Nils Einecke

Stereoscopic Depth Estimation for Online Vision Systems

Stereoscopic Depth Estimation for Online Vision Systems

Nils Einecke

Universitätsverlag Ilmenau
2012

Impressum

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Angaben sind im Internet über http://dnb.d-nb.de abrufbar.

Diese Arbeit hat der Fakultät für Informatik und Automatisierung der Technischen Universität Ilmenau als Dissertation vorgelegen.

Tag der Einreichung:	28. Oktober 2011
1. Gutachter:	Univ.-Prof. Dr.-Ing. Horst-Michael Groß (Technische Universität Ilmenau)
2. Gutachter:	Dr. rer. nat. Julian Eggert (Honda Research Institute Europe GmbH, Offenbach/Main)
3. Gutachter:	Univ.-Prof. Dr.-Ing. Darius Burschka (Technische Universität München)
Tag der Verteidigung:	21. August 2012

Technische Universität Ilmenau/Universitätsbibliothek
Universitätsverlag Ilmenau
Postfach 10 05 65
98684 Ilmenau
www.tu-ilmenau.de/universitaetsverlag

Herstellung und Auslieferung
Verlagshaus Monsenstein und Vannerdat OHG
Am Hawerkamp 31
48155 Münster
www.mv-verlag.de

ISBN 978-3-86360-044-0 (Druckausgabe)
URN urn:nbn:de:gbv:ilm1-2012000243

Titelfoto: photocase.com

Acknowledgments

First of all, I would like to thank Prof. Dr. Edgar Körner, Prof. Dr. Bernhard Sendhoff and Andreas Richter for giving me the opportunity to perform my research at the Honda Research Institute Europe. The open-minded and amiable atmosphere and the collection of people from different disciplines gave rise to many new ideas. Moreover, the critical questions and discussions after presentations were a great preparation for conference Q&A sessions and the defense of my thesis.

I want to thank Prof. Dr. Horst-Michael Groß for the constructive discussions and comments despite the spatial distance. Further thanks got to Prof. Dr. Burschka for his time to examine this thesis and to participate in the defense.

I would also like to thank Dr. Julian Eggert, my supervisor at the Honda Research Institute. He always gave me a large freedom in pursuing my research but challenged my ideas in fierce discussions which I enjoyed and valued a lot. With his insights, knowledge and wisdom he was a great guide in the jungle of the modern research landscape.

Many thanks go to Dr. Tobias Rodemann who was a member of the advisory board that supervised my PhD work at the Honda Research Institute. Since his background is a bit decoupled from the topic of my thesis he could give me some very important comments from a different point of view.

Special thanks go to Daniel Weiler and Sven Rebhan with whom I not only shared my first office room at the Honda Research Institute but also a deep friendship. Discussing problems and ideas with them always helped me to improve my work. Another special thanks goes to Volker Willert. He willingly spend a lot of time proof-reading my formulas and suggested ways to write these with clearer expressions. Additionally, his experience with optical flow shaped my understanding for stereoscopic computations, and this work would be much different if it had not been for him. Many thanks also go to Jens Schmüdderich with whom I worked together in many projects and who became a friend of mine in

the course of these. Furthermore, I would like to thank all colleagues that had to bear my skeptical character in joint projects. In particular, I would like to thank Stephan Hasler, Robert Kastner and Alexander Gepperth who shared the one or other night shift with me when we were working on the SamSys project. I would also like to thank Michael Gienger and Manuel Mühlig for the very nice collaboration work which demonstrated the large potential of combining state-of-the-art robotics and computer vision research.

Last but not least I would like to thank my beloved wife Eileen who unremittingly encouraged me to write this thesis although due to this she had to endure plenty of free time without me. Her kind words and her understanding motivated me to keep on writing in the long periods of time were thesis writing had to be done in parallel to other projects at the institute. I would also like to thank my family for their support and belief in me which bolstered my spur in pursuing a doctoral degree.

Kurzfassung

Die visuelle Wahrnehmung des Menschen wird in hohem Maße vom stereoskopischen Sehen beeinflusst. Die dreidimensionale Wahrnehmung entsteht dabei durch die leicht unterschiedlichen Blickwinkel der beiden Augen. Es ist eine nahe liegende Annahmen, dass maschinelle Sehsysteme ebenfalls von einem vergleichbaren Sinn profitieren können. Obwohl es bereits zahlreiche Arbeiten auf dem Gebiet des maschinellen stereoskopischen Sehen gibt, erfüllen die heutigen Algorithmen entweder nicht die Anforderungen für eine effiziente Berechnung oder aber sie haben nur eine geringe Genauigkeit und Robustheit.

Das Ziel dieser Doktorarbeit ist die Entwicklung von echtzeit- und realweltfähigen stereoskopischen Algorithmen. Insbesondere soll die Berechnung der Algorithmen leichtgewichtig genug sein, um auf mobilen Plattformen eingesetzt werden zu können. Dazu werden im Rahmen dieser Arbeit zwei neue Methoden vorgestellt, welche sich durch eine gute Balance zwischen Geschwindigkeit und Genauigkeit auszeichnen.

Als erstes wird die “Summed Normalized Cross-Correlation” (SNCC) vorgestellt, eine neue Kostenfunktion für blockvergleichende, stereoskopische Tiefenschätzung. Im Unterschied zu den meisten anderen Kostenfunktionen ist SNCC nicht anfällig für den qualitätsmindernden “Fattening”-Effekt, kann aber trotzdem sehr effizient berechnet werden. Die Auswertung der Genauigkeit auf Standard Benchmark-Tests zeigt, dass mit SNCC die Genauigkeit von lokaler, blockvergleichsbasierter, stereoskopischer Berechnung nahe an die Genauigkeit von global optimierenden Methoden basierend auf “Graph Cut” oder “Belief Propagation” heran kommt. Des Weiteren wird eine parallele Mehrkernberechnung für blockvergleichsbasierte, stereoskopische Algorithmen vorgestellt, welche eine lineare Beschleunigung der Berechnung mit der Anzahl der Prozessorkerne erlaubt.

Die zweite vorgestellte Methode ist das “Direct Surface Fitting”, ein neuer Algorithmus zum Schätzen parametrischer Oberflächenmodelle an Hand von Stereobildern. Dieser Algorithmus ist inspiriert vom Homographie-beschränkten Gradientenabstieg, welcher häufig dazu benutzt wird um die Lage von planaren Oberflächen im Raum zu Schätzen. Durch die Ersetzung des Gradientenabstiegs mit der direkten Suchmethodik von Hooke und Jeeves wird die planare Schätzung auf beliebige parametrische Oberflächenmodelle und beliebige Kostenfunktionen erweitert. Ein Vergleich auf Standard Benchmark-Tests zeigt, dass "Direct Surface Fitting" eine vergleichbare Genauigkeit wie Methoden aus dem Stand der Technik hat, im Gegensatz zu diesen aber eine erhöhte Robustheit in anspruchsvollen Situationen besitzt. Ergänzt wird der neue Algorithmus durch ein spärliches Berechnungsschema welches selbst große Oberflächen in Echtzeit schätzen kann.

Um die Realwelttauglichkeit der vorgestellten Methoden zu untermauern wurden diese in ein Automobil- und in ein Robotersystem integriert. Das Automobilsystem wurde in verschiedenen Wettersituationen und unter verschiedenen Lichtbedingungen getestet. Die Ergebnisse demonstrieren die hohe Robustheit und Stabilität der eingeführten Methoden. In der Umwelt des Robotersystems befinden sich viele Objekte mit homogener Farbgebung um die Erkennung dieser Objekte zu erleichtern. Unglücklicherweise, macht diese Homogenität die Tiefenschätzung dieser Objekte sehr schwer da sie nur eine schwache Textur besitzen. Greif- und Navigationsexperimente demonstrieren, dass trotz der schwierigen Bedingungen die vorgestellten Methoden dazu in der Lage sind sehr genaue Schätzungen der Tiefe zu liefern. Insgesamt wird durch die durchgeführten Experimente bestätigt, dass die in dieser Doktorarbeit entwickelten Methoden sehr gut für Echtzeit- und Realwelt-Anwendungen auf mobilen Plattformen geeignet sind.

Abstract

The human visual perception heavily depends on stereoscopic vision. By fusing the two views that our eyes provide, a 3-D sensation of the surrounding is generated. It is therefore natural to assume that machine vision systems also benefit from a comparable sense. A lot of work has been done in the area of machine stereo vision, but a severe drawback of today's algorithms is that they either achieve high accuracy and robustness by sacrificing real-time speed or they are real-time capable but with major deficiencies in quality.

The goal of this thesis is to tackle the problem of efficient, real-world, stereoscopic depth processing. In particular, the processing should be lightweight enough to be processed on mobile platforms. To this end, two new methods are introduced that have a very good balance between speed and accuracy.

First, the summed normalized cross-correlation (SNCC) is proposed which constitutes a new cost function for block-matching stereo processing. In contrast to most standard cost functions it hardly suffers from the fattening effect while being computationally very efficient. Evaluations on standard benchmarks show that SNCC brings the standard local block-matching stereo very close to the performance of global optimization methods based on graph cut or belief propagation. As an additional benefit for real-time implementation, a multi-core parallel processing scheme for block-matching stereo is discussed which allows for a runtime gain that is linear in the number of processing cores.

Second, the direct surface fitting, a new algorithm for fitting parametric surface models to stereo images is introduced. This algorithm is inspired by the homography-constrained gradient descent methods which are frequently used to estimate the orientation and depth of planar surfaces. By replacing the gradient descent search with the direct search method of Hooke-Jeeves the fitting of any parametric surface model with arbitrary cost functions becomes feasible. A comparison on standard benchmarks shows that the direct surface fitting has a depth accuracy comparable to state-of-the-art methods while being more robust in challenging situations. Additionally, a sparse matching scheme is discussed which enables real-time estimations even for very large surfaces.

In order to prove the real-world capabilities of the proposed methods, they are integrated into an automotive and a robotic vision system. The automotive system is tested in different weather and lighting conditions which demonstrates the high robustness and stability of the proposed methods for variable conditions. In the environment of the tested robotic system a lot of objects have a homogeneous color to ease their segmentation and detection. This, however, poses a grave challenge to the depth perception due to weak textures. Grasping and navigation experiments demonstrate that despite the hard challenge the proposed methods are able to estimate the depth of objects very accurately. Altogether, the experiments highlight that the developed methods are well-suited for real-time, real-world applications on mobile platforms.

Contents

1. Introduction

"When I first learned about stereopsis in college, I wondered if I could imagine this way of seeing. Now I had my answer. I could not. Stereopsis provides a distinctive, subjective sensation, a quale. [...] While I could infer indirectly a sense of depth through cues like perspective and shading, I could not synthesize stereoscopic depth from other visual attributes, such as color, position, form, or brightness. The sensation provided by stereopsis of empty space and things projecting or receding into that space is unique." [Barry, 2009, p. 101-102]

This is how neurobiologist Dr. Susan R. Barry describes her unique experience of acquiring the ability to see in 3-D through stereopsis. Born with an eye muscle deficiency she developed a strabismus when she was three months old. Strabismus is a misalignment of the visual axis of the two eyes that leads to an inability to look at the same point in space with the two eyes. Usually, when people look at something the two eyes yield a slightly different view of the very same point in space. The brain merges these two views into one, thereby creating a 3-D impression of the world which is called stereopsis. People with strabismus lack this impression because their eyes deliver strongly different views which their brain is unable to merge. Although Dr. Barry had three correcting eye-muscle surgeries when she was a child she remained stereo-blind until she enrolled in a vision therapy at her late forties and finally acquired her 3-D vision at an age of 48.

Dr. Susan R. Barry summarized her unique experience in the book "Fixing my gaze" [Barry, 2009]. There she reports how much different the human visual perception really is without a working stereoscopic vision. As a neuroscientists she is also able to relate her experience to the neuronal level thus giving new insights about the human visual perception and especially the importance of stereoscopic vision. Although Dr. Barry could cope quite well in her daily life using other modalities to fill the gap of the missing stereoscopic perception, she always had trouble with things that others could do with ease.

After having acquired her 3-D vision, she also realized that stereoscopic vision is something that cannot be replaced by other visual cues. Her insights demonstrate how important the perception of depth is for humans. As the human visual system acts as a model for most of the artificial machine vision systems, it is very natural to assume that the performance of machine vision systems is also heavily depending on a good performance of their 3-D perception.

Since the first attempts of turning the stereoscopic depth perception into an algorithm [Marr and Poggio, 1976], computer calculated stereoscopic vision has developed to one of the standard techniques in machine vision. The research activity is comparable to other fields like visual object tracking, image segmentation or object detection and classification. In the recent years, people started to assemble different vision algorithms into vision systems [Ess et al., 2008, Groß et al., 2009, Schmüdderich et al., 2010] instead of working on the single algorithms in separation. Usually, vision systems are dedicated for a certain complex task like navigating through unknown environments, visually guided object manipulation or automated scene analysis. Using several complex vision algorithms for solving such tasks also means an increased need for computational power. This need might be satisfied by a computer cluster, however, this is not an option for mobile real-time systems which posses only limited computational resources. In these cases efficient algorithms and an elaborated resource management [Rebhan et al., 2009] become inevitable. That some algorithms depend on the result of some others makes things even harder. If for example an object segmentation algorithm needs the result of the object tracking for initialization then it is no longer sufficient that each single algorithm runs in real-time. In order to prevent an undesired increase in latency, both have to run in super-real-time because they are executed one after the other. In this sense, stereo processing can become a very critical module in a vision system. Not only it is very computationally demanding but stereo calculation is usually one of the first modules in the processing chain. This means that both the speed and the accuracy of the stereo processing can be very critical for the performance of the whole system.

One might argue that machine stereo vision is unnecessary because of the development of specialized hardware for depth measurement like RADAR, LIDAR or Time-of-Flight cameras. In fact stereoscopic depth estimation is one of the major methods for acquiring depth information in robotics and the domain of intelligent vehicles. Its main advantages are the high spatial resolution and the low cost. The technique itself has even been transferred to extraterrestrial applications. In 2006 NASA launched the twin STEREO spacecraft [Kaiser et al., 2008] for researching the causes and mechanisms of coronal mass ejections (CME). The two STEREO satellites are equipped with a nearly identical setup of instruments. In contrast to former missions the dual setup allows to reconstruct CMEs in 3-D [Howard and Tappin, 2008] by means of triangulation. Aside from understanding CMEs the STEREO mission helps to improve the space weather prediction which is essential for protecting spacecrafts and astronauts but also for reducing the effects of electromagnetic storms on electric infrastructures on earth.

The target of this thesis is to provide techniques for a lightweight but at the same time good performing stereoscopic depth perception, i.e. algorithms with a very good trade-off between accuracy and speed. As such a kind of algorithms are mainly interesting for mobile platforms a side-constraint is the real-world capability of the algorithms. Furthermore, we are at the verge of a paradigm change in the computer hardware. The clock speed of modern CPUs has not increased for some years now, instead CPUs now consist of an increasing number of processing cores. Hence, the presented algorithms are also discussed in the light of their parallelizability with a particular focus on multi-core architectures.

