and are completely estimated from practical experience. The estimation of the contact point to the ground is often relatively imprecise (for example, weak contrast between the road and the tires of the object, and a large overhang at the back of the object) and we assume $\Delta s = 5\,px$. Also the estimation of the road geometry is often inaccurate, since lane markings may be hardly visible at far distances, or may be occluded by other traffic participants, or may even be not present. Unevennesses of the ground and sudden changes of the pitch angle introduce further perturbances and we assume $\Delta h = 5\,px$. The estimation of Δd is also not trivial, because the stereo matching approach, the sub-pixel technology, the dissimilarity function and also, to some extent, the image content have an impact on disparity estimation. However, it is often assumed that the error of disparity estimation ranges between $\frac{1}{16}px < \Delta d < 1\,px$, so we use $\Delta d = \frac{1}{4}px$.

Based on these quantities we can further simplify (2.31), because in practice $fH >> (\Delta s + \Delta h)Z$ and $fB >> \Delta dZ$:

$$\Delta Z_M \approx \frac{\Delta s + \Delta h}{f_y H} Z^2 \qquad\qquad \Delta Z_B \approx \frac{\Delta d}{f_x B} Z^2 \tag{2.32}$$

The most interesting observation here is that the error grows for both systems quadratically with the object distance (*i.e.* $\Delta Z \propto Z^2$). Further, all the assumptions made lead to the theoretical approximation that the monocular error is about seven times larger than the binocular – however, a good monocular system may still be better than a worse binocular.

3. State of the Art

> ***Stereo Vision*** *derives from Greek* stereos *(στερεός; solid, firm) and from Latin* vīsiō *(seeing, sight, view) which is the noun of action form from* vīsus *(the power of sight) which originates from Latin* videō *(I see, I perceive, I understand).* [2]

This chapter reviews the current state of the art in binocular stereo and multi-view reconstruction.

3.1. Matching Cost Functions

To compare two images and to establish pixelwise correspondences, cost functions are used to express the *similarity* between individual pixels or groups of pixels [5]. Probably the simplest measures are the *absolute* or *squared intensity difference*, which can be implemented efficiently on many platforms. However, it is important that the intensities of the pixels of the two images follow the same physical properties, and even small radiometric distortions (*e.g.* changes in illumination or exposure time) can lead to very different values. It is well known that this behavior can be limited by convolving the input images with the Sobel operator or by applying a mean filter to the input images [63]. Another filter is the Laplacian of Gaussian (LoG) which is known to also improve robustness to noise [61]. However, the mean and LoG filters operate with small window-based filter masks and therefore increased errors in regions near discontinuities are likely [63]. The *normalized cross correlation* and its variations are by nature more robust, but are also more time consuming to compute because for every matching template the mean and variance of image intensities must be computed. Some less demanding approximations exist, but come also with reduced robustness. The *Rank Transform* and the *Census Transform* of Zabih and Woodfill [169] are local non-parametric measures, which means that they only rely on the relative positions of the pixel intensities rather than their intensity values. Therefore, they are robust to many radiometric distortions and even to small amounts of image noise. Very universal is *mutual information* [1, 37, 59, 77] and is known to be very robust [63] even at the presence of complex and non-linear image transformations. The idea is basically to use statistics of corresponding image intensities and to use their mutual information to measure pixel-wise similarity. The integration is not always straight forward because the entropy of the input image must be known beforehand. Hirschmüller [59] proposed therefore an iterative technique which starts with a random initialization on down-scaled images.

Many stereo methods compute only integer valued disparities. To estimate *sub-pixel* values usually a function is fitted to the dissimilarity values around the minimum. It must be noted that the combination of the cost function and fitting function should be chosen carefully, due to the so-called *pixel-locking* effect, where interpolated values are often biased towards integers. Such estimation errors have been investigated thoroughly by Shimizu and Okutomi [123].

3.2. Binocular Stereo Matching

In the following, we mainly categorize stereo methods using their computation principle.

Local Methods. In traditional *local correlation-based methods*, like in the early work of Faugeras *et al.* [38], a brute force search is performed. In the spirit of a winner-takes-all decision, at every pixel the disparity which minimizes a specific dissimilarity function is used. The use of single pixels is in practice very sensitive to image noise and is highly ambiguous in regions with less or without texture. For this reason, usually a support region (*i.e.* a window) around the pixel of interest is utilized and matching is performed on multiple resolutions so that in uncertain regions matches from lower resolutions can be used.

These window-based methods can be implemented very efficiently [35, 38, 44, 61, 104, 144, 161]. This is mostly due to their simple and regular structure, which allows streamlined implementations on various processors. Also several techniques have been introduced to improve the quality of these methods [35, 61, 66, 73, 118, 146, 168] but robust ones are time consuming [66, 168]. The basic idea behind these improvements is that basically the size or the shape of the window is adapted to local image content (for example, adaptive windows of Kanade and Okutomi [73], Hirschmüller's shiftable or multiple windows [61], or Veksler's variable windows [146]), or that for every pixel of the window a *support weight* is computed from color information (for example, Yoon and Kweon's adaptive support weights [168] or the geodesic support weights of Hosni *et al.* [66]). Very famous is also the *left-right consistency check*, where two disparity maps are computed relative to each image: one using left to right matching and another one using right to left matching. Values which are inconsistent between the two disparity maps introduce holes which may be filled using interpolation or median filtering [61, 66].

To this end, window-based methods may achieve a very good quality, but always have an inherent conceptual problem in common: the use of a support region is only legitimate at fronto-parallel surfaces and must not cover depth discontinuities. In practice, these assumptions are often violated and result in blurred object boundaries or wrong depths on slanted or curved surfaces. One way to directly address this flaw is by using pixels as matching primitives which is done in global methods.

Global Methods. In global methods stereo matching is formulated as an energy minimization problem, where a function is computed (*i.e.* the disparity map) that minimizes a global energy functional, which usually uses pixel-wise dissimilarities. These combinatorial problems may then be solved efficiently using *Graph Cuts* [21, 77, 84] or *Loopy Belief Propagation* [40, 132, 159, 165]. To resolve ambiguity and spurious matches, regularization is modeled in the energy functional:

$$E(\mathcal{D}) = E_D(\mathcal{D}) + \lambda E_S(\mathcal{D}) \tag{3.1}$$

where the unary data-term E_D measures how well the disparity map $\mathcal{D}$ matches with the input images and E_S is a smoothness term that penalizes disparity variations between pairs of neighboring pixels. In general, such optimization problems are NP-hard in computational complexity. Both methods only *approximate* the global optimum, either by using the max-flow min-cut theorem (Graph Cuts), or by message passing on a Markov random field (Belief Propagation). Graph cuts is very powerful, but is only applicable to problems that can be reduced to the problem of finding the minimum cut in a graph [86].

Recently, the efficient fusion-move approach [90] has become popular and has only logarithmic complexity in the number of labels. It is able to handle continuous labelings within a discrete optimization framework. The basic idea is to *fuse* a *new* labeling with the current one using binary labels. However, for that the problem to be solved must be formulated as a binary pairwise fusion-energy and raises also the question of the generation of *new* labelings.

Belief propagation iteratively passes messages within a graph and was originally formulated as sum-product algorithm [88] with min-sum and max-product algorithms as variations [135] and it is known that for trees the solution is exact. While it is still not well understood under which conditions loopy belief propagation will converge on arbitrary graphs, it usually converges to a solution which is close to the optimum within a few iterations [101].

These global optimization approaches perform very well, also in textureless regions, but occlusions may result in wrong assignments. To improve in these situations, to enable high quality object boundaries and to capture small image details, image segmentation has been incorporated with excellent results [14, 16, 17, 79, 134, 166]. However, in practice, it is difficult to identify the optimal segmentation parameter set for a broad spectrum of image data. For stereo problems, graph cuts is in practice rather slow (several minutes) and belief propagation is more efficient (10-20 seconds for simple functionals on recent CPUs) but still not real-time on CPUs.

Cooperative Algorithms. Inspired by biology, these approaches [67, 97, 99, 118, 157, 177] formalize assumptions about continuity and uniqueness of the disparity map and iteratively diffuse disparity values through a locally connected graph and thus, explicitly handle occluded pixels. For that, they operate directly in the space of correspondences, the matching score volume[1]: each element (x_L, x_R, y) represents a pixel with a certain disparity. The matching score volume is initialized using a local similarity measure (for example, with local correlation). The cooperation between "support and inhibition" is implemented using an update routine which diffuses support among neighboring values for regularization by incorporating values along similar lines of sight: if a weight at (x_L, x_R, y) is large, the weights at $(\cdot, x_R, y)$ and $(x_L, \cdot, y)$ are "inhibited" to enforce uniqueness, and for continuity, weights at any other, non-inhibited points in the local neighborhood of (x_L, x_R, y) are excited. While [177] use a fixed window for the excitation, object boundaries may be blurred. This may be improved by incorporating an initial color segmentation [67, 157, 174].

Very good results are achievable in combination with segmentation, but due to the local update rules, the final result highly depends on a good initialization. In terms of running

[1] Also called the *disparity-space image* (DSI) [119].

time, it was reported to be twice as fast as graph cuts [119].

Non-Global Optimization Methods. To reduce the computational effort of the optimization problem, in *Dynamic Programming* [10, 12, 18, 103, 128, 145, 155] and *Scanline Optimization* [119, 98] only individual scanlines are processed. In dynamic programming, an optimal path through a cost-matrix (relating the scanlines of both images) is found and in scanline optimization only energy functionals of unary functions are minimized by handling occlusions explicitly. However, inter-scanline consistency is difficult to enforce and thus, often streaking effects can be observed. Further, most formulations of dynamic programming require an ordering constraint to be fulfilled [12] and violations may result in gross errors (which, for example, may be the case with thin foreground objects).

In some sense, Hirschmüller's *Semi-Global Matching* [59, 60] is a generalization of dynamic programming: global minimization is approximated by processing dynamic programming in multiple directions across the image. In practice, this approach keeps up with global methods, performs very well on real imagery and is also relatively efficient. In terms of running time, these methods require between 0.5 and 10 seconds on recent CPUs.

Layered Methods. Common to *plane-sweeping* methods is that the presence of specific scene planes is assumed. Collins presented in [29] an approach where a single plane partitioned into cells is swept through the volume of space along a line perpendicular to the plane. At each position of the plane along the sweeping path, the number of viewing rays that intersect each cell are tallied, and any cell with sufficient numbers of intersections is output as the likely location of a 3D scene point. The concept was then generalized to handle multiple scene planes, reflections and even translucency [46, 92, 137]. These methods were reported to be real-time on GPU hardware [46, 100], but the restriction to a priori scene planes is a limitation in general.

Recent works follow the principle of segmentation which allows the derivation of scene plane segments [14, 16, 134] and from that an estimation of depth and alpha matte information. The approaches are in principle different, but they have in common that they iteratively optimize a specific energy functional.

Phase-based Methods. Also phase-based methods have received some attention [24, 42, 43, 95, 115]. The basic idea is that a wavelet transformation is applied to the input images and then the disparity information can be determined from phase differences. These methods have a very high sub-pixel accuracy, but object separation is a weakness and the wavelet transformation may be expensive to compute.

Propagation-based Methods. Recently, some propagation-based methods have been proposed [26, 122, 156, 158]. The idea is to first identify so called *ground control points*, pixels whose disparity has been identified with high confidence, and to interpolate then into uncertain regions. This principle was first introduced in the context of dynamic programming by Bobick and Intille [18]. Not all of these formulations produce dense disparity maps, like the region-growing method of Čech and Šára [26], and may require an initial color segmentation for disambiguation, like in the work of Wei and Quan [158].

Segmentation-supported Approaches. Also referred to as *region-based*, technically, these methods [7, 11, 14, 16, 17, 28, 79, 134, 157, 165, 166, 174, 173] do not stand on their own. However, the idea to support matching by image segmentation has received such great success, that it is worth to be stated explicitly. The main idea is that first the input images are segmented and then, correspondences are estimated between regions rather than pixels. But the way how the segmentation is actually incorporated into the minimization framework is very specific.

GPU Implementations. Most of the presented methods can be implemented efficiently on dedicated hardware [25, 38, 52, 114, 155] and may even reach real-time performance. However, in our work we restrict ourselves to standard mobile CPUs and will concentrate on approaches that can be used for real-time applications with no additional dedicated hardware.

3.3. Multi-View Stereo Reconstruction

In the following we cover different aspects of multi-view stereo (MVS) methods.

Photo-Consistency. Also in the multi-view case, visual correspondences must be somehow measured to select depths which are consistent with the input views and this is often called *photo-consistency*. In principle, the same metrics as for binocular matching might be used, however it is much more difficult to control the camera behavior in the multi-view case, and therefore, traditionally a strong emphasis lies on robustness. To some extent, this usually comes at the price of reduced discriminability. In practice, the normalized cross correlation is quite robust and is often employed [53], but other measures may be used like the illumination-robust proposal of Hornung and Kobbel [65] or the work of Jin *et al*. which handles non-Lambertian reflectance [69]. However, Bonfort and Sturm [19] also presented a way to handle specular surfaces using geometric assumptions.

Scene Representation. There are several ways how different methods model the entire scene. Some methods [39, 69, 110, 111] represent the scene continuously using the so-called *level sets*, which is mainly a numerical technique for shape tracking. In 3D space, the surface of the scene is determined as a 2-dimensional implicit curve which is defined using a helper function. The set of all the positions at which the helper function is zero is the desired scene surface. In practice, such partial differential problems can be tackled numerically using the calculus of finite differences.

While level sets are more or less a continuous encoding, other works represent the surfaces of the scene using *polygonal meshes* [45, 57, 171]. In this case, a set of connected geometric primitives, like triangles, is used to construct a mesh of the scene. From this point of view it is a more discrete principle using locally linear patches. It is also quite practical, because the memory footprint is usually relatively low and such meshes are usually handled natively by graphics cards.

Very popular are *voxel grids*, which divide the entire 3-space into small equi-sized cells [19, 31, 58, 149]. Each cell may be described using different properties like occupancy, coloring or surface normal information and it is therefore a quite simple but also very powerful data structure. On the other hand, the memory requirement is extraordinarily huge for large outdoor applications.

Another branch of methods focuses on representing the scene using a set of depth maps [22, 47, 74, 75, 85, 100, 133, 173, 178] and are sometimes called *multi-depth-map* methods. Usually, for a set of input camera positions a set of depth maps is computed (or given). The positive points about this representation are that no data conversion is required and reasoning can be performed purely in the projective 2D domain.

Optimization Strategy. First to mention are local methods, for example [19, 29, 100, 102, 106], where depth values are obtained by purely localized computations either in the voxel grid [19], or in image space [100], or by simply fitting surfaces to triangulated image features [102]. Opposed to this are global approaches, for example [58, 82, 149, 173], which formulate, similar to the binocular case, the reconstruction problem as one of minimizing a global energy functional. In the discrete setting, a solution may be found using graph cuts [21, 84] or belief propagation [132]. The energy function may be defined on a volumetric MRF [58] or in 2D image space [173]. Like in binocular stereo, a smoothness prior is modeled as a pairwise energy potential in the image domain [162]. Alternatively a continuous convex relaxation scheme may be used to minimize a spatially continuous functional [82, 83]. In such works, the implicit surface of interest is represented by a characteristic function in voxel space. Usually non-convex energy functionals are reformulated as convex ones via relaxation, which allow for global optimization using gradient descent techniques. In practice, such problems are efficiently solved interatively using Gauss-Seidel, successive over-relaxation or multi-grid methods [82].

In many methods, an iterative scheme is employed. For example, in Zhang *et al*. [173] belief propagation is performed individually at every input depth map and the results are refined by repeating the optimization a few times for every depth map. Surface based methods start from an initial estimation and evolve the surfaces by, either numerically minimizing partial differential equations for level sets, or by adapting a polygonal mesh using momentums on the facets. In other works, like in Yang *et al*. [167], a voxel grid is updated iteratively by even allowing the creation and removal of voxels.

Occlusion Handling. Almost all multi-view stereo algorithms reason somehow about the visibility of individual pixels in different views. A widely incorporated idea in surface-based methods is to use an estimate of the scene structure in order to check for intersections of the viewing rays with the different surfaces [121]. In other works, occlusions and depths are maintained iteratively [74, 127], or during optimization [128]. Strecha *et al*. [128] even combined depth and occlusion estimation in a MRF framework to simultaneously reason about depth and visibility. Merrell *et al*. [100] handle occlusions directly, by suppressing depth values whose inter-view relationships are inconsistent.

3.4. Intelligent Vehicle Systems

In the following, we give a brief overview on state of the art vehicle sensors, driver assistance and in particular parking assistance systems.

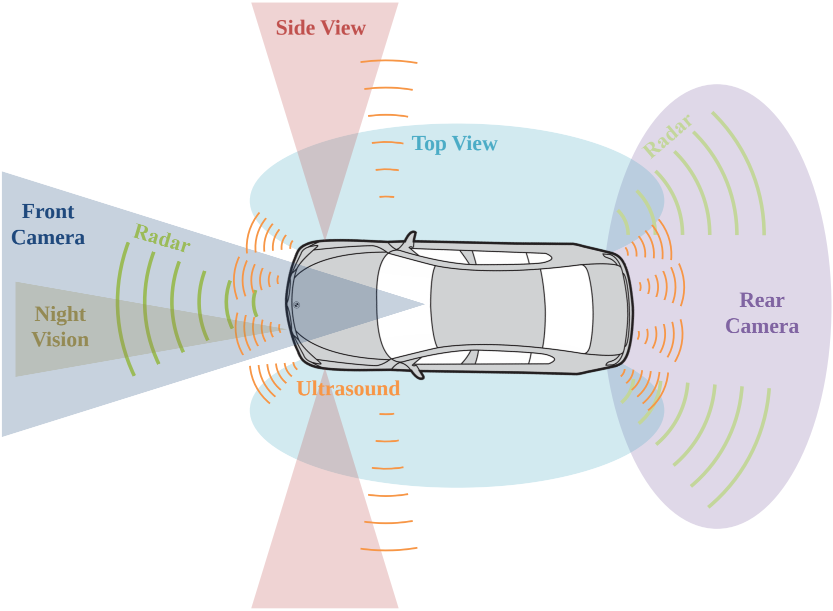


Figure 3.1. Currently available sensors on serial production vehicles include rear-, side- and front-looking cameras (Side-View and Top-View), ultrasonic sensors, RADAR, and night vision systems.

3.4.1. Vision Sensor Technologies

In the following we would like to introduce sensor systems that are suited for automotive applications. Please see Fig. 3.1 for an illustration of currently available sensors and their installation positions on serial production vehicles.

Ultrasonic Sensors emit sound waves with a high, inaudible frequency (above 20 kHz) and sense the impulse responses. Notably, the maximum range and measurement rate is influenced by the velocities of sound and the host vehicle and is thus a strong physical limitation. In practice, ultrasonic sensors have a very low angular resolution and weather conditions may have negative impacts (*e.g.* ice or water before the sensor, strong rain and wind).

RADAR is an abbreviation for Radio Detection and Ranging and is a technique to measure the distance and velocity of other traffic participants. Measurements are created by sending out modulated pulse-coded radiowave signals and by sensing incoming reflections. The distance is computed by measuring the elapsed time and the relative velocity from the phase difference (Doppler-effect). In practice, RADAR works well with many types of materials, reaches ranges up to several hundreds of meters and is less impacted by darkness or weather conditions (snow is known to affect the detection quality). However, the opening angle is limited and it is difficult to vertically and laterally associate objects with measurements. The small angular resolution makes it therefore very difficult to reason about the exact position and size of objects. In practice, to overcome these issues static objects are usually filtered out.

LIDAR is an abbreviation for Light Detection and Ranging and is a technique similar to

RADAR, where modulated laser beams are used instead of radiowaves. The reflected light of emitted laser beams (using a rotating mirror) is detected and the elapsed time is used to calculate distances. In practice, the reliable range of these sensors goes up to 50 meters due to limitations on the light intensity reflected back from objects with dark surfaces. These sensors provide a high accuracy, but vehicle integration is difficult due to the size and their performance is affected by weather conditions (*e.g.* rain and fog). Further, the mechanical design with a rotating unit is also a unfavorable property.

A **Time-of-Flight** camera is in some sense a special type of a LIDAR which does not require a rotating scanning unit. Instead, modulated light is emitted into the whole scene and the PMD (photonic mixing device) sensor measures the running time of the light waves for each pixel. Thus, 3D information is relatively rich. However, imager resolution is currently restricted (up to 200x200) and even low resolution devices are still quite expensive. Further, their usage in sun-lit scenarios is limited.

Structured Light systems use a camera to sense light patterns which were emitted using a projector. Based on the distance between the projector and the camera, depth may be estimated using triangulation. If infrared light is used the projections are even invisible for the human eye. Also in this case, the maximum range is limited by the maximally useable light intensity. In practice, the performance of these systems is deteriorated by sunlight, but in controlled environments a very high accuracy is achievable.

As already mentioned in chapter 2, **Monocular Cameras** do not actively measure distances. From a single image, distance information must be estimated using an object detector together with assumptions about the ground plane or the object size. The detectable object classes must be defined beforehand. In practice, such cameras are often integrated behind the windshield within the wiped area to solve problems arising from dirt and weather, but very adverse weather conditions still have an impact (*e.g.* heavy rain and dense fog). Further, the error of distance measurements is very large if the assumptions are violated. In this work, we use the principle of triangulation to determine depths from corresponding points identified in images acquired over time.

Stereo Cameras estimate distances directly by triangulation from corresponding image points and provide very rich depth data along with image intensities. In practice, the vehicle integration is more difficult due to a larger camera package and weather conditions also influence the performance negatively (*e.g.* rain and fog).

3.4.2. Categories of Driver Assistance Systems

Advanced driver assistance systems are electronic devices in automobiles to support the driver during the driving task. There are mainly two classes of customer functions, namely systems that focus either on *comfort* or on *safety*. In future, it is more likely that the gap between comfort and safety gets smaller, due to a higher functional integration and increased automation. These driver assistance systems are not directly related to this thesis, however, a short overview and some pointers to image processing may be interesting to some readers. A comprehensive overview can be found in [36].

Comfort Functions

These applications are designed to inform and to relieve the driver, and to support the sensation of comfort.

Adaptive Cruise Control (ACC) automatically adapts the velocity of the host vehicle based on the velocity of and the distance to the preceding one. To accomplish this, long-range RADAR or laser sensors are used, but also monocular and binocular camera-based solutions are known [126].

Using **High-Beam Automation (HBA)** the switching between low and high beam is performed automatically. Oncoming and preceding traffic is detected using a front camera. In the newest generation of these systems, the range to be illuminated is actively computed to enable a smooth transition between low and high beam, or to realize an intelligent combination of both, or to adapt the light distribution in a way such that other traffic participants are not glared.

Traffic Sign Recognition (TSR) [48] detects and classifies traffic signs (*e.g.* speed limits) using a front camera and shows the current state (*e.g.* current speed limit) to the driver.

Top-View or Surround View systems visualize a virtual bird's eye view containing the close environment around the host vehicle to give visual support during low speed or parking maneuvers. For that, up to four wide-angle cameras are required: one on the rear, two in the external mirrors and a front looking one.

Safety Functions

Safety functions are designed to avoid collisions or other dangerous situations, either passively (by information or warning) or actively (by braking or even steering).

Night Vision systems increase the vehicle driver's perception in darkness. While near infrared sensors (NIR) are relatively cheap, because standard imagers can be used, far infrared sensors (FIR) are more expensive. However, these also allow the observation of objects far beyond the reach of the vehicle's headlights. In practice, these sensors are used to detect pedestrians [96].

The **Side-View** applications are customer functions that use two side looking cameras integrated into the front bumper of the vehicle. These camera images are visualized to the driver and effectively support him in situations when the line of sight is obstructed, such as at the exit of car parks (see Fig. 7.2 for an example).

Lane Departure Warning (LDW) and **Lane Keeping Assistant (LKA)** detect lane markings to issue a warning when the vehicle begins to move out of its lane (LDW) [87] or to actively keep the vehicle within its lane (LKA). It is achieved either by putting a small momentum on the steering mechanism or by braking individual wheels. In many cases the detection of the markings is performed using a front camera, but other cameras on the vehicle may be used. In some situations, the usage of a laser scanner is also possible.

Lane Change Assistant (LCA) and **Blind Spot Detection (BSD)** use either short range RADAR sensors or cameras to detect other vehicles located to the driver's side and rear. When using RADAR fast approaching vehicles entering the blind spot can be detected.

For **Collision Mitigation by Braking (CMbB)**, **Forward Collision Warning (FCW)** [32] and **Preventive Pedestrian Protection (pPP)**, RADAR sensors, LIDAR sensors, cameras, or a combination of them is used to estimate the time-to-collision to vehicles, vulnerable road users (VRUs) [50, 113] or other dangers that lie ahead on the road. The presence of a hazardous obstacle may then result in a warning, assisted braking, automatic breaking or even steering.

Current products on the market for **Driver Drowsiness Detection (DDD)** learn driver patterns by monitoring the steering wheel, foot pedals and other control elements. This information, and possibly motions of the human [49] measured using an interior camera, is used to detect when the driver gets drowsy to issue a warning.

3.4.3. Parking Assistance

There are several different ways to perceive spatial information about the lateral space of the vehicle's environment. In the following we review some works in the context of parking assistance. Most popular are current commercial solutions based on ultrasonic sensors [125], but these are only passive systems with the purpose to inform the driver about distances to nearby objects. Recent products additionally utilize laterally mounted ultrasonic sensors to detect parallel parking slots into which the vehicle may be navigated semi-automatically [107, 109, 112]. However, depending on the required measurement range, these systems may not be able to detect cross parking slots and complex geometry makes correct interpretation of sensor signals difficult. Similar is a system developed by Schanz [117]: it uses a laterally mounted laser-scanner and, due to the relatively good measurement accuracy and range, the system is able to detect both parallel and cross parking slots. However, in practice laser-scanners are currently too expensive for mass production. Compared to these types of sensors, another benefit of our camera-based approach is that very rich depth information is acquired at low costs.

Systems that use cameras have also been investigated. Kämpchen *et al*. [72] detect parking lots using a forward looking stereo vision system: a point cloud generated from sparse stereo correspondences is analyzed to detect vehicles. Generic vehicle models are used to estimate their poses and parking slots are detected by analyzing the free space between two vehicles. However, due to the orientation and the limited field of view (FOV) of the stereo system, the detection of parallel and especially cross parking slots may be difficult.

The use of PMD cameras was evaluated by Scheunert *et al*. [120]: from the 3D data they build a local 2D grid where every cell is in one of four modes (unknown, ground, obstacle low and obstacle high), depending on the height of a point. From this grid, the curb is determined and a distance profile is computed. This information is then used to detect free spaces. However, they do not determine the envelope of the ground plane dynamically and since they assume the presence of the curb, they did not demonstrate the detection of cross parking slots. Furthermore, the PMD technology is still very expensive and thus not suitable for serial production.

It is also possible to detect parking slots using a camera and a projection of structured light [71]. However, legal restrictions in many countries render a worldwide commercialization of such solutions impossible. Other systems detect parking slots by extracting and interpreting ground markings [70, 164]. But the applicability and thus customer value is very limited, because the markings have to fulfill specific requirements on color, visibility and geometric properties.

There is also a wide range of recent methods that use the principle of motion-stereo [41, 124, 130, 131, 147, 151, 152]. However, these works address only a feature-based strategy and no one utilized dense disparity maps. The basic idea is to calculate characteristic features in subsequent images. Over time, this relatively small number of points is tracked and then a 3D reconstruction is analyzed to find parking slots. These approaches perform well in friendly conditions, *i.e.* as long as enough strong and distinctive features can be derived from the images. However, challenging are both lowly textured objects, which lead to very sparse point clouds, or also complex textures like foliage, where high ambiguity during feature matching introduces wrong distance measurements. Moreover, features are not necessarily located at the boundaries of objects. Thus the size of objects and free space might be wrongly calculated. In these situations, the accuracy and reliability of the determination of free parking areas varied in an inacceptable way.

In this thesis, we present a powerful approach that is based on our *dense* motion-stereo pipeline, where at every frame a dense disparity map is computed. This results in important advantages, namely a very high detection rate of obstacles, a high measurement accuracy, a nearly drift free environment model and the ability to display a multitude of different customer functions.

4. Efficient Stereo Vision

Natura non facit saltus.[1] (Latin for *Nature does not make jumps*)

Dense stereo matching in real-time is important for many fields of applications that require an on-line dense three-dimensional representation of the observed scene [38, 144]. Also the processing of large images or long image sequences needs computationally efficient algorithms [59]. However, for automotive and other mobile applications, the hardware requirements must be as low as possible. This usually restricts the available processing power as well as the number of cameras.

Typical commercial implementations of such systems use one or two cameras together with a computationally feasible algorithm to compute depth information [38, 144]. If two cameras are used, local methods based on correlation can be implemented very efficiently [61]. On dedicated hardware, methods such as local correlation, dynamic programming, semi-global matching or even belief propagation can be implemented for real-time application [25, 38, 114, 155]. However, optimized local methods are among the fastest ways in order to perform dense matching solely on general purpose CPUs without special hardware. In this case, decisions must be made upon the values of some parameters, particularly for the maximum disparity.

We will later demonstrate that the choice for a fixed maximum disparity influences the quality of the disparity map: setting it too high favors false matches and setting it too low will result in gross errors at close objects. In some cases, the choice of the maximum disparity is complicated. Especially at motion-stereo [93], when the camera moves at a variable velocity, a choice for a fixed maximum disparity either restricts the practical applicability or results in an increased number of errors and an inefficient use of processing power (if the maximum disparity is set too high).

Another highly important aspect for our work is the stereo calibration of the cameras. In practice, an accurate rectification is of eminent importance when applying dense stereo methods to pairs of images. The reason for this lies in practical considerations to maximize the efficiency of stereo methods, where rectification usually transforms the epipolar geometry of both images in a way such that epipolar lines are horizontal and *matched up*. This means, that after rectification the y-coordinate of corresponding image pixels is always constant and that the search-space for stereo-processing is heavily constrained. Therefore, an inaccurate rectification directly affects stereo matching. It is known that even slight

[1] This rather famous principle dates back at least until ARISTOTELE and refers to the intuition that things and processes evolve continuously in nature and not sudden. But having discontinuities of disparity maps in mind, this axiom obviously does not hold without restrictions in the projective world. However, it may be remedied if the scene is decomposed into individual scene surfaces and this is an important assumption in many stereo methods and also the driving idea for this chapter; scene surfaces do not make jumps but evolve smoothly in disparity maps.

inaccuracies of the epipolar geometry may result in significant degradation of the stereo matching performance.

In motion-stereo applications, the rectification of two consecutive camera frames must be estimated from available vehicle sensors, for example, from odometry using wheels and the levels of the dampers. However, practical experience shows that the accuracy of both odometry and damper-levels does not suffice for an accurate rectification, due to slippery or uneven ground.

Furthermore, in future, vehicles may be equipped with binocular front cameras, and this would imply the use of stereo algorithms in vehicles. For long-term installations of stereo rigs in vehicles, an adaption to decalibration issues is preferable, since there is very limited experience with vehicular stereo rigs over very long periods of time (*e.g.* 10 years). In these cases, the stability of the mounting concept (with respect to deterioration or deformation) and thermal influences on material might have a huge impact on the accuracy of rectification. Moreover, camera calibration is costly, time-consuming and critical for the quality of serial production vehicles. From this point of view, methods are preferable that do not require an exhaustive calibration procedure, but work well with rough, approximate settings, that might, for example, be computed from CAD models.

In this thesis we propose a novel method for efficient dense stereo matching without the need of the choice of the maximum disparity. We further introduce generalizations which make the matching robust to inaccurately rectified images. Moreover, we present a fast post-processing technique that is based on energy minimization and is suited to refine the obtained results.

In the rest of this chapter we first give an overview on related efficient stereo methods. After introducing the traditional way of local disparity computation in section 4.2, we formalize the idea of iterative region tracing for stereo in section 4.3 and introduce a novel disparity computation algorithm in section 4.4. In section 4.5 we present efficient generalizations of our ideas to address an inaccurate rectification. To increase the quality of disparity maps from window-based stereo methods, we present an approach for disparity refinement in section 4.6. Finally, we present an extensive evaluation in section 4.7 using well known stereo datasets with ground truth, apply the methods to real world imagery and discuss our proposals in section 4.8.

4.1. Related Methods

4.1.1. Efficient Stereo

Regarding highly efficient stereo matching there is a large number of works [35, 38, 46, 51, 61, 114, 144, 146, 155]. Most famous are local correlation methods whose cost functions are usually separable, such that an incremental computation is possible [35, 61, 144, 146]. Dynamic programming [10, 12] has also received some attention due to its efficient algorithmic structure, but the well known streaking effects are very impractical. The work of Bleyer and Gelautz [12] addresses this issue, but also comes at a much higher processing overhead. In other works dedicated hardware is used to achieve real-time performance [38, 51, 114, 155]. For example, by using an FPGA Gehrig *et al.* [51] successfully ported semi-global matching. A different principle is by making assumptions about the scene geometry. One example is the work of Gallup *et al*. [46], who use a priori scene surfaces for fast stereo reconstruction. However, in our application we are faced with complex 3D geometry (like vehicles, bushes, poles, stones, pedestrians *etc*.) where arbitrary surface orientations exist.

There is also the branch of region-based methods which extract ground control points in the first place [26, 158] and which diffuse matches into uncertain regions. From some perspective, these methods may also be run without knowledge about the maximum disparity; however, [26] does not produce dense disparity maps and [158] is far from real-time.

In this chapter we focus on efficient stereo matching on a standard CPU without any dedicated hardware or graphics unit. Different to most other works, we use complex robust matching cost functions and no a priori knowledge about the maximum disparity is required for our method. More precisely, our contribution is a novel disparity computation algorithm which replaces the famous winner-takes-all (WTA) approach.

4.1.2. Decalibration

Binocular stereo matching is a well explored direction, but to our knowledge, all of the methods presented in section 3.2 require an accurate rectification of the images. However, relaxing the epipolar constraint immediately leads to optical flow methods [4, 6, 27, 64]. While real-time GPU implementations exist [27], most of the approaches that compute a dense flow field on the CPU are far from real-time and address a different conceptual problem; many methods are usually designed to recover small displacements and do not directly address the problem of "small epipolar deviations", which happens if a pair of images is not rectified well. The method derived in section 4.5.1 directly addresses the problem of large horizontal and small vertical displacements. In particular, the methods in section 4.5 are efficient formulations using block matching and are therefore different from differential optical flow methods like [27, 64].

Multi-view reconstruction algorithms, like [3, 29, 81, 100, 121, 170, 173], also rely on some knowledge about camera positions, and usually perform the calibration by estimating the epipolar geometry from a sparse set of feature points [55, 105, 136] using epipolar or trilinear constraints. However, the extraction of feature points, their matching and the projective warping of the pair of images for rectification is also relatively time-consuming and only works well if enough correctly matched feature points are available. In other applications, the idea of *online calibration* of stereo rigs is applied [33]. But also in these works, usually a set of sparse correspondences is required to determine or to refine calibration parameters. In section 4.5, we focus on determining dense correspondences directly.

4.2. Traditional Disparity Computation

In traditional correlation-based methods, to each pixel $\mathbf{x} = (x, y)^T$, the disparity associated with the minimum dissimilarity is assigned:

$$\mathcal{D}(\mathbf{x}) = \operatorname{argmin}_{d_{\min} \leq d \leq d_{\max}} \mathcal{C}_A(\mathbf{x}, d) \tag{4.1}$$

where $\mathcal{D}$ is the disparity map, $d_{\min}$ is the minimally possible disparity value, $d_{\max}$ is the maximum disparity (MD) and $\mathcal{C}_A(\mathbf{x}, d)$ is the dissimilarity function where lower values indicate higher similarity. Without loss of generality we assume for the rest of this chapter that $d_{\min} = 0$ because it will make formulas easier to read. We consider $\mathcal{C}_A$ as aggregated matching cost by summing pixelwise costs over a rectangular support region around the pixel of interest:

$$\mathcal{C}_A(\mathbf{x}, d) = \sum_{u=-w}^{w} \sum_{v=-w}^{w} \mathcal{C}_M\big(\mathbf{x} + (u, v)^T, d\big) \tag{4.2}$$

where w parameterizes the window size and $\mathcal{C}_M(x, y, d)$ is the pixelwise matching cost (for example the squared intensity difference or the absolute intensity difference) which measures the dissimilarity between the left image pixel $\mathcal{I}_L(x, y)$ and the right image pixel $\mathcal{I}_R(x - d, y)$. These formulations of $\mathcal{C}_A$ are known as SAD or SSD; in many applications robust cost functions are preferable.

4.2.1. Robust Similarity Measures

In practice, stereo matching and in particular motion-stereo becomes difficult under sudden exposure or illumination changes (*e.g.* in garages), in low-light scenarios, during different weather conditions (rain, snow, *etc.*) or due to glare light effects. We chose to use the following robust cost functions.

Normalized Cross-Correlation (NCC).

$$\mathcal{C}_{NCC}(\mathbf{x}, d) = \frac{\sum_{u=-w}^{w} \sum_{v=-w}^{w} \Big(\mathcal{I}_L\big(\mathbf{x} + (u, v)^T\big) - \bar{\mathcal{I}}_L(\mathbf{x})\Big) \cdot \Big(\mathcal{I}_R\big(\mathbf{x} + (u - d, v)^T\big) - \bar{\mathcal{I}}_R(\mathbf{x})\Big)}{-\sqrt{\sigma_L(\mathbf{x})\sigma_R(\mathbf{x})}} \tag{4.3}$$

where w parameterizes the window size, $\mathcal{I}_L$ and $\mathcal{I}_R$ are the left and right images and $\bar{\mathcal{I}}_L$, $\bar{\mathcal{I}}_R$ are mean pixel intensities in the correlation window computed using

$$\bar{\mathcal{I}}_L(\mathbf{x}) = \frac{1}{(w+1)^2} \sum_{u=-w}^{w} \sum_{v=-w}^{w} \mathcal{I}_L\big(\mathbf{x} + (u, v)^T\big) \tag{4.4}$$

and the variance of the image intensities in the correlation window is given by

$$\sigma_L(\mathbf{x}) = \frac{1}{(w+1)^2} \sum_{u=-w}^{w} \sum_{v=-w}^{w} \Big(\mathcal{I}_L\big(\mathbf{x} + (u, v)^T\big) - \bar{\mathcal{I}}_L(\mathbf{x})\Big)^2 \tag{4.5}$$

The terms $\bar{\mathcal{I}}_R$ and σ_R are computed analogously. Then the aggregated NCC matching costs are then given by $\mathcal{C}_A(\mathbf{x}, d) = \mathcal{C}_{NCC}(\mathbf{x}, d)$.

Census Transform (Census). The Census filter computes a bit string for every image pixel. Every bit encodes a specific pixel of the local window centered around a pixel of interest. The bit is set to one if the pixel has a lower intensity than the pixel of interest. Later, the pixel-wise matching cost is defined as the Hamming distance of pairwise bit strings. In practice, we sum these Hamming distances over a small support region.

4.3. Disparity Computation by Region Tracing

The main idea is that the disparity values of adjacent pixels are often similar which happens primarily in scenarios where the scene is composed of a few smooth surfaces. Hence, if we assume that the disparity of a small image region is known, then there must be a way to infer the disparities of neighboring regions. For example, if we know the disparities for a small part of a single scanline, then our goal is to infer the disparities of the adjacent scanlines by assuming that their disparities are similar. There are three practical motivations for our approach presented here: first, we are interested in saving processing time; second, there is no optimal choice for the maximum disparity value for the whole image; third, in our application, the scene is composed of a few smooth surfaces.

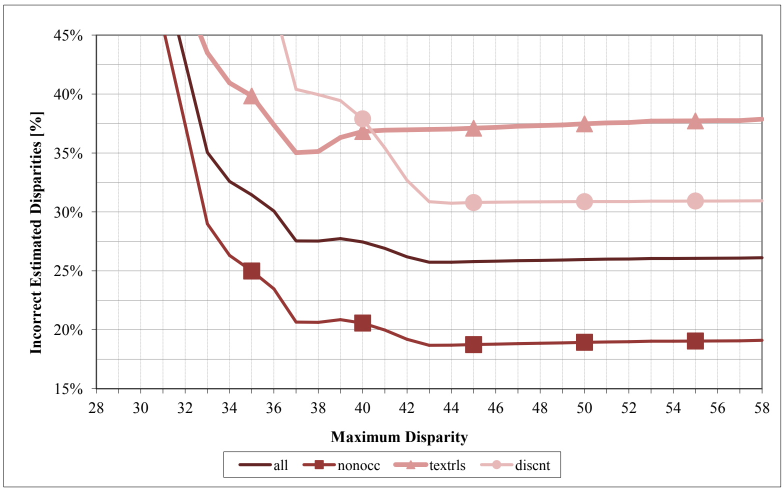


Figure 4.1. Curves plotting the percentage of incorrect disparities estimated with classical methods on the different image regions of the *Teddy* dataset as a function of the maximum disparity value used. The curves show that the optimal maximum disparity is not unique for different regions in this particular example. When considering all pixels (all), the optimal value of the maximum disparity is equal to 43. The same value is optimal when considering non-occluded regions (nonocc) and regions close to discontinuities (discnt). However, this value is not optimal when considering textureless regions (textrls) where the value 37 provides a lower percentage of wrong disparities.

Fig. 4.1 shows the relationship between the MD and matching errors for the standard dataset *Teddy*. Setting the MD too high introduces false matches and setting it too low will produce gross errors at close objects. But also a seemingly ideal MD value will not result in the best possible result. The figure depicts that there is no optimal fixed MD setting that minimizes all individual errors simultaneously. For example, if the value 37 is used the errors of textureless regions are minimized but this value will cause errors in the other regions. The optimum may be obtained if the MD is set to 37 in textureless regions and to 43 in the rest of the image. Intuitively, image regions that contain background structures reach the optimal disparity earlier than regions with foreground structures. Hence, investigating more disparities requires more processing time and increases the possibility of false matches especially in regions with weak texture. To avoid these drawbacks the MD should be variable and as close as possible to the true disparity.

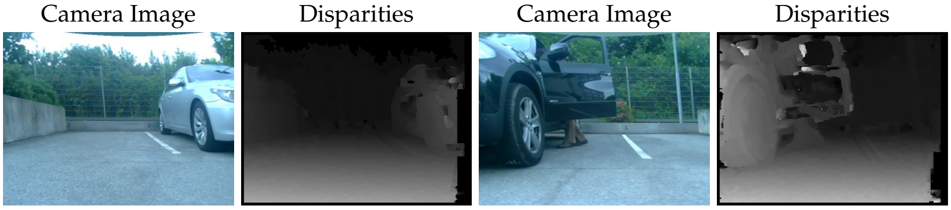


Figure 4.2. Exemplary disparity maps of our application.

In general, the relationship between the MD and matching errors depends on the structure of the scene, the image texture and the chosen image partitioning. It can be assumed that for very well and uniquely textured objects less false matches will occur at higher disparities. Also, if the weakly textured objects reside at the foreground of the scene, the tendency is that the number of false matches will decrease at higher disparities. It is very difficult to formalize the dependency between false matches, scene texture and the MD. However, the intuitive motivation remains that the likelihood of a wrong disparity increases with the size of the search region, because the discriminability decreases. Moreover, and this is depicted in Fig. 4.2, there is also the fact that some parts of the image exhibit mostly background structures (upper regions) and other parts of image foreground objects (middle and lower regions).

Fig. 4.2 illustrates exemplary disparity maps of our application. Obviously, the disparities are very inhomogeneous across the image and the scene is mainly composed of several smooth scene surfaces. This has the following implications: first, a global MD will result in waste of processing resources (the MD is large in the lower image region, but a smaller setting suffices for the upper parts). Second, disparity values of adjacent pixels are very likely to be similar (*i.e.* the difference is smaller than 1).

Our approach is to iteratively increase the MD $d_{\max}$ and to modify the disparity map incrementally. After an initialization phase, we increase the MD $d_{\max}^{n+1} = d_{\max}^{n} + 1$ until the disparity map needs no further modifications. In this way, we visit every disparity level (from back- to foreground) and add new information to the disparity map. We will show how to exploit information of one disparity level to compute the next disparity level.