Structure

This thesis is split up into four main parts. The first part describes the general idea of stereoscopic depth estimation, discusses the basic biological background and gives the mathematical foundation for machine stereo vision.

In the second part, state-of-the-art stereo algorithms are discussed with respect to their speed and accuracy. Furthermore, a novel cost function for block-matching stereo processing is proposed. In contrast to most standard cost functions it hardly suffers from the fattening effect while being computationally very efficient. Evaluations on standard benchmarks show that this novel cost function brings the standard local block-matching stereo very close to the performance of global optimization methods based on graph cut or belief propagation. Thus the proposed cost function allows for a block-matching stereo with a very good trade-off between speed and accuracy. Additionally, a multi-core parallel processing scheme for block-matching stereo is discussed which allows for a runtime gain that is linear in the number of processing cores. State-of-the-art works usually concentrate on time consuming parallel implementations on graphic cards with a low portability. As comparisons in this thesis show the graphic card implementations are only slightly faster than the proposed multi-core processing which is very easy to implement.

The third part, discusses depth perception which is constrained by geometrical models. A new method is introduced which combines the advantages of the two main existing approaches of homography estimation by gradient descent and RANSAC fitting. On the one hand the new method allows for any parametric model like RANSAC and on the other hand its estimation process is directly based on the image data like the homography estimation (and not on preprocessed disparity maps) which reduces the aperture problem. Comparisons of the three approaches demonstrate that the new algorithm has a comparable depth accuracy as the state-of-the-art methods while being more robust in challenging situations. Furthermore, a sparse processing scheme allows for real-time estimations of even very large surfaces.

The fourth part demonstrates the advantage of the newly proposed techniques by presenting two vision systems that integrate these techniques. The first is an automotive vision system for car detection which is tested in different weather and lighting conditions. The second is a robotic vision system adopted on a humanoid robot with grasping abilities, whose environment is populated by homogeneously colored objects which make the depth estimation very difficult. The good results achieved with these two systems highlight that the developed methods are well-suited for real-time, real-world applications on mobile platforms. Eventually, this thesis is closed by a summary.

2. Stereoscopic Depth Perception

This chapter explains the concept of stereoscopic depth estimation. It gives insights into the biological foundation and the algorithmic transfer to a computer program.

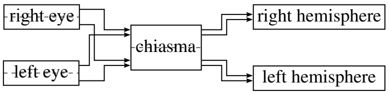
2.1. Human Stereoscopic Vision

The visual system of the human brain is a very powerful but also a very complex system. Despite many years of neurobiological research it is still not fully understood. On the other hand the insights that were gained gave rise to many inspirations of how to build a machine vision system. One key element to the depth perception of humans are the two eyes. These deliver a slightly different view of the world which the brain can merge into one consistent view thereby creating a sense of 3-D.

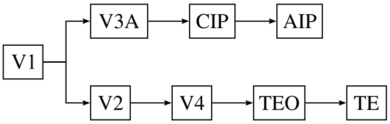
The sensing structure in the human eye is the retina. It consists of an abundance of single receptors which turn photons of light into visual information. This information is split up into two streams as shown in Fig. 2.1(a). The information from the left part of both retinas is send to the left brain hemisphere. Accordingly the information from the right part is send to the right hemisphere. Both streams pass the chiasma which can be seen as a kind of information flow junction. Once the visual information has reached the visual cortex the depth processing begins.

Depth Processing in Monkey Brains

Most neuronal insights on visual processing in the brain is gained from studies of the monkey brain. Figure 2.1(b) gives a very rough overview



(a) Flow of visual information



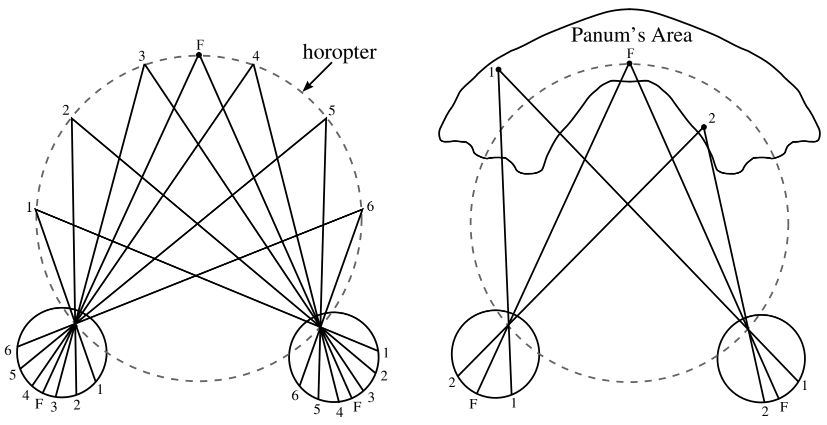
(b) Depth processing brain areas

Figure 2.1.: Depth perception in the human brain. (a) The visual information from the retina of both eyes is split up. While the information from the left side of both retinas is transferred to the left brain hemisphere the information from the right side is transferred to the right hemisphere. (b) Overview over monkey brain areas involved in depth perception. The depth processing starts at the lowest visual area V1 and continues along the ventral stream (V2, V4, TEO, TE) and the dorsal stream (V3A, CIP: caudal intraparietal, AIP: anterior intraparietal).

of the most important visual areas involved in depth processing in the monkey brain. As can be seen the processing of stereoscopic vision starts already at the lowest visual area V1. From here the visual depth information travels along the dorsal and the ventral path to the higher visual areas where it gets refined.

A general insight of the visual processing in the brain is that depth perception gets more elaborated on higher visual areas [Anzai and DeAngelis, 2010]. In V1, so-called binocular neurons locally compare the inputs from the left and the right eye. This leads already to a precise local depth measurement. However, local comparison are prone to ambiguities which can only be resolved by a global combination and reasoning on the local measurements. To this end, higher visual areas perform stereo processing on a more global level in order to improve the robustness and accuracy of the depth perception.

In V4, line segments are augmented by a 3-D orientation [Hinkle and Connor, 2002] using the information of the lower visual areas which leads to a more robust depth perception of edges. Further upwards in the area TE of the infero temporal (IT) cortex, neurons with larger receptive fields react selectively for slanted and curved surfaces [Janssen et al., 2000, Orban et al., 2006]. The IT cortex is also very important for object recognition. Hence, it is natural to assume that the 3-D surface selectivity of the IT neurons supports object recognition. Recent findings [Srivastava et al., 2009] in the anterior intraparietal (AIP) area suggest a similar 3-D slant and curve selectivity. As this is a key area for object grasping and manipulation, it shows that a good 3-D



(a) Horopter (b) Panum's Area

Figure 2.2.: (a) If a human focuses on an object F the image of this object will project to the fovea F in both eyes. Furthermore, all objects that lie on the horopter (dashed circular line) will project to exactly the same retinal position relative to the fovea F in both eyes. The brain infers the depth of these objects from their position on the horopter and the relative angle between the eyes. (b) Objects that do not lie on the horopter project to different points on the retina. The brain has to compute the relative position of corresponding points for 3-D perception. This processing is limited to Panum's area.

estimation of object surfaces is also important for object manipulation. Although the above findings were mainly made in macaque monkeys, similar findings have been made with corresponding areas in the human brain [Georgieva et al., 2009].

Horopter and Disparity

When a human focuses his vision on an object he will move his eyes such that the object is centered in both eyes. This process is called vergence. It ensures that the current object of attention is projected to the high-resolution fovea in the middle of the retina of both eyes (see points F in Fig. 2.2(a)). Thus, objects that are viewed with vergence, project on exactly the same location on the two retinas. The same is true for all objects that lie on the *horopter* (see Fig. 2.2(a)), a virtual, circular line that goes through the verged object and the two centers of projection. Regarding the fovea as a reference point on the retina, all objects that lie on the horopter will project to the same retinal location relative to the fovea in both eyes. This is shown exemplarily in Fig. 2.2(a) by the 3-D points 1-6 and their retina projections. The 3-D position of an object on the horopter can be calculated from its 2-D position on the horopter and the radius of the horopter. The horopter radius itself depends only on the relative angle between the two eyes which is directly available to the brain.

Panum's Area

In order to calculate the 3-D positions of objects that do not lie on the horopter the brain needs additional information that is hidden in the projection characteristics. Objects which do not lie on the horopter project to different points in the two retinas. The distance between those two projection points is called *disparity*. It is directly coupled to the 3-D distance between the object and the horopter. As a matter of fact, all objects on the horopter have a zero disparity. In the brain the disparity is measured by means of the binocular neurons. Each binocular neuron receives input from the left and the right retina but the origin of this input must not necessarily be the same retinal location. If the two inputs are correlated the neuron will fire spikes, thereby signaling that it found corresponding locations on the two retinas, i.e. locations that match in terms of the available visual structure. As a binocular neuron processes only one pair of inputs its spikes encode for a certain disparity at a certain retinal location. However, the binocular depth processing is limited to a maximal disparity range. This leads to a restricted area around the horopter in which the two views of an object can be merged into an 3-D perception. In neuroscience this area is called *Panum's area* (see Fig. 2.2(b)). Everything that lies outside of the Panum area cannot be merged into one consistent view and is perceived (often unrecognized) with double images which are not anchored in the 3-D world.

2.2. 3-D View Geometry

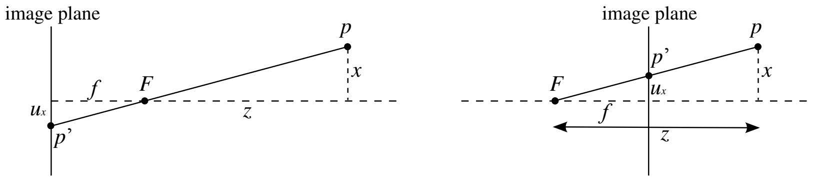
So far the biological background of stereopsis has been discussed. As explained above the main source of the 3-D depth perception of humans are the two slightly different views that our eyes provide. By finding correspondences in the left and right eye the brain is able to compute the distance of objects in the environment. In a similar fashion it is possible to derive the 3-D position of objects captured by a setup of two cameras, usually referred to as *stereo camera*. This section gives the mathematical foundation of how the 3-D position of points in the world can be inferred from their 2-D projections in two cameras. Some of the concepts discussed in the following are not necessary for understanding the techniques proposed in this thesis but are rather a prerequisite for using a stereo camera for depth perception.

2.2.1. Perspective Projection

Before formulas for calculating the depth of objects can be derived it is first necessary to describe how 3-D points in the world project onto the sensing chip (CCD or CMOS) of a camera. Unfortunately, a correct description of such a projection is much to complex, especially when the camera has multiple lenses. It turns out, however, that as long as the real camera system is well-focused it can be approximated by a pinhole model. The pinhole model actually describes the perspective projection for a degenerated thin lens with zero aperture, i.e. the lens has no physical extension and all rays of light pass through the same tiny point, the pinhole.

Figure 2.3(a) displays how a 3-D point p with coordinates $\mathbf{x} = (x, y, z)^T$ is projected to the image plane (camera sensor chip) in the pinhole model. The projected point denoted as p' is located in 2-D image plane coordinates $\mathbf{u} = (u_x, u_y)^T$. For a crisp description of the projection, the world frame is centered in the focal point F with the z-axis being aligned with the optical axis of the camera. The distance between the image plane and the focal point is called focal length f. This is the main influencing parameter for the perspective projection.

Using the intercept theorem, the formula for the perspective projec-



(a) Pinhole Model (b) Pinhole with Shifted Image Plane

Figure 2.3.: (a) Side-view of point projections under a pinhole model with focal length f and the world coordinates centered in the focal point F. The 3-D world point p is projected to the 2-D image point p'. x and u_x are the 3-D world and 2-D image x-coordinates, respectively. z is the 3-D z-coordinate of p, i.e. the depth of p. (b) Projection in accordance with (a) but with a virtual image plane in front of the focal point. Using this representation eliminates the need to work with flipped images as would be the case in (a).

tion follows directly from Fig. 2.3(a)

$$u_x = -f\frac{x}{z} \ . \tag{2.1}$$

Due to the perspective projection the image of objects is flipped upside-down and left-right. This is indicated in the formula by the minus sign. As this can be a bit cumbersome the minus is removed with a little trick. By moving the image plane in front of the focal point F (see Fig. 2.3(b)) the minus disappears and the perspective projection equation becomes

$$u_x = f\frac{x}{z} \ . \tag{2.2}$$

In a real camera system the trick is realized by flipping the image. This is usually done directly on the camera when the sensor chip is read out.

For the y-coordinate the perspective projection is exactly the same which leads to the perspective projection equation

$$\mathbf{u} = \begin{pmatrix} u_x \\ u_y \end{pmatrix} = \frac{f}{z}\begin{pmatrix} x \\ y \end{pmatrix} . \tag{2.3}$$

A more common way of writing this equation is to use homogeneous coordinates

$$z\begin{pmatrix} u_x \\ u_y \\ 1 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} . \tag{2.4}$$

The 3x4 matrix in the above equation can be split up into two separate matrices

$$\begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \mathbf{K}_f \mathbf{\Pi}_0 \ , \tag{2.5}$$

which leads to the compact form

$$z\mathbf{u} = \mathbf{K}_f \mathbf{\Pi}_0 \mathbf{x} \ . \tag{2.6}$$

In the above projection equation **u** and **x** are in homogeneous coordinates. The matrix $\mathbf{\Pi}_0$ is often referred to as the *standard projection matrix.*

Intrinsic Camera Parameters

The equations derived above work with ideal image plane coordinates which are based on the same unit length as the world coordinates. In reality coordinates on a camera sensor chip are rather given in pixels, i.e. the unit length is in accordance with the physical size of the pixels on the chip. This means that the real pixel coordinates $\mathbf{u}'$ are subject to a scale s_x in x-direction and a scale s_y in the y-direction with respect to the image plane coordinates **u**

$$\begin{pmatrix} u'_x \\ u'_y \end{pmatrix} = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix} \begin{pmatrix} u_x \\ u_y \end{pmatrix} . \tag{2.7}$$

Furthermore, the center of the pixel grid (o_x, o_y) must not necessarily be the same as the ideal center of the image plane. Incorporating this into the pixel scale equation (2.7) and using homogeneous coordinates the whole transformation from image plane coordinates **u** to pixel coordinates $\mathbf{u}'$ reads as

$$\mathbf{u}' = \begin{pmatrix} u'_x \\ u'_y \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & s_\theta & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_x \\ u_y \\ 1 \end{pmatrix} = \mathbf{K}_s \mathbf{u} \ . \tag{2.8}$$

Here s_θ is a skew factor which can be used if the pixels are not rectangular [Ma et al., 2003, sec. 3.2.2]. Using the above mapping to the pixel space, the perspective mapping equation (2.6) can be extended to

$$z\mathbf{u}' = \mathbf{K}_s \mathbf{K}_f \mathbf{\Pi}_0 \mathbf{x} = \mathbf{K} \mathbf{\Pi}_0 \mathbf{x} \ . \tag{2.9}$$

The combination of the matrices $\mathbf{K}_s$ and $\mathbf{K}_f$ is called *intrinsic parameter matrix* $\mathbf{K}$. This matrix describes the parameters that are inherent (or intrinsic) to a specific camera.