Our implementation does not explicitly maintain the history of disparity maps evolving from the iterations (*i.e.* $\mathcal{D}^0, \mathcal{D}^1, \ldots$). But, for the sake of clarity, it makes sense to introduce such notion because we want to argue about the step from $\mathcal{D}^n$ to $\mathcal{D}^{n+1}$ by incremental modifications. At this point it is also worth stating that $\mathcal{D}^n(\mathbf{x}) \leq d_{\max}^n$, $\forall \mathbf{x}$.

Critical Region Images. In every iteration, our method thresholds the current disparity map $\mathcal{D}^n$ to compute a so called *critical region image* χ^n. Fig. 4.3 is an example. We define the indicator function χ^n as follows:

$$\chi^n(\mathbf{x}) := \begin{cases} 1 & \mathcal{D}^n(\mathbf{x}) = d_{\max}^n \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

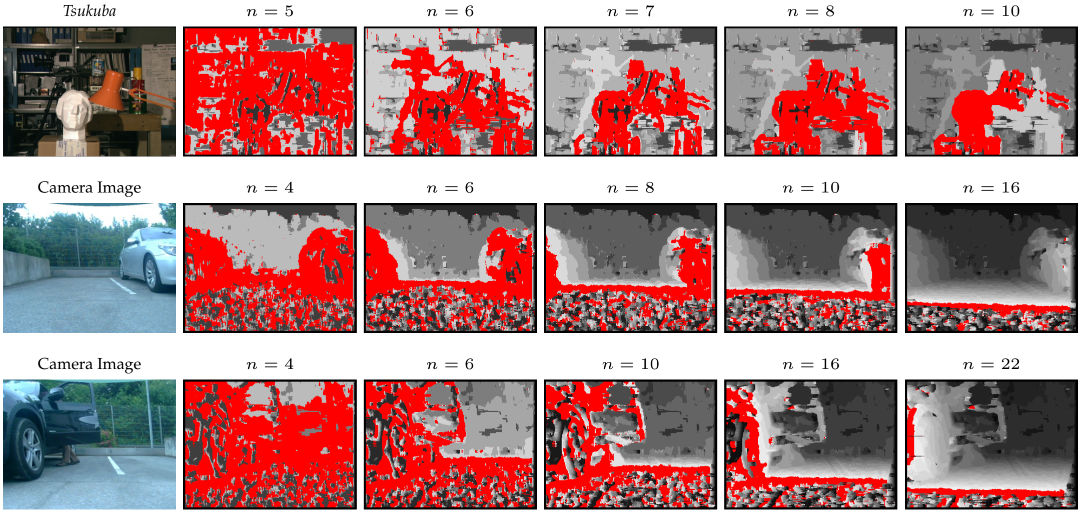


Figure 4.3. Critical region images superimposed with intermediate disparity maps: our approach applied to the *Tsukuba* dataset (top row) and images from our application (last two rows). The first column shows the camera image; the other columns show the critical regions χ^n overlaid in red over the intermediate disparity map $\mathcal{D}^n$ (grayscale).

Given the threshold parameter, a thresholded disparity map χ^{n+1} represents partial information about the disparity map $\mathcal{D}^{n+1}$, because

$$\mathcal{D}^{n+1}(\mathbf{x}) = d_{\max}^{n+1} \iff \chi^{n+1}(\mathbf{x}) = 1 \tag{4.7}$$

In the rest of the image, where $\chi^{n+1}(\mathbf{x}) = 0$, we only know that

$$\mathcal{D}^{n+1}(\mathbf{x}) < d_{\max}^{n+1} \tag{4.8}$$

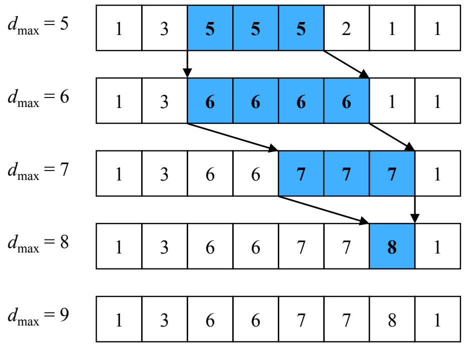
In these areas, we assume that $\mathcal{D}^{n+1} \equiv \mathcal{D}^n$ which results in the following recursive mapping:

$$\mathcal{D}^{n+1}(\mathbf{x}) := \begin{cases} d_{\max}^{n+1} & \chi^{n+1}(\mathbf{x}) = 1 \\ \mathcal{D}^n(\mathbf{x}) & \text{otherwise} \end{cases} \tag{4.9}$$

Hence, the final disparity map, say $\mathcal{D}^N$, may be built up solely from the sequence of critical region images $(\chi^0, \chi^1, \dots \chi^N)$:

$$\mathcal{D}^N(\mathbf{x}) = \max\Big(\big\{d_{\max}^k \big| 1 \leq k \leq N,\ \chi^k(\mathbf{x}) = 1\big\} \cup \big\{\mathcal{D}^0(\mathbf{x})\big\}\Big) \tag{4.10}$$

General Idea. We would like to determine the final disparity map using (4.10), but we cannot use (4.6) to determine χ^k (with $k \geq 1$, because we would have to know every $\mathcal{D}^k$ in advance, including $\mathcal{D}^N$). We rather start with an initial disparity map $\mathcal{D}^0$, compute χ^0, and use χ^0 to *estimate* χ^1, use χ^1 to estimate χ^2 and so forth. We stop this process at an unknown iteration N if $\chi^N(\mathbf{x}) = 0, \forall \mathbf{x}$. As it is not memory efficient to keep all χ^k's, we



$d_{\max} = 5$	1	3	**5**	**5**	**5**	2	1	1
$d_{\max} = 6$	1	3	**6**	**6**	**6**	**6**	1	1
$d_{\max} = 7$	1	3	6	6	**7**	**7**	**7**	1
$d_{\max} = 8$	1	3	6	6	7	7	**8**	1
$d_{\max} = 9$	1	3	6	6	7	7	8	1

Figure 4.4. A possible evolution of a single critical region as the maximum disparity increases. The boxes represent eight fixed pixels with their disparity value in it. The critical region is shaded in blue. The arrows denote corresponding region borders.

directly modify the estimate of the final disparity map using (4.7). In the following, we focus on the main goal of determining χ^{n+1} from χ^n.

Fig. 4.3 exemplifies the key observation regarding the evolution of the critical region images as the MD increases: they do not change arbitrarily but evolve rather slowly and smoothly, because usually the scene structure is composed of a few smooth surfaces.

Critical Regions. We process every scanline separately and denote the current scanline by y_0. We already introduced *critical region images* in (4.6). Now, we define:

Definition 4.1 (Critical Region). *A* critical region *of a critical region image χ^n is a non-empty interval $I = [a,\, b]$ that fulfills*

$$\forall x \in I : \; \chi^n(x, y_0) = 1 \tag{4.11}$$

and is maximal

$$\chi^n(a-1, y_0) = 0 \quad \wedge \quad \chi^n(b+1, y_0) = 0 \tag{4.12}$$

To avoid undefined conditions, we also assume $\chi^n(x, y_0) = 0$ if $(x, y_0)^T \notin \mathcal{I}$.

Using this definition, a scanline of a critical region image may be decomposed into several distinct critical regions. Please note that this is different to existing region-based methods [28, 11, 7, 157], where the input images are segmented using color or intensity information. We also do not use regions as matching primitives and we do not use them for estimating planar surface models – in our sense, a critical region is a small region of interest within the disparity map.

These critical regions play a key role in determining χ^{n+1} from χ^n: first, χ^n is decomposed into critical regions. Then, we show how the critical regions develop, if the MD is incremented. This development can be used to efficiently determine χ^{n+1}.

Relating Critical Regions. The observation of the evolution of critical region images led us to the assumption that χ^n and χ^{n+1} are in some sense similar. Let $I^n = [a, b]$ be a critical region of χ^n with the left and right boundaries a and b. We assume that there exists a critical region $I^{n+1} = [a', b']$ in χ^{n+1} which is "close" to I^n.

Fig. 4.4 shows how the evolution of a single critical region might look. The boundaries of the regions undergo slight translations. During the whole process, already visited pixels may be modified several times.

We will turn to some special cases later, but for the moment, we assume that for every I^n exists exactly one corresponding I^{n+1}, such that their intersection is not empty:

$$I^n \cap I^{n+1} \neq \emptyset \tag{4.13}$$

If (4.13) is fulfilled, then I^{n+1} can be determined by a simple local search. Let $f(t) := \chi^{n+1}(t, y_0)$. We define this search routine

$$\sigma(a) := \begin{cases} \max\{t | t < a, f(t) = 0\} + 1 & f(a) = 1 \\ \min\{t | t > a, f(t) = 1\} & \text{otherwise} \end{cases} \tag{4.14}$$

and show its validity:

Lemma 4.2. *Let $I^n = [a, b]$ and $I^{n+1} = [a', b']$ be two corresponding critical regions with $I^n \cap I^{n+1} \neq \emptyset$. Then, using Eq. (4.14) we have $\sigma(a) = a'$.*

Proof. There are only two cases. The first case is $a' \leq a$. Then we have $b' \geq a$ and $f(a) = \chi^{n+1}(a, y_0) = 1$. Using (4.14) we obtain

$$\begin{aligned} \sigma(a) &= \max\{t | t < a, f(t) = 0\} + 1 && (4.15)\\ &= \max\{t | t < a, t \notin I^{n+1}\} + 1 && (4.16)\\ &= \max\{t | t < a, t < a'\} + 1 && (4.17)\\ &= a' - 1 + 1 = a' && (4.18) \end{aligned}$$

The second case is $a' > a$. Then we have $f(a) = \chi^{n+1}(a, y_0) = 0$. Using (4.14) we get

$$\begin{aligned} \sigma(a) &= \min\{t | t > a, f(t) = 1\} && (4.19)\\ &= \min\{t | t > a, t \in I^{n+1}\} = a' && (4.20) \end{aligned}$$

Together: $\sigma(a) = a'$. □

The right border b' can be determined similarly. The only missing point is how the function f in (4.14) is computed: in accordance to (4.6), we get

$$\begin{aligned} f(t) &= \chi^{n+1}(t, y_0) && (4.21)\\ &= \begin{cases} 1 & \mathcal{C}_A\big((t, y_0)^T, d_{\max}^{n+1}\big) < \mathcal{C}_A\big((t, y_0)^T, \mathcal{D}(t, y_0)\big) \\ 0 & \text{otherwise} \end{cases} && (4.22) \end{aligned}$$

where $\mathcal{C}_A((x, y)^T, d)$ is the dissimilarity measure.

Summary. The idea of this method is that we process the individual disparity levels one by one (in ascending order, so the background is processed first). We accomplish this by stepwise increments of the MD. At every disparity level d^n_{max}, we extract all pixels whose disparity is maximal and call them *critical* (4.6). It turns out that there are mainly blocks of such pixels. We call these blocks *critical regions* because the disparity of those pixels is likely to change in the next disparity level. In addition it appears that the critical regions of two adjacent disparity levels are likely to be similar (4.13) and that the critical regions of the two disparity levels can be related to each other efficiently (4.14). Hence, (4.14) indirectly relates two adjacent disparity levels. The efficiency lies in the search routine (4.14): it allows the estimation of the next disparity level with less dissimilarity computations (only the boundaries of the critical regions need to be found).

Regarding the evolution of critical regions, some more cases need to be discussed: if condition (4.13) is violated, then the region might have disappeared or the boundaries have undergone a large translation. These cases can be treated in the implementation by limiting the search-range in (4.14), for example by requiring $a' \leq b$ or $|a' - a| \leq \Theta_\sigma$. We obtained good performance if $\Theta_\sigma = d^n_{\mathrm{max}} + 10$. Emerging regions are possible (for example, thin foreground objects which are not connected to the ground), but in our experiments we discovered that the other way round is more likely: in the beginning the number of pixels being part of a critical region is high. Then, the critical regions disappear gradually or split up (so this number decreases more and more). Therefore, it is advisable to cover the case if regions split up: Once I^{n+1} has been recovered we split I^{n+1} if necessary by ensuring that (4.11) is fulfilled. Finally, we point out that it is not important to establish *correct* correspondences between the critical regions of χ^n and χ^{n+1} – we are only interested in determining χ^{n+1} efficiently.

Discussion. Even though the presented algorithm significantly reduces the number of required correlations[2], the performance compared to local correlation is actually lower. We found out that this is due to the nature of caching in CPUs. Also [104] gave remarks on effectively using caches, specifically regarding the ordering of the loops. A structure like above is sub-optimal for cache utilization, since for every disparity value, whole scanlines must be evaluated. Following the lines of [104], the innermost loop should run over disparity values. This motivated us to formulate the minimization differently and resulted in the method presented in section 4.4.

Concluding Remarks. The most important observation on critical regions is that they are predominantly stationary. This has the consequence that a pixel is highly likely to be repeatedly part of a critical region during several consecutive iterations. Suppose there are critical regions $I^n, I^{n+1}, \ldots, I^{n+m}$ fulfilling (4.13):

$$I^k \cap I^{k+1} \neq \emptyset \qquad k = n, n+1, \ldots, n+m-1 \tag{4.23}$$

[2] For the *Teddy* dataset: the number of the required correlations necessary to obtain the disparity map was reduced from 7,621,950 (traditional method with $d_{\max} = 43$) to 3,518,829.

Then, if m is chosen appropriately we have

$$S := \bigcap_{k=n}^{n+m} I^k \neq \emptyset \tag{4.24}$$

Hence, there are pixels $(s,y)^T$, $s \in S$ which are critical during m consecutive iterations:

$$\mathcal{C}_A\big((s,y)^T, d_{max}^n\big) > \ldots > \mathcal{C}_A\big((s,y)^T, d_{max}^{n+m}\big) \tag{4.25}$$

We aim at determining the *intermediate optimal* disparity d_{max}^{n+m} *directly* for the pixels $(s,y)^T$, instead of requiring m iterations. For this, we minimize the dissimilarity using

$$\mathcal{D}(\mathbf{x}) \mapsto \min\big\{d \,|\, d \in \mathbb{N}_0,\, d \geq \mathcal{D}(\mathbf{x}),\, \mathcal{C}_A(\mathbf{x}, d+1) >= \mathcal{C}_A(\mathbf{x}, d)\big\} \tag{4.26}$$

This step resembles in some sense the minimization property of the region tracing. The second ingredient is to account for the deformations of critical regions. Here, it is mainly given by the local search routine (4.14) and our idea is to introduce a propagation-step that reasons on neighboring disparities. Thus, the basic idea is to handle the pixels individually and to iteratively optimize the disparity values.

4.4. Efficient Disparity Computation without Maximum Disparity

Also in this section we focus on an iterative algorithm that stops at the right disparity value, instead of determining disparities by a brute-force search within the whole disparity domain. Based on the ideas of the previous section, we perform two operations at each pixel: a minimization followed by a propagation-step. The minimization basically follows a line search strategy and allows us to find the "next" local minimum of the matching cost function. Since matching costs usually have many local minima we introduce a propagation-step in order to find further, better local minima using disparity values of neighboring pixels. We embedded these steps into a hierarchical setup, which will also be described in detail.

Minimization. For every pixel $\mathbf{x}$, we determine an *intermediate optimal* disparity:

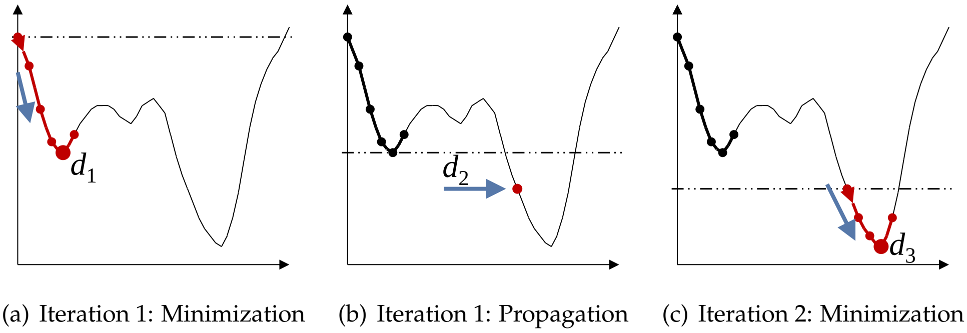
$$d_{n+1} = \operatorname{argmin}_{d \in \{d_n, d_n+1\}} \mathcal{C}_A(\mathbf{x}, d) \tag{4.27}$$

We use $d_0 = \mathcal{D}(\mathbf{x})$ and if $d_{n+1} = d_n$ the iteration is stopped and the disparity map is updated: $\mathcal{D}(\mathbf{x}) \mapsto d_n$. Using this formula, we "step down the hill" and thus search for the next optimal disparity by iteratively incrementing the current disparity.

Propagation. As the minimization-step will only return the first (and possibly not optimal) minimum of the dissimilarity function (see Fig. 4.5(a)), we additionally propagate the disparities of adjacent pixels:

$$\mathcal{D}(\mathbf{x}) \mapsto \operatorname{argmin}_{d \in N(\mathbf{x})} \mathcal{C}_A(\mathbf{x}, d) \tag{4.28}$$

with the neighboring disparities $N(\mathbf{x})$. $N(\mathbf{x})$ should at least contain the disparities of the left and right neighbors and must fulfill $\mathcal{D}(\mathbf{x}) \in N(\mathbf{x})$. Disparity values will be propagated



(a) Iteration 1: Minimization (b) Iteration 1: Propagation (c) Iteration 2: Minimization

Figure 4.5. An example for a dissimilarity function (vertical axis) over disparity (horizontal axis). (a) Only the first minimum d_1 of the function is found by the first minimization. (b) Then, the propagation selects an adjacent disparity d_2, because it leads to a lower dissimilarity. (c) The second minimization finds the optimal minimum d_3.

through their local neighborhood in this step. The idea is that the global minimum of a single dissimilarity function can be found by alternating the minimization- and propagation-steps. The propagation jumps to a disparity value with a dissimilarity smaller than the previous local minimum and the minimization navigates to the next local minimum which is nearby to the propagated value. In practice, only a few iterations are required (2-4).

Hierarchical Setup. A hierarchical implementation stabilizes execution times, because disparities can be found with an almost constant effort. However, it must be noted that a hierarchical setup may obey the drawback of loosing thin foreground objects and that errors at low resolutions may have severe impacts at higher resolutions. To reduce artifacts from false matches at low resolutions, we apply the hierarchical approach only to the horizontal dimension. Basically, at every resolution, the depth-map is initialized with the scaled up disparities from the previous resolution (in the beginning, the depth-map is initialized with zeros). For our algorithm, it is important to scale up disparities properly. Since the search direction of the minimization-step is in positive disparity orientation, it is beneficial to *underestimate* the actual disparity. Let σ be the scale factor, for example $\sigma = 2$. For scaling up, we use the following formula:

$$\mathcal{D}'(\sigma x, y) = \sigma \mathcal{D}(x, y) - \sigma + 1 \tag{4.29}$$

$$\mathcal{D}'(\sigma x + k, y) = \sigma \min(\mathcal{D}(x, y), \mathcal{D}(x+1, y)) - \sigma + 1 \quad \text{with } k = 1, \dots, \sigma - 1 \tag{4.30}$$

To summarize, the minimization- and the propagation-step is applied to every pixel of the image/scanline. This process is repeated until the disparities reach a fixed point. Then, the whole procedure is applied to the next resolution using the scaled up disparities.

Recommendations for an Efficient Implementation. In accordance to [104], we optimize scanlines individually to benefit from caching in CPUs. Further, we achieved good running times by storing the maximally tested disparity for every pixel, in order to reduce redundant computations. In this way, we discard disparities smaller than the maximally tested disparity. A similar improvement can be applied for the minimally tested disparity for every pixel.

Another important optimization is the use of SIMD[3] instructions (in all methods: the traditional local correlation-based method and our proposals). However, to keep the code maintainable, we optimized the dissimilarity measure only (in our case: sum of absolute differences over an 8×8 window). We used so called *compiler intrinsics* to avoid cryptic assembler instructions.

4.5. Generalizations to Decalibrated Stereo

To overcome problems resulting from an inaccurate stereo rectification we propose in this section generalizations of our dense correspondence computation algorithm, which is robust and efficient at the same time.

For this, we generalize and extend the concepts of the efficient disparity computation approach given in section 4.4, which was originally designed for highly efficient disparity retrieval from accurately rectified image pairs. We significantly increase the correspondence search range to a two dimensional area and, although based on window-based block matching, still maintain a surprisingly high efficiency. Consequently, our approach can be applied to decalibrated stereo image pairs and we also demonstrate the robustness by applying our method to optical flow image pairs.

4.5.1. Small Epipolar Deviations

In this section we assume that there is only a "small" decalibration with the following constraints, that the correct disparity is found within a small corridor along the epipolar line and that the horizontal displacement is always positive.

In the following, we generalize the approach presented in section 4.4, by modifying the individual processing steps. For every pixel location $\mathbf{x} = (x, y)^T$ we search for a displacement vector $\mathbf{f} = (u, v)^T$, where u and v are the displacements in x- and y-direction. The dissimilarity function $\mathcal{C}_A(\mathbf{x}, \mathbf{f})$ correlates a pixel $\mathbf{x}$ of the reference image with a pixel $(\mathbf{x}+\mathbf{f})$ of the match image. In practice, for $\mathcal{C}_A$ we use matching costs based on SAD $\mathcal{C}_{SAD}$, NCC $\mathcal{C}_{NCC}$ or Census transform $\mathcal{C}_{Census}$ and store two-dimensional displacements in a displacement field $\mathcal{F}(\mathbf{x}) = (u, v)^T$.

Optimization Procedure

Our idea was to generalize the algorithmic structure of section 4.4 in a way such that correspondences in a small corridor along the epipolar line are considered. This means that at every pixel, a 2D displacement vector is modified using the *minimization* step instead of a 1D disparity value. Again, the minimization will stop at local, suboptimal minima and to alleviate this problem, we also use the *propagation*, so that at every pixel, the displacement vectors of adjacent pixels are evaluated.

Minimization Step. Let the current displacement vector at $\mathbf{x}$ be $\mathbf{f}_0 = \mathcal{F}(\mathbf{x}) = (u_0, v_0)^T$. The mapping for the iteration is then given as:

$$\mathbf{f}_{n+1} = (u_{n+1}, v_{n+1})^T := \operatorname{argmin}_{\mathbf{f} \in M} \mathcal{C}_A(\mathbf{x}, \mathbf{f}) \tag{4.31}$$

[3] Single Instruction, Multiple Data

with the modified vectors

$$M = M_{ED} := \left\{ \begin{pmatrix} u_n \\ v_n \end{pmatrix}, \begin{pmatrix} u_n + 1 \\ v_n \end{pmatrix}, \begin{pmatrix} u_n + 1 \\ v_n + 1 \end{pmatrix}, \begin{pmatrix} u_n + 1 \\ v_n - 1 \end{pmatrix} \right\} \tag{4.32}$$

If $\mathbf{f}_{n+1} = \mathbf{f}_n$ the iteration is stopped and the flow field is updated. In practice, we perform the iteration at all pixels of the image.

Propagation Step. In the propagation at every pixel, the displacement vectors from surrounding pixels are evaluated and the displacement field is updated:

$$\mathcal{F}(\mathbf{x}) \mapsto \operatorname{argmin}_{\mathbf{f} \in N(\mathbf{x})} \mathcal{C}_A(\mathbf{x}, \mathbf{f}) \tag{4.33}$$

with the neighboring displacement vectors $N(\mathbf{x})$ (with $\mathcal{F}(\mathbf{x}) \in N(\mathbf{x})$). In this case, the horizontal component (*i.e.* the disparity) is never decreased. Then, the number of vectors to evaluate in the propagation step can also be reduced, by storing the maximally tested disparity for every pixel: only those propagated displacements are evaluated whose disparity is larger than the stored maximum. At this step, displacement vectors may be spread through their local neighborhood. In practice, we alternate minimization and propagation steps for a few iterations until convergence is achieved (2-3 repetitions from experience).

Hierarchical Iteration. In our original formulation, the image pyramid was created only by scaling the horizontal dimension to reduce ambiguity in textureless regions. Also here, the quality of disparity maps can be slightly improved, if only the horizontal dimension of the images is scaled in the image pyramid. However, this strongly reduces the maximally recoverable vertical displacement.

In the most generic formulation, both dimensions are scaled. So in every pyramid level, we perform the optimization procedure, which computes an estimated displacement field. At next resolution, the optimization uses the upscaled displacement field from the previous resolution as a starting point (in the beginning, all displacement vectors are set to $(0,0)^T$):

$$\mathcal{F}^{(k+1)}(2x + i, 2y + j) = 2\mathcal{F}^{(k)}(x, y) \quad \text{with } i, j \in \{0, 1\} \tag{4.34}$$

Epipolar Geometry. From the computed correspondences, the fundamental matrix can be estimated [136] to determine the epipolar geometry and a corrected rectification or a reconstruction using known techniques [3, 81].

If the pair of images is rectified, the updated disparity map for the rectified images must be derived. Let $\mathbf{x}_R = \mathbf{x}_L + \mathbf{d}$ be a correspondence and $\mathtt{H}_L$ and $\mathtt{H}_R$ be the rectifying homographies for the left and right frame. For every entry $\mathcal{D}'(\mathbf{x}'_L)$ of the updated disparity map we use the inverse mapping $\mathbf{x}_L = \mathtt{H}_L^{-1}(\mathbf{x}'_L)$ to compute:

$$\mathcal{D}'(\mathbf{x}'_L) = \mathtt{H}_R\left(\mathbf{x}_L + \mathcal{D}(\mathbf{x}_L)\right) - \mathbf{x}'_L \tag{4.35}$$

Please note that this formula uses inhomogeneous vectors, with $\mathtt{H}_L$ and $\mathtt{H}_R$ as projective functions. In practice, this step and the rectification is relatively time-consuming and for our application it is sufficient to simply ignore the small vertical displacements.

4.5.2. Large Epipolar Deviations

In this section we make no further conceptual restrictions and we allow arbitrarily large epipolar deviations and negative horizontal displacements. From some point of view, this problem is related to optical flow. It helps determining the motion of moving objects and has to face similar challenges as ordinary stereo. However, while for stereo the epipolar geometry can be used to constrain possible matches to epipolar lines, in optical flow a large (rectangular) search region must be considered instead.

We use the same notation as in the previous section 4.5.1.

Optimization Procedure

Also the optimization procedure is very similar to the generalization presented in section 4.5.1.

Minimization Step. At the minimization we employ a different set M of modified displacements:

$$M = M_{OF} := \left\{ \begin{pmatrix} u_n + i \\ v_n + j \end{pmatrix} \middle| \, i, j \in \{-1, 0, 1\}, i^2 + j^2 \leq 1 \right\} \tag{4.36}$$

In this case, four possible displacements are evaluated at every pixel.

Propagation Step. The propagation needs no further modification, but no computational improvements, like storing maximally tested displacements, are possible.

Hierarchical Iteration. To increase the efficiency, we scale both the horizontal and vertical dimensions using (4.34).

4.6. Disparity Refinement using Local Energy Minimization

Local correlation-based methods produce errors in regions near depth discontinuities [61, 146] due to the assumption of constant disparity within the support region. This assumption is violated at object borders. There are techniques to reduce such errors but they do not eliminate them completely and are sometimes hindering for real-time application, in terms of execution time. The most effective remedy is to abandon matching windows, and to use pixelwise matching. However, to treat instabilities caused by pixelwise matching [8], many global methods minimize an energy functional, such as $E(\mathcal{D}) = E_D(\mathcal{D}) + \lambda E_S(\mathcal{D})$, where E_D measures how well the depth-map $\mathcal{D}$ matches with the input images and E_S is a so called smoothness term, penalizing disparity variations [119].

Based on the assumption that the depth-map of a local method is a rough estimate of the ideal solution, we focus on enhancing a previously computed depth-map. To maximize the efficiency, we perform a winner-take-all optimization at every pixel (which is different to scanline optimization [119]).

The general idea is to propagate disparities through their neighborhood. This way, the computational scheme is similar to the approach presented in section 4.4.

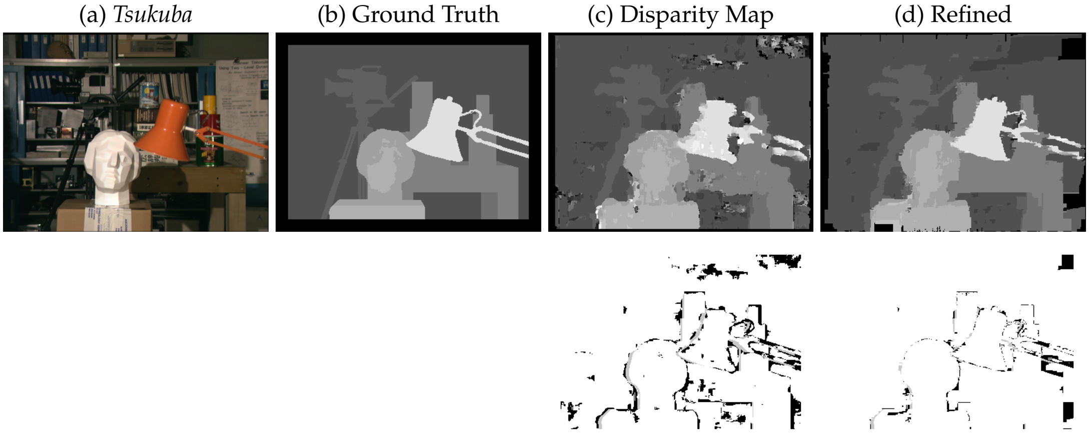


Figure 4.6. Local energy minimization applied to the dataset *Tsukuba*: (a) the left image and (b) the ground truth. (c) Shows a disparity map computed using the local method as presented in section 4.4 with the bad pixels (bottom row), where the blurred object boundaries are clearly visible. (d) Shows the refined result of our proposed post-processing routine.

Idea. The basic idea of this post-processing technique is motivated by the properties of local correlation-based stereo. These methods can be tweaked such that they perform relatively well in most image regions, but often many errors are made near discontinuities due to the use of a support region. As a result, object boundaries are often blurry and inaccurate. However, the discontinuity of a specific object boundary is present in most cases, but the actual location is wrong and is shifted by a few pixels. This behavior can be observed in Fig. 4.6 and is the motivation for the goal of our method: in the following, we aim at aligning the previously computed discontinuities on intensity edges.

Algorithm. For every pixel $\mathbf{x} = (x, y)^T$, we determine the best matching disparity value:

$$\mathcal{D}(\mathbf{x}) \mapsto \operatorname{argmin}_{d \in N(\mathbf{x})} \mathcal{C}_P(\mathbf{x}, d) \tag{4.37}$$

with the neighboring disparities $N(\mathbf{x})$ as defined in section 4.4. Our pixelwise matching cost $\mathcal{C}_P$ is defined as:

$$\mathcal{C}_P(\mathbf{x}, d) := \vartheta\big(\mathcal{C}_M(\mathbf{x}, d)\big) + \tau(\mathbf{x})\rho(d - \mathcal{D}(\mathbf{x} - \mathbf{r}_x)) + \tau(\mathbf{x})\rho(d - \mathcal{D}(\mathbf{x} - \mathbf{r}_y)) \tag{4.38}$$

with a function to truncate the pixelwise costs

$$\vartheta(x) := \begin{cases} 0 & x \leq I_T \\ x & \text{otherwise} \end{cases} \tag{4.39}$$

and

$$\tau(\mathbf{x}) := \begin{cases} \gamma & \Delta\mathcal{I}(\mathbf{x}) > \Theta_{\mathcal{I}} \\ 1 & \text{otherwise} \end{cases} \qquad \rho(t) := \begin{cases} 0 & t = 0 \\ P_L & |\,t\,| = 1 \\ P_H & \text{otherwise} \end{cases} \tag{4.40}$$

$\mathcal{C}_M$ is the (possibly truncated) absolute intensity difference or Birchfield and Tomasi's sampling invariant dissimilarity [8]. The parameters $\mathbf{r}_x$ and $\mathbf{r}_y$ point to the previously processed pixel/scanline (for example, if the image scanlines are processed from bottom to top, $\mathbf{r}_y$ may be set to $(0, -1)$). In (4.38) we use only two neighbors in order to avoid that depth discontinuities are penalized twice. The penalties P_L and P_H should be chosen such that $P_L < P_H$ to improve the recovery of slanted surfaces. The function τ helps to align discontinuities to intensity edges if $0 < \gamma < 1$ because it lowers the penalty if the intensity gradient $\Delta\mathcal{I}$ is high.

Through $\mathbf{r}_x$ and $\mathbf{r}_y$ the solution depends on the ordering in which pixels are processed. This also affects the possibilities how object borders may be adapted. In practice, we process every scanline in both directions, such that $\mathbf{r}_x \in \{(1,0)^T, (-1,0)^T\}$ and $\mathbf{r}_y = (0,1)^T$ (to support the propagation in horizontal directions equally). Therefore, the procedure processes every pixel a fixed number of times, depending on the number of directions.

Occlusion Detection. On top we try to improve disparities near depth discontinuities. Generally, there are occluded pixels in the left image, if there is a positive disparity gradient in positive x-direction. The number of occluded pixels is given by the difference of the two disparities. We implement this efficiently in a relatively simple way using an array. At every pixel $\mathbf{x} = (x, y)$ we mark the entry at index $x - \mathcal{D}(\mathbf{x})$. But, if the entry has already been marked, the pixel is considered occluded.

4.7. Results

We evaluate our and other methods using classical Middlebury stereo images with ground truth of Scharstein *et al.* [119] and present results obtained from real world stereo and motion-stereo sequences from a moving vehicle. We ran all methods with constant parameters across all Middlebury image pairs and use the same evaluation criteria as in [119]. Here, we focus on the standard datasets *Tsukuba*, *Venus*, *Teddy* and *Cones*. Our comparison comprises several methods:

1. Our region tracing method presented in section 4.3.

2. Our real-time disparity computation approach (RT) presented in section 4.4.

3. Our post-processing approach with and without occlusion detection (LEMO, LEM) presented in section 4.6.

4. Our efficient stereo matching that compensates for small (RT-SD) and large epipolar deviations (RT-LD) presented in section 4.5.

5. The traditional correlation (Trad.; section 4.2) using our own implementation.

6. Semi-Global Matching (SGM) [59] using our own implementation.

7. Belief Propagation (BP) [40] using our own implementation.

8. Graph Cuts (GC) [21] using an implementation from Yuri Boykov (available at [20]).

9. Geodesic Support Weights (GSW) [66] using our own implementation.

During the evaluation, we had several goals in mind: On the one hand, we wanted to compare our methods against efficient state of the art approaches. On the other hand, real-time performance and applicability to real world data is also very important for our application. These aspects will now be covered in the following sections.

4.7.1. Stereo Pairs with Ground Truth

In the following, we present quantitative results in Fig. 4.7, disparity maps of *Tsukuba* and *Teddy* in Fig. 4.8 and Fig. 4.9, and discuss our proposals based on these measurements.

Our Real-Time Methods. Our approach based on Region Tracing is worse than the traditional correlation, because sometimes the tracing of a region fails (for example the flowers on the right at *Teddy*). These errors are also more likely at complicated scenery, for example at thin structures. However, we found that these results were already very promising, considering that the disparity computation is completely different from the traditional approach.

Even though our efficient disparity computation scheme of section 4.4 is based on the idea of region tracing, it performs often better than the traditional one. On the first look this might be surprising – but our efficient routine does not explore the full range of possible disparities but stops adaptively based on the local distribution of disparity values and thus, avoids some errors.

Another huge benefit of our disparity computation algorithm is that it can be easily adapted to non-linear search ranges. This is underlined by the method presented in section 4.5 and the results of our implementation (there, the maximum range for the vertical displacement was set to ±30 pixels). In this case the results tend to be slightly worse than those of traditional correlation. However, the search space is in this case 61 times larger.

Local Energy Minimization. We apply our local energy minimization to local methods (*i.e.* traditional correlation and our real-time method), because it was designed to eliminate boundary errors that result from window-based matching. The bottom-line here is twofold: on the one hand, very good results are possible, but in complicated scenarios the simple occlusion detection may degrade results. On the other hand, at difficult structures a degradation is possible and also streaking effects, usually known from dynamic programming or scanline optimization, are sometimes visible. All in all, this routine helps to significantly improve the quality in real-time applications.

Robust Cost Measures. In our experiments with the Middlebury data set *Art* from [63] we performed stereo matching using image pairs with different combinations of exposures or illuminations similar to [63]. The main result depicted in the graphs of Fig. 4.10 is that the tested cost measures are less effective for different illuminations than for exposure changes. The matching error depends on the amount of the illumination change between the image pair. On the contrary, the exposure change has less influence on the error variation. The Census Transform is very effective in this case and shows only slight variations

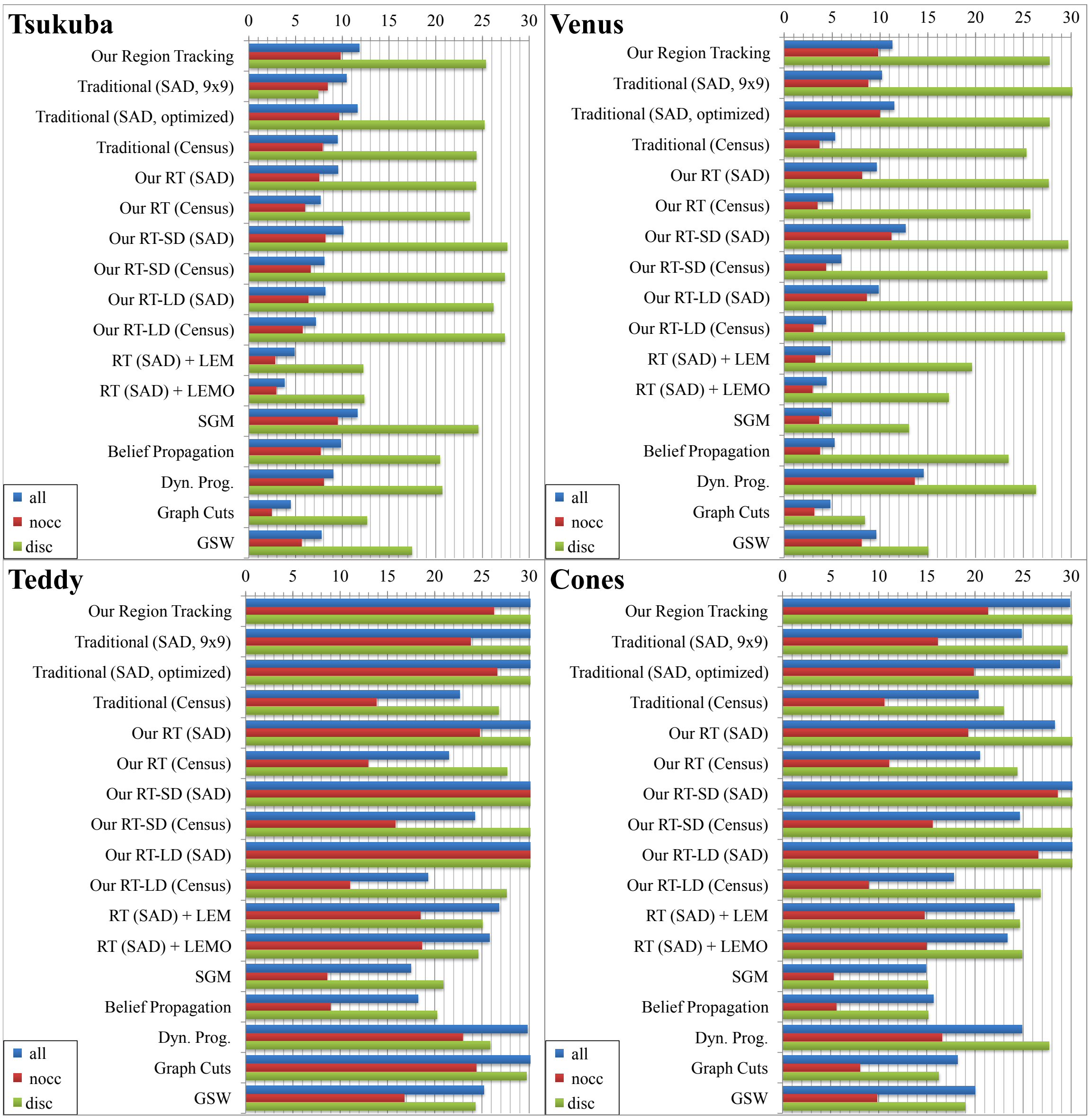


Figure 4.7. The performance of different stereo methods: the numbers are percentages of disparities that differ by more than 1 from the ground truth in regions near discontinuities (disc), non-occluded image regions (nocc) and the whole image (all). Disparity maps were computed using Traditional Correlation (*Trad*: section 4.2), our region tracing method (section 4.3), our real-time proposal (*RT*: section 4.4), our method that compensates for epipolar deviations (section 4.5), Semi-Global Matching (*SGM*: [59]), Belief Propagation [40], Dynamic Programming [119], Graph Cuts [21, 119] and Geodesic Support Weights (GSW) [66]. We activated the Local Energy Minimization without and with Occlusion Detection (*LEM* and *LEMO*: section 4.6 and section 4.6) for some of the methods.

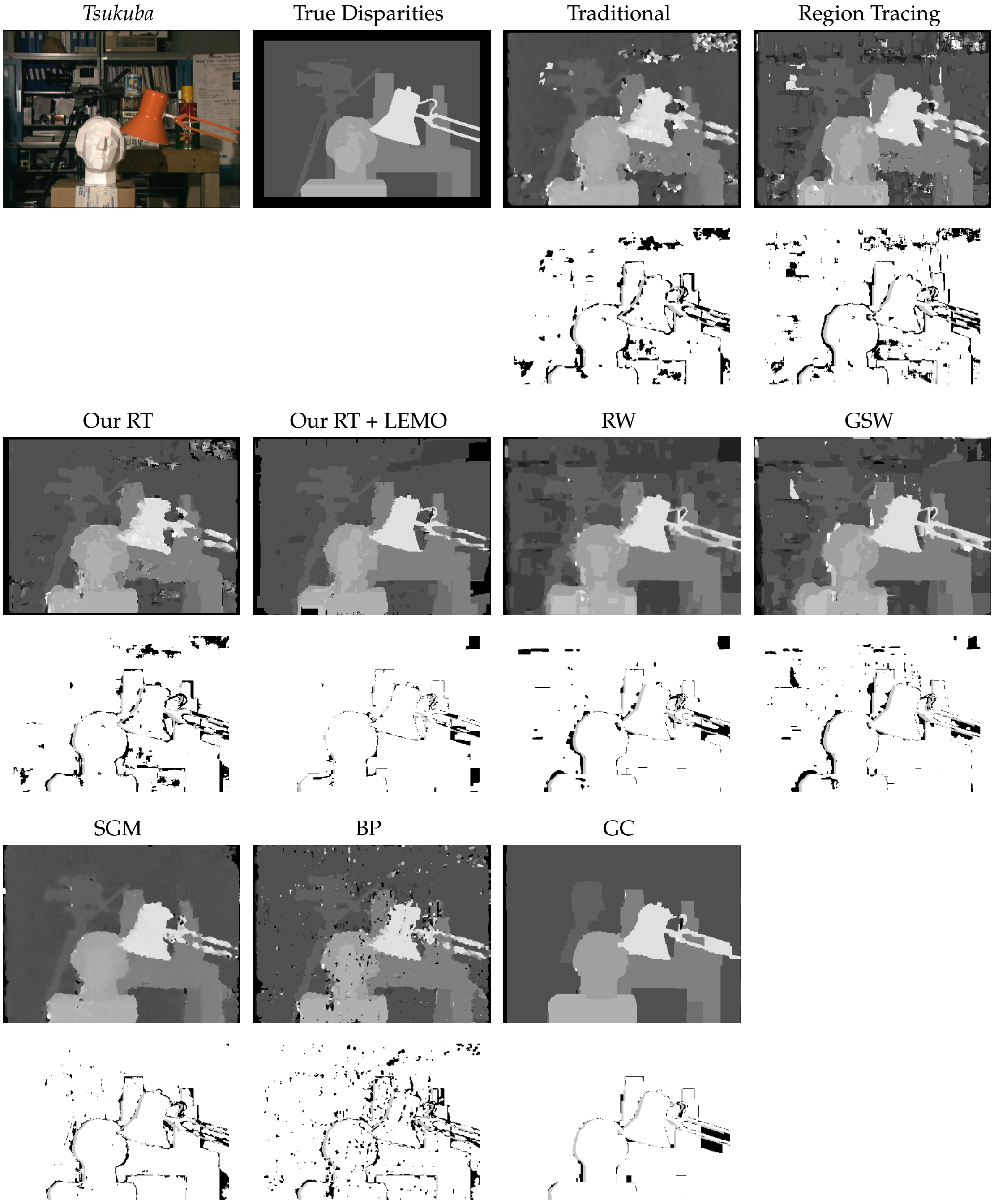


Figure 4.8. Disparity maps and bad pixels of the tested stereo methods for the dataset *Tsukuba*.

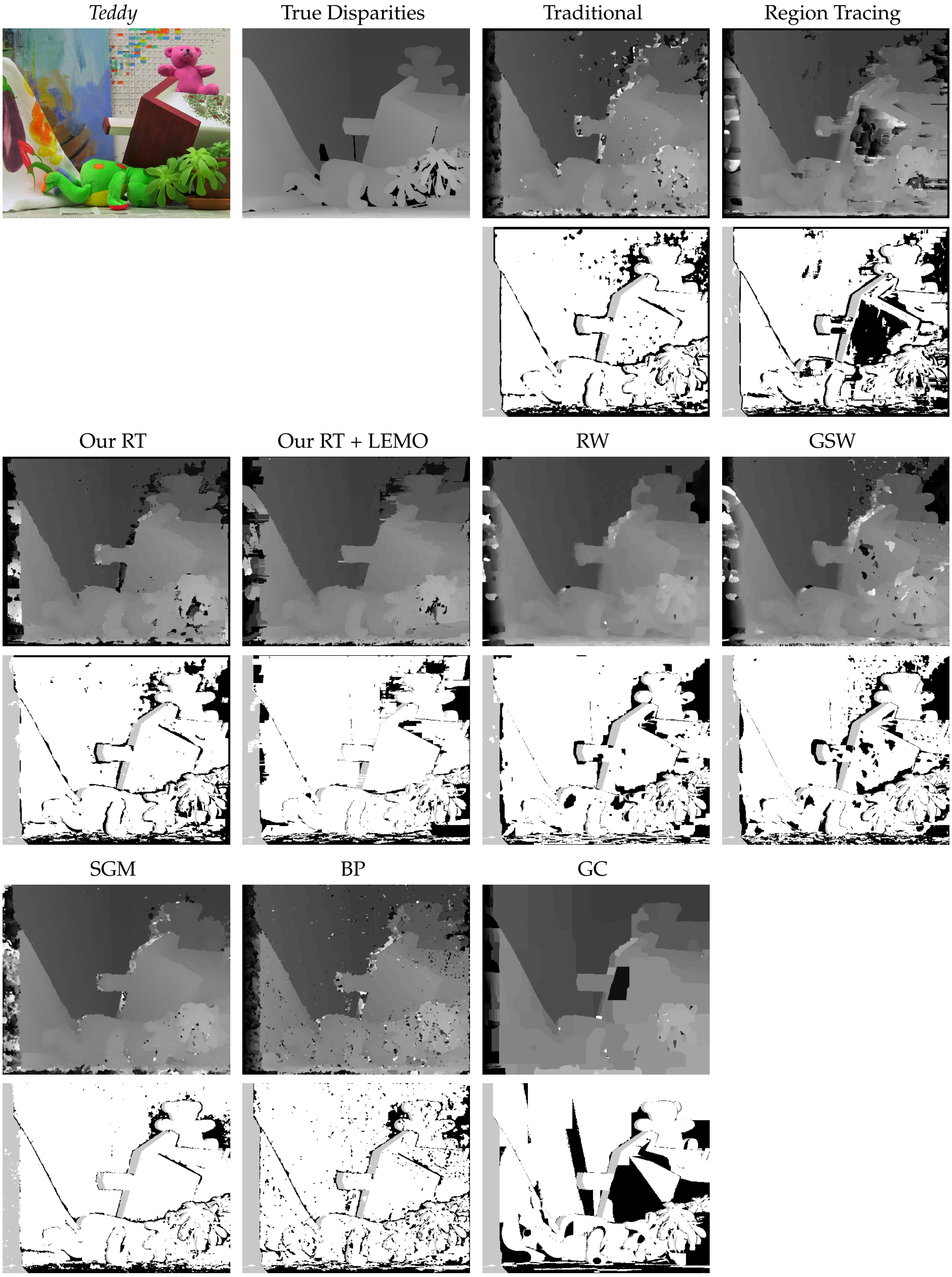


Figure 4.9. Disparity maps and bad pixels of the tested stereo methods for the dataset *Teddy*.

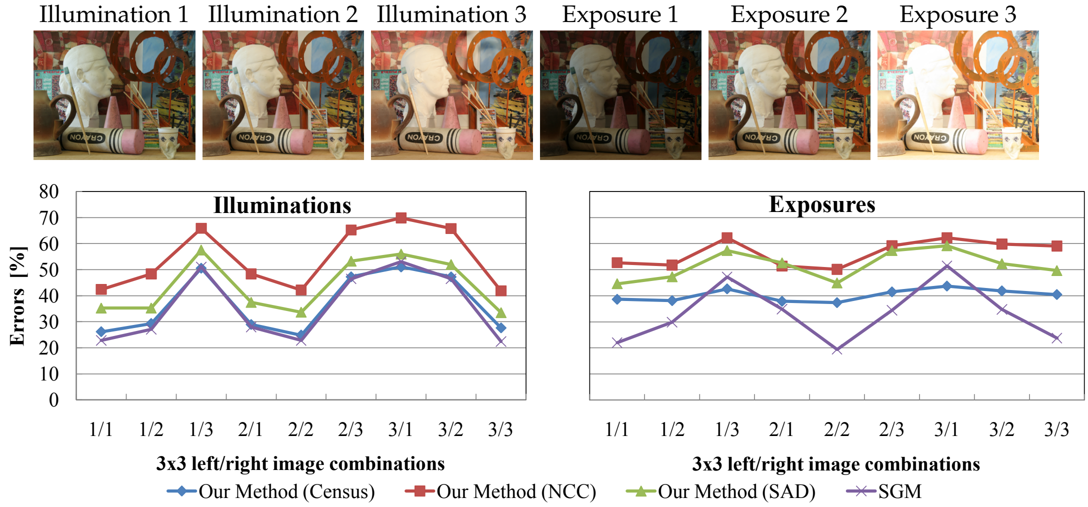


Figure 4.10. Results on the stereo dataset *Art* for different exposures and illuminations. We tested our real-time method presented in section 4.4 with different cost measures (Census Transform, NCC and SAD combined with a XSobel) and compare to semi-global matching [59] (SGM).

between the combinations. It is interesting that in many cases local matching can keep up with semi-global matching and was in two cases even better.

Pre- and Post-Processing. There are well known pre- and post-processing steps. Although they seem to be used by many authors on a regular basis to improve the quality of disparity maps, it is seldomly documented in literature. One improvement is the use of pre-processed input images. Especially applying a Sobel-filter in x-direction (XSobel) is a widely used technique (from experience, the improvement on standard datasets is roughly 3%). More often, applying a Gaussian smoothing of images was described – but from our experience, the effectiveness highly depends on image quality.

Some stereo methods tend to produce small "speckles" in the disparity map (single pixels with a wrong disparity). In these cases, post-processing disparity maps with a small median filter can – in extreme cases – reduce more than 10% errors.

In our methods, we only apply a XSobel to input images. In practice, we do not use a median filter due to the required processing resources, but apply it to disparity maps of SGM and GSW.

Other Stereo Methods. In practice, GSW preserves many fine details, which results often in high quality disparity maps, but occluded regions and slanted or curved surfaces are problematic. SGM performs very well on many kinds of imagery, but also here wrong assignments are likely in occluded areas. Similarly behaves BP, but with the exception that it works inferior on real image sequences (especially at slanted surfaces). Disparity maps of GC are usually very "clean", but also there, weaknesses can be observed at slanted surfaces.

4.7.2. More Results on Real World Sequences

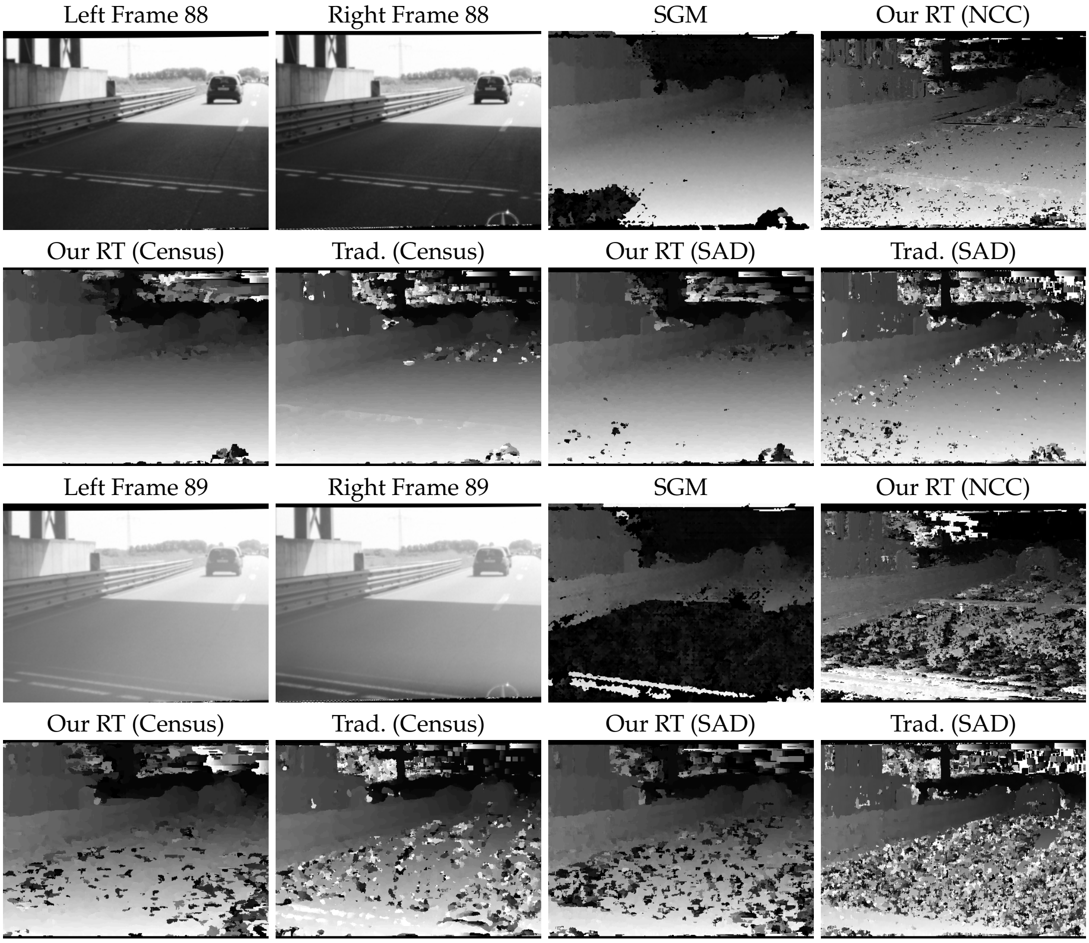


Figure 4.11. Results on the sequence *Exposure Changes*.

We performed tests on real world sequences provided by the 2011 DAGM Adverse Vision Conditions Challenge (AVCC) and on imagery from our vehicle. In particular, we present results on the sequences *Exposure Changes* (see Fig. 4.11), *Groundplane Violation* (see Fig. 4.12) and a motion-stereo video from our automotive application (see Fig. 4.13), because there are interesting differences noticeable. For the *Motion-Stereo* example we picked a very challenging sequence with incident sunlight. In practice this leads to glare light artifacts and frequent exposure changes. Particular to our sequences is the presence of non-lambertian lighting, such as specular reflections and specular highlights.

In Fig. 4.11 (*Exposure Changes*), Fig. 4.12 (*Groundplane Violation*) and Fig. 4.13 (*Motion-Stereo*) we show a comparison of traditional matching algorithms and our stereo method with different similarity measures. We performed all experiments in full image resolution. As expected, Census Transform performs in overall better than the other similarity measures, and surprisingly, SAD performs also quite well in combination with a x-Sobel operator. The quality when using NCC is relatively bad, which might be explained by a high sensitivity in homogeneous regions. Semi-global Matching of [59] produces relatively

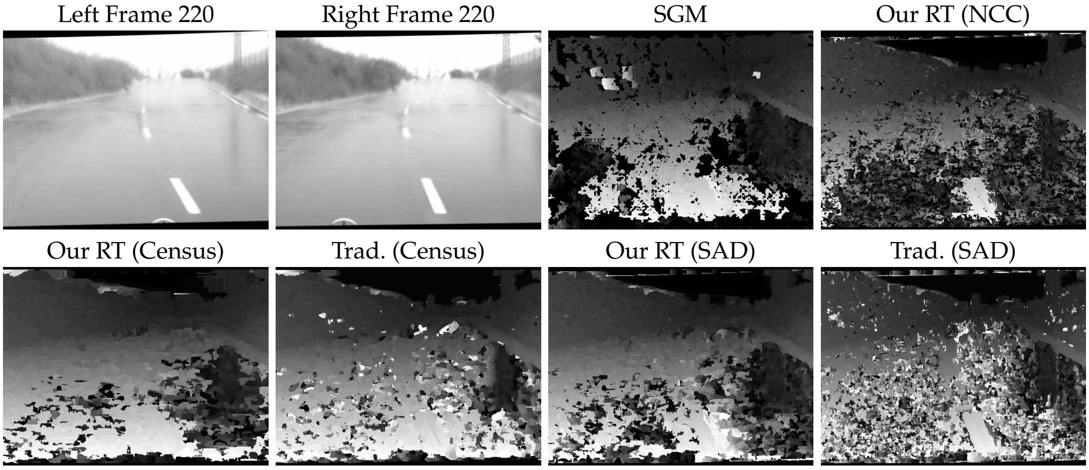


Figure 4.12. Results on the sequence *Groundplane Violation*.

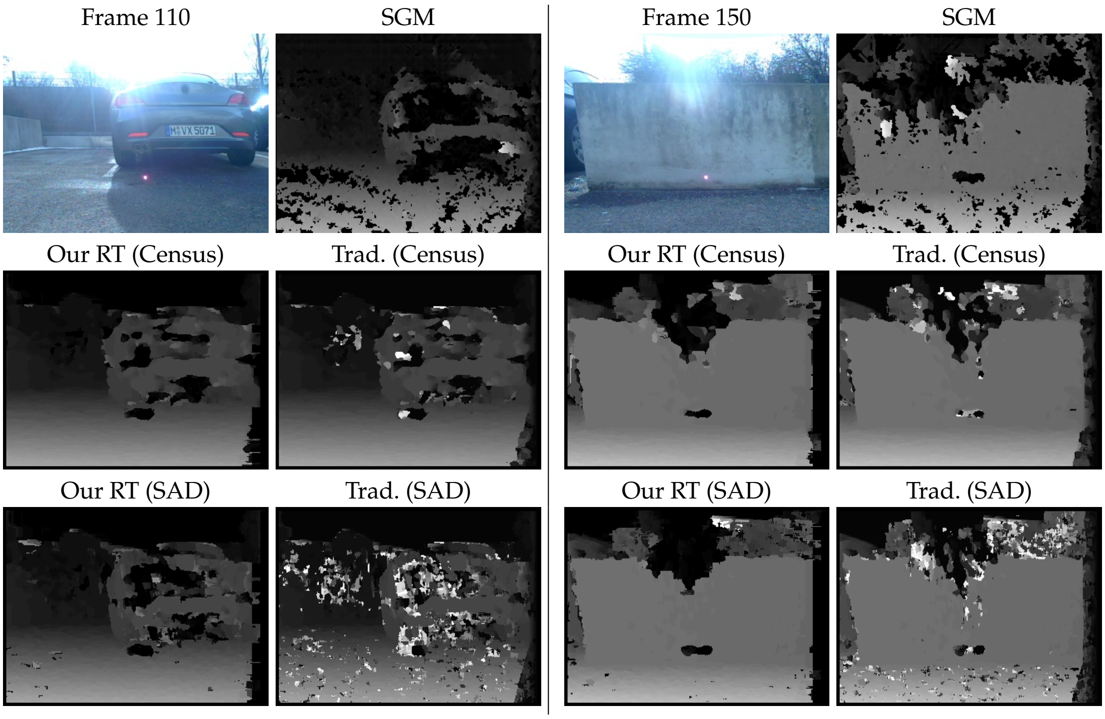


Figure 4.13. The robustness of different methods and similarity measures at a difficult motion-stereo sequence.

good results, but in some difficult situations the density of the disparity maps is reduced in homogeneous regions (see Fig. 4.11 frame 89, Fig. 4.12 and Fig. 4.13).

For the *Motion-Stereo* example we picked a very challenging sequence with incident sunlight. In practice, this leads to glare light effects and, for cameras that perform active exposure control, frequent exposure changes. Such glaring observably leads to inoptimal exposure of the imager and blooming. Furthermore, light which is scattered within the lens results in reduced contrast and lens flare patterns such as starbursts and circles.

All these effects can be reduced (but not avoided) by using our approach in combination with Census Transform. However, whether such artifacts result in false matches depends highly on the strengths of the artifact (which in turn depends on the local distribution of colors) and also the expected result. For example, if a disparity value is solely justifiable by visual correspondence, a disparity value of zero might be correct for the lens flare – if a disparity map should render the projected structure as close as possible, the disparity value should depend on depth only. For our application, the most effective approach to treat these disturbances is by using multi-view stereo fusion.

4.7.3. Experiments on Stereo with Epipolar Deviations

In this section we mainly focus on experiments where epipolar deviations are present and compare the extensions described in sections 4.5.2 and 4.5.1 to the original formulation in section 4.4. We show the performance of three different methods:

1. **Our RT**: The stereo implementation of section 4.4, which does not account for epipolar deviations.
2. **Our RT-SD**: Our implementation including the improvements given in section 4.5.1, which is optimized for fast running times and **small** epipolar deviations.
3. **Our RT-LD**: The implementation of section 4.5.2, which can handle **large** epipolar deviations.

We evaluate using modified stereo datasets of [119] and show qualitative results on real world sequences acquired with the vehicle. To simulate the effects of an inaccurately estimated epipolar geometry, we transform the right image of every dataset of [119] with a homography which does not modify the x-coordinate of transformed points. By keeping the left camera frame unchanged, we can still use the provided ground truth disparity maps without modification. At every value for the epipolar deviation v_{max}, we transformed the right image of every dataset with a random homography such that the epipolar deviation v of every pixel fulfills $-v_{max} \leq v \leq v_{max}$. To determine the overall disparity-error, we ran the algorithms and compared the estimated disparity maps to the ground truth.

Large Deviations. Fig. 4.14 shows the overall disparity-error (the percentage of disparities that differ by more than 1 from the ground truth) of the standard stereo approach (section 4.4), our RT-LD which handles large epipolar deviations (section 4.5.2) and our optimized variant RT-SD for small deviations (section 4.5.1) using growing values of the maximal epipolar deviation (x-axis). It can be seen that for the stereo approach the error grows very quickly, whereas in our generalizations, the error grows only slightly. The error of the optimized variant RT-SD also grows fast, but a slight improvement is visible at very small deviations.

Fig. 4.16 shows disparity maps of our real-time stereo approach (RT) and our generalization for large displacements (RT-LD) for different values of the epipolar deviation. The degradation of the disparity maps of the stereo approach is clearly visible from the appearance of wrongly estimated disparities. However, our generalization (RT-LD) is qualitatively very robust against epipolar deviations.

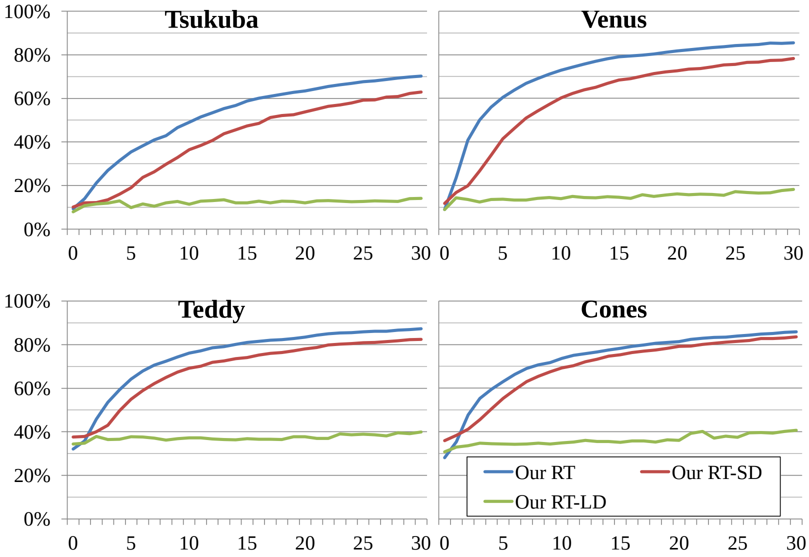


Figure 4.14. The overall disparity-error (y-axis) of different methods for growing values of the epipolar deviation (x-axis): the errors are percentages of disparities that differ by more than 1 from the ground truth. We tested our real-time stereo approach (section 4.4), our RT-LD which handles large epipolar deviations (section 4.5.2) and the optimized variant RT-SD for small deviations (section 4.5.1).

Small Deviations. For small epipolar deviations (up to a few pixels), the *optimized* variant (RT-SD) presented in section 4.5.1 is interesting. Fig. 4.15 shows the overall error of the methods for growing values of the maximal epipolar deviation.

The optimized variant (RT-SD) is slightly worse than the algorithm which can handle large deviations (RT-LD). In practice however, it gives a good compromise between quality and processing time, if the expected maximal epipolar deviation is less than three pixels.

Real Sequences with Epipolar Deviations. We selected some particular video sequences acquired with a monocular side-looking camera on the vehicle integrated into the front-bumper for our motion-stereo applications. In these examples, the pairwise rectification of camera images was not accurate due to floor unevenness. In Fig. 4.17, we present an undistorted camera image and disparity maps computed using our stereo approach and our generalizations for small and large epipolar deviations (RT-SD and RT-LD). This figure intuitively reflects our practical experience that the optimized variant (RT-SD) is a very good compromise between speed and accuracy, and is sufficient in almost all situations of our application.

4.7.4. Experiments on Optical Flow

We performed tests on the challenging real image sequences *Large Displacement* and *Exposure Changes* provided by the 2011 DAGM AVCC and show results in Fig. 4.18. We used our stereo matching method generalized for large epipolar deviations (RT-LD) and directly used the computed displacement field as the flow map. In *Large Displacement*, the vehicle

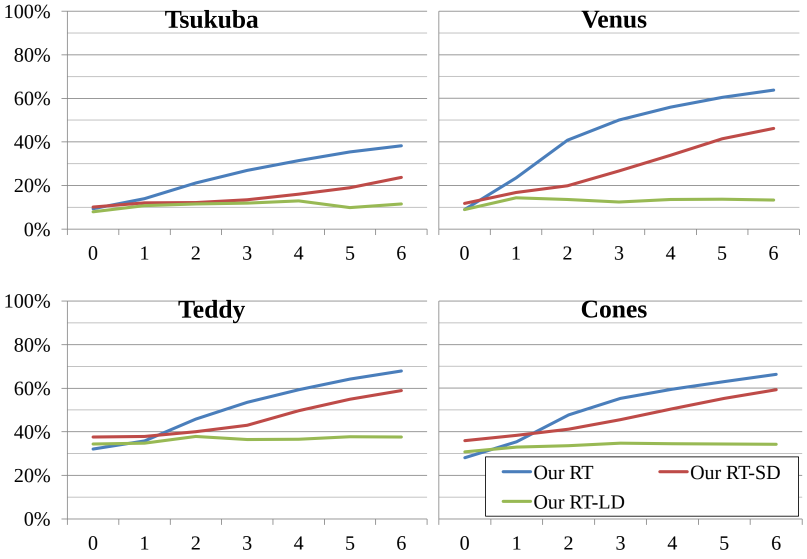


Figure 4.15. The overall error (y-axis) of different methods for small values of the epipolar deviation (x-axis): the errors are percentages of disparities that differ by more than 1 from the ground truth. We tested our real-time stereo approach (section 4.4), our RT-LD which handles large epipolar deviations (section 4.5.2) and the optimized variant RT-SD for small deviations (section 4.5.1).

in front (entering from the left) drives with a higher velocity than the cars in the background (which move from right to left). The sequence *Exposure Changes* is a video from a forward looking camera on a forward moving vehicle, where a sudden change in exposure takes place between frames 90 and 91. Our method with Census Transform shows again the best overall result, but also the SAD cost measure works surprisingly well on images that were previously filtered with a Sobel filter in x direction. The Horn and Schunck algorithm [64] recovered only very localized motion. TV-L1 of [27] works relatively well, but is highly sensitive to exposure changes. Due to this reason we use Sobel-filtered images, but in this case the smoothness is negatively affected by image noise. At dramatic changes of the exposure time, none of the methods succeeded.

4.7.5. Execution Times

We measured the execution times in Tab. 4.1 on an Intel E8200 with 2.67 GHz and 4 GiB RAM using a single threaded implementation. In practice, the running time of our region tracing method (section 4.3) depends on the image content. To some extent, this also applies to our real-time approach (section 4.4), but the localized structure and the hierarchical setup result in more stable timings. Our local energy minimization is relatively expensive, because at every pixel several comparisons with the surrounding values are utilized which is costly and not parallelizable. The generalizations of our stereo method which includes epipolar deviations preserve also a high efficiency: while the most general formulation introduces a high overhead, the optimized variant for small deviations is quite efficient. The huge performance gap when using different cost functions compared to the traditional

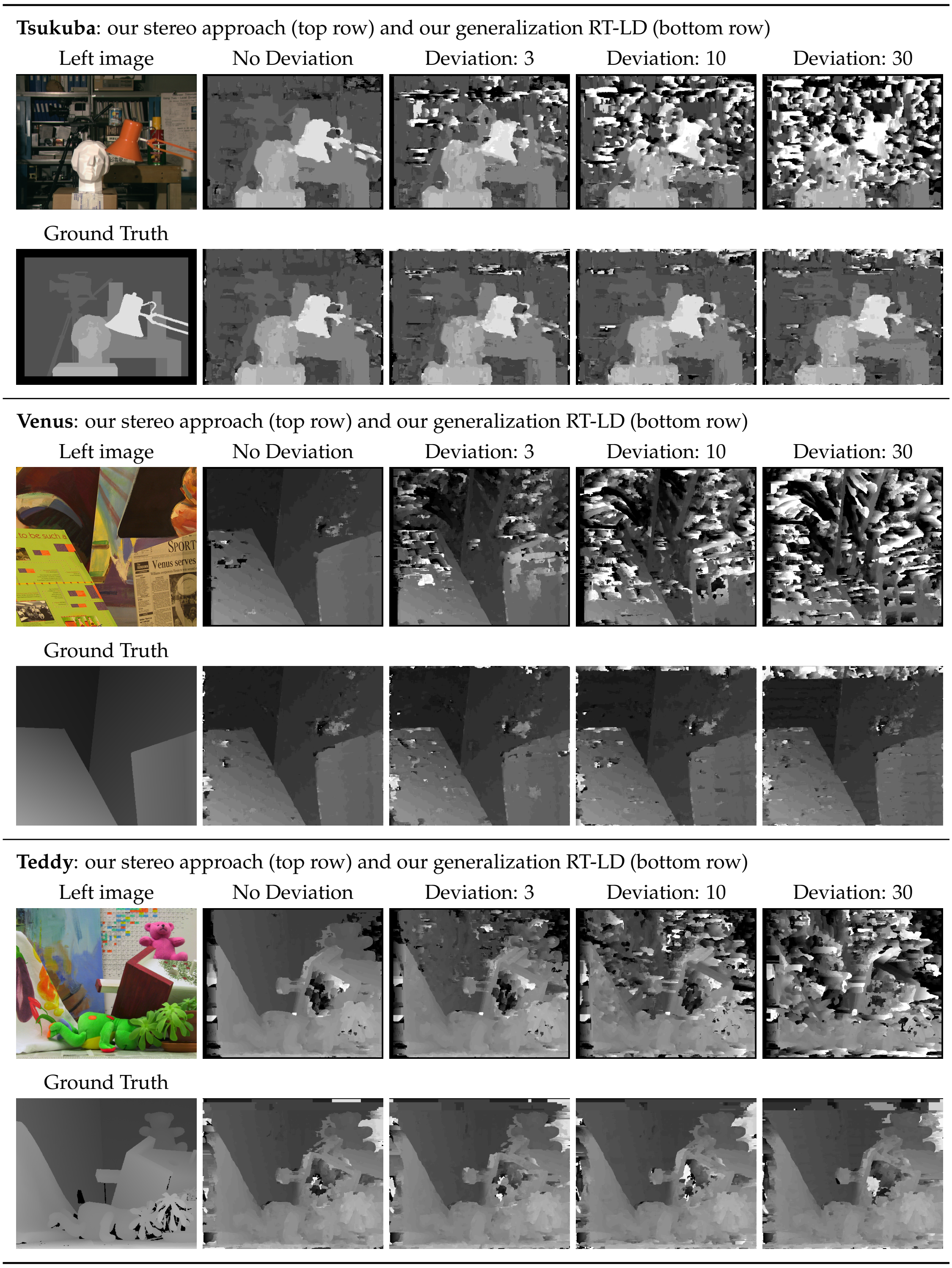

Tsukuba: our stereo approach (top row) and our generalization RT-LD (bottom row)
Left image
No Deviation
Deviation: 3
Deviation: 10
Deviation: 30
Ground Truth
Venus: our stereo approach (top row) and our generalization RT-LD (bottom row)
Left image
No Deviation
Deviation: 3
Deviation: 10
Deviation: 30
Ground Truth
Teddy: our stereo approach (top row) and our generalization RT-LD (bottom row)
Left image
No Deviation
Deviation: 3
Deviation: 10
Deviation: 30
Ground Truth

Figure 4.16. Disparity Maps of our real-time stereo approach (RT) and our generalization (RT-LD) at different epipolar deviations. In these cases, the epipolar deviation of at least one pixel was 0, 3, 10 or 30. In each block, the top row is the result of the stereo method (RT) and the bottom row shows the result of our generalization (RT-LD).

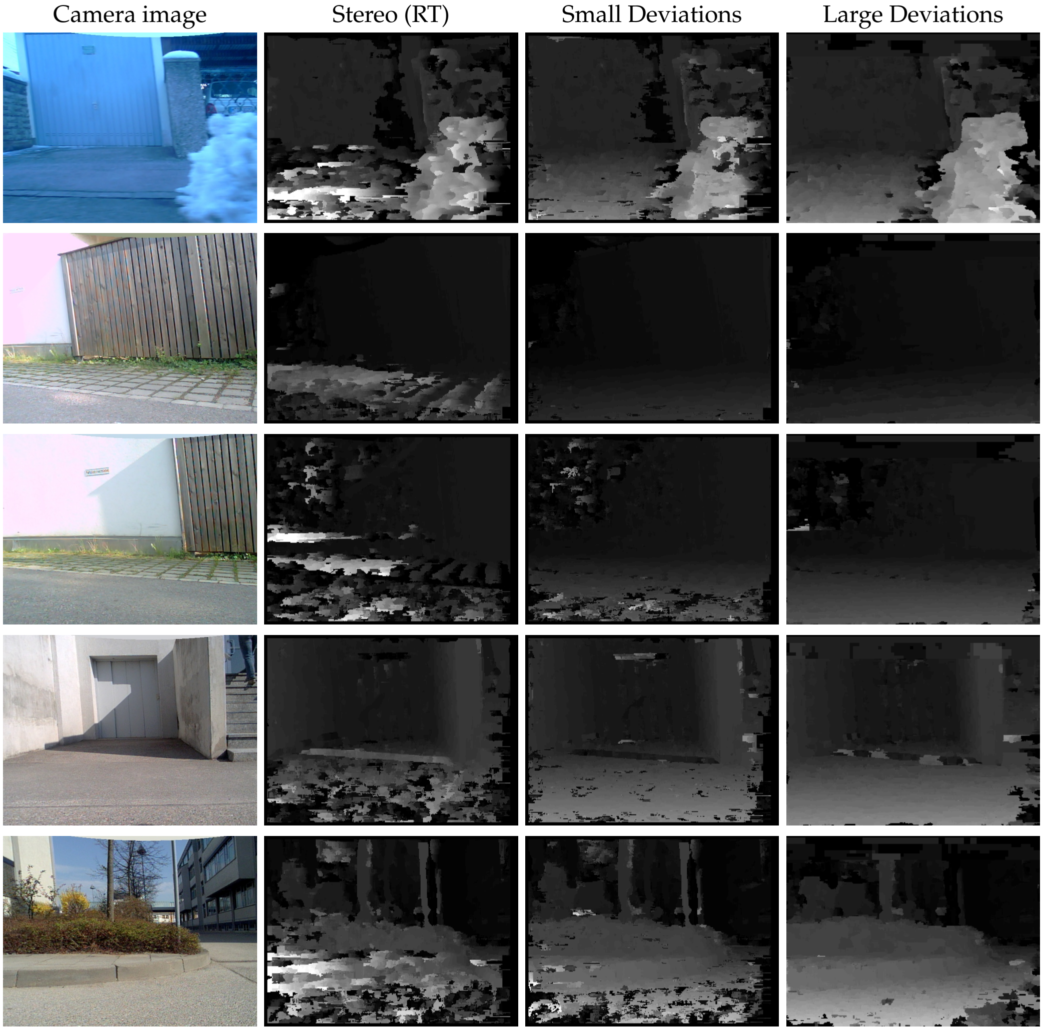
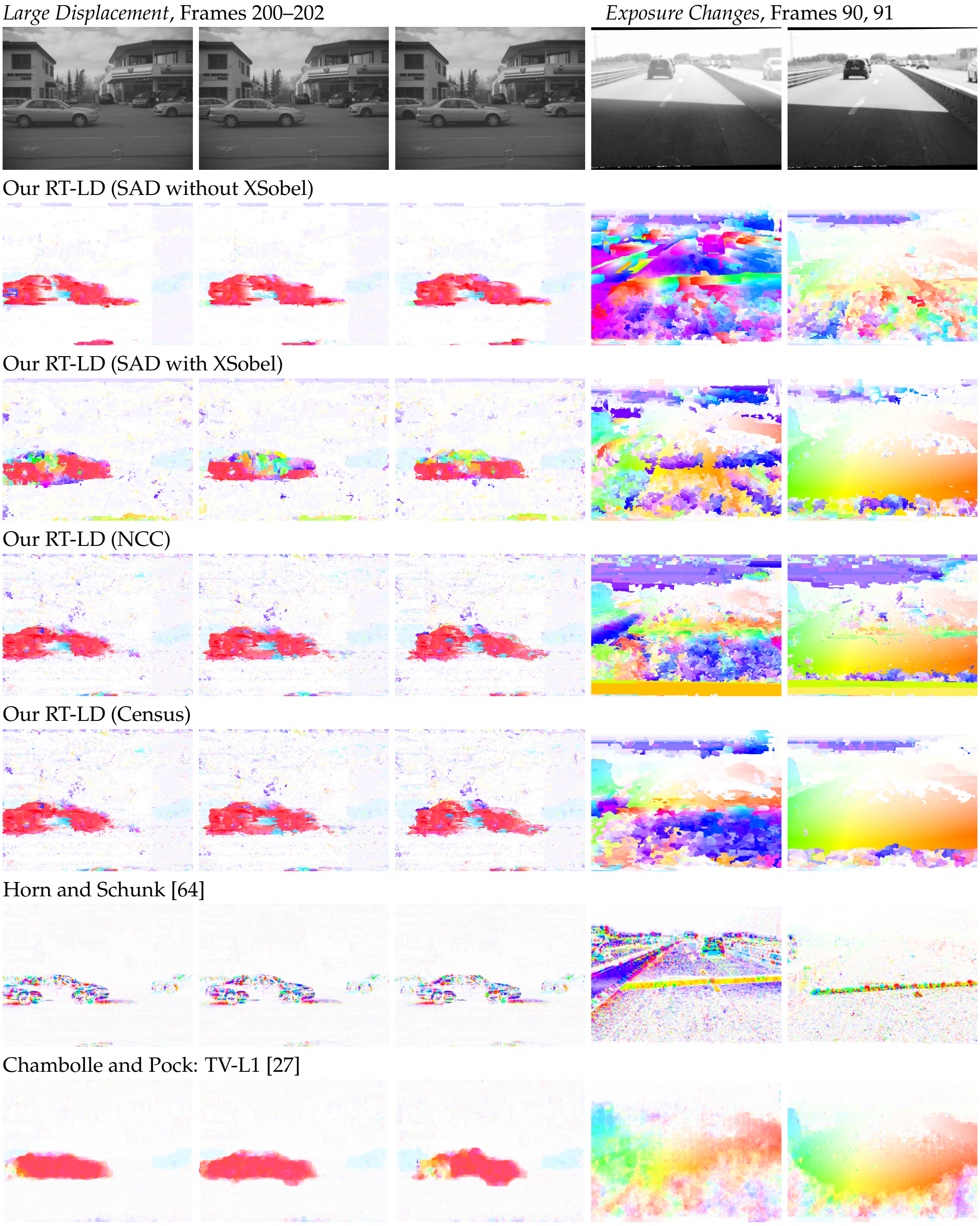


Figure 4.17. Disparity Maps of our real-time stereo approach, our variants for small (RT-SD) and large deviations (RT-LD) at real sequences.



Large Displacement, Frames 200–202

Exposure Changes, Frames 90, 91

Our RT-LD (SAD without XSobel)

Our RT-LD (SAD with XSobel)

Our RT-LD (NCC)

Our RT-LD (Census)

Horn and Schunk [64]

Chambolle and Pock: TV-L1 [27]

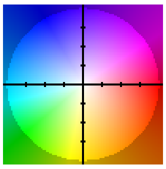


Figure 4.18. Results on the sequences *Large Displacement* (flow fields were computed for frame pairs (200, 201), (201, 202) and (202, 203)) and *Exposure Changes* (flow fields were computed for frame pairs (90, 91) and (91, 92)). The image on the right denotes the color-coding of the computed flow vectors. **Please note that this figure is best viewed in color.**

Table 4.1. The execution times of the different stereo methods in milliseconds on different datasets. The second row denotes the image resolution and maximum disparity (WxHxD)

Method	**Tsukuba**	**Teddy**	**Art**	**Real**
	384x288x12	450x375x64	463x370x84	320x240x48
Traditional (SAD)	37	154	192	80
Traditional (SAD, optimized)	32	140	189	56
Traditional (Census)	153	372	428	172
Our Region Tracing (SAD)	78	184	276	59
Our RT (SAD)	31	53	61	18
Our RT (Census)	190	311	352	204
Our LEM	23	19	44	13
Our LEMO	25	20	46	14
Our RT-SD	86	156	185	51
Our RT-LD	223	398	399	149
SGM	191	970	1250	423
Belief Propagation	500	2656	3637	1174
Dyn. Prog.	227	1264	1740	556
Graph Cuts	72 s	11 m	15 m	3 m
GSW	2 m	3 m	7 m	3 m

method results from a highly reduced number of cost function evaluations.

Optical flow methods were also tested (see Tab. 4.2) on 640x481 images and flow vectors with displacements of at most 30 pixels in each direction (the timings of [27] are not comparable, because their Matlab implementation took minutes and a GPU version reported as real-time [27] is of course not comparable to a CPU implementation). We also performed tests with down-scaled images (third of their original size). The timings underline the high efficiency of our proposed generalization RT-LD which demonstrates that with small images it is possible to compute dense optical flow with large displacements in real-time on commodity hardware, even with robust cost measures. On higher resolutions, the performance gap to traditional block matching is extremely big: our proposal is up to 90 times faster due to our efficient search algorithm and the hierarchical setup.

4.8. Discussion

In this chapter we introduced several novel concepts: we presented a new idea that performs region tracing to compute a disparity for every pixel. This is achieved by identifying a critical set of pixels for which the disparity values are updated iteratively. Based on this approach, we derive a localized, more efficient and general algorithm which retains a high efficiency by massively reducing the number of required cost function evaluations. Even though originally formulated only for stereo matching of rectified image pairs, we generalize our proposal in a way such that images from decalibrated stereo rigs can be addressed efficiently. We further introduced a novel local energy minimization for the post-processing of disparity maps, which improves the localization of depth discontinuities and improves the results of area-based methods. In practice however, this simple technique is relatively time-consuming and might be confused in difficult scenarios. The validity and efficiency of our proposals is finally underlined by exhaustive experiments on

Table 4.2. The execution times of the different optical flow methods in milliseconds. Stereo matching was tested on images with a resolution of 1024x334 and a maximum disparity of 48 was used for methods that require it a priori. Optical flow was tested on 640x481 images and flow vectors with displacements of at most 30 pixels in each direction. We also performed tests on images scaled to one third of their original size

Method	**Stereo**	**Flow**	**Flow**
	1024x334	640x481	213x160
Trad. Block Matching (SAD)	263	39744	641
Trad. Block Matching (Census)	2313	162769	2029
Trad. Block Matching (NCC)	2691	140648	1913
Our Method RT-LD (SAD)	129	466	52
Our Method RT-LD (Census)	403	1763	167
Our Method RT-LD (NCC)	560	1812	179
Horn and Schunk [64]	-	313	34
Chambolle and Pock [27]	-	N/A	N/A

stereo images with ground truth and imagery from real sequences. Our results show that our proposals are more accurate and are faster than traditional area-based methods.

In practice, our real-time approach performs very well and is very efficient. The geometric structure of the sequences from our vehicle is in most cases very simple (*e.g*. urban scenarios with many planar surfaces), which leads to an even higher performance of our method. However, at difficult lighting situations the quality of the disparity maps is impacted by temporary phenomena like glare lights, light bursts or simply image noise. But also at depth discontinuities and in occluded regions the disparity values are often not correct, which is a well known drawback of window-based matching. Methods that are based on energy minimization with pixelwise matching costs (like semi-global matching or belief propagation) perform better in these cases, but take also much more time to process and are far from real-time on our platform. We therefore chose to use the disparity maps obtained from our very fast stereo matching method introduced in section 4.4 and concentrated our efforts on improving these disparity maps by exploiting the redundancy in the depth data, because most scene points are observed more than once. The success of our investigations will be presented in the next chapter.

5. Accurate Stereo Vision

> ***The Problem of the Random Walk:*** *A man starts from a point O and walks l yards in a straight line; he then turns through any angle whatever and walks another l yards in a second straight line. He repeats this process n times. . .*[1] KARL PEARSON [108]

In this chapter we focus on improving the quality of binocular stereo matching. In practice, real-time performance is always desirable, but from a scientific point of view it is a very prohibitive restriction and therefore we relax the constraint in this chapter. Further, approaches which are not real-time today may achieve such performance in future as a result of increased clock speeds or due to novel processor architectures. A good example is Hirschmüller's semi-global matching. At the time of its presentation it was far from real-time on ordinary CPUs. However, a few years later the method was ported to an FPGA [51] and rendered the processing of megapixel images at frame rate possible.

Improvements in the quality of stereo matching have been made continuously over the past decades as a result of a considerable amount of research. This has resulted in great advances that are mainly based on segmentation supported global optimization. However, in real world applications, like automotive driver assistance, one is often faced with a wide spectrum of illumination conditions and with huge variations of the environment, which often makes the adaption of the parameters of the segmentation algorithm difficult. One well known alternative are support weighted matching windows which make the correlation function more distinctive. However, since perspective distortions of the matching windows are usually not taken into account, these approaches have huge limitations at slanted surfaces. Furthermore, since occluded pixels may be correlated with high support weights, some limitations in regions near discontinuities exist.