For a machine stereo vision it is necessary to know the intrinsic camera parameters because these describe the relation between 3-D world points and their 2-D projections. However, deriving the 3-D world coordinates of a point from a single 2-D projection is an under-determined problem. Thus for a stereoscopic depth estimation two cameras are required.

2.2.2. Epipolar Constraint in Stereo Cameras

As has been explained in section 2.1 the depth perception in the human brain is based on finding corresponding points in the left and the right eye. For a computer-implemented depth perception the same basic step has to be done. Fortunately, there is no need to compare each pixel in one camera with each pixel in the other camera. The epipolar constraint reduces the search space for each pixel in one camera image to a line of candidate pixels in the other camera image. Note that in the following, L and R will be used to indicate affiliation to the left and the right camera, respectively.

Figure 2.4 shows the projection of a point p for a setup of two cameras in general position. The two projections of p in the left and the right camera are p_L and p_R, respectively. Each camera has its own reference frame centered in the corresponding focal point F_L or F_R. One can see in Fig. 2.4 that the point p and the two focal points F_L and F_R form the triangle (p, F_L, F_R) which intersects the two image planes.

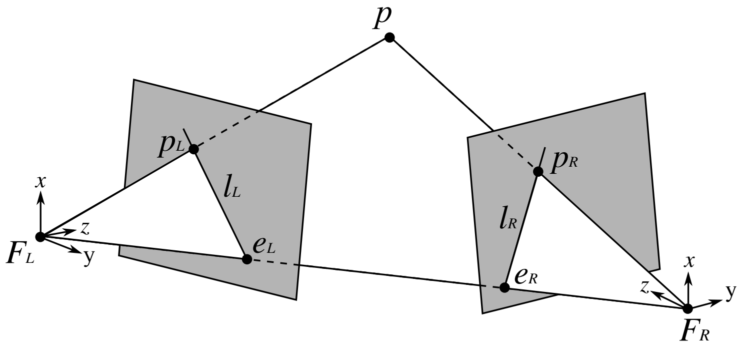


Figure 2.4.: Epipolar geometry: This figure shows two pinhole model cameras in general position. A 3-D point p projects to p_L and p_R in the left and right image plane, respectively. Each camera has its own reference frame centered in the corresponding focal point F_L or F_R. Due to the triangle built by p, F_L and F_R corresponding points in the two image planes are always located on the epipolar lines l_L and l_R. The points e_L and e_R where the line $F_L F_R$ intersects the two image planes are called epipoles.

The intersection lines between this triangle and the two image planes are called *epipolar lines*. Obviously, all 3-D points p' that lie in the plane of the triangle (p, F_L, F_R) can only project to the two epipolar lines l_L and l_R. Hence, for all pixels on the left epipolar line l_L the corresponding pixel in the right camera image is located somewhere on the right epipolar line l_R and vice versa. This is the essence of the *epipolar constraint*.

In order to make use of the epipolar constraint, one first has to find a description of the plane defined by the triangle (p, F_L, F_R) without using p because the actual 3-D position of p is not known. This can be achieved by means of the so-called *epipoles* e_l and e_r. The epipoles are the intersections of the line $F_L F_R$ with the two image planes. Note that the epipoles may lie outside of the physical extent of the sensor chip but they are unique for a given camera setup. Looking at Fig. 2.4 it is easy to see that the plane of the triangle (p, F_L, F_R) is also defined by the triangles (p_L, e_L, F_L) and (p_R, e_R, F_R). Thus the epipolar lines can be calculated without knowing the real point p. However, it is necessary to know the relative position and orientation of the two cameras. This relation can be described by

$$\mathbf{x}_R = \mathbf{R}\mathbf{x}_L + \mathbf{t} \ , \tag{2.10}$$

where matrix $\mathbf{R}$ describes the relative rotation and the vector $\mathbf{t}$ the relative position between the two cameras. Using the perspective projection equation (2.3) the above relation can be described in terms of the image plane coordinates

$$z_R\mathbf{u}_R = \mathbf{R}z_L\mathbf{u}_L + \mathbf{t} \ . \tag{2.11}$$

Note that the above equation assumes the focal length f to be 1, which can be achieved by normalization if the actual focal length is known. As the mapping between the corresponding points $\mathbf{u}_L$ and $\mathbf{u}_R$ shall be described without the 3-D position of p, z_L and z_R have to be eliminated. For doing so, a matrix $\hat{\mathbf{T}}$ is chosen such that $\hat{\mathbf{T}}\mathbf{v} = \mathbf{t} \times \mathbf{v}$ for any vector $\mathbf{v}$. The idea is to replace the cross product with an easier to handle matrix multiplication. Multiplying the coordinate transformation equation (2.11) with $\hat{\mathbf{T}}$ leads to

$$z_R\hat{\mathbf{T}}\mathbf{u}_R = \hat{\mathbf{T}}\mathbf{R}z_L\mathbf{u}_L \ . \tag{2.12}$$

Further multiplying this with $\mathbf{u}_R^T$ results in

$$z_R\mathbf{u}_R^T\hat{\mathbf{T}}\mathbf{u}_R = \mathbf{u}_R^T\hat{\mathbf{T}}\mathbf{R}z_L\mathbf{u}_L \ . \tag{2.13}$$

Exploiting the facts that $\mathbf{u}_R^T\hat{\mathbf{T}}\mathbf{u}_R = 0$ and $z_L > 0$ leads to the equation that describes the *epipolar constraint*

$$\mathbf{u}_R^T\hat{\mathbf{T}}\mathbf{R}\mathbf{u}_L = \mathbf{u}_R^T\mathbf{E}\mathbf{u}_L = 0 \ , \tag{2.14}$$

where the matrix $\mathbf{E} = \hat{\mathbf{T}}\mathbf{R}$ is called the *essential matrix*. The entries of $\mathbf{E}$ describe the relation between the cameras of a stereo camera system. These have to be determined before the epipolar lines can be calculated.

A standard way of recovering these parameters is to use the eight-point algorithm [Longuet-Higgins, 1981] which calculates $\mathbf{E}$ up to a scale from at least eight pairs of known correspondences. In order to derive the scale it is necessary to have some knowledge about the 3-D world, e.g. the real distance between two points. For calculating the epipolar lines it is not necessary to know the real scale. However, it is necessary to know the scale for computing genuine 3-D positions from found correspondences.

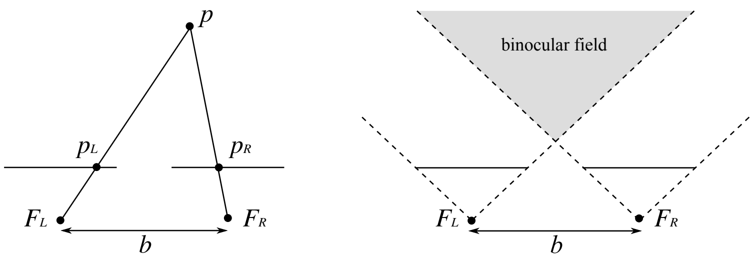
2.2.3. Parallel Stereo Camera Setup

There are two main setups for a stereo camera system. The first setup is composed of two cameras that can be moved like the eyes of a human. This is mainly done in biologically inspired camera systems [Shibata et al., 2001, Zhang and Tay, 2006, Sandini et al., 2007] in order to be able to replicate the verging movements of eyes.

The second and more frequently used setup is a parallel camera setting. In this case the cameras are aligned and fixed such that their imaging planes (sensor chips) are co-planar and vertically aligned. This has some advantageous over cameras in general position. Having again a look on Fig. 2.4 one can see that the epipoles e_L and e_R move to infinity if the image planes become co-planar. Due to this the epipolar lines get aligned with the x-axis of the image planes. Although this might seem irrelevant from a theoretical point of view, it makes the actual implementation on a computer much simpler. Furthermore, this improves the speed of the correspondence calculation because searching along the images lines is coherent with the memory alignment of images in a computer.

Due to the alignment of the image planes the essential matrix $\mathbf{E}$ of a parallel stereo camera reduces to a horizontal shift between the two cameras. This distance is called *baseline* and is the main characteristic of a parallel stereo camera system. Fig 2.5(a) shows a parallel stereo camera in a top view. As can be seen the z-distance or depth from a point p is the same for both cameras, i.e. $z_L = z_R$. The same is true for the y-coordinate because of the horizontal alignment of the two image planes. From this it follows that also the two projections have the same y-coordinate. Altogether this leads to a much simpler connection between corresponding points in the left and the right camera image as compared to a general setup (see section 2.2.2).

Let us come back to the perspective projection equation (2.3) of point p. Because the two cameras have two different coordinate systems the left coordinates of p are $\mathbf{x}_L = (x_L, y_L, z_L)^T$ and the right coordinates are $\mathbf{x}_R = (x_R, y_R, z_R)^T$. This means that the two projection equations



(a) Parallel Stereo Setup (b) Binocular Field

Figure 2.5.: Parallel Stereo Setup. (a) Projection of a point p in a parallel stereo setup. In contrast to a stereo setup in general position the image planes are co-planar and aligned vertically. They are only separated horizontally by the baseline b. (b) Only points that lie within the binocular field are projected into both image planes. For points that lie outside of this field no stereoscopic depth can be calculated.

are

$$\mathbf{u}_L = \frac{f}{z_L}\begin{pmatrix} x_L \\ y_L \end{pmatrix}, \quad \mathbf{u}_R = \frac{f}{z_R}\begin{pmatrix} x_R \\ y_R \end{pmatrix}. \tag{2.15}$$

Since in a parallel stereo camera setup $z_R = z_L$, $y_R = y_L$ and $x_R = x_L - b$, where b is the baseline, the projection equation (2.15) of the right camera can be rewritten as

$$\mathbf{u}_R = \frac{f}{z_L}\begin{pmatrix} x_L - b \\ y_L \end{pmatrix}. \tag{2.16}$$

This modified projection equation links the left 3-D world coordinates with the right projection but the actual target is to link a correspondence pair $(\mathbf{u}_L, \mathbf{u}_R)$ with the depth of the corresponding 3-D point p. In order to do so the projection equation (2.15) of the left camera is rearranged for x_L and y_L

$$x_L = \frac{u_{Lx} z_L}{f} \tag{2.17}$$

$$y_L = \frac{u_{Ly} z_L}{f}. \tag{2.18}$$

Substituting x_L and y_L in the modified projection equation (2.16) for the right camera leads to the basic mapping equation of a parallel stereo camera system

$$\mathbf{u}_R = \mathbf{u}_L - b\frac{f}{z_L}\begin{pmatrix} 1 \\ 0 \end{pmatrix} . \tag{2.19}$$

By means of the above equation a pixel from the left camera can be mapped to its corresponding pixel in the right camera using the known depth z_L. But actually we want to have it the other way around, calculating z_L from a correspondence pair ($\mathbf{u}_L$, $\mathbf{u}_R$). As was mentioned above and is also captured in the basic mapping equation (2.19) the y-coordinate of corresponding pixels is the same. Hence, the depth z can only be calculated from the difference in the x-coordinate. This is achieved by rearranging the basic mapping equation (2.19) for z_L

$$z_L = \frac{bf}{u_{Rx} - u_{Lx}} . \tag{2.20}$$

In accordance to the depth perception in the brain, the difference $d = u_{Rx} - u_{Lx}$ is referred to as *disparity*. Similar to the Panum area in the human vision, the machine depth perception is also restricted spatially. As can be seen in Fig. 2.5(b) only objects that lie within the *binocular field* are projected onto both image planes. The binocular field is the area in space where the two field of views of the two cameras overlap. Objects that lie outside of this field are either projected only to one image plane or non at all. Thus the 3-D position can only be computed for objects lying within the binocular field.

Due to the compactness of the depth equation (2.20) and the simple correspondence search along horizontal pixel lines, the parallel stereo camera setup has become the most frequently used setup in the domain of mobile systems. In the remainder of this thesis the term ”stereo camera” always refers to the parallel camera setup.

Camera Calibration and Image Rectification

In reality it is very difficult to build a perfect parallel stereo camera system. Hence, there is a need for calibrating a stereo camera before it can be used. The first thing to consider is that real lenses might cause radial distortions, especially when the focal length is very short. Fortunately, this can be removed quite easily by finding the parameters of the radial distortion and warping the captured images accordingly.

There are several algorithms (see e.g. [Ma et al., 2003, sec. 3.3.3]) that can extract the radial distortion parameters from a small set of sample images, preferably showing straight structures.

Another thing to consider is that cameras can not be build and aligned well enough so that the image planes are really co-planar. Thus, it is necessary to calibrate a parallel stereo camera system by determining the essential matrix $\mathbf{E}$ (see section 2.2.2). As soon as these parameters are known one can warp the two camera images as if the image planes were perfectly aligned. Together with the removal of the radial distortions, this process is usually referred to as *image rectification*. In this thesis the stereo images used are either already rectified (like for standard benchmark images) or an image rectification precedes the calculations on the stereo images. The rectification itself can be implemented very efficiently using a precalculated look-up table.

The last thing to consider are the intrinsic camera parameters. In order to calculate 3-D positions from found correspondence pairs it is also important to know the focal length f and the layout of the pixels on the sensor chip. Although these values may be provided by the camera manufacturer it is often nevertheless necessary to find them by camera calibration. The most common way of doing this is to use a planar checkerboard [Ma et al., 2003, sec. 6.5.3] as a calibration rig because corresponding points can be robustly detected in an automated way.

3. Dense and Accurate Stereo Correspondence Calculation

The last chapter gave an introduction on how to calculate 3-D positions of corresponding pixels seen in the two views of a stereo camera. In this chapter a novel technique for improving dense stereo calculation is discussed, which allows for a very good trade-off between speed and accuracy. The notation dense stereo relates to calculating a disparity or depth value for all pixels in the stereo images, i.e. a disparity or depth map. This chapter is divided in two main parts.

The first part gives an overview of state-of-the-art approaches for calculating dense stereo, with a focus on real-time applicability. The analysis in the overview reveals that block-matching stereo processing is the most appealing approach for real-time systems because it has a runtime that scales linearly with the number of pixels and the disparity search range, while having a very low number of operations per pixel. Unfortunately, block-matching stereo is prone to the fattening effect, i.e. among other things disparity discontinuities are not well localized.

Therefore in the second part of this chapter a new matching cost function called *summed normalized cross-correlation* (SNCC) [Einecke and Eggert, 2010b] is introduced. This new cost function is a variant of the *normalized cross-correlation* (NCC) that strongly reduces the fattening effect while keeping all the invariance properties of NCC. Evaluations on benchmark stereo images demonstrate that this new cost is much better than other standard cost functions like NCC or rank transform. The quality comes close to elaborated methods based on graph cuts or belief propagation but having a runtime that is one or two magnitudes faster. Hence, SNCC allows for a very good trade-off between speed and accuracy which makes it perfectly suited for real-time applications.