Motivated by the strengths of segmentation and support weighting methods, we introduce random walks as matching primitives that combine the strengths of both worlds. Generally, random walks are stochastic processes that traverse the image in a random way. At each step of the walk, the next adjacent pixel location is chosen based on color similarity. In our proposal, we explicitly simulate random walks and use them for matching since they are robust along discontinuities. Further, we incorporate surface orientations in order to handle perspective distortions from slanted surfaces. We also introduce a novel voting technique based on simulated random walks that serves to identify the most probable disparities. After this voting step, we propagate disparities into occluded regions using random walks and compute a probability distribution over disparities for every image pixel.

[1] In 1905 Prof. KARL PEARSON raised the question in the NATURE journal about random walks and asks: *I require the probability that after these n stretches he is at a distance between r and $r + \delta r$ from his starting point.* Lord RAYLEIGH answered that for great values of n the probability sought is $\frac{2}{n}e^{-r^2/n}r\delta r$. Later, PEARSON thanks several correspondants and comments: *The lesson of Lord Reyleigh's solution is that in open country the most probable place to find a drunken man who is at all capable of keeping on his feet is somewhere near his starting point!*

Finally, we use this distribution as a prior for global energy minimization. In this chapter, we integrate strengths of many existing stereo methods in a completely novel manner, where the use of random walks is essential not only for matching but also for selecting the most probable disparities.

We provide extensive evaluation results and give an elaborate analysis on the contributions of the different steps of our algorithm, highlighting the improvements they provide. The method presented in this chapter compares very well with state of the art and is capable of achieving top rankings at the Middlebury benchmark. As of Nov. 2012, we are able to reach the 2nd place using the more restrictive quality threshold of 0.75 between computed and ground truth disparities. The results demonstrate improvements in notoriously difficult situations like occlusions, discontinuities and slanted surfaces.

5.1. Related Methods

In this section we only discuss directly related methods. We refer to section 3.2 for a thorough overview on stereo.

Shen *et al.* [122] use the improved random walks algorithm of Grady [54] to compute reliable matches. In a second step, they interpolate matches in ambiguous regions. Their approach is very different from our work because we explicitly simulate random walks and use them as matching primitives. Further, we incorporate surface orientations and we also introduce a novel voting technique based on simulated random walks. Moreover, in our propagation model, we obtain a complete probability distribution for the disparities and use it as a prior for global energy minimization.

Local support weighted approaches like the works from Hosni *et al.* [66] or from Yoon and Kweon [168] are also to some extent related. In these methods a rectangular region is used for matching and the pixels inside the window are weighted based on color information. Among local methods, [66, 168] can achieve a very good quality, but they often produce errors in textureless regions which is due to the lack of global constraints. Since the assumptions about fronto-parallel surfaces are violated, errors also appear on slanted surfaces and near depth discontinuities.

In the context of window-based matching, Bleyer *et al.* [15] addressed slanted surfaces. However, due to the relatively large size of the matching windows, a huge number of orientations have to be evaluated. Therefore, their approach cannot explore the whole parameter space and the solution is computed iteratively. In contrast, our method is able to produce high quality results by considering only a few surface orientations and no iterative scheme is required.

Wang and Yang [156] introduced sparse ground control points at the places where the matching has been repeatedly achieved using three different local matching cost functions. From them, a dense disparity map is obtained by diffusing these reliable matches in the continuous disparity domain. Finally, the stereo problem is solved using global energy minimization with an additional unary potential which penalizes deviations from the propagated disparity map. In contrast to their work, we discretely propagate disparities into occluded regions in order to obtain a probability for every discrete disparity. Consequently, we directly improve the matching cost function instead of using an additional energy potential.

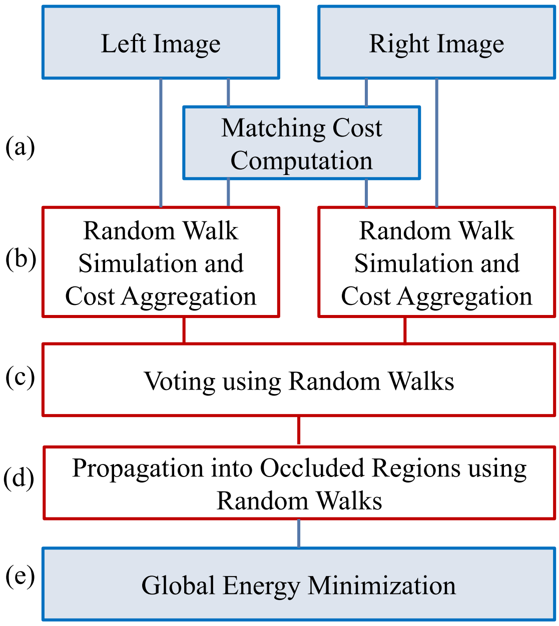

Left Image
Right Image
(a)
Matching Cost Computation
(b)
Random Walk Simulation and Cost Aggregation
Random Walk Simulation and Cost Aggregation
(c)
Voting using Random Walks
(d)
Propagation into Occluded Regions using Random Walks
(e)
Global Energy Minimization

Figure 5.1. The processing steps of our method: (a) Pixel-wise correlation of the input images; (b) Aggregation of matching costs for every random walk where occlusions and slanted surfaces are taken into account; (c) The information of all random walks is collected in the voting volume; (d) Reliable matches are propagated to occluded regions and probabilities of the disparities for every pixel are computed; (e) Finally, the probabilities serve as a prior for global energy minimization.

5.2. Accurate Stereo Vision using Simulated Random Walks

Fig. 5.1 shows an overview of our method. We first compute pixel-wise matching costs that are stored in the matching cost volume for every image location and the range of evaluated disparity values. Instead of aggregating matching costs using windows around the pixels like in local methods, we simulate random walks at every pixel to aggregate matching costs along each walk. Since random walks rarely cross large image gradients, this can be understood as a pre-segmentation step. In this way, we increase robustness at discontinuities. We further explicitly consider occlusions, by simulating random walks in both left and right images, and slanted surfaces, by evaluating different surface orientations. Once we computed a cost function for every random walk, we introduce a novel voting technique that fuses the information of all random walks into one global voting volume. After this step, we identify inconsistent regions which are mainly caused by occluded pixels and use random walks to propagate reliable matches into these regions. All these computations result in a very robust data term which is finally used as a prior for global energy minimization.

5.2.1. Computation of Matching Costs

We first compute the matching costs which is basically a cost value for all possible image correspondences. Formally, we compute a matching cost volume $\mathcal{C}_M(x, y, d)$ where (x, y) is defined for all possible image locations of the left image $\mathcal{I}_L$ and d iterates over the set of possible disparity values: $d_{\min} \leq d \leq d_{\max}$. For a given (x, y, d), a correspondence between the left image pixel intensity $\mathcal{I}_L(x, y)$ and right image pixel intensity $\mathcal{I}_R(x - d, y)$ of a rectified image pair is then computed.

We use pixel-wise dissimilarities using the sampling invariant differences of Birchfield and Tomasi [9]. One might use any other method here, *e.g.* Census or Rank transform [169], which are often preferred in practice because of their efficiency and robustness to photometric variations.

5.2.2. Random Walks

We now formally define a random walk as an ordered sequence of pixel locations $\mathcal{R}(\mathbf{x}_S) = \langle \mathbf{r}_i \rangle_{0 \leq i \leq N}$ starting at a *start pixel* $\mathbf{r}_0 := \mathbf{x}_S$, where N is the length of the random walk. At each step of the walk, *i.e.* $\mathbf{r}_i \mapsto \mathbf{r}_{i+1}$, a new pixel is randomly selected for $\mathbf{r}_{i+1}$ from the four-connected neighborhood of $\mathbf{r}_i$ based on *transition probabilities* $p_T(\mathbf{r}_{i+1} \,|\, \mathbf{r}_i)$.

Later, we want to use the set of pixels defined by a random walk to infer information about the depth of all the pixels along the walk. Therefore a random walk ideally never crosses object boundaries and covers only pixels that are defined by the same scene surface. Similar to many other works, we also assume that depth discontinuities coincide with sharp color gradients and thus, we define the transition probability for a random walk step from pixel $\mathbf{r}_i$ to pixel $\mathbf{r}_{i+1}$ as a function of the color similarity:

$$p_T(\mathbf{r}_{i+1} \,|\, \mathbf{r}_i) = \frac{1}{C(\mathbf{r}_i)} \cdot \exp\left(- \frac{d_C(\mathcal{I}(\mathbf{r}_i) - \mathcal{I}(2\mathbf{r}_{i+1} - \mathbf{r}_i))}{\sigma_C} \right) \tag{5.1}$$

where $\mathbf{r}_{i+1}$ is a pixel from the four-connected neighborhood $\mathcal{N}(\mathbf{r}_i)$ of $\mathbf{r}_i$. The value $C(\mathbf{r}_i)$ is a normalization factor, such that the transition probabilities sum up to one for every pixel. The function d_C computes the color norm for grayscale or RGB images and in practice we use $d_C(\mathbf{c}) = \sqrt{\mathbf{c}^T \mathbf{c}}$, where the vector $\mathbf{c}$ contains the differences of the individual color channels. In our experiments, we found that this choice of color similarities performs quite well, but it may be necessary to adapt it to other types of camera sensors, for example by weighting the different channels. Further, in (5.1) we used a step size of 2 pixels for the probability computation because along depth discontinuities, there is an at least 1 pixel wide region of highly unreliable color values, which is a result of pixel sampling. Therefore, by using a step size of 2 for probability computation, we reduce the risk that a random walk crosses an object boundary.

The value σ_C in (5.1) controls how likely it is that a random walk steps towards a pixel with a higher color difference. Given that d_C is a strictly increasing function, the above definition in (5.1) will always prefer pixels with a similar color (*i.e.* pixels with smaller values of d_C). This bias towards similarly colored pixels can be controlled using σ_C. For example, if σ_C is set to a very large value, then the value of p_T will depend less on the actual value of d_C. In practice, σ_C depends on image sensor noise which might be given

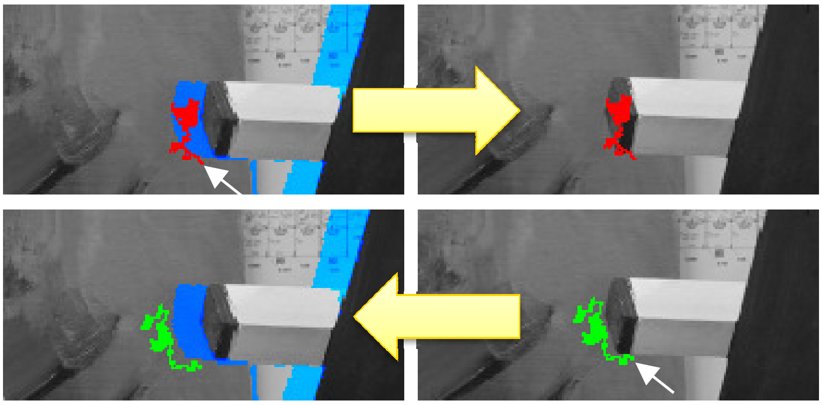


Figure 5.2. Left-right simulation of random walks for the image pair *Teddy*. White arrows indicate the pixels at which random walks were started. Occluded pixels are painted in blue. In the top left image the random walk was started at a non-occluded pixel, but crosses the area of occluded blue pixels. In the bottom right a random walk does not go into an occluded area because no occluded pixels are present at that location in the right image.

in specifications from the image sensor supplier, but it can also be estimated efficiently for a given image using the method of Immerkær [68]. In practice we obtained good results with $\sigma_C = 4\sigma_I$ with σ_I being the *standard deviation* of image noise computed by [68]. In general, σ_C also depends on the texture of the scene surfaces: even if σ_C is set according to sensor noise, a suboptimal behavior of the random walks is possible for highly textured image content – that is, random walks cover only a few pixels. This can be explained by the amount of surface texture present in the image: in such situations color variations may lead to very small transition probabilities.

5.2.3. Left-Right Simulation

In this section, we take a closer look at the behavior of random walks near occlusions. Including occluded pixels later for correlation is highly problematic, because pixels with no physical relation to the other image would be correlated with completely wrong surfaces, and their corresponding cost values would not be reliable. In Fig. 5.2, we show a magnified region of the stereo image pair *Teddy* from Middlebury [119]. The random walks are simulated in both left and right images starting at the pixel labeled with a white arrow. In the upper left image a *left random walk* painted in red is simulated. In the lower right image a *right random walk* painted in green is simulated. In the left images occluded pixels are painted in blue. The left random walk crosses occluded pixels painted in blue, but this does not happen for the right random walk because no occluded pixels are present in the vicinity in the right image.

In general, both the left and right images contain occluded pixels, however at different locations in the images. Therefore, we simulate random walks for the left and right images independently and denote them as $\mathcal{R}^L$ and $\mathcal{R}^R$ respectively. Formally, every pixel location $\mathbf{x}_L$ of the left image is assigned a random walk and so they can be "re-used" for the different disparities (the same applies similarly for the right image). We will explain in the next section how these left and right walks come together, but the basic idea is that the aggregated correlation values of a random walk are usually higher if the walk covers occluded pixels. This technique does not completely avoid the presence of occluded pixels in random walks. It may fail in regions where many thin foreground objects are present and it will fail for random walks whose start pixel is occluded.

5.2.4. Cost Aggregation using Multiple Surface Orientations

One of the big challenges for stereo methods are slanted surfaces and in the following we describe how we tackle this problem. First, we assume that the surface shape can be linearly approximated for the region covered by a random walk. While this assumption might be violated for some walks that traverse a large image region, we argue that the failure of some walks is negligible for the final result, due to our voting technique presented in the next section. Further, our experiments clearly show that our approach can handle very difficult geometries.

To compute the aggregated cost $\mathcal{C}_A(\mathbf{x}, d, \delta_k)$ for a given pixel $\mathbf{x}$ and disparity d, we use the random walks defined in the left and right image and take different a priori surface orientations $\delta_k \in \Delta$ into account. To perform left to right correlation, we use the random walk $\mathcal{R}^L(\mathbf{x}) = \langle \mathbf{r}_i \rangle_{0 \leq i \leq N}$ and sum the pixelwise costs $\mathcal{C}_M$ along the walk:

$$\mathcal{C}_A^L(\mathbf{x}, d, \delta_k) = \sum_{i=0}^{N} \mathcal{C}_M(\mathbf{r}_i, d + \delta_k^T(\mathbf{r}_i - \mathbf{r}_0)) \tag{5.2}$$

The expression $d + \delta_k^T(\mathbf{r}_i - \mathbf{r}_0)$ in (5.2) adapts the disparity value to the given surface orientation δ_k, which is defined as the disparity gradient in horizontal and vertical direction. Note that in this formulation some pixels may occur multiple times in the correlation sum. For right to left correlation we use the random walk of the right image at the corresponding position $\mathcal{R}^R(\mathbf{x} - (d, 0)^T) = \langle \mathbf{r}_i \rangle_{0 \leq i \leq N}$:

$$\mathcal{C}_A^R(\mathbf{x}, d, \delta_k) = \sum_{i=0}^{N} \mathcal{C}_M(\mathbf{r}_i + (d, 0)^T, d + \delta_k^T(\mathbf{r}_i - \mathbf{r}_0)) \tag{5.3}$$

Note that $\mathcal{C}_M$ is defined for pixels of the left image and therefore, we shift the image positions back to the left image using $\mathbf{r}_i + (d, 0)^T$. Finally, we obtain a global cost volume which assigns a dissimilarity score to every disparity and orientation given a pixel location $\mathbf{x}$ of the left image:

$$\mathcal{C}_A(\mathbf{x}, d, \delta_k) = \min(\mathcal{C}_A^R(\mathbf{x}, d, \delta_k), \mathcal{C}_A^L(\mathbf{x}, d, \delta_k)) \tag{5.4}$$

In the next section, we analyze the cost volume $\mathcal{C}_A$ to determine depths and orientations for every random walk of the left image.

5.2.5. Voting using Random Walks

In this step, we transform the volume of aggregated costs $\mathcal{C}_A$ into a *voting space* $\mathcal{V}$ using a novel technique based again on random walks. The main intuition consists of two simple observations. Firstly, the minima of the aggregated costs $\mathcal{C}_A$ provide optimal depth and orientation hypotheses for *all* pixel locations of the random walks. Secondly, every simulated random walk covers a set of different pixel locations and thus, in general, every pixel is covered by many different random walks. Therefore, for every pixel $\mathbf{x}$, there are multiple depth and orientation hypotheses that are contributed by different random walks which start in the neighborhood of $\mathbf{x}$ and which cover $\mathbf{x}$. By collecting this information in the voting space at every pixel we will obtain confirmation of all walks about optimal disparities.

The voting space $\mathcal{V}(\mathbf{x}, d)$ can be best understood as a data structure that holds a histogram for disparity values at every pixel location and is initialized to zero. First, for a given pixel $\mathbf{x}$ of the left image, we collect a set S of relevant depth and orientation hypotheses:

$$S = \{(d, \delta) \,|\, \mathcal{C}_A(\mathbf{x}, d, \delta) \leq \hat{s} + N\Theta, d_{\min} \leq d \leq d_{\max}, \delta \in \Delta\} \tag{5.5}$$

with $\hat{s} = \min_{d,\delta} \mathcal{C}_A(\mathbf{x}, d, \delta)$. The parameter Θ specifies a small corridor within which the hypotheses may be located relative to the minimum $\hat{s}$, and N is the length of the random walks.

The second step is to update the voting volume with a random walk starting at pixel $\mathbf{x}$ based on simulations using the hypotheses in S. For every hypothesis $(d, \delta) \in S$ for the start pixel $\mathbf{x}$, we update every pixel of the random walk $\mathcal{R}(\mathbf{x}) = \langle \mathbf{r}_i \rangle_i$ by simulating the random walk again with the given depth and orientation prior using:

$$\mathcal{V}(\mathbf{r}_i, d_i) \mapsto \mathcal{V}(\mathbf{r}_i, d_i) + 1 \qquad \text{with } d_i = d + \delta^T(\mathbf{r}_i - \mathbf{x}). \tag{5.6}$$

Note that for a given random walk, (5.6) is updated only once per pixel, even if a pixel occurs several times in the walk. Ideally the optimal disparity for every pixel will receive many votes from other random walks which cross that pixel.

5.2.6. Propagation into Occluded Regions

The voting space $\mathcal{V}(\mathbf{x}, d)$ may be used to infer information about non-occluded pixels but, in occluded areas, it is impossible to obtain depth information using binocular correlation. Therefore, our idea is to identify occluded pixels using the well known left-right consistency check [61] and then propagate depth information into inconsistent regions using random walks. This can be achieved by either, explicitly simulating random walks or by computing deterministic paths using Dijkstra's shortest path algorithm. However, both approaches are highly inefficient and we use the random walk framework proposed by Grady [54] to propagate disparities with high confidence to the occluded regions. Originally, Grady used random walks for image segmentation by propagating a set of seed labels into unlabeled regions using a weighted graph. In his work, he exploits an important connection between random walks and potential theory which in turn is related to the Dirichlet problem. The solution to such problems may be found by solving for the minima of the combinatorial Dirichlet problem $\frac{1}{2}\vec{x}^T L \vec{x}$, where in our case $\vec{x}$ holds label probabilities for every pixel of the image and L is the combinatorial Laplacian matrix initialized using the graph weights:

$$L_{ij} = \begin{cases} 1 & i = j \\ -p_T(\mathbf{x}_i|\mathbf{x}_j) & \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ adjacent} \\ 0 & \text{otherwise} \end{cases} \tag{5.7}$$

where p_T is computed using (5.1) with a step size of 1 to ensure $L_{ij} = L_{ji}$. By reorganizing the indices of L, the vector $\vec{x}$ can be split into unlabeled pixels $\vec{u}$ and known pixels $\vec{m}$ (*i.e.* $\vec{x}^T = (\vec{u}^T, \vec{m}^T)$). Also L can be partitioned into several submatrices where the submatrix A represents the connectivity between unlabeled nodes and B the edges between known and unlabeled pixels. If n_m is the number of known pixels then A and B are defiend by $A_{ij} = L_{i+n_m,j+n_m}$ and $B_{ij} = L_{i,j+n_m}$.

The most important result is then a sparse system of linear equations $A\vec{u} = -B^T\vec{m}$ where, in our case, $\vec{m}$ is initialized with probabilities for disparities of non-occluded pixels of the disparity map and $\vec{u}$ will contain the probabilities for the disparities of occluded pixels after solving the linear system.

Finally, to compute the probability $p(\mathbf{x}, d)$ that the disparity is d at $\mathbf{x}$ we proceed as follows. We first extract a disparity map $\mathcal{D}_C$ from $\mathcal{V}$ and filter out inconsistencies using the left-right consistency check [61]. The probability for a consistent pixel is directly computed using the histogram in $\mathcal{V}$ and for the inconsistent ones we use the propagation described above. To compute the probabilities for the inconsistent pixels for disparity d, we first initialize the vector of consistent pixels $\vec{m}$ by setting $\vec{m}_j$ to 1 if $|\mathcal{D}_C(\mathbf{x}_j) - d| < 1$ and to 0 otherwise. Then, we solve the linear system and directly obtain the probabilities for all inconsistent pixels: $p(\mathbf{x}_{i+n_m}, d) = \vec{u}_i$. Note that the system must be solved for every discrete disparity value.

5.2.7. Global Stereo Model

The steps described above compute a probability distribution $p(\mathbf{x}, d)$ for the disparities at every pixel location. To resolve ambiguity and spurious matches we enforce a smoothness constraint using an energy function $E(\mathcal{D}) = \lambda E_D(\mathcal{D}) + E_S(\mathcal{D})$ with

$$E_D(\mathcal{D}) = \sum_{\mathbf{x}} -\log p(\mathbf{x}, \mathcal{D}(\mathbf{x})) \qquad E_S(\mathcal{D}) = \sum_{\mathbf{x}} \sum_{\mathbf{y}\in\mathcal{N}(\mathbf{x})} \min(\kappa|\mathcal{D}(\mathbf{x}) - \mathcal{D}(\mathbf{y})|, \tau) \tag{5.8}$$

where the data term E_D directly uses the computed probabilities and for the smoothness term E_S we use a simple linear truncated model. In practice we use loopy belief propagation to optimize the energy $E(\mathcal{D})$.

5.2.8. Summary

To summarize shortly, we first aggregate pixel-wise costs $\mathcal{C}_M(\mathbf{x}, d)$ into the aggregated cost volume $\mathcal{C}_A(\mathbf{x}, d, \delta)$ by considering a small number of a priori orientations Δ. Using $\mathcal{C}_A$ we simulate random walks with specific depth and orientation hypotheses and collect votes in $\mathcal{V}(\mathbf{x}, d)$ which holds a histogram of disparities for every pixel. Then, we identify inconsistencies and propagate there using random walks. These computations result in a probability distribution $p(\mathbf{x}, d)$ which serves as a prior for global optimization.

5.3. Results

We tested our method using the Middlebury stereo datasets with ground truth following the methodology of Scharstein and Szeliski [119] and used the same parameters for all images. We performed tests on a dual Intel X5690 and our highly parallelized but not yet optimized C++ CPU implementation completes in just 9.7s and 13.3s for the datasets *Teddy* and *Art*, respectively.

Since our method consists of many parts and depends on a few parameters we perform a thorough analysis of the influence of these parameter and of the different parts of the algorithm to the performance of our method.

5.3.1. Parameter Analysis

The walk length N and the parameter σ_C of the random walk transition probability of (5.1) are the most critical ones for the performance of our method. Therefore we evaluate their influence to the performance of our method. In Fig. 5.3, we present exemplary results for varying values of N and σ_C using the datasets *Tsukuba* and *Cones*. We chose these two datasets because we measured the biggest differences for them.

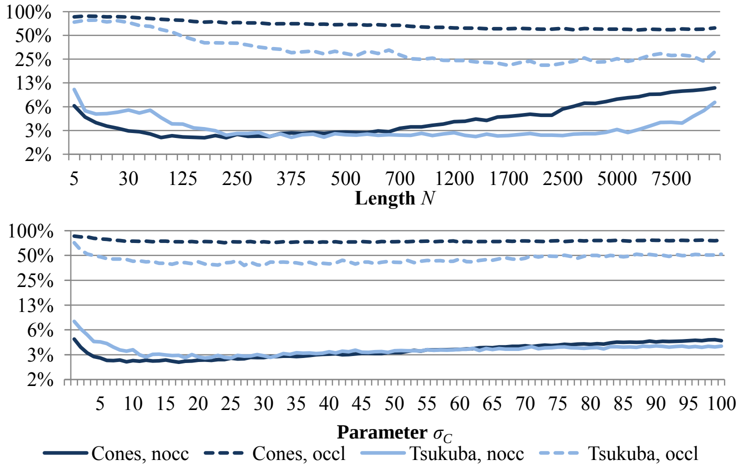


Figure 5.3. The charts display the influence of the parameters N and σ_C on the errors in non-occluded (nocc) and occluded (occl) areas. Errors are defined as percentages of disparities that differ by more than 1 from the ground truth and we used a logarithmic scale of the vertical axis for a better visualization. To generate these charts, we did not use the random walk based propagation. For the first chart we fixed $\sigma_C = 15$ and for the second chart $N = 200$.

Small values of N introduce errors because aggregated matching costs are less discriminative in this case and thus, many wrong minimum values receive votes. Larger values of N have a positive effect on the performance in occluded regions, which can be explained by the voting strategy. If a walk covers occluded pixels and if the correct disparity and orientation is contained in S then these occluded pixels will receive support for the correct disparity. At the same time, the errors increase in non-occluded regions because it becomes more likely that walks step over discontinuities. The steeper increase at *Tsukuba* might be explained by the simple geometric structure of *Tsukuba* (there are less discontinuities than in *Teddy*).

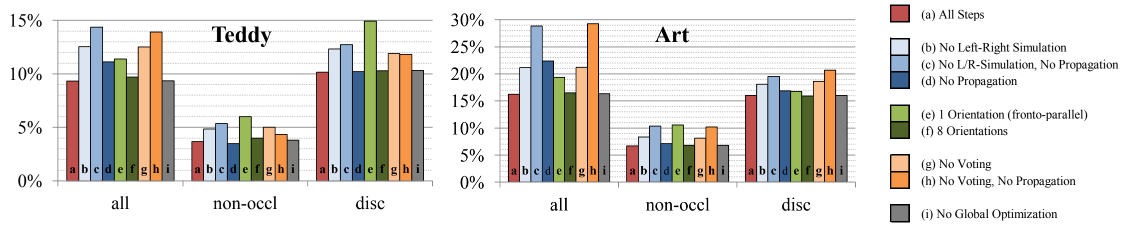


Figure 5.4. The charts display the effect of disabling processing steps of our method. Error bars show percentages of disparities that differ by more than 1 from the ground truth in the whole image (all), non-occluded pixels (non-occl) and regions near discontinuities (disc).

Very small values of σ_C lead to high errors in non-occluded regions because the walks are then very sensitive to image noise. Especially at *Tsukuba*, there are some vertical artifacts present in the images, presumably a result from a Bayer-pattern, which seem to lead to higher errors for small values of N and σ_C. Larger values of σ_C gradually increase the error because it is more likely that walks cross object boundaries. In practice, good values for σ_C and N, which minimize errors in non-occluded regions, can be obtained relatively efficiently in an iterative manner. With a dense discrete parameter exploration we found that the trends described above are still valid for other parameter combinations.

5.3.2. Method Analysis

In Fig. 5.4, we analyze the influence of the different processing steps of our method on the quality in different image regions. The red bars (a) show the performance of the full method with all processing steps.

The blue-colored bars (b-d) show the impact of the simulation in left and right images and of the propagation. The left-right simulation helps in most parts of the image because some false matches in regions near discontinuities are avoided. Due to the voting, both occluded and non-occluded regions benefit from that. The propagation clearly improves the occluded regions by comparing (a) and (d), but may also slightly degrade non-occluded areas because occasionally false matches are diffused into the neighborhood.

The green-colored bars (e-f) show the influence of the a priori surface orientations. For (e) we used only a fronto-parallel prior $\Delta_1 = \{(0,0)^T\}$, for (a) 4 orientations $\Delta_4 = \Delta_1 \cup \{(0,1)^T, (\pm\frac{1}{2},0)^T\}$, and for (f) 8 orientations $\Delta_8 = \Delta_4 \cup \{(\pm\frac{1}{3},0)^T, (\pm\frac{3}{4},0)^T\}$. It is clearly visible that the addition of only a few orientations using Δ_4 results in a huge improvement in performance on these datasets. The negative side-effects of adding more orientations using Δ_8 is surprisingly low, since we would have expected a larger degradation due to a higher matching ambiguity. However, there is nearly no improvement of using eight orientations at these datasets, mainly because the disparity gradients on surfaces are relative small at these datasets and thus, fewer orientations suffice. At the dataset *Flowerpots* of Fig. 5.5 larger gradients occur and in non-occluded regions we measured an error of 7.8%, 6.5% and 4.8% for 1, 4 and 8 orientations respectively. These experiments also provide some evidence for our assumption that random walks usually cover only a small region and thus, perspective distortions have to be considered only for larger disparity gradients.

The orange-colored bars (g-h) display the effect of the voting technique. In this case, the data term was initialized using $E_D(\mathcal{D}) = \sum_{\mathbf{x}} \min_\delta C_A(\mathbf{x}, \mathcal{D}(\mathbf{x}), \delta)$. For (g) we only disabled the voting but left the propagation enabled and for (h) we disabled both voting and propagation. When comparing (g) to (h) it is noticeable that for *Teddy* in non-occluded regions the propagation reduced the quality, which was due to a false match which was propagated into the neighborhood. But also here the picture is clear that the propagation mainly improves in occlusions. The impact of voting is best measured by comparing (h) to (d) because in both cases no propagation is performed. From that, a considerable influence on the quality can be observed in all image regions. This can be explained by the random walks: if a random walk covers occluded and non-occluded pixels, all of them will receive support for the correct depth if the true disparity and orientation is contained in the set S.

The gray-colored bars (i) show the influence of global energy minimization. In our experiments we measured the best performance when using a relatively small strength of regularization, which might explain the small influence.

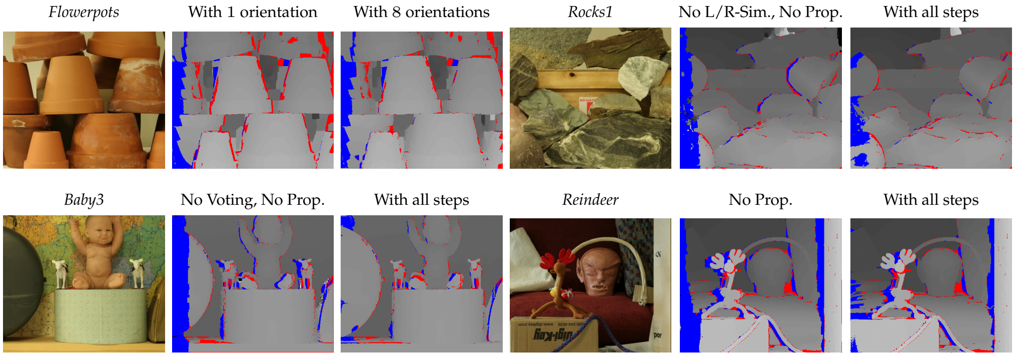


Figure 5.5. A qualitative comparison to visualize the effect of disabling processing steps of our method. For each dataset we show the left image, a disparity map where specific steps of the algorithm were disabled and a disparity map of the full method. Blue and red pixels are wrong disparities in occluded and non-occluded regions respectively (*i.e.* the disparity error is greater than one). We processed *Flowerpots* with one and eight a priori surface orientations. At *Rocks1* we disabled the simulation in left and right images (sec. 5.2.3) and the propagation of sec. 5.2.6. At *Reindeer* we disabled the voting and at *Baby3* the propagation additionally.

In Fig. 5.5, we give a qualitative impression using difficult Middlebury datasets which underline the previous observations. In the supplementary material we provide more elaborate results which show that our method produces quite robust results with the same parameter set for all images. In general, our method has some limitations in textureless regions, however, in most cases ambiguities are handled correctly by belief propagation and the random walk based propagation.

5.3.3. Comparison to Other Methods

Fig. 5.6 and Tab. 5.1 show the results for the standard Middlebury stereo pairs. From our perspective it is very important to underline that our method performs very well for smaller values of the threshold for the absolute disparity error. With a threshold of 1.0 we currently achieve the 23rd place on the Middlebury evaluation, and with values of 0.75 and 0.5 we reach the 2nd and 6th place respectively.

We also compared to results of [156]. On the datasets *Art*, *Bowling2*, *Flowerpots*, *Reindeer* and *Rocks2* the authors achieved in non-occluded areas errors of 15.3%, 14.6%, 20.1%, 10.4% and 4.8%, whereas our method performed better with 7.0%, 4.6%, 6.5%, 4.1% and 1.9%. Details can be found in the supplemental material.

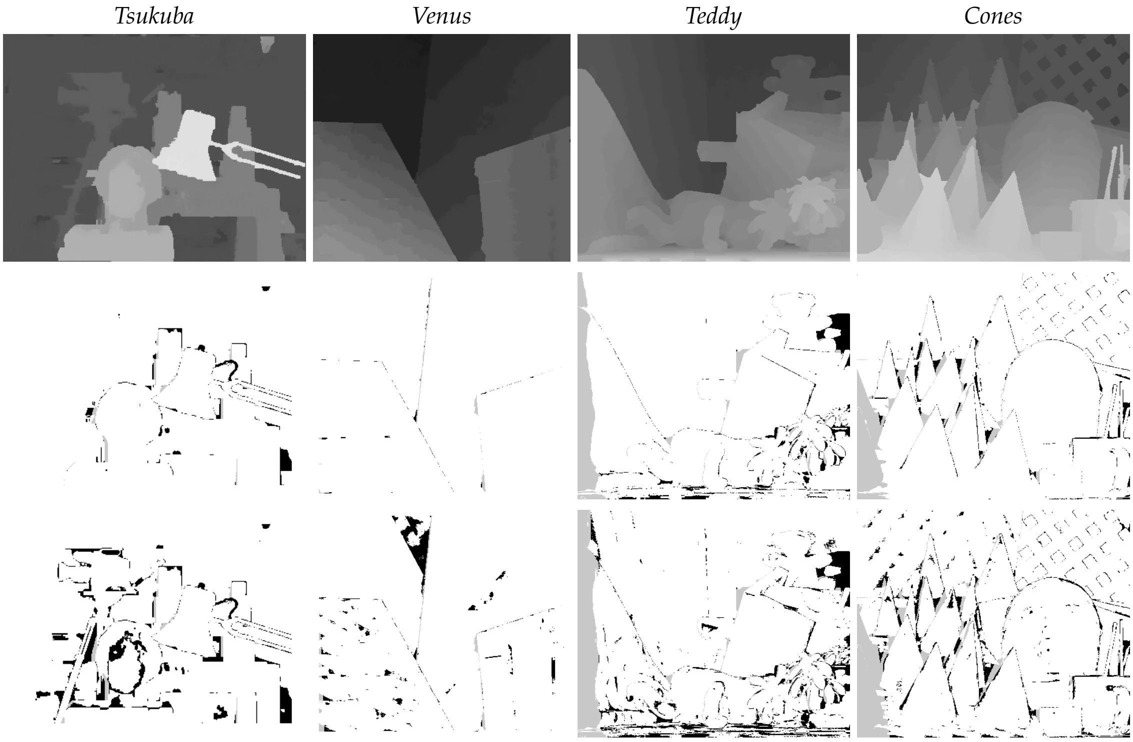


Figure 5.6. The disparities and bad pixels of our proposal for the standard Middlebury images [119]. The rows show the disparity map of our method and the bad pixels for the disparity error thresholds 1.0 (middle row) and 0.5 (last row). White pixels denote correct disparities, black and grey pixels incorrect ones (*i.e.* the disparity difference is greater than the threshold) in non-occluded and occluded regions, respectively.

5.3.4. Real World Sequences

We tested our method also on real world sequences from a moving vehicle. Fig. 5.7 shows the rectified camera frame and the disparity maps obtained using our method based on random walks. We also show disparity maps of our real-time method presented in section 4.4. The disparity maps are shown as grayscale images and as color overlays (warmer colors indicate higher disparities).

Table 5.1. The performance of our method at the Middlebury benchmark [119]. At the time of writing this thesis, we achieved places 6, 2 and 23 for error thresholds 0.5, 0.75 and 1.0 respectively.

Algorithm	Error Thresh.	Rank	Avg. Error	Tsukuba			Venus			Teddy			Cones		
				nocc	all	disc	nocc	all	disc	nocc	all	disc	nocc	all	disc
Our Method	0.5	6	9.68	7.20	7.94	13.9	2.84	3.51	9.04	7.71	14.8	17.6	5.81	11.9	13.9
Bleyer *et al.* [15]	0.5	8	9.91	15.0	15.4	20.3	1.00	1.34	7.75	5.66	11.8	16.5	3.80	10.2	10.2
Bleyer *et al.* [16]	0.5	23	13.1	22.8	23.1	18.7	3.70	3.85	8.79	10.6	13.8	22.6	5.47	11.0	13.4
Hosni *et al.* [66]	0.5	52	15.9	22.9	23.1	20.4	6.81	7.11	11.5	13.5	20.4	26.8	8.17	15.0	15.5
Wang and Yang [156]	0.5	71	17.5	22.5	23.9	18.5	6.81	7.41	10.7	16.1	22.2	30.6	12.2	18.3	20.1
Our Method	0.75	2	6.66	5.93	6.70	13.1	0.31	0.85	3.17	4.80	11.4	11.7	3.33	9.20	9.45
Bleyer *et al.* [15]	0.75	12	8.39	15.0	15.4	20.3	0.41	0.64	4.38	3.84	9.48	12.0	2.81	8.55	7.95
Bleyer *et al.* [16]	0.75	19	9.87	22.8	23.1	18.7	0.42	0.55	3.87	5.62	8.02	13.5	3.48	8.73	9.74
Hosni *et al.* [66]	0.75	44	11.5	22.9	23.1	20.4	0.67	0.89	3.57	8.36	15.1	19.3	3.73	10.1	9.86
Wang and Yang [156]	0.75	59	12.0	22.5	23.9	18.5	0.66	1.18	4.30	8.38	14.3	19.9	5.62	11.7	12.8
Bleyer *et al.* [16]	1.0	9	4.06	1.28	1.65	6.78	0.19	0.28	2.61	3.12	5.10	8.65	2.89	7.95	8.26
Bleyer *et al.* [15]	1.0	17	4.59	2.09	2.33	9.31	0.21	0.39	2.62	2.99	8.16	9.62	2.47	7.80	7.11
Our Method	1.0	23	4.83	2.02	2.77	8.58	0.21	0.68	2.31	3.87	9.47	9.34	2.67	8.28	7.74
Wang and Yang [156]	1.0	30	5.60	0.87	2.54	4.69	0.16	0.53	2.22	6.44	11.5	16.2	3.59	9.49	8.95
Hosni *et al.* [66]	1.0	33	5.80	1.45	1.83	7.71	0.14	0.26	1.90	6.88	13.2	16.1	2.94	8.89	8.32

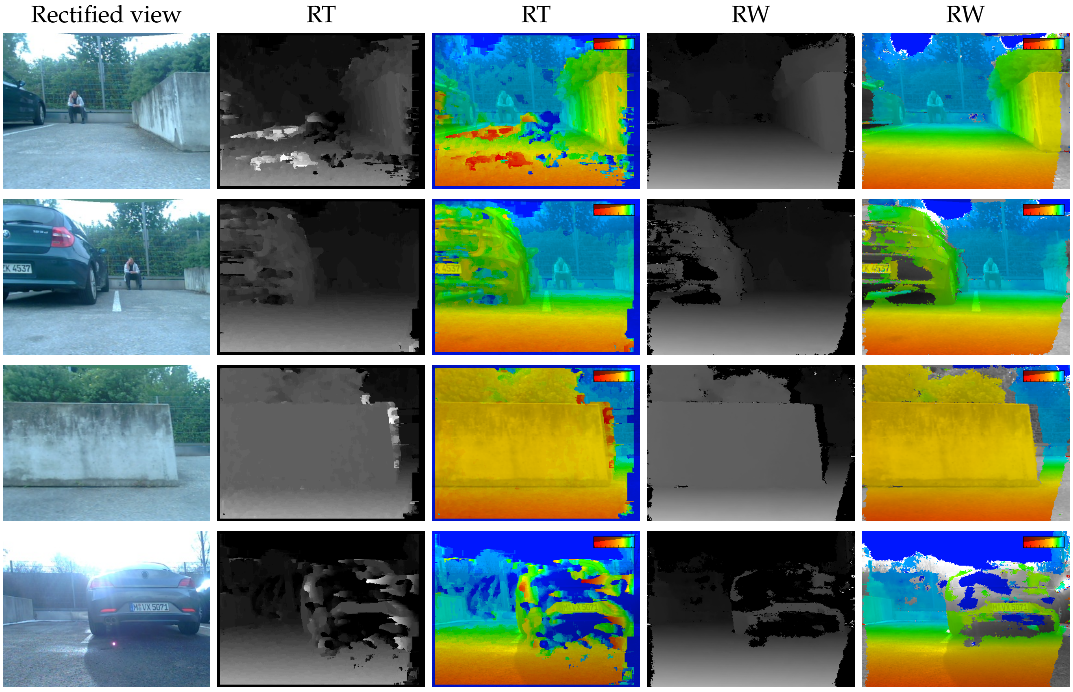


Figure 5.7. Our method applied to sequences from our vehicle using our method based on random walks (RW) and our real-time stereo method (RT). The disparity maps are shown as grayscale images and as color overlays (warmer colors indicate higher disparities).

5.4. Discussion

In this chapter, we introduced a novel approach for accurate stereo matching. In particular, we propose to use simulations of random walks for stereo vision. We make the matching process systematically robust to challenging problems like discontinuities, occlusions and slanted surfaces. This is mainly achieved by using random walks as matching primitives because they, in some sense, perform a localized soft segmentation. Further, we introduce a few a priori surface orientations for cost aggregation to handle slanted surfaces and by using left-right random walk simulations we increase robustness in occluded regions. Our novel voting strategy increases the general robustness in all image regions. Finally, we perform a propagation of confident disparities into inconsistent regions and use global optimization on a probability distribution over disparities to handle ambiguities. We demonstrated experimentally that these measures lead to very reliable and very accurate disparity maps which is strengthened by achieving the 2nd place at the Middlebury benchmark.

In practice, our proposed method works very well on many difficult images, however, the biggest challenge for our method are currently homogeneous regions. In these cases, the voting space is not able to aggregate meaningful votes and the resulting votes are not very useful. To reduce the ambiguity, we introduced global optimization, but it does not resolve successfully in all situations. The main reason for this is the conflict of goals between the preservation of small image details and the regularization in homogeneous regions.

6. Probabilistic Stereo Fusion

> *If you want to inspire confidence, give plenty of statistics. It does not matter that they should be accurate, or even intelligible, as long as there is enough of them.*
>
> LEWIS CARROLL

In this chapter we present the second, important component of our motion-stereo pipeline which directly uses the disparity maps of the binocular stereo matching. The main motivation are the findings of the previous chapters where we evaluated that even very sophisticated global stereo methods do not perform very well in occluded regions or in areas near discontinuities. Also robustness in adverse vision conditions is difficult to achieve using binocular methods, because in some situations with glare lights the imagers are often physically limited in local image regions. The positive aspect of our motion-stereo application is that almost all scene points are observed multiple times when passing by.

Thus, to increase the robustness and the accuracy of our depth sensing, we chose to exploit the redundancy contained in disparity maps computed at different viewpoints. Such methods are also known as *multi-view stereo* approaches and although a large amount of research has been devoted to the stereo problem using multiple cameras [29, 85, 106, 116, 121, 133], obtaining dense high-quality depth maps in real-time is still a challenging problem. A few multi-view stereo methods [100, 170] may achieve real-time performance, but only by using the enormous processing power of graphics cards. But such hardware is not available on our platform and therefore it is absolutely necessary that all calculations can be performed in real-time on a standard mobile CPU at video frame rate.