3.1. State-of-the-Art Stereo Algorithms

The area of stereoscopic depth processing has been and still is one of the most actively researched fields in computer vision. Today there is a vast amount of algorithms that use very diverse principles for finding the pixel correspondences that are necessary to calculate 3-D depth (see chapter 2). The major applications are 3-D scene vision for mobile platforms and 3-D reconstruction of object surfaces. While mobile platforms usually employ a parallel stereo camera setup, the cameras used for 3-D reconstruction are often in a general position. It has been observed in [Scharstein and Szeliski, 2002] that all stereo matching algorithms typically consist of four main steps:

1. matching cost computation
2. cost aggregation
3. disparity computation / optimization
4. disparity refinement

In the first step a matching cost between all potentially corresponding pixels is calculated. Common pixel-based matching costs include squared difference, absolute difference, binary matching costs or gradient-based measures [Scharstein and Szeliski, 2002]. As explained in section 2.2.3, the corresponding pixels in rectified stereo images are located on the same image line. This enables to store the matching costs in a condensed data structure called Disparity Space Image (DSI). The DSI is a three-dimensional structure whose entries are indexed by x-position, y-position and disparity (see Fig. 3.1).

Unfortunately, a single pixel matching is prone to noise and ambiguity. Hence, in a second step follows the aggregation of the matching costs. This can be done by filtering the DSI with spatial filters like a Gaussian or a box filter. In most cases the aggregation is restricted to the two-dimensional x-y planes but might also be done for the full three-dimensional DSI.

After the DSI has been generated and improved by aggregation, the pixel correspondences have to be determined. The most simple form is a maximum selection, i.e. for each pixel in one image the best matching pixel in the other image is selected. More complex selection mechanisms

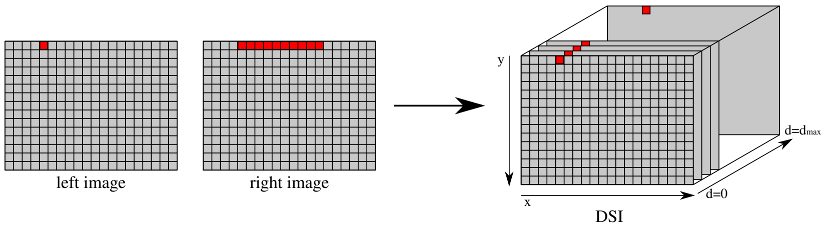


Figure 3.1.: The Disparity Space Image (DSI) stores the matching results for all pixels and all disparities. For example, the matching cost between the red pixel in the left image and all the red pixels in the right image are stored at the red positions in the DSI.

incorporate smoothness constraints for neighboring pixels to reduce ambiguities and increase the robustness to noise.

The last step of a stereo computation is a refinement of the calculated disparities. These encompass interpolation of disparities to sub-pixel accuracy or a left-right consistency check for detecting occlusions.

In the following, some of the currently most prevalent stereo computation algorithms are introduced and analyzed with respect to their real-time capability and quality. Usually, the main idea of these algorithms is to focus on one of the first three steps of the stereo computation, while being quite unconstrained with respect to the other steps. The analysis will reveal that only a few approaches solely scale linearly with the number of pixels n and number of disparities d, which is a prerequisite for real-time stereo processing.

3.1.1. Correlation-based Stereo

Correlation-based stereo processing is one of the oldest and most simple form of stereo computation [Barnard and Fischler, 1982]. The idea is to compute the correlation between all potentially corresponding pixels. Instead of comparing only pixel-by-pixel, a neighborhood around each pixel is used for comparison.

Many cost functions used for correlation-based stereo are equivalent to first constructing a DSI with a pixel-wise matching and then aggregating the costs in the x-y planes of the DSI. For example *summed squared difference* (SSD) and *summed absolute difference* (SAD) work that way. The aggregated matching cost $\hat{C}(p,d)$ at a pixel p for a disparity d reads as:

$$\hat{C}(p,d) = \sum_{q \in N_p} w(q,p) C(q,d) \ . \tag{3.1}$$

In the above equation N_p denotes the neighborhood of p and $C(q,d)$ denotes the pixel-wise cost for a pixel q and disparity d. Furthermore, w is a function that weights the pixel-wise costs according to their spatial distance to the anchor pixel p. Usually, this spatial weighting is a Gaussian or an equal weighting.

Other cost functions directly compute the cost from the neighborhood which does not allow for a separation into pixel-wise cost calculation and aggregation. The most prominent example is the *normalized cross-correlation* (NCC). This cost function will be discussed in more detail in section 3.2.

After the costs have been calculated, corresponding pixels are determined by a maximum (or minimum) selection

$$d_c(p) = \arg\max_i \hat{C}(p,i) \ , \tag{3.2}$$

where $d_c(p)$ is the disparity of pixel p.

Due to its simple nature, correlation-based stereo is one of the most frequently used stereo methods. Unfortunately, the quality is usually inferior to the quality of more elaborated methods. One major problem is the so-called *fattening effect*. The fattening effect is a tendency of correlation-based stereo to smear disparities. This is most apparent at depth discontinuities where foreground disparities are smeared over background disparities (see Fig. 3.2 for an example).

For naive implementations of the correlation-based stereo, the time complexity is $\mathcal{O}(wnd)$, where w is the size of the neighborhood, n the number of image pixels and d the number of searched disparities. This renders correlation-based stereo impractical for real-time applications. However, usually the neighborhoods in correlation-based stereo have a rectangular shape. In this case the correlation-based stereo is referred to as block-matching stereo. If the block-matching is done with an equal spatial weighting (i.e. box filters) then integral images (summed-area tables) [Crow, 1984, Viola and Jones, 2004] or separated filtering

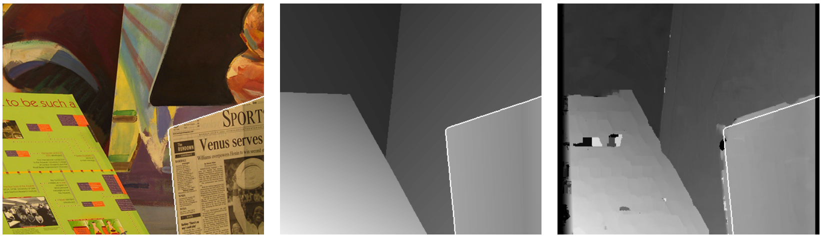

(a) Venus (b) Ground Truth (c) NCC Disparity

Figure 3.2.: Fattening effect: (a) Left image of the Venus scene from the Middlebury benchmark [Scharstein and Szeliski, 2002]. (b) The ground truth disparity map. (c) Disparity map calculated with NCC. The disparity values are encoded by means of intensity. Near pixels are bright and far pixels are dark. The white line in all three images shows the outline of the newspaper. As can be observed in (c) the disparity values of the newspaper are smeared over the background. This is the essence of the foreground fattening effect.

[McDonnell, 1981, Sun, 1997] reduce the computational time to $\mathcal{O}(nd)$. Moreover, the actual number of operations is very low making block-matching stereo computation very appealing for real-time implementations [Brown et al., 2003, Humenberger et al., 2010].

Another thing to consider is memory usage. In the naive implementation first all costs are calculated and stored in the DSI. Then the best disparity for each pixel is selected via maximum search. Unfortunately, storing the complete DSI has a space complexity of $\mathcal{O}(nd)$ which could surpass the memory of a modern computer for large images and search ranges. Thus a better way is to compute the maximum on the fly while calculating the aggregated matching costs D. This reduces the space complexity to $\mathcal{O}(n)$.

Adaptive Weight

Adaptive Weight (AW) approaches [Mattoccia et al., 2009, Yoon and Kweon, 2006, Heo et al., 2008] are a variant of the correlation-based stereo which involve the feature similarity of neighboring pixels. The idea is lend from Bilateral Filtering (BF) [Tomasi and Manduchi, 1998] which is used for smoothing images while preserving edges. Instead of aggregating the matching cost of the vicinity by a simple summation, the matching cost of the neighboring pixels are weighted according to their distance in the spatial and feature domain

$$\hat{C}(p,d) = \sum_{q \in N_p} w_s(p,q) w_f(p,q) C(q,d) \; . \tag{3.3}$$

In the above equation $\hat{C}(p,d)$ is the weighted matching cost of pixel p for disparity d, N_p are the neighbors of p that influence the weighted cost of p, w_f and w_s are weighting filter kernels and $C(q,d)$ is the plain matching cost of pixel q for disparity d. The kernels w_f and w_s weight the matching cost of the pixels in the surrounding of p according to their spatial and feature distance to p. For the spatial kernel w_s usually a Gaussian is used. The feature kernel w_f depends on the features used for matching. In most cases w_f is an intensity or color distance measure.

One attribute of AW is the high precision of depth discontinuities. The reason for this is that the weighted aggregation takes the pixel-similarity within one stereo image into account. This strongly reduces the fattening effect at depth discontinuities because pixels from the two different sides of a depth discontinuity often have very dissimilar features. Hence, pixels from a different depth have very little influence on the cost. However, there is a big downside of AW. It is computationally very expensive which excludes this method for real-time applications. In contrast to the simple box-filter aggregation, AW kernels are in general not separable which leads to a runtime complexity of $\mathcal{O}(wnd)$, with w being the number of pixels within one aggregation window, n the number of image pixels and d the number of disparities. There are some possibilities to reduce the runtime to $\mathcal{O}(nd)$ by using integral histograms [Porikli, 2005; 2008] or piecewise-linear bilateral filtering [Yang et al., 2009]. However, in general, these methods constitute only an approximation to the actual bilateral filter and still take several seconds for processing standard stereo images.

Rank and Census Transform

Although the rank transform (RT) [Zabih and Woodfill, 1994] is usually referred to as a cost function for correlation-based stereo it actually constitutes a pre-processing step for intensity images. The principal idea of RT is to replace each pixel intensity with its rank among a certain

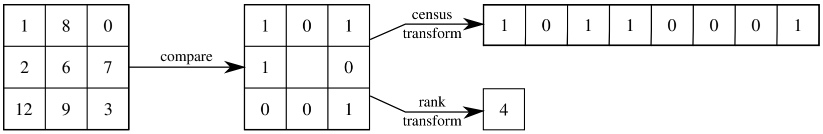


Figure 3.3.: Working of rank and census transform. For an image patch the intensities of all pixels are compared to the intensity of the center. A rank transform replaces the center pixel intensity by the number of intensities smaller than the center. The census transform replaces the center pixel by a bitstring describing the result of the comparison.

neighborhood (see Fig. 3.3). This is done for both stereo images and removes most of the lighting changes that can occur between these images. The actual stereo processing is typically done by a standard SAD or SSD correlation. A nice side-effect of RT is a decreased fattening.

A variant of RT is the so-called census transform (CT) [Zabih and Woodfill, 1994]. This transformation does not replace the pixel intensities with their rank but rather with a binary fingerprint that encodes which pixel intensities of the neighborhood are smaller than the anchor pixel. For computing a matching cost the hamming distance between these fingerprints is calculated. Usually, CT has a better quality than RT but at a slightly higher computational and memory cost.

In a large comparison of 15 cost functions [Hirschmüller and Scharstein, 2009], census transform, NCC and rank transform were shown to be the best cost functions for correlation-based stereo with respect to several radiometric changes. The runtime of stereo with rank transform is $\mathcal{O}((w+d)n)$, where n is the number of pixels, d the number of disparities and w the number of pixels in a pixel neighborhood. Thereby the runtime of just the rank transformation is $\mathcal{O}(wn)$. For typical setups in real-time applications (e.g. $d = 50$ and $w = 121$) this means that the rank transformation of the two stereo images can easily take two to four times as long as the actual stereo calculation. For the census transform the runtime is similar but depends on the actual implementation of the bit-string calculation, storage and comparison.

3.1.2. Global Stereo Matching

Global stereo matching approaches usually turn the stereo matching problem into an energy-minimization problem. The goal is to find a disparity function D which minimizes a global energy function

$$E(D) = E_{\text{data}}(D) + \lambda E_{\text{smooth}}(D) \ , \tag{3.4}$$

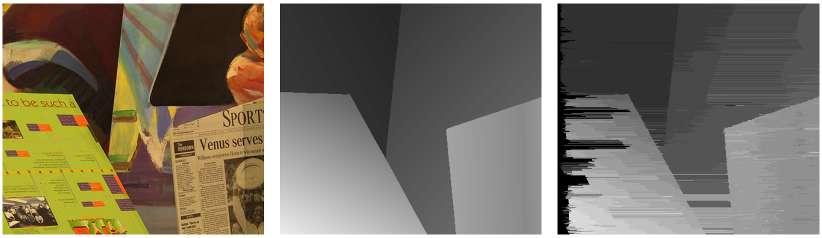
where $E_{\text{data}}(D)$ reflects the matching quality of the two stereo images given the disparity map D. Hence, $E_{\text{data}}(D)$ constitutes the aggregation of the matching costs at a pixel-level. The second term $E_{\text{smooth}}(D)$ makes the energy function favor smooth transitions between the disparity of neighboring pixels (*continuity constraint*). In general the smoothness term adds some increasing penalty for increasing difference of disparity that may also depend on the intensity or color difference of the pixels.

Unfortunately, finding the global optimum of such an energy function is NP-hard and, hence, intractable due to the large number of elements. This means that one has to soften the problem by either trying to find a local optimum or by reducing the constraints of the energy function (3.4). Typically used algorithms to achieve this are: graph cut, belief propagation or dynamic programming.

Dynamic Programming

As mentioned above, finding the global optimum of the global energy (3.4) is NP-hard. One way to tackle this problem is to solve the optimization for each image line independently. By doing so the global 2-D optimization is split up into multiple 1-D optimizations. These 1-D optimizations can be done efficiently in polynomial time using Dynamic Programming (DP) [Cormen et al., 2009, chap. 15]. The optimization scheme follows Bellman's *principle of optimality* [Bellman, 1952] and constitutes a *shortest-path* search in a graph, where the nodes of the graph are the elements of the DSI and the path lengths are the costs defined by the energy function (3.4).

Dynamic programming is one of the techniques that are frequently used in real-time implementations [Forstmann et al., 2004, Wang et al., 2006, Salmen et al., 2009] because it allows for an efficient improvement of the results from a correlation-based stereo. Unfortunately, DP



(a) Venus (b) Ground Truth (c) DP Disparity

Figure 3.4.: Streaking effect of Dynamic Programming (DP). (a) Left image of the Venus scene from the Middlebury benchmark [Scharstein and Szeliski, 2002]. (b) The ground truth disparity map. (c) Disparity map calculated with DP [Scharstein and Szeliski, 2002]. It can easily be observed that DP leads to horizontal streaks in the disparity map. The disparity values in both maps are encoded by means of intensity. Near pixels are bright and far pixels are dark.

has some drawbacks that limit its usage. Firstly, DP has an inherent demand for the ordering constraint, i.e. DP assumes that the ordering of pixels stays the same for corresponding image lines of the two stereo views. This may be violated by narrow foreground objects. Secondly, DP leads to the so-called *streaking effect*, which is a tendency of DP to produce fine streaks in the disparity map, in particular at depth discontinuities (see Fig. 3.4).