Although we are particularly interested in parking assistance using motion-stereo, our proposed solution is generic and can be applied to any video with known camera motion. In Fig. 6.1 we show an example where the camera is mounted laterally on a vehicle. The disparity maps can be computed from consecutive image frames over time when the vehicle moves. From these disparity maps, we build a model of the environment, in order to avoid collisions or to find lateral parking space. Even at higher velocities, the disparity maps exhibit a large overlap and thus depth information is highly redundant. At the same time, due to the real-time stereo method used, disparities are very error prone. The question is how to fuse all those disparity maps to improve the accuracy of the disparity map defined by a reference image pair, for example, the last two images in case of motion-stereo.

The problem of fusing disparity maps was addressed by [100, 170, 173] in order to produce either dense disparity maps of the scene from video [173], or to do a dense surface reconstruction [100, 170]. These methods can either globally fuse disparity maps obtained from different views or locally fuse them from the overlapping views. We restrict ourselves to the local fusion of the disparities among overlapping views, because it is more typical for our application. Since the methods of Merrell *et al.* [100] and Zhang *et al.* [173] currently provide impressive results, we extensively compare our method to theirs. However, both

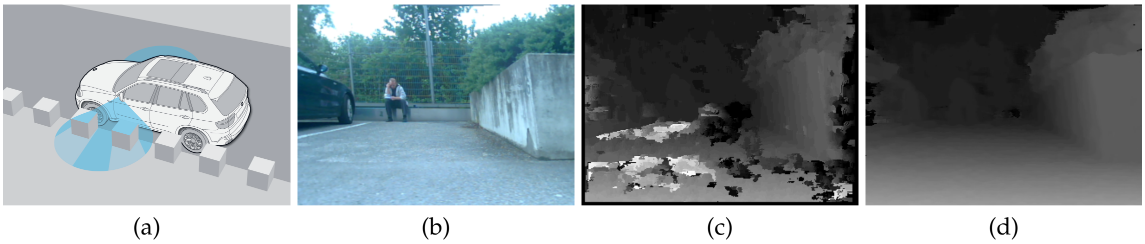


Figure 6.1. Real-Time motion-stereo for automotive driver assistance. When the vehicle moves, depth is inferred via motion-stereo. (a) A camera mounted on the side of the vehicle observes the lateral space. (b) One frame captured by the side-view camera. (c) A disparity map obtained by pairwise real-time stereo matching. (d) The result of our proposed fusion method which removes outliers and improves the quality of the disparity maps.

methods are not real-time with our hardware: [100] requires a GPU to be real-time and [173] performs expensive energy minimization with belief propagation which is far from real-time.

In this chapter, we assume that a set of disparity maps is available and that they were computed using any available short baseline stereo technique. Then, given any other reference view pair we propose a novel *probabilistic disparity fusion* method to produce an accurate disparity map of the given reference view pair by fusing all available disparity maps. We first project all disparity maps to the reference view pair. After maintaining visibility constraints, we estimate a probability density function over all valid disparities in the reference view using uncertainties of these reprojections and their photo-consistencies. Finally, this allows us to select the most probable disparity map from this distribution. In addition we model occlusions which produce holes in the reprojected disparity maps and define reliable areas by checking visibility in the reference and input views. This contributes to the overall statistics of the disparity and provides better pdf estimation.

We tested our method on the challenging datasets of Middlebury [119] and compared it to the fusion methods of [100] and [173]. The experiments show that our technique is very robust and that the quality is significantly improved, especially in occluded regions and at discontinuities. We also show results on real-world sequences acquired from a camera attached to a vehicle. A very important fact is that our method allows real-time operation on the CPU without dedicated hardware.

In the remainder we will first review the state of the art, then present our probabilistic method and finally show an exhaustive experimental evaluation.

6.1. Related Methods

In recent years, multi-view stereo methods have been extensively studied and tested using the available benchmarks [121, 129]. While resulting in a large amount of excellent results, little attention has been spent on computational performance. However, when that was the case and real-time stereo methods were proposed [61, 138], the reconstruction quality was significantly decreasing. While multi-view stereo approaches introduce assumptions on shape priors and use robust photo-consistency measures, there are others which aim to produce consistent disparity maps [47, 85, 100, 133, 173, 178]. In many cases, disparity maps that are produced locally using a number of overlapping views are later fused into either a global disparity video [173], or a full 3D model [100, 170]. Again, the vast majority of works aim at high quality reconstructions of single objects and only very few try to minimize the computational overhead.

Some works try to enforce temporal depth consistency using various smoothing approaches. For example, Bleyer and Gelautz [13] collect a number of depth values for every fixed pixel location (ignoring the 3D geometry) and use in this sense temporal median filtering. In other approaches [89, 91] temporal smoothness is formulated as a global energy functional.

Since the main motivation of our work comes from motion-stereo we tend to fuse locally overlapping disparity maps and do not aim to produce full 3D models. Works of Merrell *et al*. [100] and Zhang *et al*. [173], which explicitly deal with fusion of the disparity maps, are thus directly related to our approach.

Merrell *et al*. [100] compute depth maps between neighboring views and fuse this information based on the *stability* of every depth. In order to keep track of occlusions, the stability is determined for every depth hypothesis and is defined by counting occlusions in the reference and other views. A valid depth is defined as the first depth hypothesis which is stable. However outliers affect the stability and such hard decisions may produce incorrect depth estimates. Further, the computational complexity grows quadratically with the number of disparity maps and in practice real-time operation is only possible with GPU hardware. In our paper, we overcome these problems. Our probabilistic approach employs reprojection uncertainties, handles outliers robustly and depth-accuracy gets improved compared to this approach.

Zhang *et al*. [173] impressively generalized the fusion problem by formulating it as an energy minimization problem. In their *bundle optimization* framework all disparity maps are optimized iteratively using belief propagation. In contrast to Merrell *et al*. [100] they do not model occlusions or visibility constraints explicitly. In their work these constraints are handled by the simultaneous use of *geometric coherence* and *color-similarity* as well as the regularization of belief propagation. The minimization of the energy functional is in practice very time consuming and thus, this method is not an option for mobile real-time applications.

Koch *et al*. [80] introduced the efficient *correspondence linking algorithm*: by *chaining* correspondences across many views outliers are rejected and accuracy is improved. However, no solution was provided for multiple disparity maps per view and disparities in occluded regions or outliers near the beginning of the chain are problematic.

Finally the method of Zach [170], which fuses multiple depth maps to obtain a full volumetric 3D reconstruction, was formulated as a relatively efficient method that uses the GPU and produces very good results. However, the hardware requirements are too high and the volumetric representation is problematic for our application.

Compared to other fusion methods, our work focuses on real-time performance, but also offers high quality depth maps. This is demonstrated through exhaustive experimentation and comparison to prior art where we obtained better depth maps, especially in occluded and discontinuity areas.

6.2. Probabilistic Stereo Fusion

The major problem in motion and multi-view stereo are occlusions and discontinuities. Here we consider a reference view pair (RVP) in which we want to improve disparities, especially in occluded and discontinuity areas by bringing the information from other view pairs to this view. For this we propose to compute a probability density function defining the probability of the disparities in the reference image. It is done from the re-projection of all disparity maps of all available view pairs to this RVP. This allows us to select the most probable disparity at certain pixel locations of the RVP. Since the probability density function (pdf) is sampled from a relatively large number of measurements coming from other view pairs reprojected to the RVP, we demonstrate in the results section, that our approach significantly improves disparities at occlusions and areas near discontinuities.

6.2.1. Left-Right and Right-Left Distinction

We assume that there is a set of N input disparity maps between pairs of views. Disparity maps between two, pairwise rectified views $\mathcal{I}_A$ and $\mathcal{I}_B$ are denoted by $\mathcal{D}_{A,B}$ (using left-right stereo-matching) and $\mathcal{D}_{B,A}$ (using right-left matching). In the rest of the section we will discuss the computation of only one pdf for left-right disparity maps $\mathcal{D}_{A,B}$. The computation of the pdf for right-left disparity maps is identical. The intuition behind this is that in most stereo methods left object boundaries are usually very stable when performing right to left matching (because no occluded pixels are present there in the right image). The same can be said for right object boundaries and left to right matching. This implies that the left-right consistency check only removes information and can introduce errors at occluded areas around discontinuities. To avoid loss of information, we instead directly use unfiltered disparity maps to compute two separate pdfs and combine the information later.

6.2.2. Goal

Our goal is to compute an improved disparity map $\hat{\mathcal{D}} = \hat{\mathcal{D}}_{R_1,R_2}$ for a given RVP $(\mathcal{I}_{R_1}, \mathcal{I}_{R_2})$. To do this we transfer disparities from all input disparity maps (*e.g.* $\mathcal{D}_{0,1}$, $\mathcal{D}_{2,3}$) to the RVP $(\mathcal{I}_{R_1}, \mathcal{I}_{R_2})$. A simple triangulation and projection is sufficient [173] to perform this transfer. Independent from the transfer method used, we refer to it using the transfer function $\Theta_k^{A,B} : (\mathbf{x}_A, d_{A,B}) \mapsto \mathbf{x}_k$, which transfers a point $\mathbf{x}_A$ using input disparity $d_{A,B} = \mathcal{D}_{A,B}(\mathbf{x}_A)$ into view $\mathcal{I}_k$. So, we use functions $\Theta_{R_1}^{A,B}$ and $\Theta_{R_2}^{A,B}$ to compute a *reprojected* disparity map $\tilde{\mathcal{D}}_{A,B}$ by applying the transfer to every disparity in $\mathcal{D}_{A,B}$: $\tilde{\mathcal{D}}_{A,B}(\mathbf{x}_{R_1}) = \Theta_{R_1}^{A,B}(x_A, \mathcal{D}_{A,B}(x_A)) - \Theta_{R_2}^{A,B}(x_A, \mathcal{D}_{A,B}(x_A)) = \mathbf{x}_{R_1} - \mathbf{x}_{R_2}$.

In practice, all available disparity maps are transferred to the RVP. Later, all these disparity maps are used to compute the pdf of the disparities in the RVP. From this pdf, the most probable values are extracted and they define the improved disparity map $\hat{\mathcal{D}}_{R_1,R_2}$.

6.2.3. Handling Occlusions

When performing the reprojection, depending on the occlusions and discontinuities in $\mathcal{D}_{A,B}$, there are in general zero, one or even multiple disparity estimates for every pixel

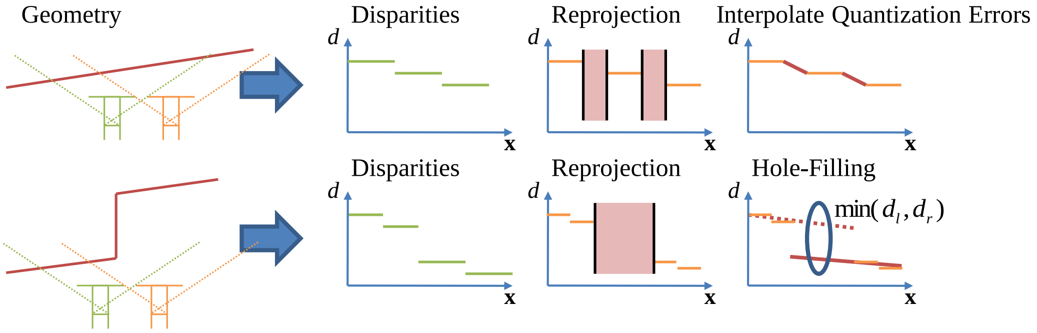


Figure 6.2. Holes as a consequence of reprojection. In these two cases, a 2D-scene is observed by two different stereo cameras. In the first row, errors from disparity quantization are interpolated and in the second row, holes at depth discontinuities are filled using extrapolation.

of a reprojected disparity map. In an ideal world, the case with only one disparity occurs when cameras of the reference and input views observe only non-occluded scene points. Multiple disparities occur due to depth discontinuities where several input disparities of different scene surfaces reproject to the same location in the reference view with different disparities. However, in our method we must make sure that there is only one disparity per pixel and thus we choose the closest depth estimate (*i.e.* maximal disparity) from these values to maintain correct visibility. Zero disparities can occur due to disparity quantization in the input view or due to occlusions on surface discontinuities. It means that at a particular pixel location in the reprojected disparity map there is no information about the disparity, resulting in holes in the disparity map. We eliminate those holes by filling them with approximated values based on surrounding disparities. This is important to be done because it improves the estimation of the probability density function in many cases. Below we discuss proposed solutions for the hole-filling. In addition it is very important to check whether close parts of the scene are visible in the input views: we have to ensure that background disparities are used for the pdf, if and only if the input disparity map may contain information about the presence of foreground disparities (*i.e.* if close parts of the scene are visible in both cameras). We perform this check (the *reliable area*) using the maximum disparity (the closest possible depth) and invalidate areas which are not visible in both, reference and input view pairs. If we did not perform this check, background disparities would receive too much support and chances would be high that background is visible in spite of the presence of foreground objects.

Holes from Disparity Quantization

In the first row of Fig. 6.2 we show an example where missing disparities in the reprojected disparity map are artifacts of the disparity quantization. This usually happens on slanted surfaces or on discontinuities. In Fig. 6.2 we used an example of a slanted surface. The discontinuity of the input disparity map shown in the first row of Fig. 6.2 is an artifact of disparity quantization. When reprojected to the reference view, holes are created whose sizes vary depending on the camera motion between the reference and input frames. We detect and interpolate those holes by checking if the difference of the left and right neighboring disparities is less than a small threshold. In practice, we set this value to two and interpolate holes smaller than five pixels.

Holes at Occlusions

The remaining holes occur at depth discontinuities. In the second row of Fig. 6.2 we show an example of an occlusion where part of the surface is visible in the RVP, but is not visible in the input view pair. The reprojected disparities near the discontinuity (*i.e.* the occluded area) will create a hole at that place. To fill this hole we chose to extrapolate the left and right neighboring surfaces. To avoid using all left and right disparities, which can belong to multiple objects in the scene, we segment those disparities and take only those that potentially create the same surface. The interpolation is done by linear regression, *i.e.* we fit the line to segmented left and right disparities as shown in Fig. 6.2. At a specific point $\mathbf{x}$, where the disparity is missing, the extrapolation gives two estimated disparities d_l and d_r and we use the background (the occluded surface) as the final disparity, *i.e.* $\min(d_l, d_r)$.

Reliable Area

We must ensure that every reprojected disparity comes from the surface visible in both, reference and input camera pair. If that is not the case it means that the point corresponding to this disparity is potentially occluded or not visible in the reference view. To check this we verify if a point on the surface defined by the maximum disparity is outside the frustum of $\mathcal{I}_A$ and $\mathcal{I}_B$. In practice, for every point $\mathbf{x}_{R_1} \in \mathcal{I}_{R_1}$ we compute $\mathbf{x}_k = \Theta_k^{R_1,R_2}(\mathbf{x}_{R_1}, d_{\max})$ and check if $\mathbf{x}_k \in \mathcal{I}_k$ for $k \in \{A, B\}$. If $\mathbf{x}_k \notin \mathcal{I}_k$, then the disparity at $\mathbf{x}_{R_1}$ is invalidated, meaning that either it is occluded in the reference view or it is not visible in the input view. Here, $d_{\max}$ is the maximum disparity of view pair $\mathcal{I}_{R_1}$ and $\mathcal{I}_{R_2}$.

6.2.4. Probability Density Function of Disparity

We reproject all input disparities to the RVP and use them as measurements to compute a probability density function of the disparity in the reference image. Later we draw from this pdf the most probable disparity at every pixel location of the reference view as illustrated in Fig. 6.3.

First we build the set $\mathcal{S}$ of reprojected disparity maps by reprojecting all N input disparity maps to the RVP. Now we use these disparity maps as measurements to sample the pdf of disparity d at every given pixel location $\mathbf{x}$ in the reference image. The unknown pdf p can be modeled as:

$$p(\mathbf{x}, d) = \sum\nolimits_{\tilde{\mathbf{x}} \in \mathcal{I}_{R_1}} \sum\nolimits_{\tilde{d} \in \mathcal{S}(\tilde{\mathbf{x}})} p(\mathbf{x}, d \,|\, \tilde{\mathbf{x}}, \tilde{d})\, p(\tilde{\mathbf{x}})\, p(\tilde{d}) \tag{6.1}$$

where $p(\mathbf{x}, d|\tilde{\mathbf{x}}, \tilde{d})$ is the joint probability of disparity d at pixel location $\mathbf{x}$ given a measurement $\tilde{d} \in \mathcal{S}(\mathbf{x})$ at measured location $\tilde{\mathbf{x}}$ of a reprojected disparity map in $\mathcal{S}$. We assume that all the measurements of locations and disparities are equally probable. Therefore we

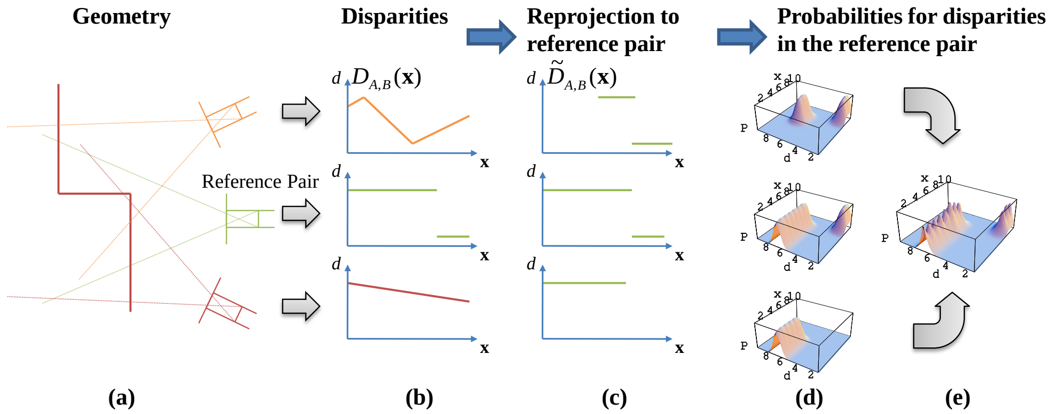

Geometry
Disparities
Reprojection to reference pair
Probabilities for disparities in the reference pair
Reference Pair
$D_{A,B}(\mathbf{x})$
$\tilde{D}_{A,B}(\mathbf{x})$
d
x
P
(a)
(b)
(c)
(d)
(e)

Figure 6.3. The pdf estimation: (a) The 2D-geometry is observed from three different stereo cameras. (b) Disparity maps from input stereo pairs are determined. (c) The reprojected disparity maps to the reference view pair lead to (d) three pdfs for each reprojected disparity map which are finally (e) combined in one pdf.

consider them constant and write after marginalization:

$$p(\mathbf{x}, d) \sim \sum\nolimits_{\tilde{\mathbf{x}} \in \mathcal{I}_{R_1}} \sum\nolimits_{\tilde{d} \in \mathcal{S}(\tilde{\mathbf{x}})} p(\mathbf{x}, d \,|\, \tilde{\mathbf{x}}, \tilde{d}) \tag{6.2}$$

The probability $p(d, \mathbf{x} \,|\, \tilde{\mathbf{x}}, \tilde{d})$ depends on the reprojection uncertainty defined by the probability $p_L(\mathbf{x}, d \,|\, \tilde{\mathbf{x}}, \tilde{d})$ that the scene point $\mathbf{X}(\tilde{\mathbf{x}}, \tilde{d})$ (computed from the uncertain correspondence $\mathbf{x}_A \leftrightarrow \mathbf{x}_B$ before reprojection) projects to the location $\mathbf{x}$ in the image $\mathcal{I}_{R_1}$. It further depends also on the probability $p_D(\mathbf{x}, d \,|\, \tilde{\mathbf{x}}, \tilde{d})$ that the disparity of $\mathbf{X}$ is d in the RVP. So we write it as:

$$p(\mathbf{x}, d \,|\, \tilde{\mathbf{x}}, \tilde{d}) = p_L(\mathbf{x}, d \,|\, \tilde{\mathbf{x}}, \tilde{d}) \cdot p_D(\mathbf{x}, d \,|\, \tilde{\mathbf{x}}, \tilde{d}) \cdot p_C(\tilde{\mathbf{x}}, \tilde{d}) \tag{6.3}$$

These uncertainties are naturally coming from the input image pairs and can be directly estimated there. In the following, we use the transfer function Θ to relate the uncertainties to the RVP. The location uncertainty at pixel position $\mathbf{x}$ is measured by the discrepancy between the true location $\mathbf{x}_A = \Theta_A^{R_1,R_2}(\mathbf{x}, d)$ and the measured location $\tilde{\mathbf{x}}_A = \Theta_A^{R_1,R_2}(\tilde{\mathbf{x}}, \tilde{d})$ in the input image obtained by back-projections of the true $\mathbf{x}$ and measured $\tilde{\mathbf{x}}$ locations from the reference image. Thus, p_L has its maximum value when the true $\mathbf{x}_A$ and measured $\tilde{\mathbf{x}}_A$ back-projections coincide and it decreases with increasing distance. So we use:

$$p_L(\mathbf{x}, d|\tilde{\mathbf{x}}, \tilde{d}) \sim \exp\left(-\frac{1}{2\sigma_x^2}\left\|\Theta_A^{R_1,R_2}(\mathbf{x}, d) - \Theta_A^{R_1,R_2}(\tilde{\mathbf{x}}, \tilde{d})\right\|_2^2\right) \tag{6.4}$$

Similarly, p_D is maximal at $\tilde{d}$ and decreases for differing depths:

$$p_D(\mathbf{x}, d|\tilde{\mathbf{x}}, \tilde{d}) \sim \exp\left(-\frac{1}{2\sigma_d^2}\left\|(\Theta_B(\tilde{\mathbf{x}}, \tilde{d}) - \Theta_A(\tilde{\mathbf{x}}, \tilde{d})) - (\Theta_B(\mathbf{x}, d) - \Theta_A(\mathbf{x}, d))\right\|_2^2\right) \tag{6.5}$$

where $\Theta_A = \Theta_A^{R_1,R_2}$, $\Theta_B = \Theta_B^{R_1,R_2}$ and σ_x is the location uncertainty defined by pixelwise sampling and σ_d is the accuracy of the disparity estimation. Note that $\tilde{\mathbf{x}}$ and $\tilde{d}$ are taken from the set of reprojected disparity maps. If the disparities d and $\tilde{d}$ are the same, the point defined by $(\mathbf{x}, d)$ will project to exactly the same input locations $\tilde{\mathbf{x}}_A$ and $\tilde{\mathbf{x}}_B$ and define the same disparity in the input view, which will result in the maximum value. Otherwise, points with different disparities or locations will back-project to locations away from the measurement $(\tilde{\mathbf{x}}, \tilde{d})$ and get lower values.

The color-similarity measure between $\mathbf{x}_A$ and $\mathbf{x}_B$ is given as in [173]:

$$p_C(\tilde{\mathbf{x}}, \tilde{d}) \sim \frac{\sigma_C}{\sigma_C + |\mathcal{I}_A(\tilde{\mathbf{x}}_A) - \mathcal{I}_B(\tilde{\mathbf{x}}_B)|} \tag{6.6}$$

where σ_C is the color variance which we obtained experimentally. The value of p_C is 1 for points with identical intensities and decreases as a function of the color dissimilarity.

6.2.5. Disparity Selection

Finally we estimate the most probable disparity map from the estimated pdf. Assuming that image positions $\mathbf{x}$ are equiprobable and with $p(\mathbf{x}, d) = p(d \,|\, \mathbf{x}) p(\mathbf{x})$ we get:

$$\hat{d} = \mathrm{argmax}_d \, p(d \,|\, \mathbf{x}) = \mathrm{argmax}_d \, p(\mathbf{x}, d) \tag{6.7}$$

We compute two different probability density functions p_l and p_r each corresponding to the reprojection of left-right $\mathcal{D}_{A,B}$ and right-left disparity maps $\mathcal{D}_{B,A}$. In our experiments we found out, that the final disparity should be defined as $\hat{d} = \min(\hat{d}_l, \hat{d}_r)$, where $\hat{d}_l$ and $\hat{d}_r$ are obtained from p_l and p_r. When determining $\hat{d}$, using the $\max$-function instead of the $\min$-function appears to significantly degrade the final disparity map (see Fig. 6.6). This is due to the bad performance of most stereo methods at object boundaries (regions near discontinuities). Distinguishing between left-right and right-left matching will separate good from bad left-boundaries (and bad from good right-boundaries). In ambiguous situations, the $\min$-function will favour the background (which is usually occluded).

6.2.6. Aligned Cameras

In the special case of linearly aligned cameras, the transfer function Θ is a linear relationship depending on the baselines between the views:

$$\mathbf{x}_k = \Theta_k^{A,B}(\mathbf{x}_A, d_{A,B}) = \mathbf{x}_A + \lambda_{A,B,k} \cdot \mathcal{D}_{A,B}(\mathbf{x}_A) \tag{6.8}$$

with

$$\lambda_{A,B,k} = \frac{B_{k,A}}{B_{B,A}} \tag{6.9}$$

where $B_{i,j}$ is the signed baseline between views $\mathcal{I}_i$ and $\mathcal{I}_j$, *i.e.* $B_{i,j} = -B_{j,i}$. In practice, these factors may be numerically estimated using (6.9), even if the cameras are not perfectly aligned (for example, at motion-stereo).

As described in the previous section, we propose to add two disparity maps per input view-pair: namely $\mathcal{D}_{A,B}$ **and** $\mathcal{D}_{B,A}$. In practice, we first split the set of disparity estimates

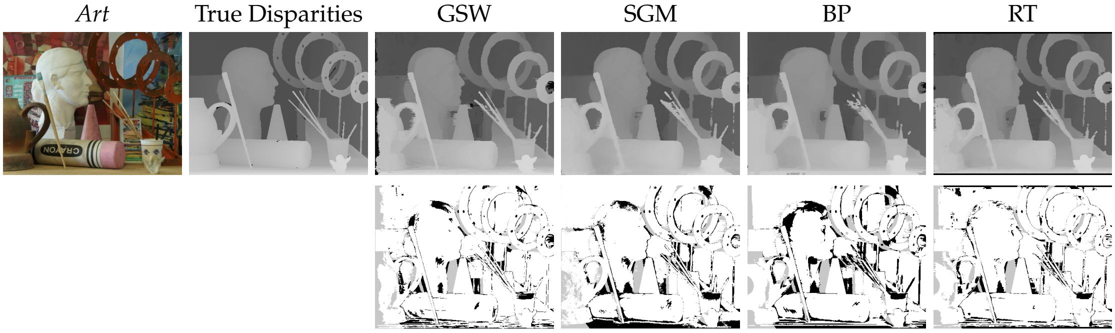


Figure 6.4. The disparities and bad pixels of our fusion method when using different stereo algorithms on the dataset *Art*.

in $\mathcal{S}(\mathbf{x})$ into two disjunct sets $\mathcal{S}_l(\mathbf{x})$ and $\mathcal{S}_r(\mathbf{x})$, depending on the factors defined in (6.9). We define

$$\mu_{A,B} := \lambda_{A,B,R_2} - \lambda_{A,B,R_1} = -\mu_{B,A} \tag{6.10}$$

and distribute the values in $\mathcal{S}(\mathbf{x})$ to $\mathcal{S}_l$ if $\mu_{A,B} < 0$ and to $\mathcal{S}_r$ if $\mu_{A,B} > 0$. We then apply the probabilistic fusion efficiently using the two different probability density functions p_l and p_r as described.

While the use of these factors is only viable for special camera configurations, a simple approximation of μ is possible, if an ordering on the cameras exists:

$$\mu_{A,B} = \begin{cases} -1 & A \text{ left of } B \\ 1 & B \text{ left of } A \end{cases} \tag{6.11}$$

6.3. Results

We evaluate our method using classical stereo datasets with ground truth from Middlebury [119] and demonstrate the applicability to real world data. In our experiments we used $\sigma_d = 1$, $\sigma_x = 1$ and $\sigma_C = 5$. The standard two-frame stereo datasets from Middlebury [119] contain up to 9 images from which we computed 72 (*Venus*, *Teddy*, *Cones*) or 42 (*Art*, *Moebius*, *Aloe*) disparity maps from all possible image combinations. After that, we fused these disparity maps to the standard reference view pair (*e.g.* $(2, 6)$ for *Teddy*) and computed the percentage of erroneous pixels (disparities that differ by more than 1). For stereo processing we used Belief Propagation [40] (BP), Semi-Global Matching [59] (SGM), Geodesic Support Weights [66] (GSW) and our local real-time stereo method of section 4.4 (RT). We used constant parameters for the stereo methods among all baselines and datasets.

We found out that fusion results are relatively independent of the actual stereo method used (see Fig. 6.4 and Fig. 6.5), but characteristic systematic errors of each stereo method are still visible after fusion (*e.g.* bad object boundaries for RT). The overall improvement of hole-filling is below 2% – it helps mainly in occluded regions. The use of projection uncertainties is relatively important (see Fig. 6.6): even for very well calibrated sequences,

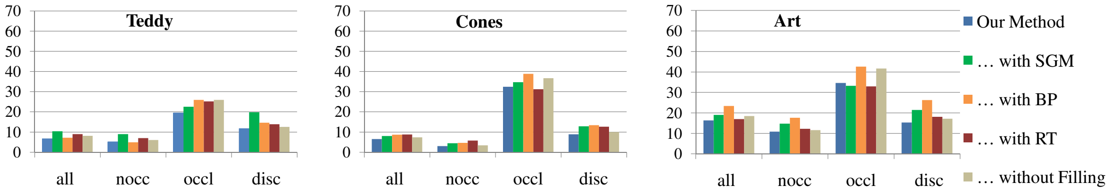


Figure 6.5. The performance of our fusion method with different stereo methods (GSW, SGM, BP and RT) or when using GSW and no hole-filling. Error bars show percentages of disparities that differ by more than 1 from the ground truth in the whole image (all), non-occluded (nocc) or occluded pixels (occl) and regions near discontinuities (disc). We fused up to 72 disparity maps. The overall improvement of hole-filling is below 2% and replacing the stereo method makes only a slight difference.

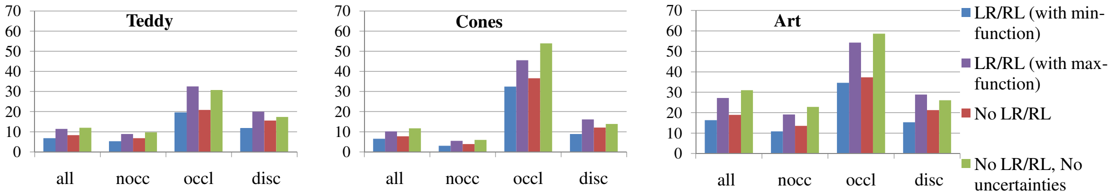


Figure 6.6. The impact of the different steps of our method: the use of projection uncertainties is important and the distinction between left-right and right-left matching (LR/RL) using the min-function helps at discontinuities. See Fig. 6.5 for a description of the bars.

there is always an uncertainty in matching which influences reprojection. The distinction between left-right and right-left matching (see also section 6.2.5) is important in regions near discontinuities (see Fig. 6.6).

6.3.1. Comparison to other Fusion Methods

We compare our method to other fusion algorithms, in particular the stability-based algorithm of Merrell *et al.* [100] using our own implementation running on CPU and the bundle optimization of Zhang *et al.* [173] using their own implementation (without their stereo-matching and without final bundle adjustment). We used the same input data (*i.e.* disparity maps) for all fusion methods. We also perform hole-filling when using the method of [100] (which was not described in [100]), because it leads to slightly better results. During our benchmark, we got the feeling that the method of [173] is optimized for short baselines (the video sequences of [173] have much smaller baselines than the datasets of [119]). Our method works better with larger baselines, which is our target application.

For our comparisons we focused on recent, challenging datasets. Difficult occlusions such as in *Art* [62] are ideal for testing fusion methods. We expect that fusion methods perform well in occluded areas, due to information contributed by other views. Further, performance in regions near discontinuities is also very interesting, because regularization may blur discontinuities.

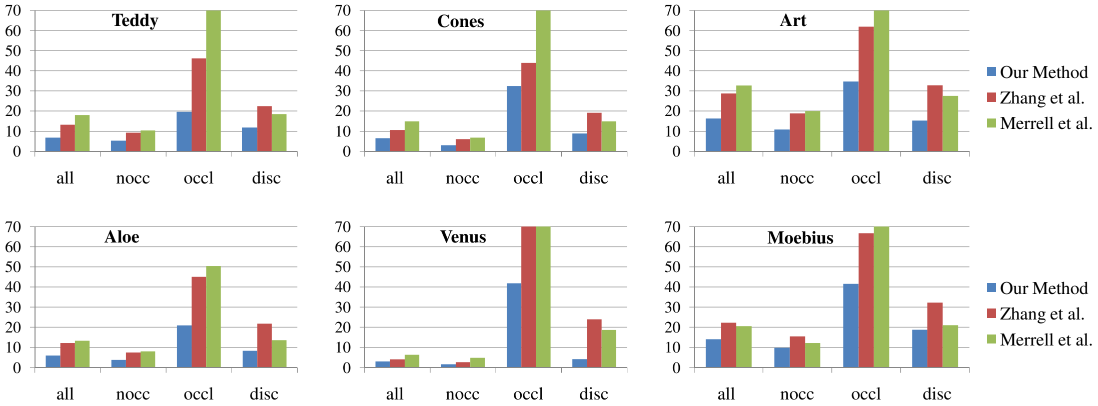


Figure 6.7. The performance of different fusion methos. Disparity maps were computed using GSW [66]. See Fig. 6.5 for a description of the bars.

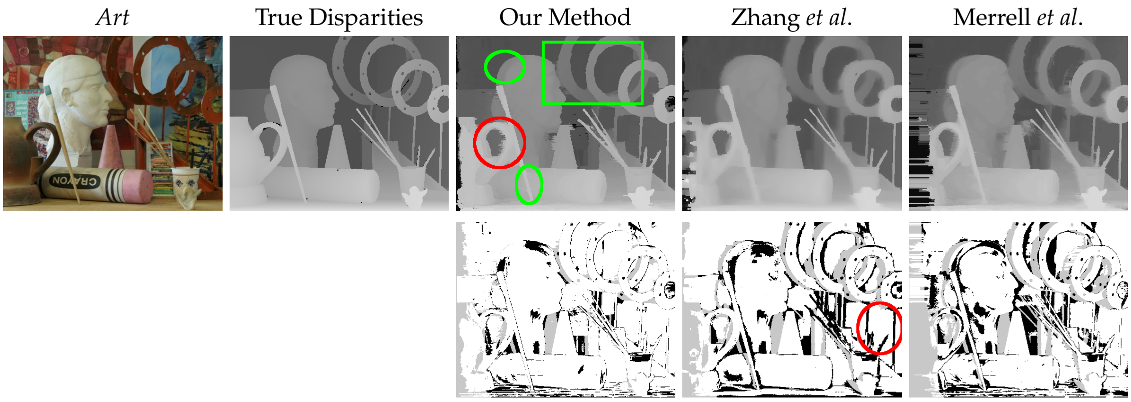


Figure 6.8. The disparities and bad pixels of different fusion methods for the dataset *Art*.

Regions Near Discontinuities

Our method preserved sharp object boundaries and thin structures: obvious when looking at the error bars in Fig. 6.7 (*e.g.* at *Art*) and disparity maps in Fig. 6.8, Fig. 6.9 (*e.g.* the house in *Teddy*) and Fig. 6.10 (*e.g.* the pyramid tops in *Moebius*). Hole-filling helps only slightly, but the distinction between left-right and right-left matching is important: left object boundaries are stable in right to left matching (and vice versa). The approach of Merrell *et al.* appears to perform better at discontinuities than the method of Zhang *et al.*, but behaves less robust if the set $\mathcal{S}(\mathbf{x})$ contains many outliers. On the first sight, the actual values of Merrell *et al.* near discontinuities may look odd, when taking the performance in non-occluded and occluded regions into account, where Zhang *et al.* is better in most cases. However, the disc-value is computed within a smaller portion of the image.

Occluded and not Occluded Regions

In these regions, we benchmarked our method better than the other methods which is mostly due to our probabilistic model (which is robust to outliers), the explicit visibility computation (using reprojection and the reliable area; it reduces the number of hypotheses) and also hole-filling (it reduces artifacts from reprojection and helps in occluded regions). In general, the method of Zhang *et al.* seems to excel at planar surfaces and the fused disparity maps tend to be slightly oversmoothed (visually very obvious in Fig. 6.9 at the *Moebius* dataset), which might be optimized by parameter tuning. There is also another interesting artifact, which we were not able to explain: thin objects in Fig. 6.8 are extracted, but with an incorrect disparity value. The robustness of Merrell *et al.* seems to be impacted by the number of outliers in the set $\mathcal{S}(\mathbf{x})$.

How can this proposal be better than prior art in these experiments?

In Merrell, visibility-constraints are enforced using their expensive definition of stability (having a complexity of $\mathcal{O}(m^2)$ – please note that the computation of $\mathcal{S}(\mathbf{x})$ is $\mathcal{O}(m)$ and that for every element of $\mathcal{S}(\mathbf{x})$, m projections are performed). However, visibility can be maintained more efficiently using reprojection and the reliable area (having $\mathcal{O}(m)$, because at every entry of $\mathcal{S}(\mathbf{x})$ we only update the global pdf by summation). This also has the big advantage that projection uncertainties can be used later, whereas in Merrell it is not possible. Moreover, for optimal stability calculation it is important that the number of outliers having a negative stability is equal to the number of outliers with positive stability. Our experiments suggest that this assumption is suboptimal in occluded regions, where usually many outliers are present.

In Zhang's method, the correct disparity is supported by the **simultaneous** combination of *geometric coherence* and *color similarity*. Geometric coherence alone supports also background disparities of surfaces occluded by foreground objects in the reference view, because visibility is not determined and this is problematic in cases where fore- and background objects are of similar color. The optimization using belief propagation ensures smoothness in these ambiguous situations but seems to perform suboptimally in regions near discontinuities. Due to the results we obtained during our experimental evaluation (our method does not use any kind of energy minimization), we believe that our pdf will also bring a huge advantage to the method of Zhang, especially near discontinuities and when using wide-baseline sequences. While the probabilistic model in (6.3) is similar the one in [173], we explicitly compute visibility to disambiguate depth hypotheses at an early stage and model projection uncertainties.

Further, we would like to stress that in our method *from every single input disparity a global pdf is computed* (parameterized using projection uncertainties and color-similarity). The final pdf of the reference disparity map results from summing up all those single pdfs. Efficiency is preserved by computing the final disparity map "bottom up" and the probabilistic model ensures robustness. Hole-filling helps slightly in occluded regions and the distinction between left-right and right-left matching is important for sharp object boundaries.

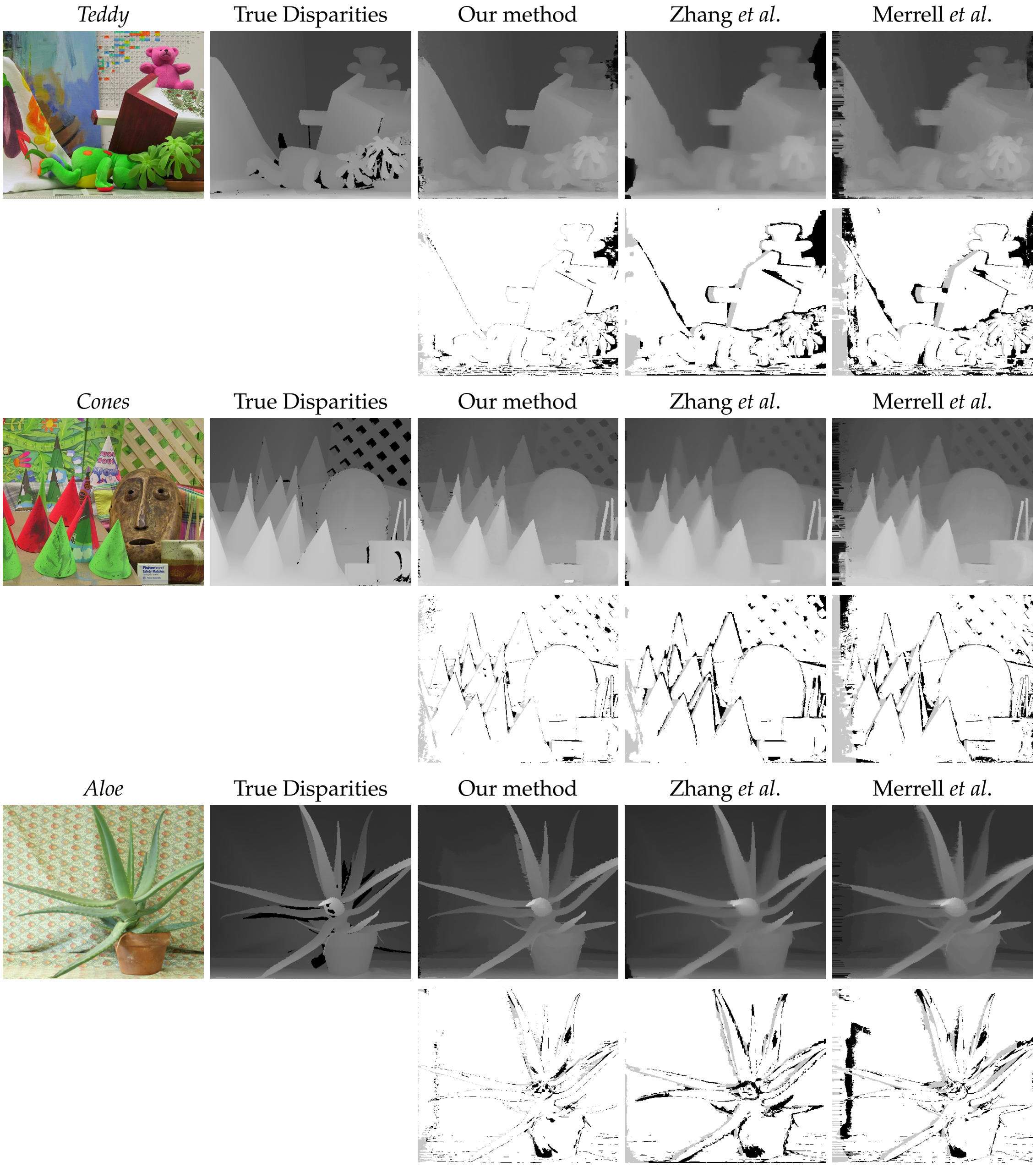


Figure 6.9. The disparity maps and bad pixels of different fusion methods for the datasets *Teddy*, *Cones* and *Aloe*.