Graph Cut

Graph Cut (GC) approaches for global stereo matching [Roy and Cox, 1998, Boykov et al., 2001, Kolmogorov and Zabih, 2001, Miyazaki et al., 2009] have become very popular for the optimization of the energy function (3.4). In its true nature GC actually is a labeling technique that tries to find an optimal label assignment f based on the energy function:

$$E(f) = E_{\text{data}}(f) + E_{\text{smooth}}(f) \quad . \tag{3.5}$$

In case of stereo matching the labels f become disparities D.

An efficient variant of GC was proposed in [Boykov et al., 2001]. The main difference of this approach compared to other label assignment approaches is that the algorithm allows for a simultaneous change of the labels of a large number of pixels. These changes are done by so-called α-β-swaps or α-expansions. The first one is able to exchange two labels α and β and the latter one is able to change any label to the label α. Using one of these moves in an iterative fashion leads to a locally optimal solution. When using the α-expansions the found local optimum is within a known factor of the global optimum.

The actual trick behind the α-β-swaps and the α-expansions is that these can be reformulated as a maximum flow or minimum cut problem. For this purpose a graph $\mathcal{G} = \langle \mathcal{E}, \mathcal{V} \rangle$ is constructed. The graph $\mathcal{G}$ contains a vertex for each pixel in the image as well as two vertices representing the labels. For α-β-swaps these two vertices represent the two labels α and β and for α-expansions these two vertices represent the label α and every label except α, respectively. The edges $\mathcal{E}$ of the graph $\mathcal{G}$ connect the pixel vertices with the label vertices as well as neighboring pixel vertices. After the construction of the graph the minimum cut is computed (see e.g. [Cormen et al., 2009, chap. 26] for some efficient algorithms). Then the edges of the minimum cut are removed from $\mathcal{G}$. After this operation all pixel vertices are only connected to one of the two label vertices which indicates the new label for each pixel.

In general GC produces spatially dense and precise depth maps. The downside is the large runtime compared to standard correlation-based stereo. The theoretical worst-case runtime for the GC approaches is hard to pinpoint because it depends on the actual variant of GC. However, in an experimental comparison of major minimum cut (maximum flow) algorithms [Boykov and Kolmogorov, 2004] it has been demonstrated that all evaluated algorithms scale near-linear with the number of pixels n but almost quadratic with the number of labels (or disparities) d. Thus in practical applications the time complexity is roughly $\mathcal{O}(nd^2)$.

Belief Propagation

Similar to GC, Belief Propagation (BP) solves low-vision problems like image restoration or segmentation by reinterpreting these as a label assignment problem on a graph structure. The label assignment problem constitutes a minimization of the energy function (3.5). For stereo calculation these labels turn into disparities. The difference between BP and GC is the way the graph structure is created and used. While GC tries to map the problem to a minimum cut, BP uses a more direct mapping. For BP each pixel becomes a vertex and edges are usually drawn to the direct neighbors, i.e. in the form of a four-connected image grid. The algorithm optimizes the energy function (3.5) by starting with a random label assignment to the nodes and then improves the assignment by passing messages around the graph. The messages are vectors of a size given by the number of labels. They encode probability distributions of the label assignment of neighboring pixels. The messages are passed in an iterative manner (loopy belief propagation) by

$$m_{pq}^{t}(f_q) = \min_{f_p} \left(V(f_p, f_q) + D_p(f_p) + \sum_{s \in N(p)-q} m_{sp}^{t-1}(f_p) \right) , \qquad (3.6)$$

where p and q are nodes in the graph and $N(p)-q$ are all neighbors of p except for q. The terms $V(f_p, f_q)$ and $D_p(f_p)$ describe the smoothness of labels and the label assignment cost, respectively. After a number of iterations T the belief vector for each node is computed by

$$b_q(f_q) = D_q(f_q) + \sum_{p \in N(q)} m_{sp}^{T}(f_q) \ . \qquad (3.7)$$

The final labels f_q^* are those that minimize $b_q(f_q)$.

In a standard implementation of BP the runtime is quite large with $\mathcal{O}(nd^2T)$, where d is the number of labels or disparities and n the number of pixels. However, there are some means of reducing this runtime. One efficient variant has been proposed in [Felzenszwalb and Huttenlocher, 2006]. By cleverly exploiting the fact that in vision the smoothness $V(f_p, f_q)$ is usually based on a difference measure, the authors manage to cut the runtime down to $\mathcal{O}(ndT)$. Furthermore, they apply a hierarchical scheme that improves the convergence speed dramatically, leading to a very low number of iterations (five to ten).

One major drawback of BP is the large amount of data that has to be stored. As BP works with probability distributions of the labels, the space needed is at least $\mathcal{O}(nd)$. This easily surpasses the fast cache memory of a modern computer leading to a massive communication with the much slower main memory which significantly decreases the overall speed. For very large images even the main memory might not have sufficient space which limits the application of BP in stereo vision to smaller images.

3.1.3. Semi-Global Matching

As the name suggest Semi-Global Matching (SGM) [Hirschmüller, 2005] constitutes a compromise between global and local stereo correspondence matching. The motivation was to combine the efficiency of local methods with the accuracy of global optimization approaches. Like global matching approaches SGM uses a pixel-wise cost and a smoothness constraint for defining an energy $E(D)$ that has to be optimized for the disparity image D:

$$\begin{aligned} E(D) = & \sum_p C(p, D_p) + \\ & \sum_{q \in N_p} P_1 \delta[|D_p - D_q| == 1] + \\ & \sum_{q \in N_p} P_2 \delta[|D_p - D_q| > 1] \ . \end{aligned} \tag{3.8}$$

Here p denotes the pixels in an image, C is the cost function that defines the pixel matching cost for the disparities D and q are the pixels in the neighborhood N_p of p. Furthermore, P_1 and P_2 are constant penalties for enforcing the smoothness constraints and δ is a threshold function that returns one if its statement is true and zero otherwise.

The energy (3.8) of SGM is split into three terms. The first one sums up the matching cost for the disparities of D. The second and third term express the smoothness by adding a small penalty P_1 for neighboring pixels that differ only by one disparity from the disparity of pixel p and a larger penalty P_2 for larger disparity differences. The idea behind the smaller penalty is to permit for the small changes that occur on slanted and curved surfaces. The penalty for larger changes restricts large disparity jumps to large intensity jumps in the image.

A global optimization of D with respect to the energy $E(D)$ is NP-complete. On the other hand, the minimization of such an energy function can be done in polynomial time if one performs the minimization along all rows individually, using Dynamic Programming (DP). As

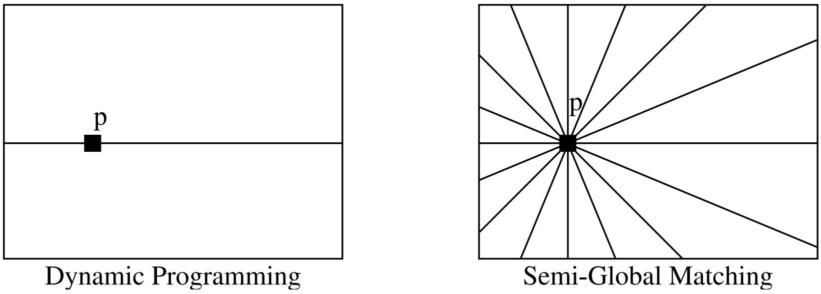


Figure 3.5.: Dynamic programming optimizes the disparity of pixel p according to the horizontal, one-dimensional path along the image. In contrast, Semi-Global Matching considers multiple one-dimensional paths through the image. This picture has been redrawn from [Hirschmüller, 2005].

was already pointed out above, DP suffers from a streaking effect (see Fig. 3.4). Therefore, the major idea of SGM is to aggregate the cost of multiple 1-D path directions (see Fig. 3.5). Paths that are traversed in direction r are denoted by L_r. For each pixel p and disparity d the cost $L_r(p,d)$ for path L_r is defined recursively,

$$\begin{aligned} L_r(p,d) = C(p,d) + & \\ & \min[L_r(p-r,d), \\ & \quad L_r(p-r,d-1)+P_1,\ L_r(p-r,d+1)+P_1, \\ & \quad \min_i L_r(p-r,i)+P_2]\ . \end{aligned} \tag{3.9}$$

The overall cost $S(p,d)$ for pixel p and disparity d is the summation of all path costs

$$S(p,d) = \sum_r L_r(p,d) \tag{3.10}$$

It was suggested in [Hirschmüller, 2005] to use at least eight paths but better 16 paths for having a good coverage of the 2-D image.

It has been shown in [Hirschmüller, 2005] that the runtime of SGM is $\mathcal{O}(lnd)$, where l is the number of paths, n the number of pixels and d the number of disparities. As l is usually eight or 16 the runtime of SGM is much smaller than the runtime of global optimization approaches but still higher than for standard block-matching stereo. One disadvantage of SGM is the necessity of storing the full disparity distribution for all pixels which leads to a space complexity of $\mathcal{O}(nd)$. As it has been discussed for BP this can lead to a significant decrease in speed on modern computers and limits the application to smaller stereo images.

3.1.4. Discussion

The overview of the current approaches and techniques in dense stereo processing revealed that most of the approaches have a large runtime or space requirement. Especially problematic with respect to scalability is GC because it has a non-linear runtime complexity. Other techniques like AW, BP or SGM have a runtime complexity that is linear in the number of pixels but it depends on other variables like a number of iterations or a neighborhood size which results in a very high number of operations per pixel. The fastest technique is block-matching stereo which has a linear runtime with a small number of operations per pixel. The major drawback of block-matching stereo is its tendency for smearing disparity values (fattening effect), in particular at depth discontinuities. This problem will be discussed and tackled in the next sections.

3.2. A Two-Stage Correlation Method for Stereoscopic Depth Estimation

This section introduces a new cost function called *summed normalized cross-correlation* (SNCC) [Einecke and Eggert, 2010b] which is an extension of the normalized cross-correlation (NCC). While NCC exhibits a strong fattening effect, SNCC is nearly uninfluenced by this effect but on the other hand has the same robust invariance properties like NCC.

3.2.1. The Fattening Problem of NCC

For two patches from the two camera images I^L (left) and I^R (right) the normalized cross-correlation (NCC) is defined as:

$$\rho_x(d) = \frac{\frac{1}{|p(x)|}\sum_{x' \in p(x)} (I^L_{x'} - \mu^L_x)(I^R_{x'+d} - \mu^R_{x+d})}{\sigma^L_x \sigma^R_{x+d}} , \tag{3.11}$$

where

$$\mu_x = \frac{1}{|p(x)|} \sum_{x' \in p(x)} I_{x'} \ , \quad \sigma_x = \sqrt{\frac{1}{|p(x)|} \sum_{x' \in p(x)} (I_{x'} - \mu_x)^2} \ . \qquad (3.12)$$

In the above equations $I_{x'}$ is the intensity of pixel x', x is the pixel position of the anchor point of the left patch, $p(x)$ is the set of pixel coordinates of the left image patch and $p(x+d)$ is the set of pixel coordinates of the right image patch, i.e. d denotes the disparity between the left and right image patch.

As was stated above NCC exhibits a strong fattening, i.e. in particular depth discontinuities are badly localized because of smearing (see Fig. 3.2). Although the fattening effect is well-known, there is hardly any discussion about the cause. In [Heo et al., 2008] it is argued that the fattening is a result of the perspective view changes between the left and the right image. However, this is inconsistent with the observation that the fattening effect also occurs at fronto-parallel surfaces. The author of [Falkenhagen, 2001, sec. 3.2] states that the effect is caused by a change in local intensity variance. At first sight this seems to be a good explanation but the fattening effect occurs also at borders of areas with different mean intensity where the variance must not necessarily be different.

In this thesis, it is argued that the fattening is caused by strong intensity contrasts [Einecke and Eggert, 2010b] within an image patch. These high contrasts dominate the correlation value by suppressing the low-contrast structures in a patch. This explains all the previous observations of the fattening effect. Depth discontinuities are affected because pixels from different depths usually show different world structures which are likely to have a different intensity. Areas of different variance as a matter of fact also have pixels of strongly different intensity. Thus the strong contrasts are consistent with the observation of the fattening effect. In a very recent work [Blanchet et al., 2011] a similar line of argumentation has been drawn for proving that the fattening effect of the SSD cost function can occur everywhere in an image. Here, the following discussion will reveal that the normalization in the correlation equation (3.11) is responsible for the rather strong fattening effect of NCC compared to other cost functions.

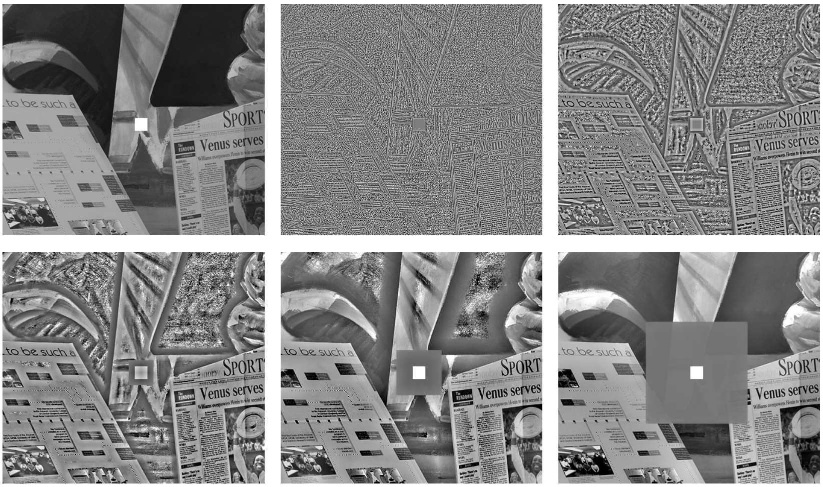
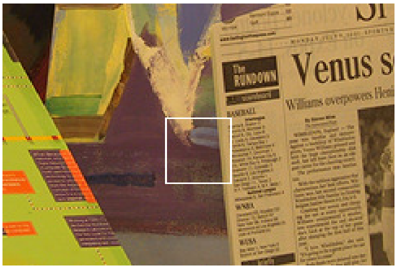


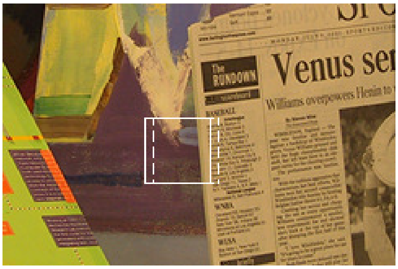
Figure 3.6.: The top left image shows the left Venus image altered by a rectangle of very high intensity, thus introducing a high contrast. From top middle to the bottom right, mean and standard deviation normalized images are shown using filter sizes of 3x3, 9x9, 21x21, 55x55, 149x149. Especially, for the larger filter sizes the suppression of the structure in the surrounding of the high intensity rectangle is clearly observable.

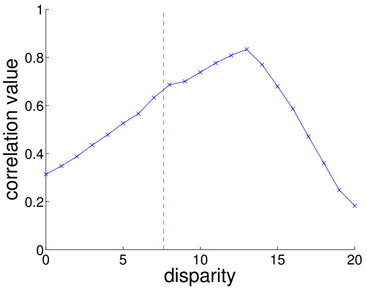
In the correlation equation (3.11) the intensity values in each patch $p(x)$ are normalized by means of the mean value and the standard deviation

$$I_{x'}^{\text{norm}} = \frac{I_{x'} - \mu_x}{\sigma_x} \ , \quad \text{where} \ \ x' \in p(x) \ . \tag{3.13}$$

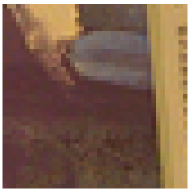
Because of this normalization, high-contrast structures suppress the low-contrast structures in their vicinity. In order to visualize this suppression, a strongly contrasting rectangle (intensity value 10000) was added to the left Venus image of the Middlebury stereo benchmark [Scharstein and Szeliski, 2002]. Then the normalization equation (3.13) was applied to this image using different patch sizes. The resulting normalized images for patch sizes 3x3, 9x9, 21x21, 55x55 and 149x149 are shown in Fig. 3.6. They demonstrate that the high contrast rectangle suppresses the structure in its surrounding according to the size of the patches used for the normalization filter.


(a) Venus left


(b) Venus right



(c) NCC correlation


(d) patch

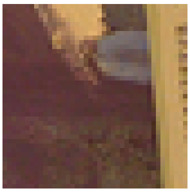

(e) match

Figure 3.7.: (a) and (b) show a cutout of the left and the right stereo image of the Venus scene. The square patch in (a) (see also (d)) is correlated with the right image for a set of disparities. (c) Correlation plot using NCC where the vertical line denotes the ground truth disparity. The best match is shown in (e) and depicted by the solid square in (b).

Due to this suppression effect all patches in the vicinity of a high contrast edge favor the disparity of this edge because it is the dominant structure. Not fitting this structure would lead to a large error or small correlation value. In Fig. 3.7(a) a cutout of the left image of the Venus scene is shown. The white square patch (see also Fig. 3.7(d)) is correlated with the right image, shown in Fig. 3.7(b), for several disparities (shifts). Figure 3.7(c) plots the correlation values for these disparities. The plot shows that the best match is roughly at 13 pixels disparity while the ground truth depth is roughly at 8 pixels disparity (depicted by the vertical line). The patch that corresponds to the peak is depicted as the solid square in Fig. 3.7(b) and shown in Fig. 3.7(e).

The reason for the wrong match is the large contrast edge between the bright newspaper and the dark background. By comparing Fig. 3.7(d) and Fig. 3.7(e) it can be directly seen that the matching was dominated by this strong contrast edge. Since the contrast edge itself has the depth of the occluder (newspaper), which has roughly a disparity of 13 pixels, all patches that encompass the border of the newspaper will also have the best correlation at 13 pixels disparity.

In summary the above observations indicate that the normalized cross-correlation is biased by strong contrasts. This leads to the conclusion that the correspondence search with NCC in block-matching stereo processing should be done with small patch sizes (filter size) to keep the distorted area small. However, decreasing the filter size would lead to very noisy depth images. Thus the next section introduces a new cost function which tackles this dilemma.

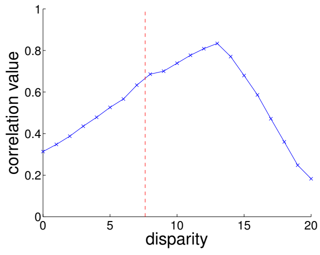
3.2.2. Summed Normalized Cross-Correlation (SNCC)

As has been discussed in the last section, stereo matching with NCC leads to a strong fattening effect caused by high contrasts within the images. The countermeasure would be to use very small patches for the correspondence search. Unfortunately, this leads to very noisy depth images. To overcome the dilemma, a two-stage correlation is proposed.

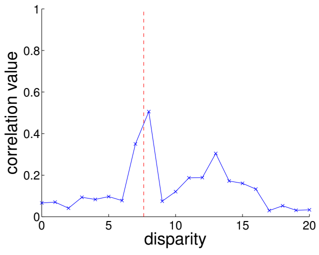
In the first stage NCC is computed according to the correlation equation (3.11) but using a very small filter size of 3x3 or 5x5. Then in the second stage a summation filter is applied directly to the result of the NCC filtering. This averages the correlation values over the neighborhood of each pixel at each disparity. As the correlation values are constrained to [-1..1] they exhibit much smaller differences between pixels as compared to pixel intensities. Thus the summation in the second stage of the filter does not lead to a fattening. As a consequence, the fine structure of the image is preserved and at the same time the noise in the depth estimation is reduced. The formula of the novel cost function *summed normalized cross-correlation* (SNCC) is defined as

$$\bar{\rho_x}(d) = \frac{1}{|p'(x)|} \sum_{x'' \in p'(x)} \rho_{x''}(d) \ , \tag{3.14}$$

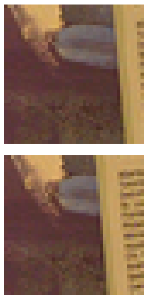
where $\rho_{x''}(d)$ is defined by the correlation equation (3.11) and $p'(x)$ is



(a) NCC correlation



(b) SNCC correlation



(c) match

Figure 3.8.: Comparison of the results of (a) the NCC and (b) the SNCC correlation for the patch at the top of (c) from the setup explained in Fig. 3.7. The best matching patch for SNCC is shown at the bottom of (c) and depicted in Fig. 3.7(b) by means of a dashed rectangle.

the set of pixel coordinates of the summation filter. It is important to understand that the summation step in SNCC is applied for every disparity d for averaging the correlation values $\rho_{x''}(d)$ before e.g. a maximum selection is done to find the best match (correspondence). In particular, this is very different from averaging a resulting depth or disparity map which indeed would lead to a fattening due to smearing.

The newly proposed matching cost was applied to the same patch from Fig. 3.7(a) like NCC. In this case a 3x3 patch size was used for the first stage (NCC) of SNCC while the summation of the second stage is defined by the image patch size. Comparing the resulting correlation values (Fig. 3.8(b)) with the results of the standard NCC (Fig. 3.8(a)) demonstrates the improvement of SNCC over NCC. For the new measure the influence of the strong contrast edge is dramatically reduced, though it is still visible by the second peak. The best matching patch for SNCC is depicted by the dashed rectangle in Fig. 3.7(b) and shown at the bottom of Fig. 3.8(c). Comparing the best patch with the template patch (top of Fig. 3.8(c)) reveals that SNCC was not distracted by the high contrast edge of the newspaper but concentrated on finding a good match for most part of the patch. This can be seen for example by the light triangular structure at the top left of the target patch.

3.2.3. Computational Complexity of SNCC

The following pseudocode summarizes the working of SNCC for block-matching stereo processing:

```
SNCC Init
    (1) Get left and right images $L$, $R$
    (2) $\mu_L = \text{boxfilter}(L, 3\text{x}3)$
    (3) $\mu_R = \text{boxfilter}(R, 3\text{x}3)$
    (4) $\sigma_L = \sqrt{\text{boxfilter}(L^2, 3\text{x}3) - \mu_L^2}$    ($L^2$ := pixelwise squared $L$)
    (5) $\sigma_R = \sqrt{\text{boxfilter}(R^2, 3\text{x}3) - \mu_R^2}$    ($R^2$ := pixelwise squared $R$)
    (6) $\rho_{\max} = -1$    confidence map in image size
    (7) $D = 0$    disparity map in image size
SNCC Loop
    (1) $\forall$ disparities $d$ do
        (1.1) Get $R_d$, $\mu_{R_d}$, $\sigma_{R_d}$
              (right image $R$ and maps $\mu_R$, $\sigma_R$ shifted by disparity $d$)
        (1.2) $\rho(d) = \frac{\text{boxfilter}(L*R_d,\, 3\text{x}3) - \mu_L \mu_{R_d}}{\sigma_L \sigma_{R_d}}$
        (1.3) $\bar{\rho}(d) = \text{boxfilter}(\rho(d), \text{averaging\_filtersize})$
        (1.4) $\forall$ positions $x$: if($\bar{\rho}_x(d) > \rho_{\max}(x)$)
              (1.4.1) $\rho_{\max}(x) = \bar{\rho}_x(d)$
              (1.4.2) $D(x) = d$
    (2) return $D$
```

This implementation exploits the fact that the correlation equation (3.11) can be reformulated to

$$\rho_x(d) = \frac{\frac{1}{|p(x)|}\sum_{x' \in p(x)} (I^L_{x'} I^R_{x'+d}) - \mu^L_x \mu^R_{x+d}}{\sigma^L_x \sigma^R_{x+d}} \,. \tag{3.15}$$

This way the mean values can be calculated in advance and need not to be mingled with the image for every disparity shift. In [Tsai and Lin, 2003] it has already been shown that NCC filtering can be calculated in linear time with respect to the number of pixels n and the number of disparities d, i.e. $\mathcal{O}(nd)$. In order to achieve the linear runtime, a fast box filter implementation based on integral images (summed-area tables) [Crow, 1984, Viola and Jones, 2004] is used. However, in this work the separated filtering of the box filter [McDonnell, 1981, Sun, 1997] is used instead of an integral image approach. There are three reasons for this:

Firstly, the separated box filtering takes two subtractions, two additions and one division per pixel, while integral images take three additions (two for creating the integral image and one for summation), three subtractions (one for creating the integral image and two for summation) and one division per pixel. Hence, the separated filtering saves one addition and one subtraction per pixel. Secondly and more importantly, the separated filtering is numerically less demanding. The maximum value needed to be stored for an integral image is $n \cdot 2^b$, where b is the encoding depth (e.g. 8-bit for standard gray valued images) and n the number of image pixels. In contrast, in a separable filtering scheme the maximum value is $w \cdot 2^b$, where w is the number of pixels in a filter window (patch). Since usually $w \ll n$ the computations can be carried out with smaller, less precise data types. Thirdly, the separated filtering spares one additional buffer (namely the integral image). These characteristics of separable filtering allow for a slightly faster implementation compared to integral images.

The extension of the efficient implementation of NCC to SNCC is straightforward, as there is only one additional box filter used for the summation in the second stage. Altogether the calculation of the matching cost of one pixel for one disparity takes six multiplications, four additions and five subtractions (not including the precomputation phase). Due to the efficient box filter implementation these numbers of operations are independent of the used filter sizes. Hence, the runtime is $\mathcal{O}(nd)$.

As mentioned in section 3.1, the linear runtime with a low number of operations per pixel makes block-matching stereo approaches very appealing for real-time applications. In contrast, other state-of-the-art algorithms have a less efficient runtime because they are based on non-linear runtime algorithms or the number of operations per pixels is very high. The next sections will show that the new cost function is not only fast to compute but also has a quality that comes close to elaborated global stereo algorithms.

3.3. Evaluation

3.3.1. Stereo Algorithm

For the evaluations in this thesis, a pre- and post-processing chain of block-matching stereo similar to the approach in [Fua, 1993] is used. This variant of block-matching stereo comprises the following five steps:

Firstly, the matching cost is calculated for all pixels and all disparities. From these the best matching is selected (winner-takes all). In a second step follows the interpolation of the disparity to sub-pixel accuracy by fitting a quadratic curve to the matching scores in the neighborhood of the optimum. The third step is a left-right consistency check [Fua, 1993, sec. 2.1] for detecting occlusions and mismatches, i.e. matches that do not pass the left-right check are considered as wrong. Furthermore, in a fourth step small disparity segments are removed by means of a simple region growing on the disparity map. This step can be regarded as a smoothness constraint because large and smooth disparity segments are kept. In the following experiments, segments with an area smaller than 200 pixels are considered to be invalid. In the final step, invalidated pixels are filled by interpolating from the first left and first right neighbor that have a valid entry.

Another important thing to note is that colored stereo images are always transformed to gray. The reason for this is that color rather deteriorates than improves the quality of correlation-based stereo computations [Bleyer and Chambon, 2010]. A positive side-effect is an increase in speed because the amount of pixel data reduces to one third.

3.3.2. Experimental Setup

In order to evaluate the newly proposed SNCC cost function, the stereo data sets Venus, Tsukuba, Teddy and Cones from the Middlebury benchmark [Scharstein and Szeliski, 2002] are used. Figure 3.9 displays the left stereo image of the four scenes. These scenes have been chosen because the online table of the Middlebury benchmark enables to compare the results of the proposed method to many other stereo algorithms (even future ones) for different criteria. In the Middlebury benchmark, stereo algorithms are assessed by means of the percentage

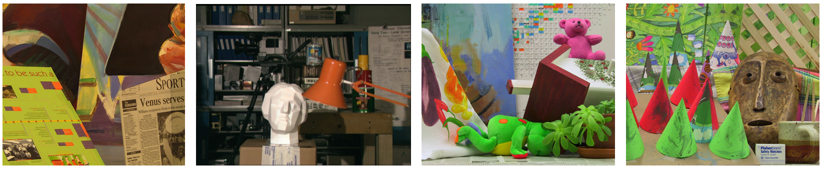


Figure 3.9.: The left stereo images of the Venus, Tsukuba, Teddy and Cones scene.

of bad pixel b_p

$$b_p = \sum_x |D_x - GT_x| > \delta \ , \tag{3.16}$$

where D is the disparity map computed by an algorithm, GT is the ground truth disparity, x are the image coordinates of the area of interest and δ the error threshold. The quality assessment of the Middlebury online table encompassed three different areas of interest (all, non-occluded, discontinuities) and five different error thresholds (0.5, 0.75, 1.0, 1.5, 2.0). For each scene and each of the three areas of interest the creators of the Middlebury benchmark have defined a binary map.

In the following evaluation the proposed *summed normalized cross-correlation* (SNCC) is compared to the standard cost functions *sum of absolute differences* (SAD), *normalized cross-correlation* (NCC) and *rank transform* (RT) [Zabih and Woodfill, 1994] using the block-matching stereo algorithm explained in section 3.3.1. Please note that for all cost functions the same pre- and post-processing steps and the same parameters of the stereo algorithm are used. Furthermore, the first stage (NCC-stage) of SNCC is always run with a 3x3 filter size, i.e. only the second stage summation area is varied. Similarly, the filter size of the rank transform is always 11x11, i.e. filter sizes are only varied for the SAD filtering that follows the rank transform. A filter size of 11x11 was chosen because this gave the best overall results for RT.