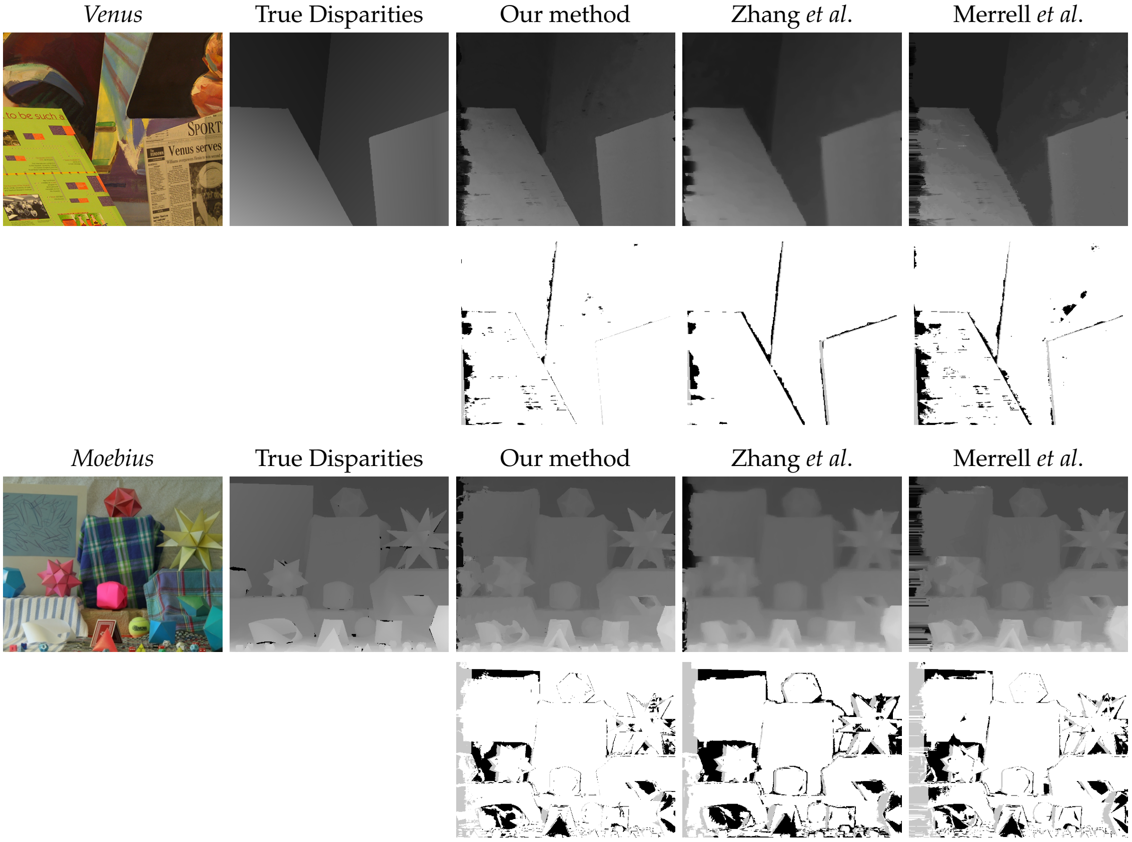


Figure 6.10. The disparity maps and bad pixels of different fusion methods for the datasets *Venus* and *Moebius*.

Execution Times

At the dataset *Teddy* (72 disparity maps) our method took 8.7 s (not optimized), the method of [100] took 40.7 s and the method of [173] 175 minutes (*i.e.* 146 s/disparity map). These times do not include stereo matching and were measured on a Intel E8200 dual-core with 2.66 GHz (for our method and [100]) or a Intel E5405 quad-core Xeon CPU with 2.00 GHz (for [173]).

For our real-time implementation, we use SIMD-instructions of the SSE2 instruction set and simplified the reprojection for motion-stereo (in this case, we assume that the y-coordinate of projected scene points is constant over time). Using pre-computed kernels, we are able to fuse 16 disparity maps in just 20 ms on a mobile CPU (2 GHz).

We also implemented an incremental variant that needs 3 ms per frame. We use the pdf of the previous frame and reproject it to the current vehicle position. This pdf is then updated using the disparity maps computed using the current camera image and the fused disparity map is determined.

6.3.2. Real World Sequences

We tested our method on real world sequences from a moving vehicle. Fig. 6.11 shows a rectified camera frame, one input disparity map (computed using our real-time stereo method of section 4.4) and one fused disparity map. For fusion we used a highly optimized implementation (using SIMD instructions) to fuse 16 adjacent input disparity maps.

Fig. 6.12 shows fused disparity maps of a sequence provided by [173], along with the camera frame and their fused depth map. For stereo matching we used GSW [66] and ensured a minimal baseline of 5 and a maximal baseline of 7 frames (the baseline of adjacent frames was too small for robust matching with GSW). We fused disparity maps of only 20 adjacent frames and this explains why some disparities which are outside of the field of view are missing (black regions at the left and right). Please have a look at the supplemental material for a complete video sequence.

6.4. Discussion

We proposed a novel probabilistic method for fusing disparity maps in classical stereo or motion-stereo setups. We achieve this by computing a probability density function from all provided disparity maps. From this distribution, we determine the most probable disparity map for a given reference view pair.

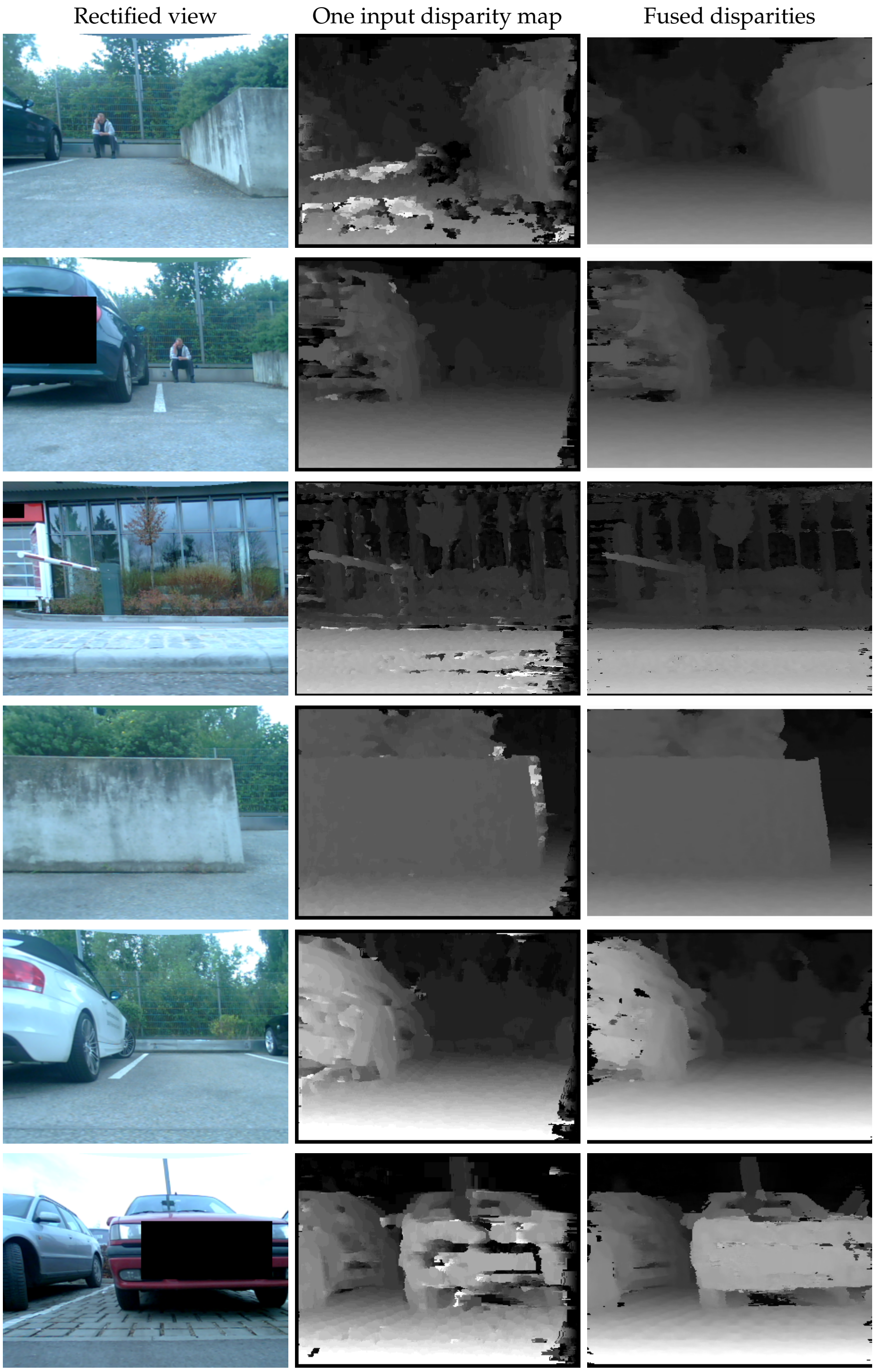
We introduce several novel concepts:

- Reprojection using the reliable area (for efficient visibility determination),
- A generic probabilistic model that uses projection uncertainties (for robustness to outliers),
- A distinction between left-right and right-left matching (for sharp object boundaries), and
- Hole-filling (for improved quality in occluded regions).

We compare our method to the current art using real scenes and disparity maps generated from datasets with ground truth. Our experiments show clearly that our proposal appeals with good results and efficiency.

In practice, the fusion of 16 adjacent disparity maps runs in real-time and performs very well at avoiding false matches and at improving the accuracy of depth discontinuities. This results in a much better performance and availability of the application, especially in very difficult situations, for example in low light or backlighting scenarios, when glare lights are present, at illumination or exposure changes, with inaccurate camera positions, and also with dirty camera lenses. Therefore, the fusion approach presented here is a fundamental building block of our robust depth sensing and is more important than the stereo matching itself. One important insight of our quantitative and qualitative results is that the actual choice of the stereo method is only of minor importance. The redundancy and massive amount of depth data is what leads to reliable and accurate results.

We found that the framework presented in this chapter is quite powerful and our idea was to further generalize the concept towards multi-view fusion of image segmentations.



[redacted] [redacted] [redacted]

Figure 6.11. Our method applied to sequences from our vehicle using our real-time stereo method.

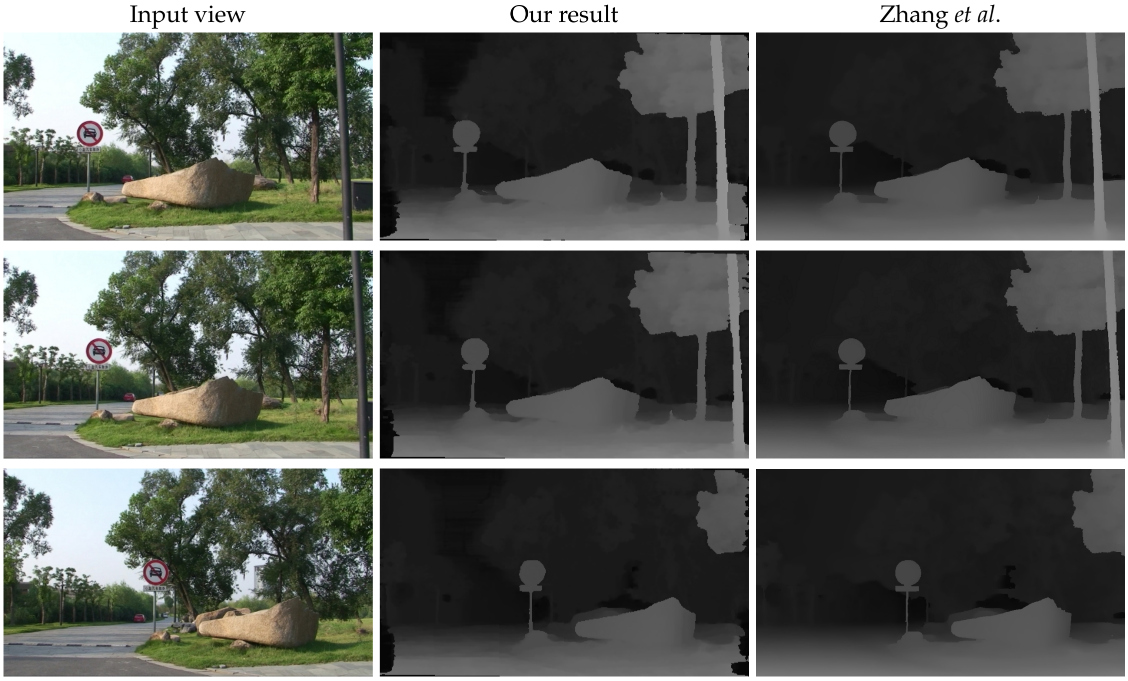


Figure 6.12. Our method applied to the sequence *Road* provided by [173].

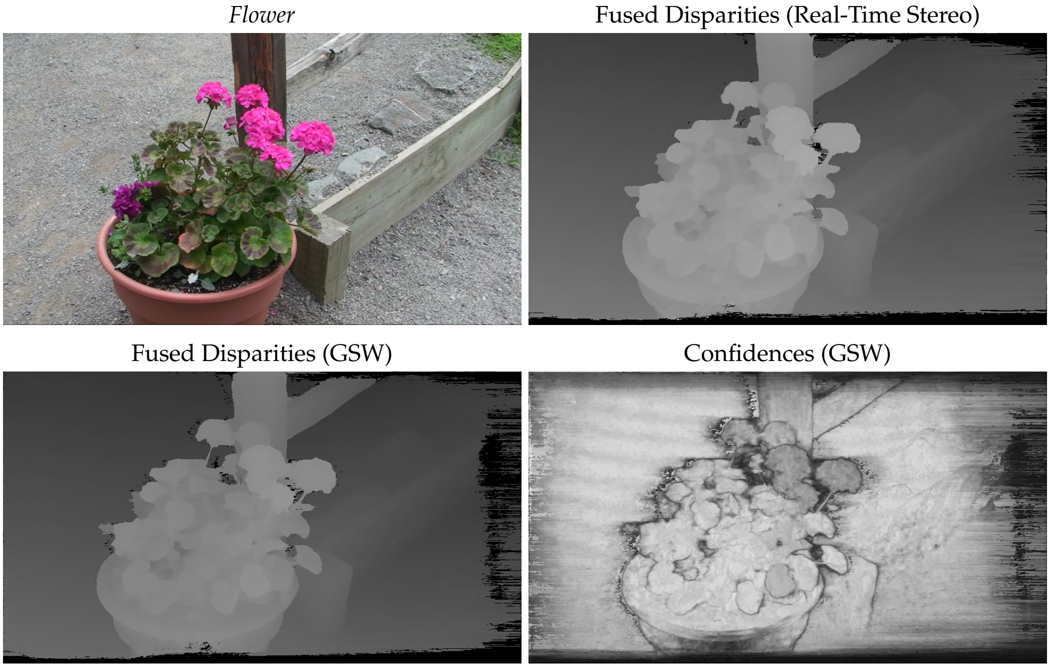
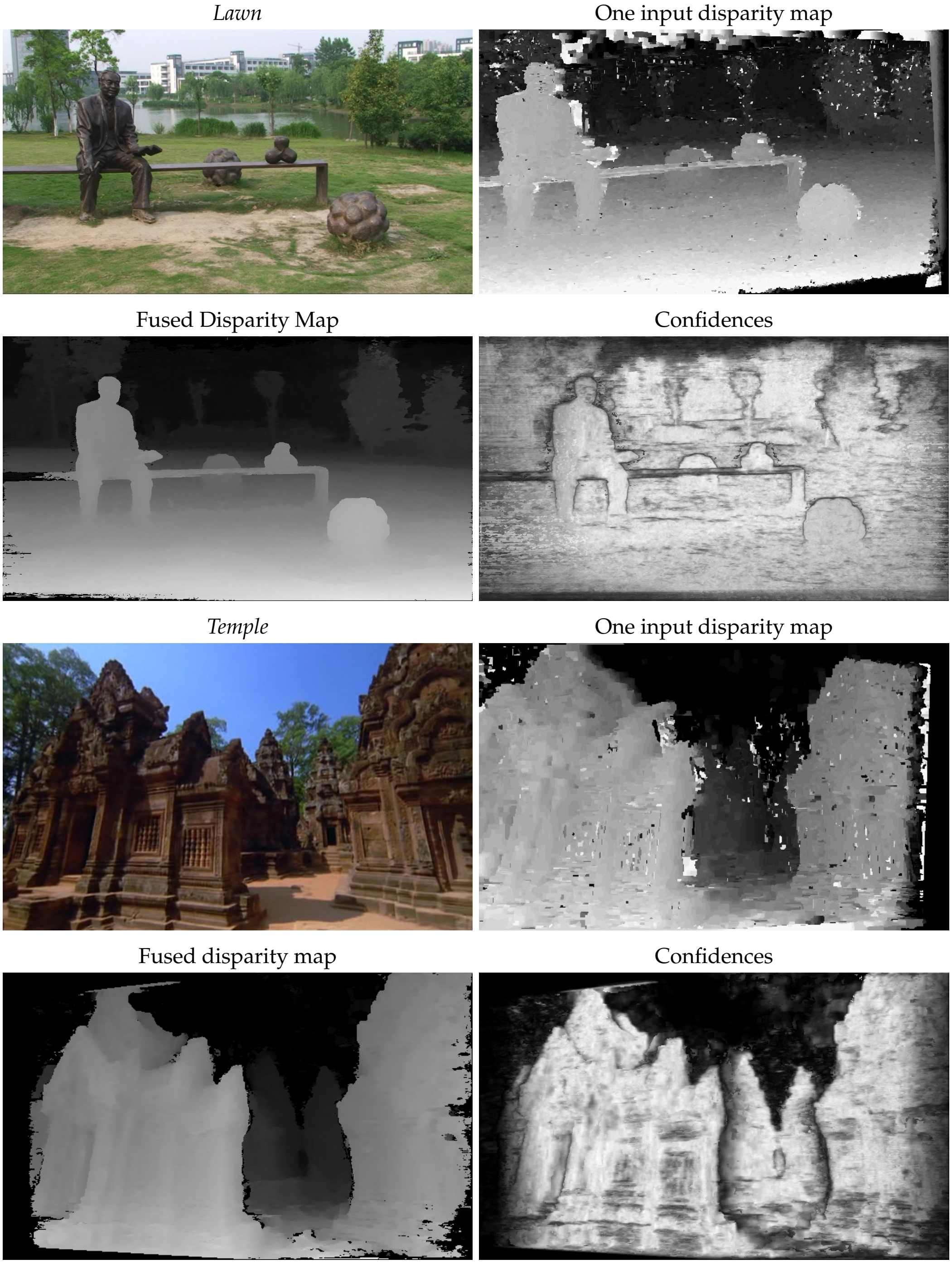


Figure 6.13. Our method applied to the sequence *Flower* provided by [173].

While for disparity maps the fusion of numerical values from continuous domain (disparities) is relatively straightforward, the problem is much more difficult for image segmenta-



Lawn

One input disparity map

Fused Disparity Map

Confidences

Temple

One input disparity map

Fused disparity map

Confidences

Figure 6.14. Our method applied to the sequences *Lawn* and *Temple* provided by [173].

tions, because the values are now discrete (the individual segment labels) and the labels of segmentations from different images are also not coherent. In this sense, the next chapter will address the problem of video segmentation.

7. Camera-based Parking Assistance

> ***Automobile*** *(French) originates from ancient Greek* autós *(αυτός; self) and from Latin* mobilis *(movable);* auto *refers to the fact that the vehicle is powered by an engine rather than being pulled by horses.* [2]

In this chapter we finally close the circle of this dissertation and present a camera-based parking assistant that makes use of the algorithms described in the previous chapters. Using our robust real-time depth sensing we introduce a streamlined approach for the interpretation of the depth data, so to detect and measure parking slots, to perform a collision analysis of specific regions around the vehicle and to visualize the acquired surroundings of the vehicle. After a brief motivation and a review of related work we describe all image processing and interpretation algorithms in detail. Towards the end we show results of our exhaustive evaluation using a huge amount of very challenging video sequences from our vehicle that comprises over 700 parking slots and different environmental conditions. Since we concentrate on the applications, a brief summary of the implementation of the motion-stereo method will be given (*i.e.* the stereo matching and probabilistic stereo fusion). The customer functions, which rely on the recovered depth maps, will be discussed in detail. There, we also compare to feature-based motion-stereo [151, 152] and a solution based on an ultrasonic sensor [112]. A number of qualitative results and illustrations are also provided.

7.1. Related Methods

In section 3.4 we give a broad overview on intelligent vehicle systems and parking assistance. Directly related are vision-based methods which also use the principle of motion-stereo [41, 124, 130, 131, 147, 151, 152]. However, these works address only a feature-based strategy and no one utilized dense disparity maps. The basic idea is to calculate characteristic features in subsequent images. Over time, this relatively small number of points is tracked and then a 3D reconstruction is analyzed to find parking slots. These approaches perform well in friendly conditions, *i.e.* as long as enough strong and distinctive features can be derived from the images. However, challenging are both lowly textured objects, which lead to very sparse point clouds, or also complex textures like foliage, where high ambiguity during feature matching introduces wrong distance measurements. Moreover, features are not necessarily located at the boundaries of objects. Thus the size of objects and free space might be wrongly calculated. In these situations, the accuracy and reliability of the determination of free parking areas varied in an inacceptable way.

In this chapter, we present a powerful approach that is based on our *dense* motion-stereo pipeline, where at every frame a dense disparity map is computed. This results in important advantages, namely a very high detection rate of obstacles, a high measurement accuracy, a nearly drift free environment model and the ability to display a multitude of different customer functions.

7.2. Environment Modeling

Our goal is to support the driver and eventually other occupants of the car at parking related tasks. In the first place, this includes assistance for finding a parking slot (*i.e.* automatic detection and measurement of free space). This also calls for an adequate interface to the driver, which provides a visualization of the found parking place. In practice, we generate a bird's eye view of the ground plane with the host vehicle, parking space and obstacles overlaid. Finally, we want to inform the driver and occupants if obstacles are located in the pivoting ranges of the doors in order to prevent minor damage.

7.2.1. Method Overview

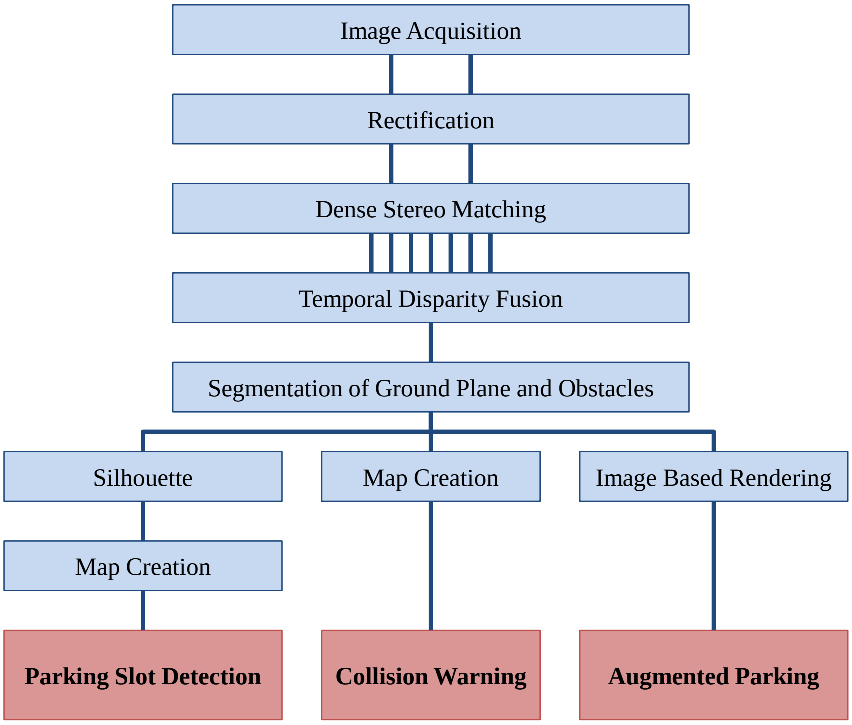


Figure 7.1. The processing pipeline of our approach: we rectify images acquired at different positions from a monocular camera and use them to compute dense disparity maps using real-time stereo and probabilistic stereo fusion. From these measurements we segment the ground plane and obstacles. This information is the foundation for the different applications: the *parking slot detection* computes silhouettes of the observed area from the segmented disparity maps, creates a map out of them and detects and measures free spaces. The *collision warning* uses obstacle points to examine the pivoting ranges of the doors, and *augmented parking* takes advantage of segmented disparity data to render an image of the ground plane with the host vehicle, parking space and obstacles overlaid.

Following Fig. 7.1, our approach is composed of several processing steps. Based on the principle of dense motion-stereo, we determine a depth for every pixel in every camera image using stereo. In our case, a rectified pair of camera images is used for the stereo matching (for example, the last two images acquired from the camera). Since these calculations are relatively expensive in terms of processing power, only efficient methods can be applied, such as our proposals of chapter 4. After that, the history of disparity maps is fused probabilistically using our multi-view method presented in chapter 6 in order to obtain for every camera image the most probable disparity map that exposes a minimum amount of outliers. In every fused disparity map we detect the ground plane, obstacles and from that a silhouette which defines the free space. Then, we combine all these partial silhouettes so that over time a global two-dimensional model of the environment is created incrementally. Within this model, we detect parallel and cross parking slots. If a free space region provides enough space for the vehicle and is bounded by obstacles, then it is a candidate for a parking slot and the exact metric size is computed.

Further, we use the disparity maps to obtain a local 3D reconstruction of specific regions of interest (for example, the pivoting range of a door). Using such a local 3D reconstruction, we perform a collision analysis and, if necessary, issue a warning to occupants to prevent minor damages. Another application is *Augmented Parking* and uses image-based rendering to compute a virtual bird's eye view to visualize the positions of the host vehicle, obstacles and the parking slot to the driver. In the following we will discuss all necessary elements of the system.

7.2.2. Camera Sensors

The position and orientation of a camera with respect to the vehicle are important parameters. Two categories of cameras are relevant for dense motion-stereo with respect to their orientation and applied functions.

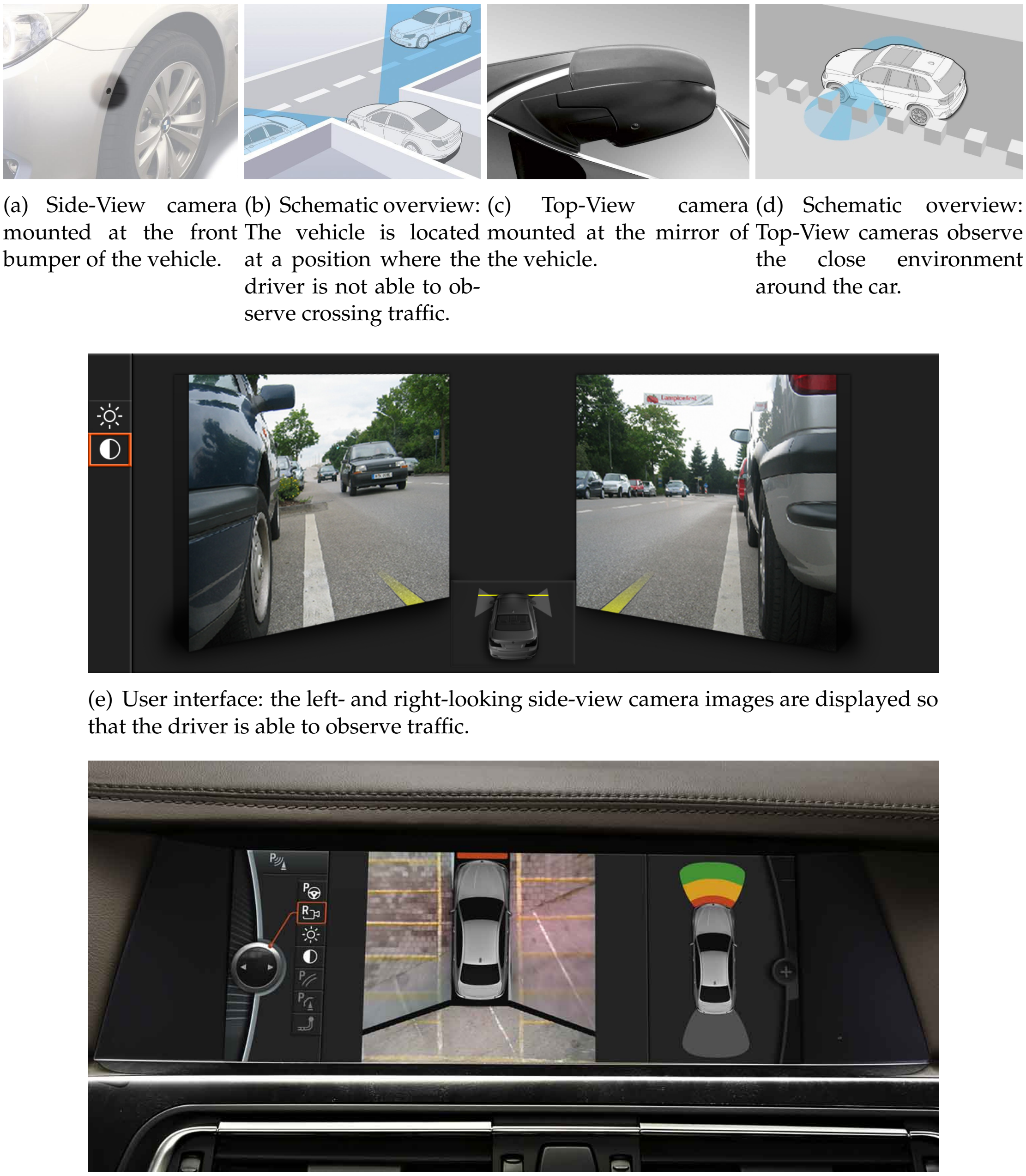
The first class is the family of side-view cameras (see Fig. 7.2), which is located in the front part of a vehicle. The optical axis of these cameras is parallel to the ground and orthogonal to the orientation of the vehicle so that they are well suited for "first views" in situations when the driver has an obstructed line of sight such as at the exit of car parks. Accordingly, side-view cameras are mostly equipped with standard lenses.

Another class of lateral cameras is the family of top-view cameras (see Fig. 7.2). Here the goal is to provide the driver with a virtual *surround view* containing the close environment around his car to give visual support during low speed or parking maneuvers. Respectively, these cameras are positioned in central parts of the body shell, where a wide angle lens allows displaying the right and left areas.

In our experiments we used both types of cameras: The diagonal FOV of the side-view and top-view camera is 68 and 170 degrees, respectively and they operate at VGA-resolution (640x480 pixels) at 30 frames per second. For stereo matching, we down-sample the images to a resolution of 213x160.

7.2.3. Calibration and Rectification

For stereo, a correct camera calibration and rectification is of eminent importance. In particular, a correction of the radial lens distortion is indispensable [34, 175]. Further, the



(a) Side-View camera mounted at the front bumper of the vehicle.

(b) Schematic overview: The vehicle is located at a position where the driver is not able to observe crossing traffic.

(c) Top-View camera mounted at the mirror of the vehicle.

(d) Schematic overview: Top-View cameras observe the close environment around the car.

(e) User interface: the left- and right-looking side-view camera images are displayed so that the driver is able to observe traffic.

(f) User interface: the images of the top-view and backward-looking cameras are undistorted and displayed together.

Figure 7.2. **Side-View** Cameras: (a) the mounting position, (b) schematic overview of the use case and (e) the user interface. **Top-View** Cameras: (c) the mounting position, (d) schematic overview of the use case and (f) the user interface.

poses of a camera at two different points in time have to be determined, in order to rectify pairs of images. In our case, odometry information was sufficient for that. However, if no odometry is available or if it is too inaccurate, then this rectification may be estimated from image correspondences [136]. The yaw-angle and movement in x- and y-direction can be determined relatively well from odometry information, if the position of the camera relative to the vehicle-origin is known. In practice, the recovery of pitch and roll angles as well as the movement in z-direction is relatively imprecise with current vehicle sensors. Due to this reason, we ignore these values in the first place. We rather use a simplified, approximated rectification (using only the yaw-angle and movement in x- and y-direction) and resort to our extended stereo method presented in section 4.5.1, which is robust to inaccurately estimated epipolar geometry. Therefore, it is not important that odometry information is highly accurate: we only use it to approximate the rectification. Another benefit of the simplified rectification is that the warping of images can be implemented in an optimized way.

7.2.4. Stereo Matching

For stereo matching we use our local method of section 4.4 that runs significantly faster than traditional real-time implementations [61, 104]. In particular, our method is suited very well for motion-stereo setups, as it does not require a priori knowledge about the maximum disparity, which depends on the motion model, the camera intrinsic parameters and on the depths of the observed scene. In our implementation, the use of SIMD-instructions allows us to compute a disparity-map for a 320x240 image in less than 30 milliseconds.

Frame Decimation. Since the vehicle moves with different velocities, the baseline of adjacent frames is not constant. Especially for low velocities, the baseline becomes too small for accurate depth computations. In practice, we use a simple frame decimation [105] technique to improve depth estimation: for every reference frame, we select the matching frame in a way such that the baseline is always greater than 10 cm. We obtain the baseline from odometry information.

Histogram Equalization. To some extent, the high efficiency results from using the sum of absolute differences (SAD) as a matching cost. However, when the camera moves through its environment, lighting conditions will change constantly and sometimes very abruptly. Because of the characteristics of the camera, we decided not to work with a constant exposure time, so to always allow optimal exposure. But this results in stereo pairs with different exposures, which has adverse effects on the matching using SAD. To reduce these problems we chose to use histogram equalization.

7.2.5. Improved Matching for Motion-Stereo

In real-world situations, the ground is not perfectly flat and will cause the vehicle to pitch and roll (*e.g.* due to road holes). In such scenarios the rectification would be more complex than the simplified one described in section 7.2.3. However, in practice the simplified rectification is sufficient, if the search area of stereo matching is slightly increased, so that a set of neighboring scanlines is taken into account, too. This is achieved by the method presented in section 4.5.1. In practice, the extended method takes roughly twice the time, so that real-time processing is still possible at a resolution of 213x160. In contrast, if the search range is increased in traditional real-time stereo methods like [61], the execution time is multiplied by the number of scanlines that have to be taken into account.

Please note that the vertical displacements may be used to update the rectification. In practice however, we discard the vertical displacements.

7.2.6. Temporal Fusion of Disparity Maps

In chapter 6, we have already pointed out that real-time stereo matching is error prone and is known to have weaknesses in regions near discontinuities [61, 119]. Therefore, we use the method proposed in chapter 6 to obtain high quality disparity maps since it allows real-time operation on standard CPUs and provides a very good accuracy, especially in occluded parts and in regions near discontinuities (see Fig. 7.3 for an example). Popular alternatives are [100, 170, 172], but [100, 170] require a GPU for real-time operation and [172] is far from being real-time.

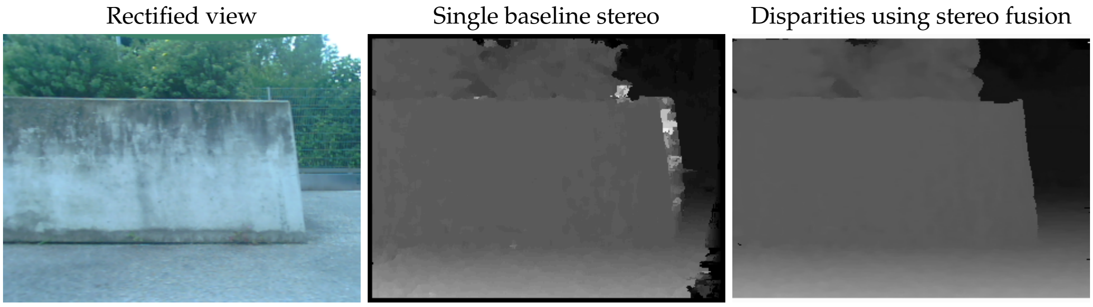


Figure 7.3. Example for the temporal fusion. We show the camera frame, a single baseline disparity map computed using our real-time stereo method and a fused disparity map. More examples are shown in Fig. 6.11.

7.2.7. Ground Plane Segmentation

One goal of our system is to retrieve information about areas that are *free* for parking and parts of the environment that are *occupied* by obstacles. To accomplish this, our strategy is to analyze individual disparity maps to identify points that belong to the ground plane and points that belong to the obstacles. We perform this classification solely in disparity maps and therefore, we look at the plane plus parallax homography induced by the ground plane:

$$\mathbf{H}_G = \mathbf{I} + s\,\mathbf{e}_1\mathbf{v}^T \tag{7.1}$$

where s is proportional to the traveled distance, $\mathbf{e}_1 = (1, 0, 0)^T$ is the baseline and $\mathbf{v} = (v_1, v_2, v_3)^T$ is the normal vector of the ground plane. In our case, the camera is mounted on a ground vehicle, so the baseline is parallel to the ground plane and therefore we can assume that $v_1 = 0$. The other two unknown components v_2 and v_3 depend on the slope of the ground. These we want to recover from the disparity map (we could also use odometry information for that, but our experiments showed that it is less reliable). With $\mathbf{p} = (x, y, 1)^T$, we obtain the following linear relationship for the disparities of points belonging to the ground plane:

$$\begin{aligned} \mathcal{D}(\mathbf{p}) &= \mathbf{e}_1^T(\mathbf{H}\mathbf{p} - \mathbf{p}) = \mathbf{e}_1^T \left(s\,\mathbf{e}_1\mathbf{v}^T\right)\mathbf{p} \\ &= s\left(\mathbf{e}_1^T\mathbf{e}_1\right)\left(\mathbf{v}^T\mathbf{p}\right) = \underbrace{(s\,v_2)}_{=:A}\,y + \underbrace{(s\,v_3)}_{=:B} \end{aligned} \tag{7.2}$$

This implies that the disparity of the ground plane pixels is constant within a scanline and motivates us to determine the parameters A and B by analyzing the histograms of disparities created for each scanline.

The rough idea is to obtain initial guesses of A and B using a small set of scanlines and then use a greedy algorithm to refine them by looking at more and more scanlines. We start this estimation at the bottom of the image and successively use scanlines above. We do it this way, because in practice in most cases the ground plane is visible in the bottom part of the image. To reject scanlines containing a high amount of outliers, we use the confidence measure which is provided by the temporal fusion for every disparity. By assuming that disparities with a small confidence are wrong, we estimate the number of outliers for every scanline. Then, during the whole estimation-process of A and B, we use only those *reliable* scanlines whose outlier-count is below a threshold (in practice, a scanline should not contain more than 50% outliers).

Initial Estimation. We start with an initial estimation of A and B, and iteratively refine these values. For the initial estimation, we use the bottommost four reliable scanlines and compute the parameters by L_2-regression: for each scanline, we compute a separate histogram of disparities and take the predominant value. Then these four values are used for the regression using (7.2). The initial estimates are then filtered using a Kalman-filter, together with values of the ground plane estimation of the previous camera frame.

Iterative Refinement. Once we obtained a first guess of the ground plane model, we add more scanlines to refine the model: we visit each scanline from bottom to top and look at the peaks of the histogram of each scanline. If the predominant value of the histogram fits to the ground plane model we use the value to refine the parameters A and B by adding the value to the regression: we do this by comparing the predominant value to the value predicted by the current ground plane model, and if the absolute difference is below a certain threshold, we update the ground plane model. During this process only those scanlines have to be considered for which the predicted disparity is positive, *i.e.* for $y > \frac{-B}{A}$ ($\mathcal{D}(x, \frac{-B}{A}) = 0$ is the horizon line). Finally, we are able to segment the ground plane by checking the disparity of every individual pixel against the ground plane model – independent from whether their scanline was used for the estimation of A and B. Fig. 7.4 is an example for such a segmentation: blue pixels belong to the ground plane and the blue line is the estimated horizon line. The red ticks indicate the theoretical position of the horizon line (computed using the camera intrinsic and extrinsic parameters and a canonical, perfectly flat ground plane).

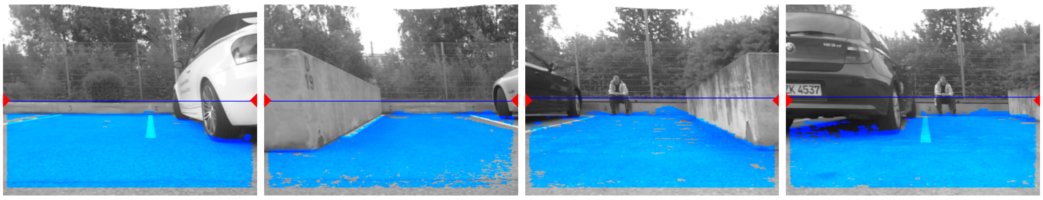


Figure 7.4. Superimposed ground plane segmentation: Blue pixels belong to the ground plane, the blue line is the estimated horizon line and red ticks indicate the theoretical position of the horizon line. Please note that the segmentation also finds points under the car.

Maximum Likelihood Estimation. After the iterative refinement we identified a set of disparities that fulfill a simple linear model and the corresponding matrix $\mathbf{H}_G$ has 2 degrees of freedom. To further improve accuracy, we randomly select 1000 points of the ground plane segmentation and compute the parameters of $\mathbf{H}_G$ using the QR decomposition.

7.2.8. Computation of Silhouettes

Once we obtained information about obstacles and the ground plane, our goal is to compute a *silhouette* that limits the free space. The free space is bounded by obstacles and by the region borders of the ground plane. For example, in the majority of cases, the curb is not detected as an obstacle, but the ground plane segmentation stops there. We define that the silhouette is represented in image coordinates.

Obstacle Silhouette. In practice the obstacles in our scenes can mainly be approximated by large fronto-parallel planar patches (at least for side-view cameras – with top-view cameras such a requirement can be enforced by rotating the camera virtually). Our practical tests confirmed that this assumption still holds for plants (like bushes), motorcycles and curved parts of other vehicles. To some extent, this can be explained by the quantization of disparity values.

Due to these properties, we first compute the histograms of disparities within single image columns, but only from those disparities which are not part of the ground plane. Building such histograms increases robustness against outliers (in difficult scenes, more stability can be attained by computing the histograms over multiple columns). From these histograms we collect the first N_{OBS} predominant entries, but only if their count is greater than a specific threshold θ_{OBS} (to exclude disparities caused by noise)[1]. From these disparities we take the largest one (minimal obstacle distance) and project the value onto the ground plane by solving (7.2) for y. This results in a single silhouette-point for an image column. By applying these steps to every image column we obtain a silhouette of the obstacles.

Ground Plane Silhouette. Often, the curb is not detected as an obstacle but the ground plane segmentation stops at the curb. Due to this reason, we also compute a silhouette

[1] In our experiments we set $N_{\text{OBS}} = 3$ and $\theta_{\text{OBS}} = 10$.

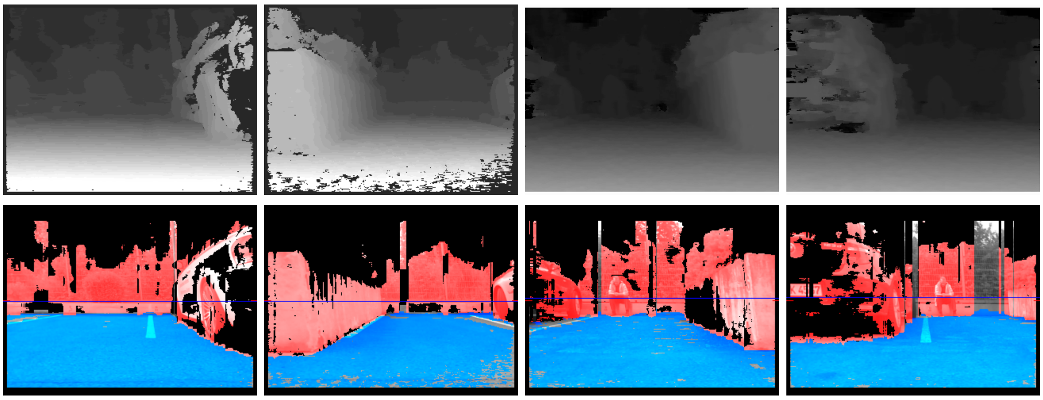


Figure 7.5. Superimposed ground plane and obstacle segmentation. First row: fused disparity maps. Second row: corresponding rectified camera frame superimposed with the segmentation result. Blue pixels belong to the ground plane; the blue line is the estimated position of the horizon line and red pixels belong to obstacles.

of the ground plane by analyzing image columns of the ground plane segmentation. For a specific image column x, the location of the silhouette point is defined as the topmost pixel that belongs to the ground plane. If the ground plane is not visible in column x we invalidate this silhouette point, because then there is no information available about the ground plane.

7.2.9. Cumulative Map Creation

We will use this map later for the parking slot detection and will propose slight modifications for other customer functions. We define our map to represent a specific region of the ground plane around the host vehicle from a bird's eye view as shown in Fig. 7.6b. We build this map of the environment incrementally and we divide it into small equi-sized cells where every cell stores the likelihood that the cell is occupied. We continuously update this map by "adding" the silhouettes. Both the obstacle and ground plane silhouettes are transformed from image coordinates into the bird's eye view using a transformation $\mathbf{H}_{BEV}$, which warps the camera image onto the ground plane. This is equivalent to mapping points of the horizon line to infinity. This transformation may be computed using the horizon line computed using the ground plane segmentation. Every cell of the map has associated a likelihood which is increased every time a silhouette projects there.