3.3.3. Experimental Results

In the first evaluation, the quality of block-matching stereo with the four different cost functions SAD, NCC, RT and SNCC for different filter sizes is assessed. Of course, a good cost function should have a minimum error for some optimal filter size but in order to generalize well to different scenes the error should degrade gracefully when deviating from this optimum. For all four cost functions squared filters of odd sizes from 3x3 to 31x31 are used. Figure 3.10 shows the error plots of the four cost functions for an error threshold of 0.5.

The first column depicts the results for the non-occluded area of interest, i.e. parts of the scene that are seen in both stereo images. One can observe that SAD and NCC perform similar on Tsukuba, but for the other scenes NCC is better than SAD for small filter sizes. Comparing RT and SNCC with SAD and NCC shows that the former two outperform the latter two for Venus, Teddy and Cones. It strikes that all four cost functions perform similar for Tsukuba. One reason could be that the Tsukuba scene is overall more homogeneous than the other scenes. Therefore, matching costs taking structure into account cannot extract additional information. Another thing to note is the minor difference between the optimal performance of RT and NCC. However, RT degrades much more gracefully with increasing filter size. At first glance this difference in quality for the non-occluded area of interest seems puzzling. However, as was shown in section 3.2.1 strong contrast edges can influence their surrounding depending on the filter size. This also explains why NCC has a similar quality like RT for smaller filter sizes but gets worse for larger filter sizes. RT and SNCC both reduce the influence of high contrast edges on the surrounding and hence degrade much more gracefully beyond their optimal point. Comparing the error curves of RT and SNCC reveals that SNCC and RT are similar for scenes with few occlusions (Venus, Tsukuba) but that SNCC is substantially better for scenes with many occlusions (Teddy, Cones). In particular SNCC is much better than NCC which confirms the improvement of this approach.

The observations made for the non-occluded area of interest are even more pronounced for the depth discontinuity area of interest (disc) shown in the second column of Fig. 3.10. The characteristics for SAD and NCC are similar to the non-occluded area but the disparity error increases more rapidly for increasing filter sizes and seems to settle for the largest filter sizes. Comparing again RT with SNCC shows that SNCC is clearly better at preserving the correct position of depth discontinuities for all scenes. Moreover, SNCC is significantly better than NCC, which confirms the argumentation in sections 3.2.1 and 3.2.2.

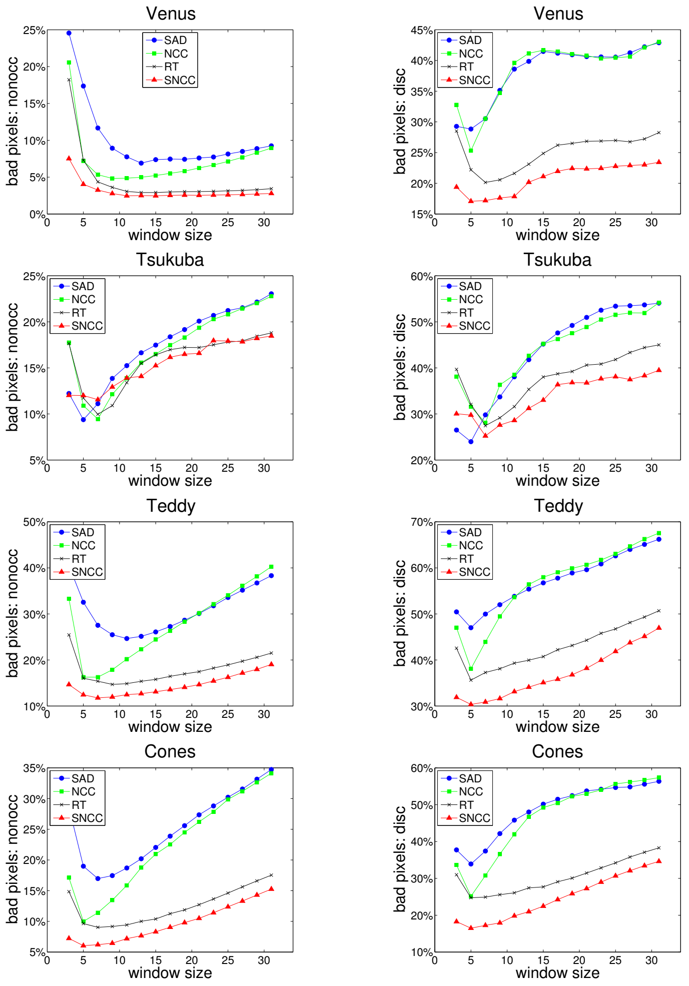
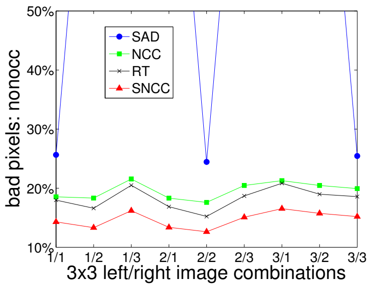
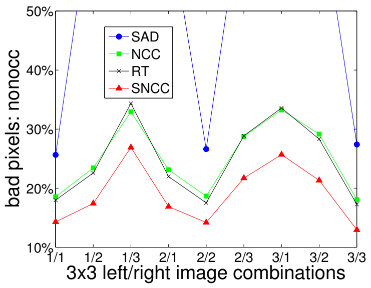


Figure 3.10.: The performance of block-matching stereo for the non-occluded (nonocc) and depth discontinuity (disc) areas of interest. The plots show the percentage of bad pixels against the patch window size for the proposed two-stage matching SNCC compared to SAD, NCC and RT. The first column shows results obtained for the non-occluded regions. The second column shows results obtained for depth discontinuities. It can be seen that SNCC (red curves) outperforms all the other cost functions considerably.



(a) Different exposure



(b) Different lighting

Figure 3.11.: The performance of SAD, NCC, RT and SNCC under lighting and exposure changes using a filter size of 11x11. i/k denotes that the left image had lighting or exposure i and the right image lighting or exposure k. The results clearly demonstrate that SNCC retains the invariance properties of NCC and that SNCC outperforms the other cost functions.

In a second evaluation, it was investigated to which extent SNCC retains the illumination invariance properties of NCC. For doing so, the Art, Books, Dolls, Laundry, Moebius and Reindeer stereo images [Hirschmüller and Scharstein, 2007] of the Middlebury database were used. These are stereo scenes taken under three different lighting conditions and three different exposures. In accordance to [Hirschmüller and Scharstein, 2007], all possible combinations of lighting for the left and right image were used, i.e. nine different lighting pairs for each scene. The same was done for the different exposure setups. In this experiment, for all cost functions a filter size of 11x11 was used. Figure 3.11 shows the result for the nine lighting and nine exposure pairs averaged over all six scenes. It demonstrates that SNCC indeed has the same invariance properties with respect to lighting like NCC. Furthermore, the results highlight that SNCC performs better than RT.

The results of the proposed SNCC cost function were submitted to the online Middlebury stereo evaluation. For this submission a non-square filter was used because it turned out that for SNCC vertical elongated filters produce slightly better results. The best overall results were achieved with a 5x9 filter for the second stage but the first stage was left untouched with a filter size of 3x3. Table 3.1 shows a small snapshot of the online Middlebury benchmark for an error threshold of 0.5 comparing SNCC to some selected stereo algorithms. For a full overview, which features also other error thresholds, please visit (`http://vision.middlebury.edu/stereo/eval`). The small numbers next to the percentage of bad pixel indicate the rank among the 87 algorithms that have been in the online table at the time of the submission (July 2010) to the benchmark. For many of the criteria, the newly proposed SNCC for block-matching stereo is among the top ten algorithms. For the Cones scene, the proposed approach yields the best overall result for the non-occluded area of interest. Usually, the top ranks are held by global stereo matching approaches based on belief propagation, graph cut, bilateral filtering or dynamic programming. In contrast the results highlight that by means of SNCC the block-matching stereo approach is able to get closer to the quality of state-of-the-art global approaches.

However, there is one exception to the good performance of SNCC and that is the Tsukuba scene. Although the block-matching stereo approach with SNCC has a high ranking for the whole scene and the non-occluded area of interest, it has a very bad rank of 74 for the depth discontinuities. One reason for this might be that the Tsukuba scene is overall more homogeneous than the other test scenes. The Tsukuba test scene is already quite old and the cameras at that time could not distinguish small intensity difference as good as current cameras. This means that the normalized cross-correlation in the first stage yields low matching values for many parts of the scene. Here approaches that constrain homogeneous areas to have one disparity (like many global stereo methods) are advantageous because they can robustly fill-in large gaps.

For a visual impression and comparison, Fig. 3.12 depicts the disparity maps (first column) that correspond to the SNCC results in Table 3.1 together with the ground truth maps (second column). What strikes most are the local artifacts that are mainly due to the simple fill-in mechanism used. This is seen best in the disparity map of the Tsukuba scene. For example, a part of the table leg is smeared to the right. These artifacts could be reduced by using an appearance-based fill-in that takes the original pixel information into account. Nevertheless, the quality of the disparity maps is already very good which again highlights the improvements that can be achieved using SNCC for block-matching stereo computations. For more results on other

Table 3.1.: A small snapshot of the online Middlebury benchmark at the time of submission (July 2010). It gives a comparison of the proposed SNCC approach to Belief Propagation (BP) [Banno and Ikeuchi, 2009], Semi-Global Matching (SGM) [Hirschmüller, 2005], Fast Bilateral Stereo (FBS) [Mattoccia et al., 2009] and the Adaptive Weight approach (AW) [Yoon and Kweon, 2006]. The numbers denote the percentage of bad pixels (pixels whose disparity differs more than 0.5 from the ground truth data) and the subscript numbers denote the rank among the 87 algorithms compared in the online benchmark [Scharstein and Szeliski, 2002] at the time of submission.

	Venus			Tsukuba			Teddy			Cones		
cost	noocc	all	disc	noocc	all	disc	noocc	all	disc	noocc	all	disc
BP	1.59_{6}	2.34_{7}	6.94_{5}	6.21_{4}	7.96_{7}	24.5_{64}	7.78_{2}	17.3_{10}	22.5_{3}	4.73_{2}	14.6_{22}	$\mathbf{10.7}_{1}$
SGM	4.55_{22}	5.38_{23}	15.7_{41}	13.4_{30}	14.3_{29}	20.3_{38}	11.0_{11}	18.5_{14}	26.1_{12}	4.93_{4}	12.5_{11}	13.5_{8}
FBS	5.71_{26}	6.66_{30}	14.9_{37}	21.5_{53}	22.4_{54}	22.9_{55}	16.2_{40}	23.3_{42}	32.1_{46}	9.10_{25}	15.8_{29}	18.1_{32}
AW	7.77_{42}	8.40_{43}	15.8_{42}	18.1_{37}	18.8_{36}	18.6_{24}	17.6_{47}	23.9_{44}	34.0_{59}	14.0_{49}	19.7_{45}	20.6_{40}
SNCC	2.35_{9}	3.23_{10}	15.4_{40}	11.3_{20}	12.3_{18}	27.5_{74}	10.6_{9}	15.2_{4}	28.6_{24}	$\mathbf{4.71}_{1}$	11.1_{4}	13.2_{6}

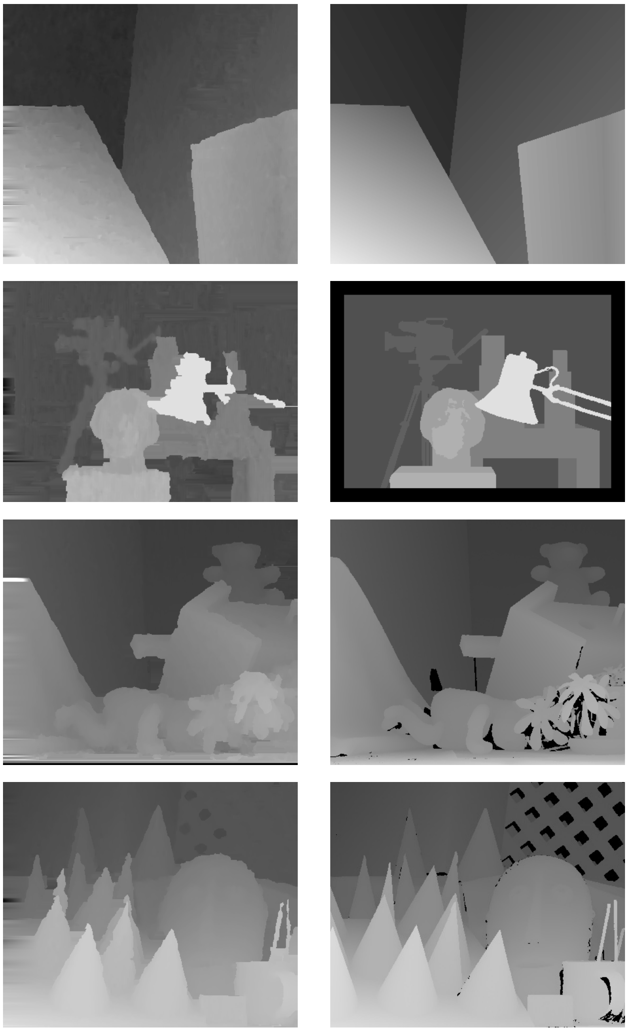


Figure 3.12.: From top to bottom these are disparity maps of the Venus, Tsukuba, Teddy and Cones scene. The first column shows the result of the proposed SNCC for block-matching stereo and the second column the ground truth disparity maps. The disparity values are encoded by means of intensity. Near pixels are bright and far pixels are dark. Black pixels in the ground truth disparity maps indicate unavailable ground truth disparity.

Table 3.2.: A comparison of the absolute runtime for the Teddy scene of block-matching stereo with SNCC and selected algorithms: Semi-Global Matching (SGM), Belief Propagation (BP), Fast Bilateral Stereo (FBS), Graph Cut (GC) and the bilateral filtering approach Adaptive Weight (AW).

Algorithm	CPU Speed	Runtime (sec)
SNCC	1x3.0 GHz	0.14
SGM [Hirschmüller, 2005]	1x2.8 GHz	1.3
BP [Hirschmüller, 2005]	1x2.8 GHz	4.2
FBS [Mattoccia et al., 2009]	2x2.14 GHz	32
GC [Hirschmüller, 2005]	1x2.8 GHz	55
AW [Mattoccia et al., 2009]	2x2.14 GHz	3226

standard benchmark scenes please refer to appendix B.

3.3.4. Runtime

In section 3.2.3, it was argued that linear runtime stereo algorithms are best suited for real-time applications and that SNCC can be implemented as such. In order to support this statement, Table 3.2 compares the absolute runtime of block-matching stereo with SNCC to some selected state-of-the-art stereo approaches. For this comparison the Teddy scene is used because it is quite common to give the absolute runtime for this stereo scene. Please note that such a comparison, although quite common, is only a very rough one. The problem is that the actual runtime of a piece of code depends among other things on the CPU speed and cache, programming language, hardware architecture and most importantly on the skill of the programmer. Nevertheless, the comparison confirms that state-of-the-art stereo approaches cannot cope with the speed of the block-matching stereo approach. The next fastest approach is SGM which still is one order of magnitude slower than block-matching with SNCC.