Hence, over time more and more silhouettes are added. It must be noted that the ground plane segmentations may have different locations of the horizon line. Sudden large changes of the horizon line should be avoided, or otherwise the distance measures in the map become inconsistent. Therefore we assume that the variation of the horizon line is small, in case a parking slot is in the FOV. While an imprecision within a specific range is normal, large variations of the ground plane indicate either large changes in the slope of the ground, or inaccuracies in the ground plane segmentation. Both situations are easily detected using the variance of the location of the horizon line and then the parking slot detection should disregard the corresponding region.

Position Estimation. Our goal is to use the plane plus parallax homography $\mathbf{H}_G$ obtained from the ground plane segmentation for the position estimation. The map represents a defined portion around the host vehicle (the host vehicle has a constant position and orientation). Since the vehicle moves over time, the map must be updated by a translation and rotation. These updates have to happen continuously at every camera frame, because of the movement of the vehicle. With the transformations $\mathbf{H}_{BEV}$ and $\mathbf{H}_{BEV} \cdot \mathbf{H}_G^{-1}$, the movement of the vehicle can be taken into account. However, at this point we also have to take care of the rectification of the stereo pair. Since the transformation $\mathbf{H}_G$ maps points of the current rectified camera frame to the previous rectified camera frame via the ground plane, we have to include the pair of rectification matrices $\mathbf{H}_{RC}$ and $\mathbf{H}_{RP}$ of the current and previous camera frames. Now, we obtain the plane plus parallax homography in image coordinates which warps the previous camera frame to the current one taking into account rectification:

$$\mathbf{H}_{RC}^{-1}\,\mathbf{H}_G^{-1}\,\mathbf{H}_{RP} \tag{7.3}$$

Finally, the transformation with which the map has to be updated may be written as:

$$\mathbf{H}_{BEV} \cdot \left(\mathbf{H}_{RC}^{-1}\,\mathbf{H}_G^{-1}\,\mathbf{H}_{RP}\right) \cdot \mathbf{H}_{BEV}^{-1} \tag{7.4}$$

Intuitively, we first warp the map to the previous image using $\mathbf{H}_{BEV}^{-1}$ and then transform the map using (7.3) according to the camera motion. Finally, we warp the transformed map from image coordinates back to the bird's eye view using $\mathbf{H}_{BEV}$.

Cumulative Map Update. We need to update the likelihoods of the occupancies of the map, according to the current obstacle and ground plane silhouettes. For that, we transform the silhouette-points, given in image coordinates of the current rectified frame, into the bird's eye view using

$$\mathbf{H}_{BEV} \cdot \mathbf{H}_{RC}^{-1} \tag{7.5}$$

In general, the warped silhouette points in the bird's eye view have a specific covariance, which depends on the accuracy of disparity estimation and the uncertainty of camera locations. We take this uncertainty into account and for every silhouette point we update the occupancy of the surrounding cells according to that uncertainty. The uncertainty is spread using a Gaussian kernel around each warped silhouette point. The variance of the Gaussian in x-direction is constant, while the variance in y-direction depends on the depth uncertainty (this is, because we defined that the orientation of the vehicle always points into the negative x-direction in the bird's eye view). Taking the uncertainty into account will help in difficult situations (for example, when using top-view cameras) and will make the model independent of quantization.

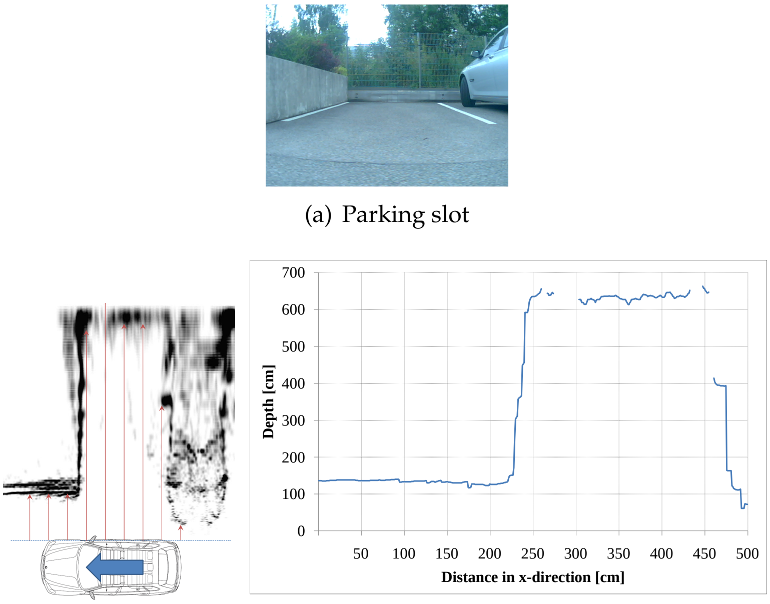
In practice, we implemented these map updates very efficiently: these assumptions lead to a rectangular region of the bird's eye view, in which the map must be updated. We implemented these operations very efficiently by using saturated additions and by scaling precompiled Gaussian kernels.

7.3. Applications

In the following, we demonstrate three different customer-oriented functionalities. These applications utilize the proposed motion-stereo processing pipeline but use disparity information in slightly different ways.

7.3.1. Automatic Parking Slot Detection

For the detection of parking slots we use the cumulative map of the environment and compute a global silhouette. Since the orientation of the vehicle is defined to be aligned with the x-axis of the bird's eye view, we compute one silhouette point for every column of the map as shown in Fig. 7.6. For that, we first determine a line which runs through the position of the camera in the bird's eye view and is aligned with the orientation of the vehicle (we assume that parking slots are either parallel or orthogonal to the orientation of the vehicle). As indicated in Fig. 7.6, in every image column x of the cumulative map, we find the closest occupied cell to that line (a cell is occupied, if its likelihood is greater than a specific threshold) and store the distance in a global distance profile $S(x)$.



(a) Parking slot

(b) Cumulative map

(c) Distance profile

Figure 7.6. Parking Slot Detection: the car passes by a parking slot (a) and builds the cumulative map (b). From this map, a distance profile is derived and the parking slot detection is performed (c).

Based on this global distance profile S, we define that a parking slot is an interval $[x_1, x_2]$ which fulfils

$$S_- < S(x) < S_+ \quad \forall x \in [x_1, x_2] \tag{7.6}$$

We use the parameters S_- and S_+ to constrain the size of the parking slot (as described below). In practice, we use a small seed interval in which we check (7.6) and then we grow the interval to the left and right.

Orientation. To detect cross and parallel parking slots, we perform the check using (7.6) with different parameters:

- To detect parallel slots we set S_- and S_+ to 2 m and 4 m, and
- For cross parking slots we use 4 m and ∞ for S_- and S_+.

Depth. To compute the depth, we pick the camera frame from the history when the camera was at position x_1 (*i.e.* the nearest neighbor). Then we select two subsets of the segmented obstacle disparities:

1. The set of close disparities: all disparities of the obstacle segmentation whose silhouette position x fulfils $x < x_1 - 25\text{cm}$.
2. The set of far disparities: all disparities of the obstacle segmentation whose silhouette position x fulfils $x > x_1 + 25\text{cm}$.

We compute the mean values of these disparity values and compute the depth of the parking slot as the difference of these two depth values. We perform the same computations for the position x_2 and use the maximal value as the final depth for the parking slot.

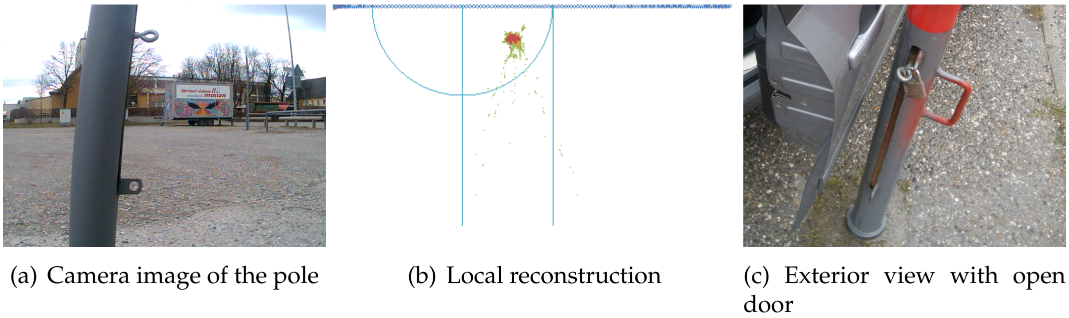
Validation. We use several rejection cues for the validation of a free space. Every free space has associated a specific interval of camera frames in which the free space is (partially) visible. We reject a free space as a parking slot, if

- the *steering angle* exceeds defined thresholds, or
- the *velocity* is greater than a defined speed, or
- the variance of the location of the *horizon line* (of the ground plane segmentation) exceeds defined thresholds.

7.3.2. Collision Warning

The sportive exterior design of cabriolets and coupés makes the pivoting ranges of doors difficult to observe, because often the door-edge is located behind the driver's head. Small objects or unthoughtful opening of a door may lead to expensive minor damage. The goal of this application is to prevent such damage by checking for possible collisions with static objects in the pivoting ranges of the doors. If such a possible collision is detected, occupants may be warned visually, acoustically or even haptically. In practice, we keep the history of disparity maps and segmentations and perform these checks in the moment when the vehicle stops.

We realized the collision detection in a slightly different way, however with the same algorithmic components. Instead of using the silhouettes generated from the obstacle segmentation, we directly use all disparities of the obstacle segmentation. For every disparity, we compute the position on the ground plane (by solving (7.2) for y) and transfer that point to the bird's eye view using (7.5). Then, we update the corresponding cell of a cumulative map at this position. This time however, we do not use a Gaussian kernel and we increment only one single cell per obstacle disparity. The collision detection is then performed by analyzing defined regions of the map (for example, the regions corresponding to the pivoting ranges of the doors). If a region contains cells whose counter is greater than a specific value, a warning is issued.



(a) Camera image of the pole

(b) Local reconstruction

(c) Exterior view with open door

Figure 7.7. Collision detection within the pivoting area of the right front door: the car passes by a pole (a) and in the moment when the vehicle stops, a local reconstruction (b) created from the history of disparity maps is analyzed for possible collisions (c).

7.3.3. Augmented Parking

Once a parking slot has been detected, the question is how to visualize the actual location to the driver. One idea is to generate a bird's eye view displaying the image of the ground plane with the position of the host vehicle and obstacles overlaid as shown in Fig. 7.8.

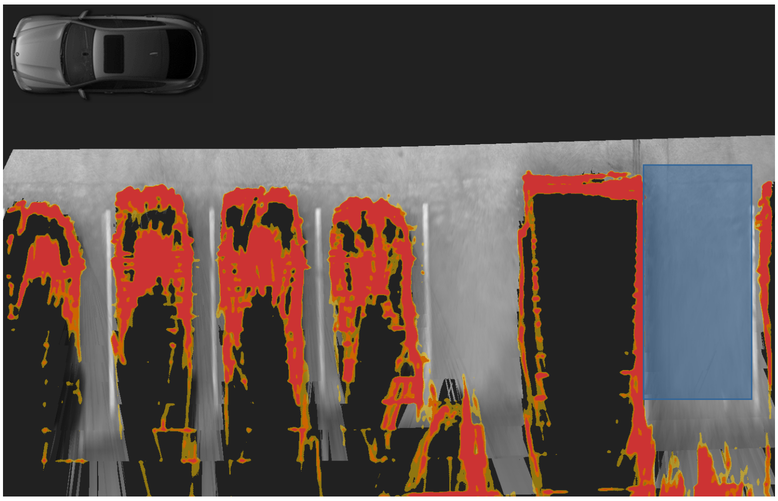


Figure 7.8. Augmented Parking: the host vehicle, the detected parking slot and surrounding obstacles are displayed over the image of the ground plane.

Also this application can be implemented with the described components. We use a slightly modified map: at this time, it represents a specific region of the ground plane, which we want to visualize. Every cell of the map holds a pair of values (c, n), with c being an intensity value and n being a counter. In the beginning, all cells of the map are initialized to $(0, 0)$. To obtain the optimal quality of this image-based rendering, we use backward-warping: we iterate over all cells of the map and compute the location in every relevant camera frame using the inverse transformation of (7.5) and by chaining the history of plane plus parallax homographies (7.4). In practice, we keep a history of camera frames and transformations in memory. For such an image pixel $\mathbf{p}$, we check whether it is part of the ground plane segmentation and obtain the intensity value c' from the camera frame using bilinear interpolation. Let the current cell of the map be (c, n), then we update it according to

$$(c,\, n) \mapsto \left(\frac{c\,n + c'}{n+1}, n+1 \right) . \tag{7.7}$$

Once all cells have been visited, we render the host vehicle and visualize obstacles using the silhouettes as described above (see section 7.2.9).

7.4. Results

In the following section we present practical results of our system, measured on the application level. We concentrated mainly on the performance at daylight conditions using the side-view camera, but we also performed tests with the top-view camera and at different environmental conditions.

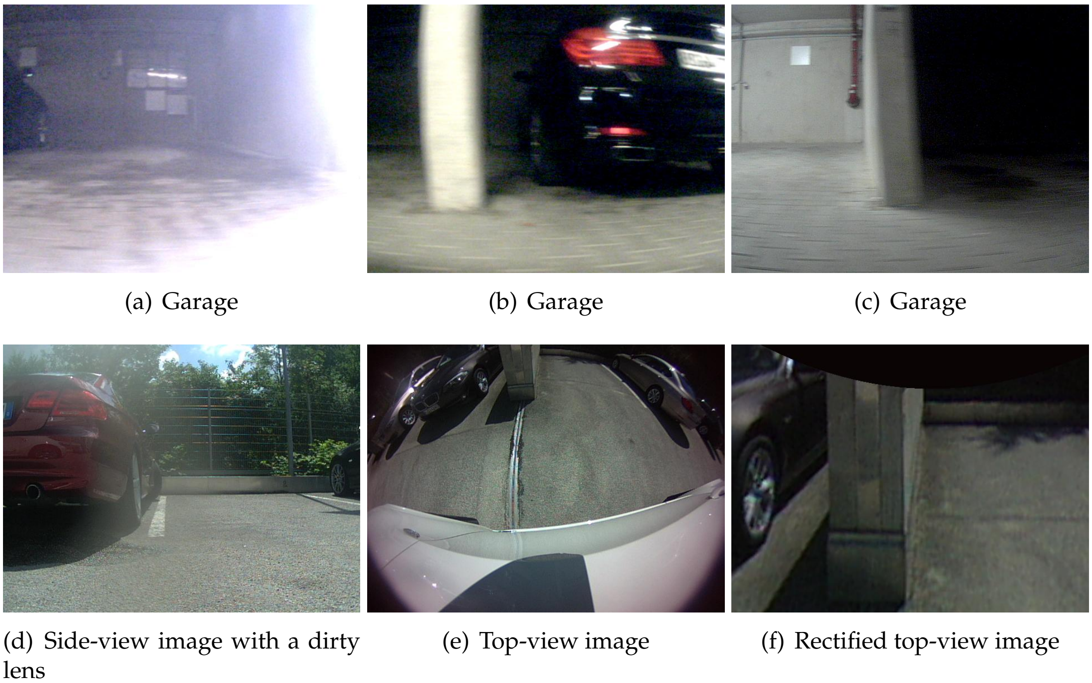
For the parking slot detection, the accuracy and the false detection rates (false positives and missed slots) are most important. For assessing the collision warning, we measured only the detection rates. And finally, for the augmented navigation we present pictures of the image-based rendering.

7.4.1. Test Methodology

Our goal was to test our method extensively in a large set of relevant scenarios. In our case, this included quantifying the performance in terms of measurement accuracy and detection rates of different algorithms and parameterizations. Especially for false detection rates, the number of test cases should be quite large. Thus, practical experimentation is usually very time-consuming and environmental influences make results of different test sessions hard or even impossible to compare. Further, at every test case, ground truth data must be acquired which requires time consuming labeling. Due to these reasons, we decided to resort to software-in-the-loop techniques [154, 150]: we recorded videos synchronized with data from the vehicle (*e.g.* odometry information). This allowed us to execute different configurations of our method on exactly the same set of scenarios. After recording the sequences we associated ground truth measurements to them. In every video we mark all frames where a parking slot is, at least, in half of the image visible. To each interval where a parking slot is visible we associate its ground truth size measured using the laser distancemeter. We compare the results of our and those of other methods to ground truth data and identify false detections.

The whole database contains over 120 GBytes of uncompressed video data (approximately 2 hours) of the side-view and top-view cameras and contains sequences of 718 parking slots. When recording the sequences we varied different parameters:

- Velocity: from idle speed up to 35 km/h, either with constant or varying speeds (by braking and accelerating).
- Yaw-rate: in most cases we drove on a straight line, but in some sequences we modified the steering angle (*e.g.* driving on a sinuous line).



(a) Garage

(b) Garage

(c) Garage

(d) Side-view image with a dirty lens

(e) Top-view image

(f) Rectified top-view image

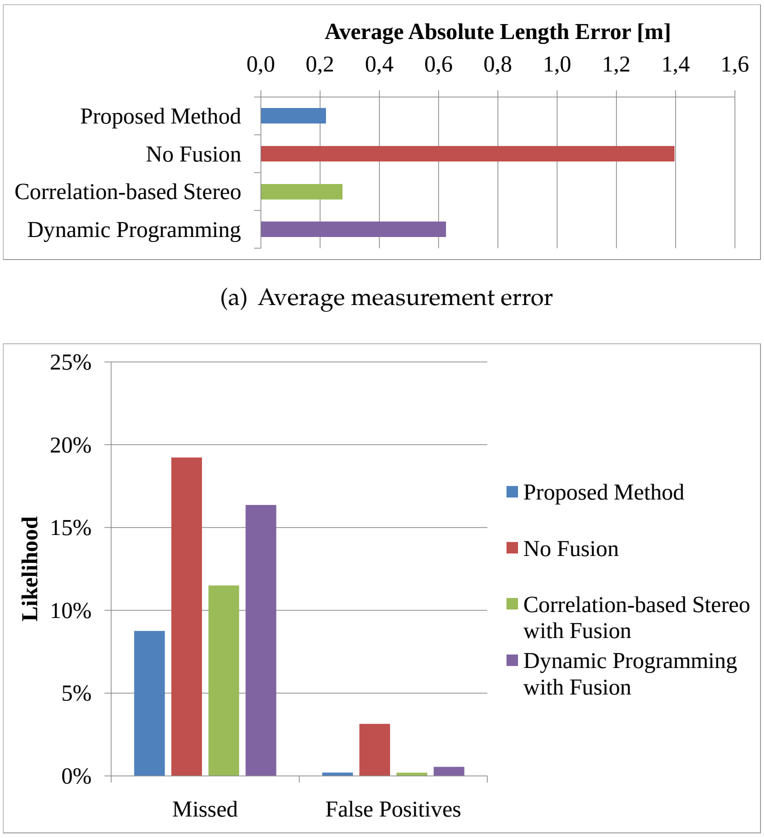
Figure 7.9. Examples of camera images used: (a-c) sequences recorded in the garage have very difficult lighting conditions due to inhomogeneous illumination (sun and neon light), (d) an image of the side-view camera with a dirty lens and (e) a top-view image before and (f) after undistortion.

- Slot length and depth (parallel and cross parking slots): lengths varied between 1.8 and 13.2 meters; depths between 2 and 10 meters. We configured the system in a way such that a parking space should provide at least an area of 5.5 × 2.5 meters.
- Illumination: daylight (sunny, cloudy and rainy) and a subterranean garage (inhomogeneously illuminated by sun and neon light; see Fig. 7.9 for example images).
- Dirtiness of the lens: the database also contains a few recordings where the lens was dirty (see Fig. 7.9 for example images).

We define that the *length* of a slot is the dimension being roughly parallel to the direction of the host vehicle and that the *depth* is orthogonal to the length. We measured every parking slot manually using a *Leica DISTO classic* laser distancemeter, which is very accurate in practice. When measuring these dimensions by hand, we could only achieve an accuracy of ±5 cm. The length is the minimal longitudinal size between two obstacles, but the depth is more difficult to define: in general, the depth is the distance between a close and a far boundary, both delimiting the parking space into which the vehicle must fit. The far boundary is usually given by some structural installations (the curb, fences, walls or other vehicles) but it may even not be present (then it may be given by ground markings or implicitly by a change in the asphalt). The close boundary is even more complicated. In some cases it is given by ground markings or a change in the asphalt, but in many other situations such indications are missing and often it is defined by a "thought line" which is tangential to other parking vehicles or may even depend on complex scene understanding. Therefore, we followed a two-fold strategy for the evaluation:

- The accuracy of length-measurements is evaluated using the whole database and the associated ground truth measurements.
- The accuracy of depth-measurements is evaluated on a smaller set of parking slots, where the far boundary was given by a wall or the curb. For the close boundary of a parking slot we used the maximum depth of the bounding obstacles to the left and right.

7.4.2. Analysis of the Stereo Pipeline



(a) Average measurement error

(b) False detection rates

Figure 7.10. The average measurement error (a) and false detection rates (b) of our method with different variations of the stereo pipeline at daylight conditions using the side-view camera: our real-time stereo with our temporal fusion, our real-time stereo without fusion, traditional real-time stereo [61, 104] with our fusion and dynamic programming [119] with our fusion.

The most important part of our method is the disparity computation stage which includes the single baseline stereo method and the temporal disparity fusion step. Fig. 7.11

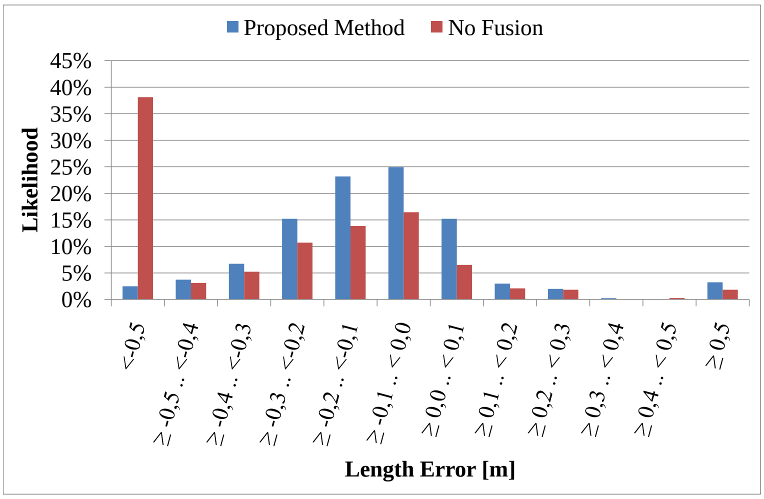


Figure 7.11. The distribution of measurement errors of our method with and without our temporal fusion at daylight conditions using the side-view camera. The fusion removes many outliers and therefore, less false matches are accumulated in the map. Hence, the measurements are more accurate.

shows that the temporal fusion is very important for accurate measurements and that the amount of outliers is very critical for the overall performance (see also Fig. 7.10). If no temporal fusion is performed, the disparity maps contain significantly more outliers and imprecise object boundaries. These characteristics lead to errors in the cumulative map and result in increased measurement errors. This becomes also obvious in Fig. 7.10 by an increase in the false detection rates. The temporal fusion makes the whole system much more reliable.

In Fig. 7.10 our proposed method, consisting of our real-time stereo method (section 4.4) with our fusion (chapter 6), is slightly better than traditional real-time stereo [61, 104] with fusion and is at the same time twice as fast. We also included dynamic programming [119] in our evaluation (also with temporal fusion enabled), because it also belongs to the class of efficient methods. However, the well known problem of streaking effects [119] limits the practical use. The different configurations of the stereo pipeline effect also the detection rates (see Fig. 7.10). The amount of outliers is a direct indicator for the robustness and thus a measure for the customer value.

7.4.3. Performance when using Top-View Cameras

Our test vehicle was equipped with side- and top-view cameras. When we recorded the sequences, we recorded the video streams of both cameras simultaneously and thus, we were able to determine the performances in exactly the same conditions. Fig. 7.12 shows the performance when using side- and top-view cameras: both systems offer the same potential of accuracy (in 40% of all measurements the error was between ±10 cm, for both systems) and it is more likely that the estimated size of a parking slot is too small, which is a desirable property. However, it is more robust to use side-view cameras: the disparity maps computed from rectified images of top-view cameras contain much more errors than

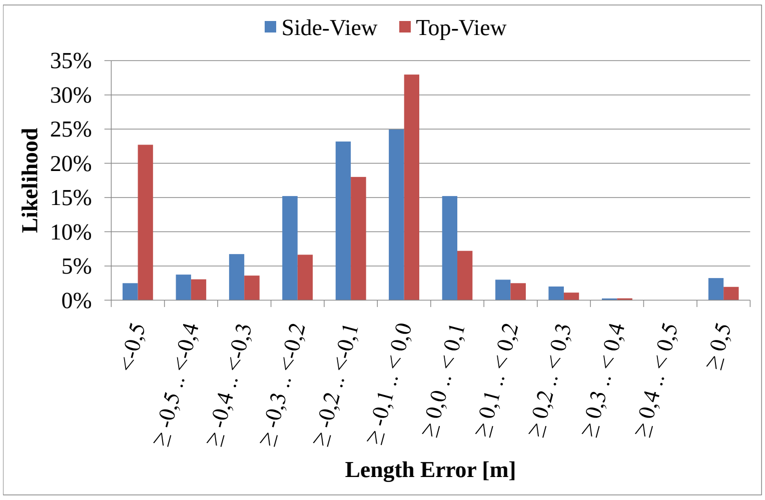


Figure 7.12. The performance of our proposal with side- and top-view cameras at daylight conditions. The bars show the distribution of measurement errors.

the ones from side-view cameras. This is due to the wide angle lenses used (less FOV and lower resolution in the region of interest) and a worse signal-to-noise ratio (SNR).

7.4.4. Analysis of Environmental Influences

We also recorded scenes with different environmental influences:

- **Daylight**: sequences recorded at daytime (1 hour after dawn and 1 hour before dusk) with sunny, cloudy or rainy conditions.
- **Dirty Lens**: we also recorded videos where the lens had (natural) dirt on it (composed of the remains of a dead insect; see Fig. 7.9 for example images).
- **Garage**: sequences recorded in a garage with artificial lighting. In these scenes, lighting conditions were very difficult (see Fig. 7.9 for example images). To some extent, this is due to the fact that sunlight is sometimes visible and the exposure control of the camera adapts permanently and switches between day- and night-mode.

Fig. 7.13 shows that the performance of the system is different for these use cases. Large mismeasurements are more likely in difficult situations but one important property always remains: it is very unlikely that the parking slot is measured too large. In only 5% of all cases, the length of the parking slot is over-estimated by more than 20 cm. This implies that if the system has found a free space, there is a very high reliability that the vehicle actually fits into it.

7.4.5. Comparison to other Methods

We compare our parking slot detection application to a feature-based method [151, 152] and a solution based on an ultrasonic sensor [112]. For the camera-based approaches, we used daylight sequences from the side-view camera. The feature-based method failed

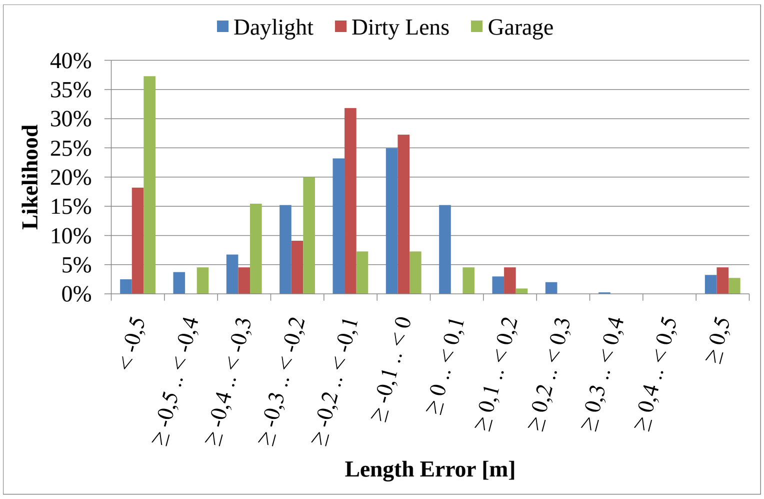


Figure 7.13. The distribution of measurement errors with different environmental influences: scenes recorded at daylight (sunny, cloudy or rainy), with a dirty lens and in a garage with very difficult lighting conditions.

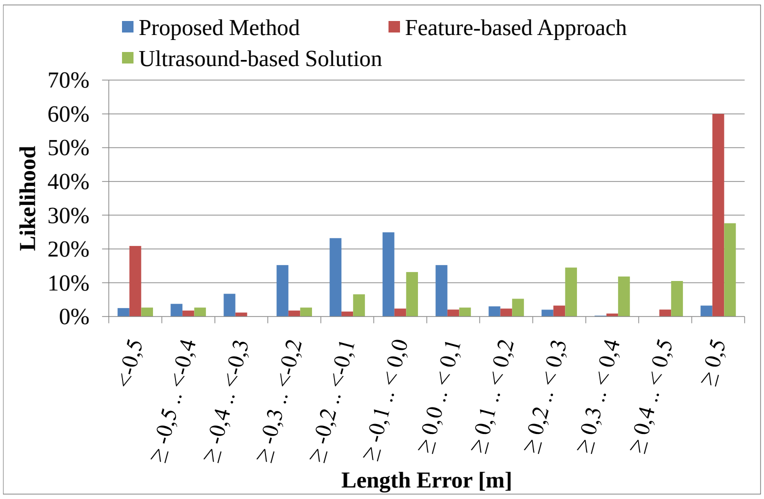


Figure 7.14. The distribution of measurement errors of different methods at daylight conditions: we compare our method based on dense motion-stereo to a feature-based method [151, 152] and a solution based on an ultrasonic sensor [112].

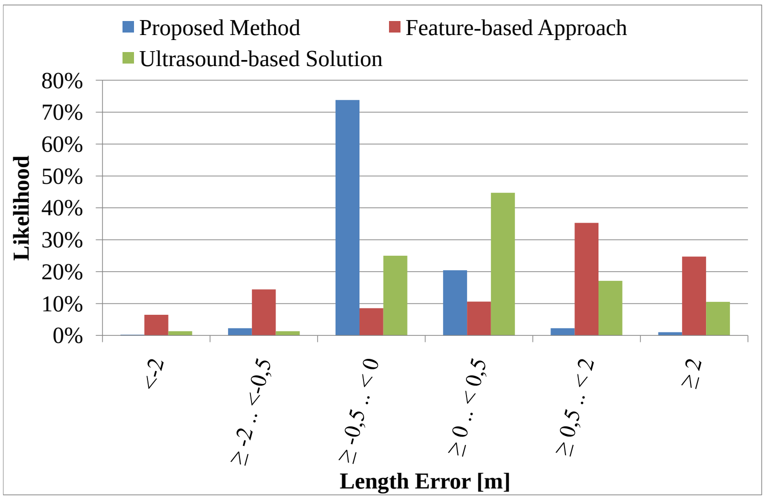


Figure 7.15. The distribution of measurement errors of different methods at daylight conditions: we compare our method based on dense motion-stereo to a feature-based method [151, 152] and a solution based on an ultrasonic sensor [112].

completely on the top-view videos. The measurement errors of the feature-based approach are much larger than the errors of our dense method. Only a few measurements have an absolute error less than 0.5 meters (see Fig. 7.14). Fig. 7.15 shows the same error distributions with coarser intervals and shows that our approach achieves a very high accuracy. The reason for the large errors of [151, 152] lies in the fact that the feature extractor often fails to detect features on object boundaries. In many cases, features are mainly detected at rims and license plates and this easily introduces an error of roughly 2 meters per parking spot. In other cases, features are completely missing at textureless objects like walls and this leads to much larger errors. Further, sometimes features are matched incorrectly (for example, at repetitive structures) and this usually results in measurements that are too small.

The approach based on the ultrasonic sensor [112] was only evaluated on parallel parking slots (in total, 114 test cases), because due to a limited depth range, it does not detect cross parking slots. This method had mainly problems when the obstacles had a complex 3D structure. For example, bushes are not detected reliably and curved object boundaries led to large errors. Also trailer hitches and small objects seem to negatively impact the measurement accuracy. There were a few false positives (only 4 cases), but probably due to measurement errors, the number of false negatives (*misses*) was with 28% quite high (in 32 cases). However, the overall impression is that it is a very reliable system, whose performance is invariant to illumination.

7.4.6. Accuracy of Depth Measurements

The accuracy of depth measurements is difficult to compare. The feature-based system of [151, 152] does not directly determine the depth of parking slots, so we cannot compare to them. The ultrasound-based solution [112] has a limited depth range (in practice, the maximum depth is between 3 and 4 meters) and so we evaluated it only on parallel parking slots.

Theoretical Discussion. In our approach, the depth Z is computed from a disparity d. The relative depth-error is according to Eq. (2.27) given by

$$Z_{err} = \frac{d_{err} \cdot Z}{f \cdot B + d_{err} \cdot Z} \tag{7.8}$$

with the focal length f, the baseline B and the error of disparity estimation d_{err}. In practice, the quantity fB is much larger than $d_{err}Z$, and therefore the error Z_{err} is approximately a linear function of Z. This means that for our system, range measurements using large values for Z are most challenging. Thus, to evaluate our proposal, we used large values for Z (*i.e.* only cross parking slots), in order to obtain an upper bound for the error.

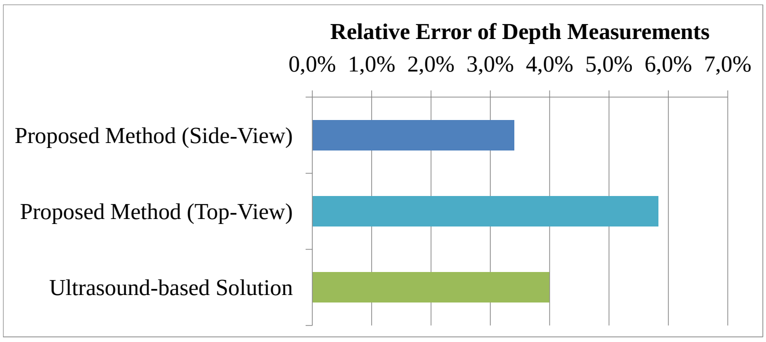


Figure 7.16. The relative error of depth measurements of different approaches: for the ultrasound-based approach, the depth of the parking spots was on average 2 meters and for our method we used only cross parking slots with depths between 5 and 6 meters (the distance between the camera and the far boundary was between 6 and 7 meters).

Fig. 7.16 shows the results of the depth accuracy: for the ultrasound-based approach, the depth of the parking spots was on average 2 meters and for our method we used only cross parking slots with depths between 5 and 6 meters (the distance between the camera and the far boundary was between 6 and 7 meters). Notably, if we assume that $d_{err} = 0.125$ then the theoretical error for a depth measurement at 6.5 meters is at 3.5% (for our side-view camera). The measurements of [112] were relatively accurate in practice. For the motion-stereo approach, the biggest challenge is given by repetitive structures: in one case the absolute error was 48 cm (10%) which was due to mismatches at a fence (repetitive structure). Also the characteristics of the camera play an important role: the worse SNR of the top-view camera nearly doubles the error.

7.4.7. Performance of the Collision Warning

We also evaluated the collision warning application. We tested it on 123 obstacles (see Fig. 7.18 for examples) and checked whether the warning was correct or not: if there is an object in the pivoting range of a door, a warning should be issued – if it is save to open all doors, then the warning should be suppressed. Based on these tests we determined the detection rates (see Fig. 7.17). In 100 cases the decision for the warning was correct

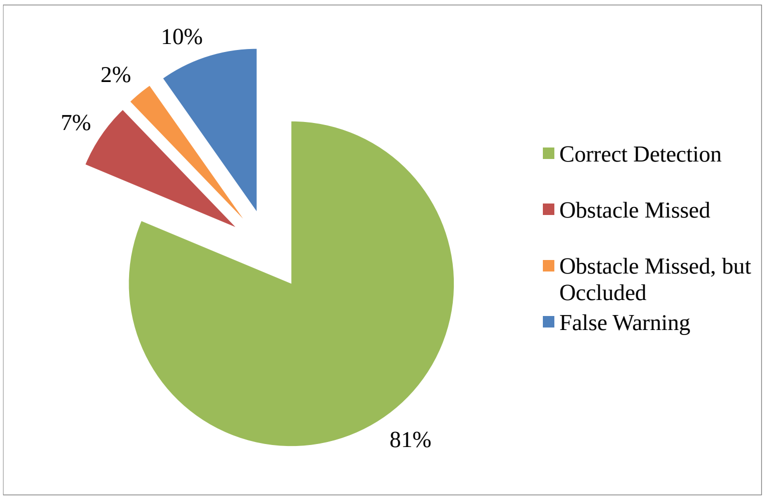


Figure 7.17. The performance of the collision warning at daylight conditions. We evaluated whether the system warned correctly for the presence of obstacles in the pivoting ranges of doors. In most cases, the decision upon the warning was correct (*Correct Detection*). In other cases, the position of the obstacle was estimated falsely outside or inside of pivoting ranges (*Obstacle Missed* or *False Warning*), or the collision-relevant part of the object was outside the FOV of the camera (*Obstacle Missed, but Occluded*).

(*i.e.* issuing the warning or not; see *Correct Detection*). In 8 cases the location of the obstacle was estimated too inaccurate (*Obstacle Missed*) and in 3 cases, the critical part of the obstacle was out of the FOV (*Obstacle Missed, but Occluded*; *e.g.* an attachable trailer and in one case the door would have hit a pole at a very high position). A *False Warning* happened in 12 cases: there was always an obstacle present, but the system falsely estimated that a collision is possible.

7.4.8. Qualitative Impressions of Augmented Parking

We present generated bird's eye views from different sequences in Fig. 7.19 and Fig. 7.20. Challenging were situations in which the ground was not completely flat: in the bottom example of Fig. 7.19 small distortions are visible in the rendered image. Also reflections lead to artifacts which may be irritating in the first place. Lastly, since we did not introduce photometric registration, coloring is often inconsistent within a rendering.

7.4.9. Execution Times

The execution times of the different processing steps can be found in Tab. 7.1. Dense stereo matching and the temporal fusion of disparity maps consume most time. We partitioned the whole system into several threads:

1. Acquisition-Thread: acquires video frames from the camera and performs the undistortion.

2. Stereo-Thread: runs the dense stereo matching algorithm.

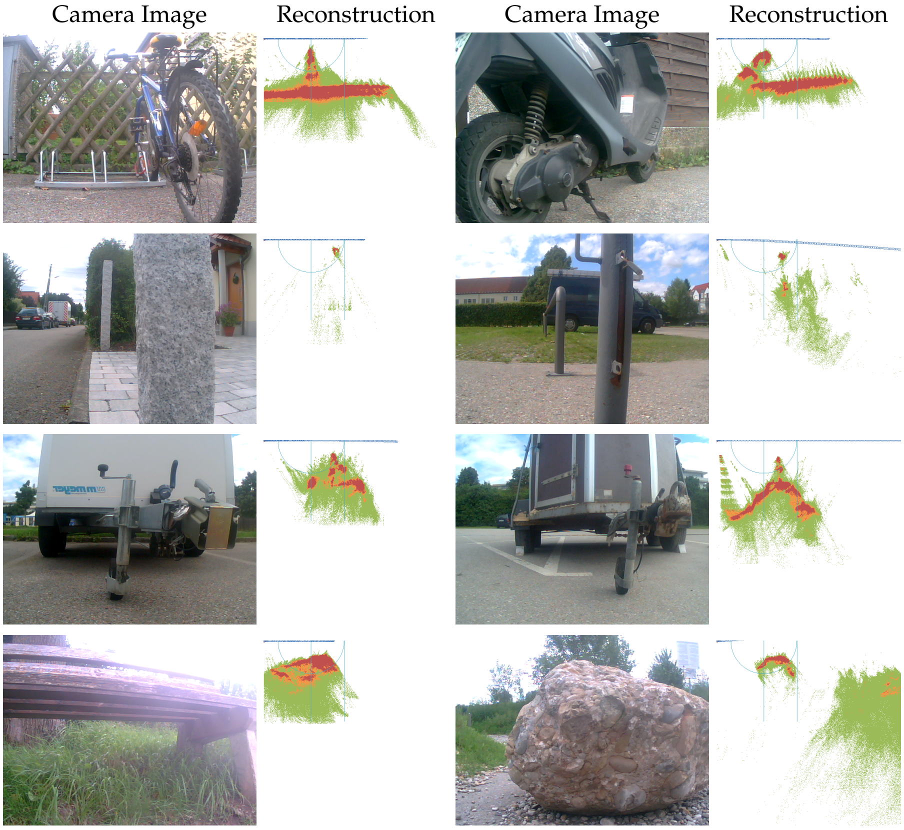


Figure 7.18. Examples for the Collision Warning: we present one camera image and the reconstruction obtained from segmented disparity maps. The objects (linewise from left to right and from top to bottom): a bicycle (no warning), a motorcycle (no warning), a pole made of stone, a pole made of steel, trailer 1, trailer 2, a bank and a stone.

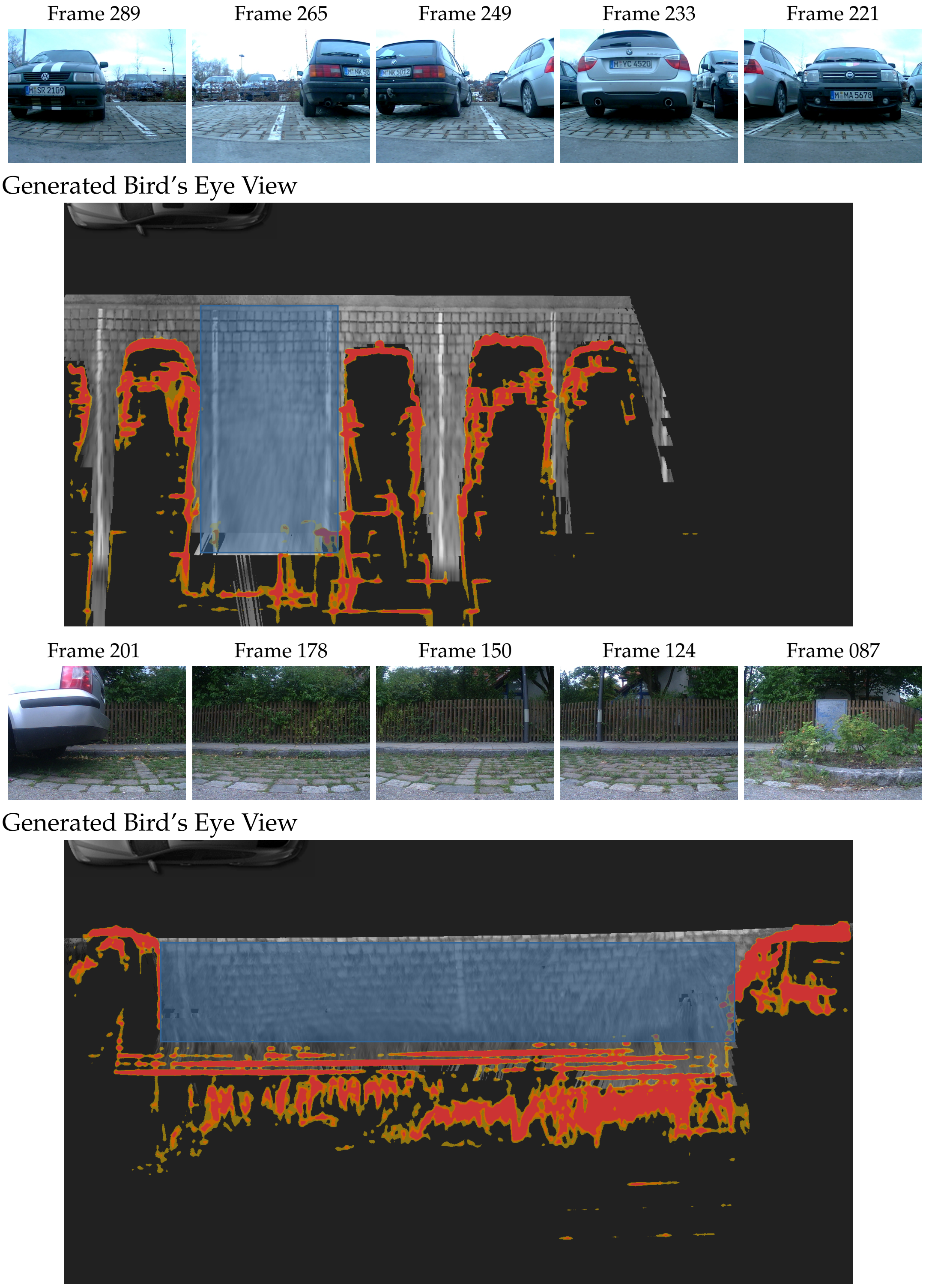


Figure 7.19. Examples for the bird’s eye views on different sequences: we show selected camera frames and the generated bird’s eye view. In the bottom example, the ground plane was not flat and caused the vehicle to pitch and movements in z-direction. This leads to distortions in the rendering.

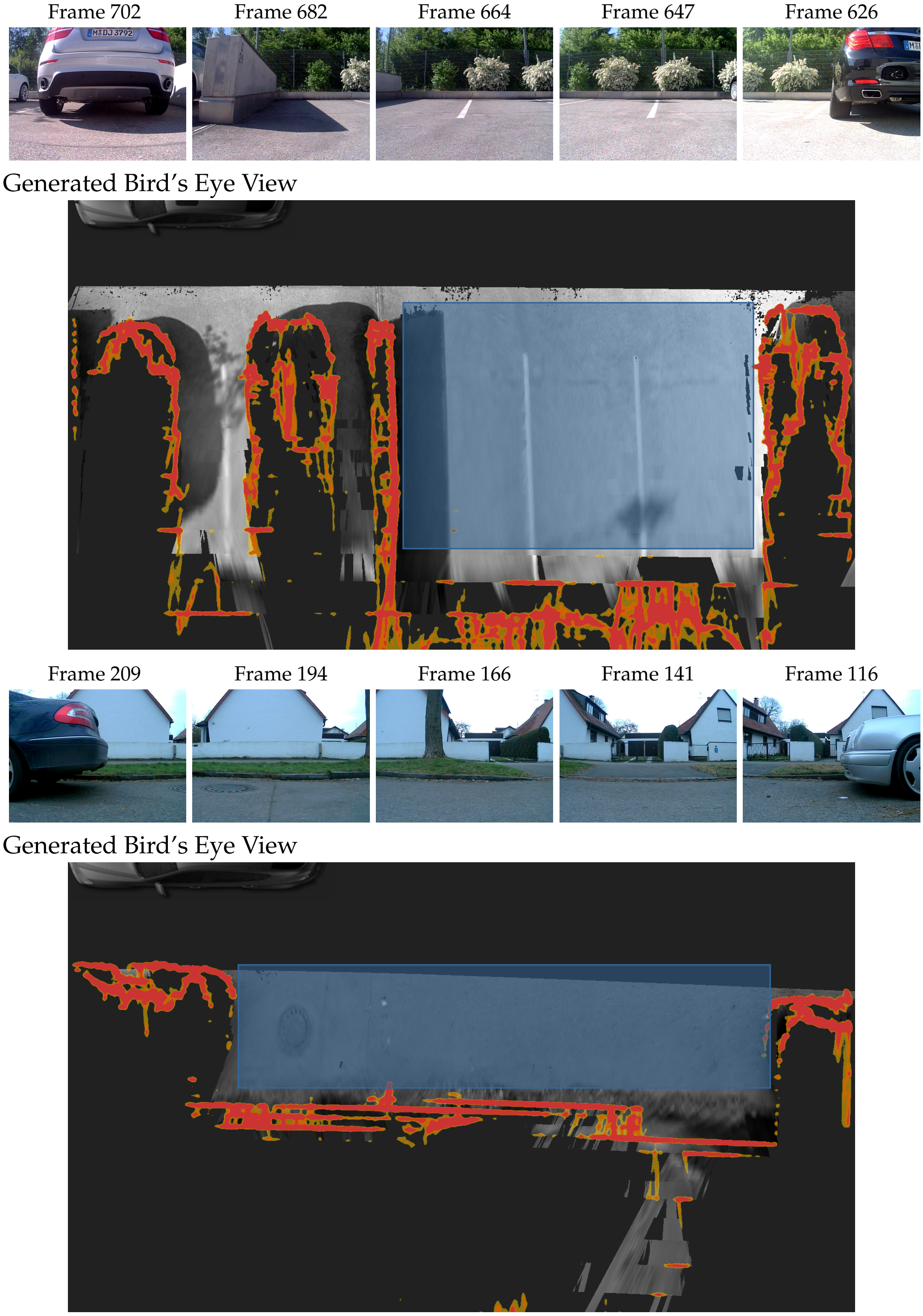


Figure 7.20. Examples for the bird's eye views on different sequences: we show selected camera frames and the generated bird's eye view. The top row is a good example for difficult lighting conditions.

Table 7.1. Execution times of the different processing steps on different processors: on the E8200 sub-pixel interpolation and many debug visualizations were disabled. The collision warning and augmented parking are only run if required (*i.e.* event-triggered).

Step	**Time: Q9300**	**Time: E8200**
	2.53 GHz, 4 cores	2.66 GHz, 2 cores
Undistortion	2 ms	2 ms
Rectification	0.5 ms	0.5 ms
Stereo Matching	15 ms	12 ms
Temporal Fusion	31 ms	29 ms
Segmentation	2 ms	2 ms
Parking Slot Detection	4-11 ms	1-9 ms
Collision Warning	*120 ms*	*80 ms*
Augmented Parking	*≈10,000 ms*	*≈10,000 ms*

3. Fusion-Thread: performs the temporal fusion of disparity maps.
4. Interpretation-Thread: executes the segmentations and the applications.
5. Visualization-Thread: cares about the user-interface.

The collision warning and the augmented parking are not real-time, which is tolerable: the collision detection is only run in the moment when the vehicle stops. The latency of the augmented parking can be easily reduced to a minimum, if processing is started as soon as a parking slot is found. Further, at the augmented parking most time is spent for perspective warping and might be accelerated with dedicated hardware. Usually, we use development settings for the interpretation thread, which includes many debug visualizations. In this case, the interpretation may consume up to 20 milliseconds, but by turning off all unnecessary outputs it can be tweaked to 3 ms (including the segmentation). Moreover, we assigned the highest priority to the fusion- and acquisition-threads, the stereo- and interpretation-threads ran with lower priority and the visualization-thread received the lowest priority.

The whole system is implemented in C++ (using Microsoft Visual C++ 9.0) and runs on Microsoft Windows. We achieved the best performance on a quad-core CPU with 2.53 GHz (Intel Core2 Extreme Q9300) and also on a dual-core CPU with 2.66 GHz (Intel Core2 Duo E8200). However, to allow real-time operation on the dual-core, we had to disable sub-pixel interpolations in the stereo and fusion algorithms. Further, time-critical parts are implemented using SIMD[2] instructions. We also performed tests with another mobile dual-core processor (in particular, a Intel T7600 with 2.33 GHz), but this CPU was not sufficient for real-time processing.

7.5. Discussion

Some of the challenges one has to face are shearing effects when using rolling shutter cameras, smearing with global shutter, and misalignments whenever interlaced images

[2] Single Instruction, Multiple Data: in particular, the SSE2 instruction set.

are involved. Moreover, the current cameras suffer from weak sensitivity in low light conditions. If an application is expected to work at night, some kind of active illumination would be required. This would involve additional costs, installation space, and often leads to legal conflicts in some countries. However, since parking maneuvers are performed with relatively low speeds and having upcoming high dynamic range imagers in mind, weaknesses of current technology are to some extent tolerable. Moreover, the temporal fusion of disparity maps turned out to be highly effective against these issues.

Furthermore, it must be noted that there are mathematical limitations for monocular systems in non-rigid scenes: in certain cases, if the motion-vectors of the host vehicle and an obstacle are collinear then the motion of the obstacle is hard or impossible to recover without additional knowledge or interpretation of data. In the worst case, this leads to wrong distances. However, our focus lies on comfort functions: the detection of parking slots and avoidance of minor damage is not safety critical. Therefore, we assume that the scene is static and there are also techniques available to detect moving obstacles by introducing other constraints [78].

Compared to the feature-based approach [151, 152], our approach based on our dense motion-stereo pipeline has important advantages:

- Redundancy of measurements due to overlap of images: redundancy can be systematically utilized to detect wrong measurements and to improve accuracy.

- Higher detection rate of obstacles: while feature-based approaches are usually specialized in a specific class of features (*e.g.* corners and edges), problems arise if such features are absent (*e.g.* regions with low texture). In our experiments the dense approach turned out to be much more flexible and detects almost all obstacles.

- Higher measurement accuracy of the measured dimensions of parking slots: high accuracy requires a precise detection of object boundaries. Feature-based approaches may miss detecting features which lie exactly on object boundaries. Such behavior introduces large measurement errors.

7.6. Summary

This chapter presents a generic method for environment modeling based on dense motion-stereo and demonstrates its flexibility using different applications for parking assistance. Our processing pipeline exploits the principle of *dense* motion-stereo and relies on the binocular and multi-view stereo methods presented in the previous chapters: after stereo matching, we fuse the history of disparity maps probabilistically to obtain for every camera location the most probable disparity map that exposes a minimum amount of outliers. In every fused disparity map we detect the ground plane, obstacles and from that a silhouette which limits the free space. Then, we combine all these partial silhouettes so that over time a global model of the environment is created incrementally.

Using this model, we perform an *Automatic Parking Slot Detection* by examining the free space. Further, we use the disparity maps to obtain a local 3D reconstruction of specific regions of interest (for example, the pivoting ranges of doors). Using such a local 3D reconstruction, we perform a collision analysis and, if necessary, issue a *Collision Warning* to occupants to prevent minor damages. Another application is *Augmented Parking* and uses image-based rendering to compute a virtual bird's eye view to visualize the positions of the host vehicle, obstacles and the parking slot to the driver.

The accuracy and reliability of our approach is demonstrated via exhaustive experimentation and comparison to solutions based on an ultrasonic sensor and feature-based matching. The results show clearly that our proposal achieves high reliability, measures accurately and is very flexible.

8. Conclusion

Do not disturb my circles! Attributed to ARCHIMEDES of Syracuse

In the course of this thesis, we contributed novel computer vision methods for efficient and accurate binocular stereo matching, for multi-view stereo reconstruction and we developed innovative functionalities for automotive parking assistance.

In particular, we introduced a novel disparity computation algorithm for efficient stereo vision. It is mainly based on the observation that in dense stereo matching adjacent pixels have similar cost functions and that these cost functions can be minimized more efficiently than in traditional approaches. In essence, our real-time stereo method performs a pixelwise minimization combined with a propagation step that diffuses disparity values through the local neighborhood. The iterative computation principle results in less required processing power and the ability to process images without knowledge about the maximum disparity. The efficiency of this idea is emphasized by several generalizations including robust cost functions like NCC or Census Transform and robust stereo matching that accounts for decalibrated stereo rigs. We also introduce a simple post-processing routine to enhance the localization of depth discontinuities in disparity maps computed using local window-based stereo methods. To summarize, we introduced a powerful disparity computation algorithm with various generalizations which replaces the simple winner-takes-all strategy found in many local matching approaches.

We also presented a novel stereo method based on simulated random walks for accurate stereo vision. The main idea is that pixels with a similar color usually also have a similar depth and that simulations of random walks will help in determining small localized segments with similar color. We introduced several measures to make the matching process robust to challenging problems like discontinuities, occlusions and slanted surfaces. The strengths of our method are mainly achieved by using random walks as matching primitives because they, in some sense, perform a localized soft segmentation. Further, we introduce a few a priori surface orientations for cost aggregation to handle slanted surfaces and by using left-right random walk simulations we increase robustness in occluded regions. Our novel voting strategy increases the general robustness in all image regions. Finally, we perform a propagation of confident disparities into inconsistent regions and use global optimization on a probability distribution over disparities to handle ambiguities. We showed experimentally that our proposed method computes very reliable and very accurate disparity maps which is strengthened by achieving the 2nd place at the Middlebury benchmark.

Chapter 6 focuses on the probabilistic fusion of disparity maps, in order to mainly improve the quality and robustness of depth measurements. Our proposal is quite efficient and allows real-time operation in our vehicle. Our efforts resulted in a novel probabilistic approach for the fusion of disparity maps in classical multi-view or motion-stereo configurations. For that, we compute a global probability density function and use several new concepts like a reprojection using a reliable area for efficient visibility determination, a generic probabilistic framework that uses projection uncertainties for robustness against outliers, a distinction between left-right and right-left stereo matching for sharp object boundaries, and hole-filling for improved quality in occluded regions. In our comparisons to the current state of the art we showed that our stereo fusion achieves very good results and a high efficiency.

Finally, we presented our camera-based parking assistant that makes use of the algorithms introduced for binocular and multi-view stereo. We implemented three prototypical functionalities, namely an automatic detection of parallel and cross parking slots, a collision warning that monitors the pivoting ranges of the doors, and an image-based rendering technique called augmented parking which visualizes the surroundings of the host vehicle. Due to the high efficiency of our stereo matching and our disparity fusion methods, a real-time implementation only on a CPU was possible. Finally, the tests of these customer-oriented functionalities in many difficult scenarios underline once more that the probabilistic stereo fusion is an integral part for robust motion-stereo processing.

A. List of Symbols

Mathematical Entities

x	Lower-case italic character: a scalar value
d	A disparity value
Z	The depth of a 3D point
B	The baseline between two cameras
f	The focal length of a camera
$\mathbf{x}$	A 2D point in the image plane
$\mathbf{X}$	A 3D point in 3-space
$\mathbf{P}$	The projection matrix of camera $\mathbf{P}$
$\mathbf{K}$	The intrinsic calibration matrix of a camera
$\mathbf{C}$	The optical center of a camera
$\mathbf{R}$	A rotation matrix
$\mathbf{I}$	The identity matrix
$\mathbf{H}$	A homography
$\mathbf{F}$	A fundamental matrix

Structures and Functions

$\mathcal{I}$	An image
$\mathcal{D}$	A disparity map
$\mathcal{F}$	A flow field
$\mathcal{R}$	A random walk (see section 5.2.2 on page 66)
$\mathcal{V}$	The voting space (see section 5.2.5 on page 68)
$E(\cdot)$	An energy function
$\mathcal{C}_M(\cdot)$	Pixelwise matching costs
$\mathcal{C}_A(\cdot)$	Aggregated matching costs
$\Theta_C^{A,B}(\mathbf{x}, d)$	A transfer function (see section 6.2.2 on page 80)
$p(A \mid B)$	The probability of A given B

Operators

$\mathbf{x}_1 \times \mathbf{x}_2$	The cross-product of $\mathbf{x}_1$ and $\mathbf{x}_2$

Bibliography

[1] François Alter, Yasuyuki Matsushita, and Xiaoou Tang. An intensity similarity measure in low-light conditions. In *European Conference on Computer Vision (ECCV)*, pages 267–280, 2006.

[2] Various Authors. Wiktionary, the free dictionary. `http://de.wiktionary.org/`, 2012. [Online; accessed 1-November-2012].

[3] Ali J. Azarbayejani and Alex P. Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 17:562–575, June 1995.

[4] Simon Baker, Stefan Roth, Daniel Scharstein, Michael J. Black, J.P. Lewis, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Conference on Computer Vision (ICCV)*, 0:1–8, 2007.

[5] Jasmine Banks and Peter I. Corke. Quantitative evaluation of matching methods and validity measures for stereo vision. *International Journal of Robotics Research (IJRR)*, 20(7):512–532, 2001.

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

[7] Stanley T. Birchfield, Braga Natarajan, and Carlo Tomasi. Correspondence as energy-based segmentation. *Image and Vision Computing*, 25(8):1329–1340, 2007.

[8] Stanley T. Birchfield and Carlo Tomasi. Depth discontinuities by pixel-to-pixel stereo. In *International Conference on Computer Vision (ICCV)*, pages 1073–1080, 1998.

[9] Stanley T. Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 20(4):401–406, 1998.

[10] Stanley T. Birchfield and Carlo Tomasi. Depth discontinuities by pixel-to-pixel stereo. *International Journal of Computer Vision (IJCV)*, 35(3):269–293, 1999.

[11] Michael Bleyer and Margit Gelautz. A layered stereo matching algorithm using image segmentation and global visibility constraints. *Photogrammetry and Remote Sensing*, 59:128–150, 2005.

[12] Michael Bleyer and Margit Gelautz. Simple but effective tree structures for dynamic programming-based stereo matching. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, pages 415–422, 2008.

[13] Michael Bleyer and Margrit Gelautz. Temporally consistent disparity maps from uncalibrated stereo videos. In *Proceedings of the 6th International Symposium on Image and Signal Processing and Analysis (ISPA 2009), Special Session on Stereo Analysis and 3D Video/TV*. IEEE, 2009. Vortrag: International Symposium on Image and Signal Processing and Analysis 2009, Salzburg; 2009-09-16 – 2009-09-18.

[14] Michael Bleyer, Margrit Gelautz, Carsten Rother, and Christoph Rhemann. A stereo approach that handles the matting problem via image warping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[15] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo - stereo matching with slanted support windows. In *British Machine Vision Conference (BMVC)*, pages 14.1–14.11, 2011.

[16] Michael Bleyer, Carsten Rother, and Pushmeet Kohli. Surface stereo with soft segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[17] Michael Bleyer, Carsten Rother, Pushmeet Kohli, Daniel Scharstein, and Sudipta Sinha. Object stereo: Joint stereo matching and object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[18] Aaron F. Bobick and Stephen S. Intille. Large occlusion stereo. *International Journal of Computer Vision (IJCV)*, 33(3):181–200, 1999.

[19] Thomas Bonfort and Peter Sturm. Voxel Carving for Specular Surfaces. In *International Conference on Computer Vision (ICCV)*, volume 1, pages 691–696, Nice, France, 2003. IEEE Computer Society.

[20] Yuri Boykov. http://vision.csd.uwo.ca/code/, August 2011.

[21] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. In *International Conference on Computer Vision (ICCV)*, pages 377–384, 1999.

[22] Derek Bradley, Tamy Boubekeur, and Wolfgang Heidrich. Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[23] Duane C. Brown. Decentering distortion of lenses. *Photogrammetric Engineering*, 32(3):444–462, 1966.

[24] E. Bruno and D. Pellerin. Robust motion estimation using spatial gabor-like filters. *Signal Processing*, 82:297–309, February 2002.

[25] Alan Brunton, Chang Shu, and Gerhard Roth. Belief propagation on the gpu for stereo vision. In *Canadian Conference on Computer and Robot Vision*, pages 76–81, 2006.

[26] Jan Čech and Radim Šára. Efficient sampling of disparity space for fast and accurate matching. In *CVPR Workshop Towards Benchmarking Automated Calibration, Orientation and Surface Reconstruction from Images*, 2007.

[27] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. Technical report, Graz University of Technology, 2010.

[28] L. Cohen, L. Vinet, P. T. Sander, and A. Gagalowicz. Hierarchical region based stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1989.

[29] Robert T. Collins. A space-sweep approach to true multi-image matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 358–363, 1996.

[30] A. E. Conrady. Decentered lens-systems. *Monthly Notices of the Royal Astronomical Society*, 79:384–390, 1919.

[31] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, pages 303–312, 1996.

[32] Erez Dagan, Ofer Mano, Gideon P. Stein, and Amnon Shashua. Forward collision warning with a single camera. In *Proceedings of IEEE Intelligent Vehicles Symposium*, 2004.

[33] Thao Dang, Christian Hoffmann, and Christoph Stiller. Continuous stereo self-calibration by camera parameter tracking. *IEEE Transactions on Image Processing*, 18(7):1536–1550, 2009.

[34] Frederic Devernay and Olivier D. Faugeras. Straight lines have to be straight. *Machine Vision and Applications (MVA)*, 13(1):14–24, 2001.

[35] Luigi Di Stefano, Massimiliano Marchionni, and Stefano Mattoccia. A fast area-based stereo matching algorithm. *Image and Vision Computing*, 22(12):983–1005, Oct 2004.

[36] Karl-Heinz Dietsche and Thomas Jäger. *Kraftfahrtechnisches Taschenbuch*. Friedr. Vieweg & Sohn Verlag, Wiesbaden, 25 edition, 2003. ISBN 3-528-23876-3.

[37] Geoffrey Egnal. Mutual information as a stereo correspondence measure. Technical Report MS-CIS-00-20, University of Pennsylvania, 2000.

[38] Olivier Faugeras, B. Hotz, Hervé Mathieu, T. Viéville, Zhengyou Zhang, Pascal Fua, Eric Théron, Laurent Moll, Gérard Berry, Jean Vuillemin, Patrice Bertin, and Catherine Proy. Real time correlation-based stereo: algorithm, implementations and applications. Technical Report RR-2013, INRIA, 1993.

[39] Olivier D. Faugeras and Renaud Keriven. Variational principles, surface evolution, pdes, level set methods, and the stereo problem. *IEEE Transactions on Image Processing*, 7(3):336–344, 1998.

[40] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient belief propagation for early vision. *International Journal of Computer Vision (IJCV)*, 70(1):41–54, 2006.

[41] Katia Fintzel, R. Bendahan, and Sylvain Bougnoux. 3d parking assistant system. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 881–886, 2004.

[42] David J. Fleet and Allan D. Jepson. Computation of component image velocity from local phase information. *International Journal of Computer Vision (IJCV)*, 5:77–104, September 1990.

[43] David J. Fleet, Allan D. Jepson, and Michael R. M. Jenkin. Phase-based disparity measurement. *CVGIP: Image Underst.*, 53:198–210, March 1991.

[44] Pascal Fua. A parallel stereo algorithm that produces dense depth maps and preserves image features. *Machine Vision and Applications (MVA)*, 6(1):35–49, December 1993.

[45] Pascal Fua and Yvan G. Leclerc. Object-centered surface reconstruction: Combining multi-image stereo and shading. *International Journal of Computer Vision (IJCV)*, 16(1):35–56, 1995.

[46] David Gallup, Jan-Michael Frahm, Philippos Mordohai, Qingxiong Yang, and Marc Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[47] Pau Gargallo and Peter Sturm. Bayesian 3d modeling from images using multiple depth maps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 885–891, 2005.

[48] Dariu Gavrila. Traffic sign recognition revisited. In *Pattern Recognition (DAGM)*, pages 86–93, 1999.

[49] Dariu Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82–98, 1999.

[50] Dariu Gavrila. Pedestrian detection from a moving vehicle. In *European Conference on Computer Vision (ECCV)*, pages 37–49, 2000.

[51] Stefan K. Gehrig, Felix Eberli, and Thomas Meyer. A real-time low-power stereo vision engine using semi-global matching. In Mario Fritz, Bernt Schiele, and Justus H. Piater, editors, *Computer Vision Systems*, volume 5815 of *Lecture Notes in Computer Science*, pages 134–143. Springer Berlin Heidelberg, 2009.

[52] Joel Gibson and Oge Marques. Stereo depth with a unified architecture gpu. *Computer Vision and Pattern Recognition Workshop*, 0:1–6, 2008.

[53] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. Multi-view stereo for community photo collections. In *International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

[54] Leo Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(11):1768–1783, 2006.

[55] Richard Hartley. Lines and points in three views and the trifocal tensor. *International Journal of Computer Vision (IJCV)*, 1997.

[56] Richard I. Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[57] Carlos Hernández and Francis Schmitt. Silhouette and stereo fusion for 3d object modeling. *Computer Vision and Image Understanding*, 96(3):367–392, 2004.

[58] Carlos Hernández, George Vogiatzis, and Roberto Cipolla. Probabilistic visibility for multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[59] Heiko Hirschmüller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 807–814, 2005.

[60] Heiko Hirschmüller. Stereo vision in structured environments by consistent semi-global matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2386–2393, 2006.

[61] Heiko Hirschmüller, Peter R. Innocent, and Jon Garibaldi. Real-time correlation-based stereo vision with reduced border errors. *International Journal of Computer Vision (IJCV)*, 47(1-3):229–246, 2002.

[62] Heiko Hirschmüller and Daniel Scharstein. Evaluation of cost functions for stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007.

[63] Heiko Hirschmüller and Daniel Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 31(9):1582–1599, 2009.

[64] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, 1981.

[65] Alexander Hornung and Leif Kobbelt. Robust and efficient photo-consistency estimation for volumetric 3d reconstruction. In *European Conference on Computer Vision (ECCV)*, pages 179–190, 2006.

[66] Asmaa Hosni, Michael Bleyer, Margit Gelautz, and Christoph Rhemann. Local stereo matching using geodesic support weights. In *IEEE International Conference on Image Processing (ICIP)*, 2009.

[67] Xiaofei Huang. Cooperative optimization for energy minimization: A case study of stereo matching. *CoRR*, abs/cs/0701057, 2007.

[68] John Immerkær. Fast noise variance estimation. *Computer Vision and Image Understanding*, 64(2):300–302, 1996.

[69] Hailin Jin, Stefano Soatto, and Anthony J. Yezzi. Multi-view stereo beyond lambert. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 171–178, 2003.

[70] Ho Gi Jung, Dong Suk Kim, and Pal Joo Yoon. Parking slot markings recognition for automatic parking assist system. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 106–113, 2006.

[71] Ho Gi Jung, Dong Suk Kim, Pal Joo Yoon, and Jaihie Kim. Light stripe projection based parking space detection for intelligent parking assist system. In *Proceedings of IEEE Intelligent Vehicles Symposium*, 2007.

[72] Nico Kämpchen, Uwe Franke, and Rainer Ott. Stereo vision based pose estimation of parking lots using 3d vehicle models. In *Proceedings of IEEE Intelligent Vehicles Symposium*, 2002.

[73] Takeo Kanade and Masatoshi Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 16(9):920–932, 1994.

[74] Sing Bing Kang and Richard Szeliski. Extracting view-dependent depth maps from a collection of images. *International Journal of Computer Vision (IJCV)*, 58(2):139–163, 2004.

[75] Sing Bing Kang, Richard Szeliski, and Jinxiang Chai. Handling occlusions in dense multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 103–110, 2001.

[76] Juho Kannala and Sami S. Brandt. A generic camera model and calibration for conventional, wide-angle and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(8):1335–1340, August 2006.

[77] Junhwan Kim, Vladimir Kolmogorov, and Ramin Zabih. Visual correspondence using energy minimization and mutual information. In *International Conference on Computer Vision (ICCV)*, pages 1033–1040, 2003.

[78] Jens Klappstein, Fridtjof Stein, and Uwe Franke. Monocular motion detection using spatial constraints in a unified manner. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 261–267, 2006.

[79] Andreas Klaus, Mario Sormann, and Konrad Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *International Conference on Pattern Recognition (ICPR)*, pages 15–18, 2006.

[80] Reinhard Koch, Marc Pollefeys, and Luc J. Van Gool. Multi viewpoint stereo from uncalibrated video sequences. In *European Conference on Computer Vision (ECCV)*, pages 55–71, 1998.

[81] Reinhard Koch, Marc Pollefeys, and Luc J. Van Gool. Robust calibration and 3d geometric modeling from large collections of uncalibrated images. In *Pattern Recognition (DAGM)*, pages 413–420, 1999.

[82] Kalin Kolev, Maria Klodt, Thomas Brox, and Daniel Cremers. Continuous global optimization in multiview 3d reconstruction. *International Journal of Computer Vision (IJCV)*, 84(1):80–96, August 2009.

[83] Kalin Kolev, Maria Klodt, Thomas Brox, Selim Esedoglu, and Daniel Cremers. Continuous global optimization in multiview 3d reconstruction. In *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, volume 4679 of *LNCS*, pages 441–452, Ezhou, China, August 2007. Springer.

[84] Vladimir Kolmogorov and Ramin Zabih. Computing visual correspondence with occlusions using graph cuts. In *International Conference on Computer Vision (ICCV)*, pages 508–515, 2001.

[85] Vladimir Kolmogorov and Ramin Zabih. Multi-camera scene reconstruction via graph cuts. In *European Conference on Computer Vision (ECCV)*, pages 82–96, 2002.

[86] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 26(2):147–159, 2004.

[87] Chris Kreucher, Sridhar Lakshmanan, and Karl Kluge. A driver warning system based on the lois lane detection algorithm. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 17–22, Stuttgart, 1998.

[88] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[89] E. Scott Larsen, Philippos Mordohai, Marc Pollefeys, and Henry Fuchs. Temporally consistent reconstruction from multiple video streams using enhanced belief propagation. In *International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

[90] Victor S. Lempitsky, Carsten Rother, and Andrew Blake. Logcut - efficient graph cut optimization for markov random fields. In *International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

[91] C. Leung, B. Appleton, B.C. Lovell, and Changming Sun. An energy minimisation approach to stereo-temporal dense reconstruction. In *International Conference on Pattern Recognition (ICPR)*, volume 4, pages 72–75, 2004.

[92] Michael H. Lin and Carlo Tomasi. Surfaces with occlusions from layered stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[93] Ye Lu, Jason Z. Zhang, Q. M. Jonathan Wu, and Ze-Nian Li. A survey of motion-parallax-based 3-d reconstruction algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 34(4):532–548, 2004.

[94] Quang-Tuan Luong and Thierry Viéville. Canonical representations for the geometries of multiple projective views. *Computer Vision and Image Understanding*, 64:193–229, September 1996.

[95] Julian Magarey and Anthony Dick. Multiresolution stereo image matching using complex wavelets. In *Proceedings of the 14th International Conference on Pattern Recognition-Volume 1 - Volume 1*, International Conference on Pattern Recognition (ICPR), page 4, Washington, DC, USA, 1998. IEEE Computer Society.

[96] M. Mählisch, M. Oberländer, O. Löhlein, D. M. Gavrila, and W. Ritter. A multiple detector approach to low-resolution fir pedestrian recognition. In *Proceedings of IEEE Intelligent Vehicles Symposium*, Las Vegas, USA, 2005.

[97] David Marr and Tomaso Poggio. Cooperative computation of stereo disparity. *Science*, 194(4262):283–287, 1976.

[98] Stefano Mattoccia, Federico Tombari, and Luigi Di Stefano. Stereo vision enabling precise border localization within a scanline optimization framework. In *Asian Conference on Computer Vision (ACCV)*, 2007.

[99] Helmut Mayer. Analysis of means to improve cooperative disparity estimation. In *In International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, volume XXXIV*, pages 25–31, 2003.

[100] Paul Merrell, Amir Akbarzadeh, Liang Wang, Jan-Michael Frahm, Ruigang Yang, and David Nistér. Real-time visibility-based fusion of depth maps. In *International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

[101] Joris M. Mooij and Hilbert J. Kappen. Sufficient conditions for convergence of the sum-product algorithm. *IEEE Transactions on Information Theory*, 53(12):4422–4437, 2007.

[102] Daniel D. Morris and Takeo Kanade. Image-consistent surface triangulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1332–1338, 2000.

[103] Mikhail Mozerov, Vitaly Kober, and Tae-Sun Choi. Improved motion stereo matching based on a modified dynamic programming. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2501–2505, 2000.

[104] Karsten Mühlmann, Dennis Maier, Jürgen Hesser, and Reinhard Männer. Calculating dense disparity maps from color stereo images, an efficient implementation. *International Journal of Computer Vision (IJCV)*, 47(1-3):79–88, 2002.

[105] David Nistér. Frame decimation for structure and motion. In *in: 3D Structure from Images-SMILE 2000, LNCS*, pages 17–34. Springer-Verlag, 2001.

[106] Masatoshi Okutomi and Takeo Kanade. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 15(1):353–363, 1993.

[107] Wan-Joo Park, Byung-Sung Kim, Dong-Eun Seo, Dong-Suk Kim, and Kwae-Hi Lee. Parking space detection using ultrasonic sensor in parking assistance system. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 1039–1044, 2008.

[108] Karl Pearson. The problem of the random walk. *Nature*, 72(1865):294, July 1905.

[109] Josef Pohl, M. Sethsson, P. Degerman, and J. Larsson. A semi-automated parallel parking system for passenger cars. In *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, volume 220, pages 53–65, 2006.

[110] Jean-Philippe Pons, Renaud Keriven, and Olivier D. Faugeras. Modelling dynamic scenes by registering multi-view image sequences. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 822–827, 2005.

[111] Jean-Philippe Pons, Renaud Keriven, Olivier D. Faugeras, and Gerardo Hermosillo. Variational stereovision and 3d scene flow estimation with statistical similarity measures. In *International Conference on Computer Vision (ICCV)*, pages 597–602, 2003.

[112] Alfred Pruckner, Frank Gensler, Karl-Heinz Meitinger, Harald Gräf, Helmut Spannheimer, and Klaus Gresser. Der parkassistent – ein weiteres innovatives fahrerassistenzsystem zum thema connecteddrive aus der bmw-fahrzeugforschung. In *Braunschweiger Symposium*, 2003.

[113] Marcus Rohrbach, Markus Enzweiler, and Dariu M. Gavrila. High-level fusion of depth and intensity for pedestrian classification. In *Pattern Recognition (DAGM)*, pages 101–110, 2009.

[114] Ilya D. Rosenberg, Philip L. Davidson, Casey M. R. Muller, and Jefferson Y. Han. Real-time stereo vision using semi-global matching on programmable graphics hardware. In *SIGGRAPH 2006 Sketches*, 2006.

[115] Terence D. Sanger. Stereo disparity computation using gabor filters. *Biological Cybernetics*, 59(6):405–418, October 1988.

[116] Tomokazu Sato, Masayuki Kanbara, Naokazu Yokoya, and Haruo Takemura. Dense 3-d reconstruction of an outdoor scene by hundreds-baseline stereo using a hand-held video camera. *International Journal of Computer Vision (IJCV)*, 47:119–129, 2002.

[117] Alexander Schanz. *Fahrerassistenz zum automatischen Parken*. Number 607 in 12. VDI Verlag, 2005.

[118] Daniel Scharstein and Richard Szeliski. Stereo matching with nonlinear diffusion. *International Journal of Computer Vision (IJCV)*, 28(2):155–174, 1998.

[119] Daniel Scharstein, Richard Szeliski, and Ramin Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision (IJCV)*, 47:7–42, 2002.

[120] Ulrich Scheunert, Basel Fardi, Norman Mattern, Gerd Wanielik, and N. Keppeler. Free space determination for parking slots using a 3d pmd sensor. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 154–159, 2007.

[121] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 519–528, 2006.

[122] Rui Shen, Irene Cheng, Xiaobo Li, and Anup Basu. Stereo matching using random walks. In *International Conference on Pattern Recognition (ICPR)*, pages 1–4. IEEE, 2008.

[123] Masao Shimizu and Masatoshi Okutomi. Sub-pixel estimation error cancellation on area-based matching. *International Journal of Computer Vision (IJCV)*, pages 207–224, 2005.

[124] Kai-Tai Song and Hung-Yi Chen. Lateral driving assistance using optical flow and scene analysis. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 624–629, 2007.

[125] W. Stahl and Jürgen Hötzel. Parktronic-system (pts), aktueller stand und ausblick. Technical Report 1287, VDI-Berichte, 1996.

[126] Gideon P. Stein, Ofer Mano, and Amnon Shashua. Vision-based acc with a single camera: Bounds on range and range rate accuracy. In *Proceedings of IEEE Intelligent Vehicles Symposium*, Columbus, OH, June 2003.

[127] Christoph Strecha, Rik Fransens, and Luc J. Van Gool. Wide-baseline stereo from multiple views: A probabilistic account. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 552–559, 2004.

[128] Christoph Strecha, Rik Fransens, and Luc Van Gool. Combined depth and outlier estimation in multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2394–2401, 2006.

[129] Christoph Strecha, Wolfgang von Hansen, Luc J. Van Gool, Pascal Fua, and Ulrich Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[130] Jae Kyu Suhr, Kwanghyuk Bae, Jaihie Kim, and Ho Gi Jung. Free parking space detection using optical flow-based euclidean 3d reconstruction. In *Machine Vision and Applications (MVA)*, pages 563–566, 2007.

[131] Jae Kyu Suhr, Ho Gi Jung, Kwanghyuk Bae, and Jaihie Kim. Automatic free parking space detection by using motion stereo-based 3d reconstruction. *Machine Vision and Applications (MVA)*, 21(2):163–176, 2010.

[132] Jian Sun, Nan-Ning Zheng, and Heung-Yeung Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 25(7):787–800, 2003.

[133] Richard Szeliski. A multi-view approach to motion and stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 1157, 1999.

[134] Yuichi Taguchi, Bennett Wilburn, and C. Lawrence Zitnick. Stereo reconstruction with mixed pixels using adaptive over-segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[135] Marshall F. Tappen and William T. Freeman. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. In *International Conference on Computer Vision (ICCV)*, pages 900–907, 2003.

[136] Philip H. S. Torr and David W. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal of Computer Vision (IJCV)*, 24:271–300, 1997.

[137] Yanghai Tsin, Sing Bing Kang, and Richard Szeliski. Stereo matching with reflections and translucency. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 702–709, 2003.

[138] Christian Unger, Selim Benhimane, Eric Wahl, and Nassir Navab. Efficient disparity computation without maximum disparity for real-time stereo vision. In *British Machine Vision Conference (BMVC)*, London, September 2009.

[139] Christian Unger, Martin Groher, and Nassir Navab. Image based rendering for motion compensation in angiographic roadmapping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[140] Christian Unger, Eric Wahl, and Slobodan Ilic. Efficient stereo and optical flow with robust similarity measures. In Rudolf Mester and Michael Felsberg, editors, *Pattern Recognition (DAGM)*, volume 6835 of *Lecture Notes in Computer Science*, pages 246–255. Springer Berlin Heidelberg, 2011.

[141] Christian Unger, Eric Wahl, and Slobodan Ilic. Efficient stereo matching for moving cameras and decalibrated rigs. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 417–422, 2011.

[142] Christian Unger, Eric Wahl, and Slobodan Ilic. Parking assistance using dense motion-stereo. *Machine Vision and Applications (MVA)*, pages 1–21, 2011.

[143] Christian Unger, Eric Wahl, Peter Sturm, and Slobodan Ilic. Stereo fusion from multiple viewpoints. In Axel Pinz, Thomas Pock, Horst Bischof, and Franz Leberl, editors, *Pattern Recognition (DAGM)*, volume 7476 of *Lecture Notes in Computer Science*, pages 468–477. Springer Berlin Heidelberg, 2012.

[144] Wannes van der Mark and Dariu M. Gavrila. Real-time dense stereo for intelligent vehicles. *IEEE Transactions on Intelligent Transportation Systems (ITS)*, 7(1):38–50, 2006.

[145] Geert Van Meerbergen, Maarten Vergauwen, Marc Pollefeys, and Luc Van Gool. A hierarchical symmetric stereo algorithm using dynamic programming. *International Journal of Computer Vision (IJCV)*, 47:275–285, April 2002.

[146] Olga Veksler. Fast variable window for stereo correspondence using integral images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 556–561, 2003.

[147] Christophe Vestri, Sylvain Bougnoux, R. Bendahan, Katia Fintzel, S. Wybo, Francisco J. Abad, and T. Kakinami. Evaluation of a vision-based parking assistance system. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 131–135, 2005.

[148] Paul A. Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision (IJCV)*, 57(2):137–154, 2004.

[149] George Vogiatzis, Philip H. S. Torr, and Roberto Cipolla. Multi-view stereo via volumetric graph-cuts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 391–398, 2005.

[150] Eric Wahl, Florian Oszwald, Artur Ruß, Armin Zeller, and Dirk Rossberg. Evaluation of automotive vision systems: Innovations in the development of video-based adas. In *FISITA World Automotive Congress*, 2008.

[151] Eric Wahl, Tobias Strobel, Artur Ruß, Dirk Rossberg, and Rolf-Dieter Therburg. Realisierung eines parkassistenten basierend auf motion-stereo. In *16. Aachener Kolloquium*, 2007.

[152] Eric Wahl and Rolf-Dieter Therburg. Developing a motion-stereo parking assistant at bmw. *MATLAB Digest*, 2008.

[153] Eric Wahl, Christian Unger, Armin Zeller, and Dirk Rossberg. 3d-environment modeling as an enabler for autonomous vehicles. *ATZ Automobiltechnische Zeitschrift*, Februar 2010.

[154] Eric Wahl and Wolgang Zeitler. Video-based driver assistance systems put to test: Comparison - evaluation - series production. In *13th International Conference: Electronic Systems for Vehicles*, 2007.

[155] Liang Wang, Miao Liao, Minglun Gong, Ruigang Yang, and David Nister. High-quality real-time stereo using adaptive cost aggregation and dynamic programming. In *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, pages 798–805, 2006.

[156] Liang Wang and Ruigang Yang. Global stereo matching leveraged by sparse ground control points. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3033–3040, 2011.

[157] Zeng-Fu Wang and Zhi-Gang Zheng. A region based stereo matching algorithm using cooperative optimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[158] Yichen Wei and Long Quan. Region-based progressive stereo matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 106–113, 2004.

[159] Oliver M. C. Williams, Michael Isard, and John MacCormick. Estimating disparity and occlusions in stereo video sequences. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 250–257, 2005.

[160] Ludwig Wittgenstein. *Tractatus logico-philosophicus, Logisch-philosophische Abhandlung*. Suhrkamp, 2003.

[161] John Iselin Woodfill, Gaile Gordon, and Ron Buck. Tyzx deepsea high speed stereo vision system. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 3 - Volume 03*, page 41, Washington, DC, USA, 2004. IEEE Computer Society.

[162] Oliver J. Woodford, Philip H. S. Torr, Ian D. Reid, and Andrew W. Fitzgibbon. Global stereo reconstruction under second order smoothness priors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[163] Gang Xu and Zhengyou Zhang. *Epipolar Geometry in Stereo, Motion and Object Recognition: A Unified Approach*. Kluwer Academic Press, 1996.

[164] Jin Xu, Guang Chen, and Ming Xie. Vision-guided automatic parking for smart car. In *Proceedings of IEEE Intelligent Vehicles Symposium*, pages 725–730, 2000.

[165] Qingxiong Yang, Liang Wang, Ruigang Yang, Henrik Stewénius, and David Nistér. Stereo matching with color-weighted correlation, hierarchical belief propagation and occlusion handling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2347–2354, 2006.

[166] Qingxiong Yang, Liang Wang, Ruigang Yang, Henrik Stewénius, and David Nistér. Stereo matching with color-weighted correlation, hierachical belief propagation and occlusion handling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 31(3):492–504, 2009.

[167] Ruigang Yang, Marc Pollefeys, and Greg Welch. Dealing with textureless regions and specular highlights - a progressive space carving scheme using a novel photo-consistency measure. In *International Conference on Computer Vision (ICCV)*, pages 576–584, 2003.

[168] Kuk-Jin Yoon and In-So Kweon. Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(4):650–656, 2006.

[169] Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. In *European Conference on Computer Vision (ECCV)*, pages 151–158, 1994.

[170] Christopher Zach. Fast and high quality fusion of depth maps. In *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, 2008.

[171] Andrei Zaharescu, Edmond Boyer, and Radu Horaud. Transformesh : A topology-adaptive mesh-based approach to surface evolution. In *Asian Conference on Computer Vision (ACCV)*, pages 166–175, 2007.

[172] Guofeng Zhang, Jiaya Jia, Tien-Tsin Wong, and Hujun Bao. Recovering consistent video depth maps via bundle optimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[173] Guofeng Zhang, Jiaya Jia, Tien-Tsin Wong, and Hujun Bao. Consistent depth maps recovery from a video sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 31(6):974–988, 2009.

[174] Ye Zhang and Chandra Kambhamettu. Stereo matching with segmentation-based cooperation. In *European Conference on Computer Vision (ECCV)*, pages 556–571, 2002.

[175] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22:1330–1334, 2000.

[176] Zhengyou Zhang and Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22:1330–1334, 1998.

[177] C. Lawrence Zitnick and Takeo Kanade. A cooperative algorithm for stereo matching and occlusion detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(7):675–684, 2000.

[178] Lawrence C. Zitnick, Sing B. Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. In *SIGGRAPH*, pages 600–608, 2004.