Cache Efficiency and Parallelization

Even though the runtime of block-matching stereo with SNCC is very good it is still too slow for many real-time vision systems. On the one hand some applications like intelligent vehicles might demand for a processing speed of 20Hz and more. On the other hand, larger vision systems incorporate a multitude of different algorithms which altogether have to run in real-time. One might argue that using a computer with multiple processing cores solves the latter problem because each algorithm can run on a separate processing unit. However, this is only half the truth. In general, the algorithms of a larger system are not independent, i.e. there are algorithms that depend on the output of other algorithms. For stereo processing this is a critical issue because it is typically one of the first algorithms in a processing chain and also one of the computationally most demanding which can severely delay the subsequent processing. Fortunately, there are some means to speed the processing up.

The first thing one should consider is the cache structure of a modern x86 CPU. The cache is a fast temporal memory which is used to store local copies of the much slower main memory (RAM). If a program uses a part of the main memory that fits into the cache then the program can run very fast because it can work directly with the fast cache memory. If, however, the memory used by the program does not fit into the cache then there is a constant data transfer from and to the slower main memory which increases the processing time. Knowing this effect, block-matching stereo can be accelerated by splitting the stereo image. Instead of processing the whole stereo images at once, they are split into smaller horizontal stripes which have a much smaller memory footprint and thus are more likely to fit into the cache.

Figure 3.13 shows an example of splitting an image into two horizontal stripes. As can be seen the stripes have an overlapping area. This overlap is necessary due to the patch-based correspondence search. For a vertical patch length of n pixels the overlap is $n-1$ pixels. The resulting disparity map can easily be constructed from the partial results of the single stripes. In order to measure the cache effect, the splitted block-matching stereo with SNCC is run on a Xeon X5355 with 2.66GHz and 4096kb of cache. Figure 3.14(a) shows the processing time in ms for an increasing number of image stripes. For this measurements only the steps 1-3 of the stereo algorithm (see section 3.3.1) were applied because it is not straightforward to modify the region growing to work with splitted images. Processing the full stereo images takes 240ms. For four stripes the processing time decreases to

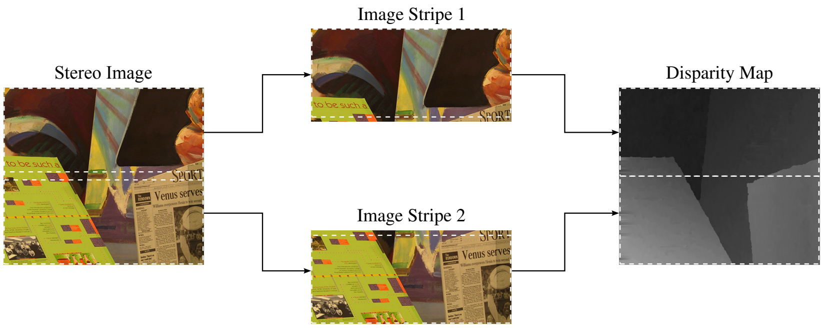


Figure 3.13.: On a modern x86 CPU programs run faster if their memory fits into the cache. Often the memory used by stereo processing exceeds the cache size. Splitting up the stereo images into horizontal stripes and processing each stripe separately increases the processing speed due to a more efficient cache usage. As block-matching stereo is a local processing the only overhead is that the stripes have an overlapping area that corresponds to the patch size used for the correspondence search.

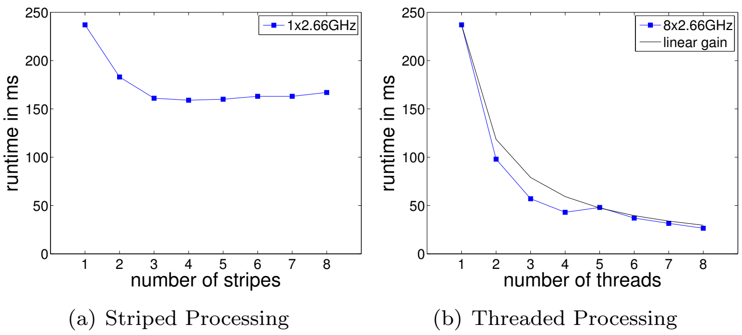


Figure 3.14.: (a) Performance of cache efficient block-matching stereo with SNCC which processes the stereo images in stripes (see Fig. 3.13). (b) Performance of parallel block-matching stereo with SNCC for an increasing number of threads. The performance measurement was done on an eight-core machine (two Xeon X5355 with 2.66GHz). Due to positive cache effects the speedup for multiple threads is super-linear. Note that for both evaluations only steps 1-3 of the stereo algorithm (see section 3.3.1) are applied.

151ms, i.e. a reduction of almost 40%. Beyond four stripes the processing time starts to increase again due to the overlapping of the stripes. However, the increase in runtime with respect to the number of stripes is very small. Even for eight stripes the processing time is still 160ms. Thus the speed up is very robust with respect to the number of stripes.

A second way of improving the processing speed is to use multiple cores for stereo processing. We are currently at the verge of a paradigm change. CPUs are not getting faster any more but the number of processing cores increases. Hence, it is necessary to adapt algorithms to this more parallel paradigm. For doing so, the splitted stereo approach can easily be extended to a multi-threaded one which can exploit multi-core architectures. It is easy to see that the single stripes in Fig. 3.13 are independent of each other and thus can be processed by one thread each. A big advantage of this threaded approach is that there is no need for synchronization except for waiting for all threads to finish. In contrast to the stereo images the resulting disparity map has no overlap areas which means that the threads can write non-blocking into their exclusive memory part.

For testing the parallel block-matching stereo a machine with two Xeon X5355 was used. Each Xeon X5355 has four cores, i.e. the machine has eight cores altogether. Figure 3.14(b) shows the runtime of the parallel block-matching stereo with SNCC for the Teddy stereo scene for an increasing number of threads. Again for this speed measurement only the steps 1-3 of the stereo algorithm (see section 3.3.1) are applied. The reason for this is that a parallelized segmentation is not straightforward and using a non-parallel implementation would give a negative bias to the processing time of multiple threads. The evaluation shows that the proposed parallel algorithm has a super-linear gain due to the positive cache effect caused by the reduction of memory per thread for an increasing number of threads (like for the striped processing). Since two cores share one cache they hamper each other if both are under heavy load. Hence, the processing time for five to eight threads is less efficient than for up to four threads. Nevertheless, the speed of the parallel block-matching stereo with SNCC for the Teddy scene increases from 4.2Hz for one thread to nearly 37.7Hz for eight threads, an increase by factor 9.

In contrast to the threaded implementation introduced in this work, the current trend for implementing algorithms in real-time is rather

Table 3.3.: A comparison of real-time stereo implementations on GPU and the proposed threaded approach using multiple CPUs for block-matching stereo with SNCC. Compared algorithms are RT Census [Humenberger et al., 2010], Realtime BFV (Bitwise Fast Voting) [Zhang et al., 2009], Realtime BP [Yang et al., 2006], Realtime GPU (a DP approach) [Wang et al., 2006] and Reliability DP [Gong and Yang, 2005]. The names of the algorithms correspond to the ones used in the online table of the Middlebury comparison. From the same online table the average ranks were taken for a disparity error threshold of 0.5 and 1.0. The speed of the algorithms is measured by means of MDE/s (Million Disparity Estimations per second).

Algorithm	Avg. Rank $\delta = 0.5$	Avg. Rank $\delta = 1.0$	Platform	MDE/s
SNCC	22.5	61.1	2 × Xeon 5355	337.2
SNCC	22.5	61.1	2 × Xeon 5550	470.4
RT Census	31.1	69.0	GeForce 9800GT	534.6
RT Census	31.1	69.0	GeForce GTX 280	1067.0
Realtime BFV	64.8	53.8	GeForce 8800GTX	129.6
Realtime BP	68.1	54.5	GeForce 7900GTX	22.2
Realtime GPU	81.2	69.4	Radeon XL1800	53.0
Reliability DP	80.0	73.9	Radeon 9800 XT	20.0

to use graphic cards, i.e. graphical processing units (GPU). Table 3.3 compares stereo algorithms that were implemented on a GPU with the threaded SNCC on two different types of eight-core machines. The speed is given by MDE/s (Million Disparity Estimations per second) which is the product of the image width and height, the disparity range and the frames per second. The advantage of using MDE/s is that it enables to compare the speed of stereo algorithms that were tested in different setups (image size and disparity range) by means of a scalar value. A drawback of using MDE/s is the implicit assumption that MDE/s changes linearly with respect to the single factors. Unfortunately, this is not always the case, e.g. on GPUs the MDE/s usually increases with the image size because of a better utilization of the processing cores. Nevertheless, the comparison in Table 3.3 highlights that also for parallel implementations the block-matching stereo approaches are much faster than other stereo algorithms. For example the real-time GPU implementation of block-matching stereo with census transform [Humenberger et al., 2010] is ten times faster than any of the other GPU implementations while having even a higher depth accuracy. Comparing the GPU census approach to the threaded SNCC approach shows that SNCC is slightly better in depth accuracy while being a third or half as fast as the census implementation, depending on the used platform.

However, the threaded implementation has some advantages compared to the GPU implementation. First, the effort for writing parallel CPU code is much smaller than to write parallel GPU code. For the CPU a simple wrapper that splits the stereo images into smaller parts and starts a thread with the original stereo code for each part is enough. On the other hand the GPU code has to be written from scratch because hardware architectural differences have to be taken into account. Thus a change of the algorithm has to be applied to both the CPU and GPU code which increases the maintenance effort. Second, the portability of GPU code is currently quite bad. The technique is rather new and the compilers are far from the performance of CPU compilers. Hence, it is necessary to write code that takes hardware specifics into account to reach high performance computing. This means that using a newer graphic card model might require to rewrite parts of the GPU code in order to use the full potential of the new card. In contrast CPU compilers are able to produce good code across different CPUs, even for CPUs of different vendors.

3.4. Summary

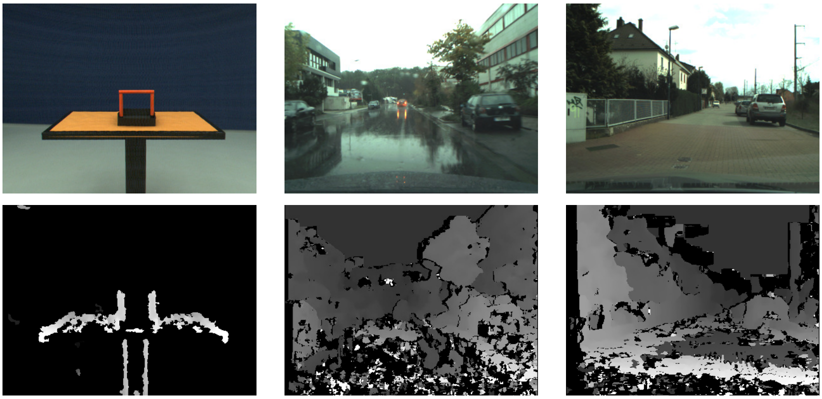
In this chapter the novel matching cost function *summed normalized cross-correlation* (SNCC) was introduced. SNCC was motivated by the fattening problem of the *normalized cross-correlation* (NCC). Among other things, this problem leads to an imprecise location of depth discontinuities. It was shown here that the underlying reason for the fattening effect of NCC are large intensity differences (strong contrasts) between pixels within a matching window. By reducing the correlation-window size and introducing a second summation filter stage the fattening effect is strongly reduced.

The improvement of SNCC over NCC was shown by means of several evaluations using standard benchmark stereo images. A comparison to a large set of state-of-the-art approaches revealed that block-matching with SNCC has a depth accuracy that is close to elaborated global stereo approaches based on graph cut or belief propagation. Further experiments with stereo images taken under varying lighting and exposure conditions proved that SNCC inherits the excellent illumination invariance properties of NCC. Moreover, theoretical considerations and practical measurements show that block-matching stereo with SNCC is able to produce disparity images in real-time. By splitting the stereo images in horizontal stripes the block-matching approach can easily be distributed to multiple cores. This makes the approach ready for the current change in CPU development towards more and more processing cores and enables to reach super real-time performance which is necessary for large vision systems. A comparison to state-of-the-art real-time implementations of stereo algorithms on GPUs showed that the gain in speed is comparable but that the multi-core block-matching with SNCC is much easier to implement and to maintain. Altogether, SNCC has proven to give block-matching stereo a very good trade-off between speed and accuracy by having a linear runtime with small constants like other block-matching cost functions and at the same time having an accuracy that comes close to elaborated global stereo approaches.

4. Model-Based Depth Estimation

In the previous chapter an improved block-matching stereo was presented. Although, the experiments have shown that this approach leads to accurate depth estimations with real-time speed, its local processing scheme is prone to the so-called aperture problem. The term aperture summarizes all kinds of ambiguities that arise from a local matching correspondence search. One way of reducing the aperture problem is to incorporate models into the depth estimation process. This may be done as some kind of post-processing by fitting models into the estimated depth or disparity data, or by directly integrating models into the correspondence search itself.

In this chapter a novel method [Einecke et al., 2010] for integrating parametric surface models into the correspondence search is presented. The main idea is to replace the usual gradient descent optimization by a Hooke-Jeeves [Hooke and Jeeves, 1961] optimization. This new approach overcomes the restriction to planar models of current approaches which are based on the homography mapping. Furthermore, experiments [Einecke and Eggert, 2010a] show that the fitting based on Hooke-Jeeves is more robust than state-of-the-art methods. As a matter of fact model-based stereo approaches are much slower than block-matching stereo. However, it is not the goal to replace block-matching stereo by a model-based depth estimation. The idea is rather that a model-based approach is used as a complementary step by applying it only to critical parts of the scene where the block-matching approach is likely to fail.



(a) Weak Texture (b) Depth Overlay (c) Repetitive Patterns

Figure 4.1.: Aperture problem of block-matching stereo. (a) Weakly textured areas lead to ambiguous correspondences because of a strong impact of the image noise. (b) Transparent or partly mirroring surfaces introduce ambiguities because of multiple correspondences. (c) Repetitive patterns like the shown cobble stones lead to multiple matches which can only be resolved by additional constraints or prior knowledge.

4.1. Limits of Block-Matching Stereo Processing

In chapter 3 a new cost function for block-matching stereo has been introduced which leads to a drastic improvement compared to common cost functions. Unfortunately, local matching is always prone to the aperture problem. Figure 4.1 shows three examples where the standard block-matching depth estimation is prone to fail.

The first example are weakly textured surfaces. Due to the weak texture the correspondence search is easily confused by image noise and hence can not find correct matches. If the amount of false correspondences caused by a weakly textured surface is not too high then post-fitting a surface model into the depth data can improve the depth estimation. Otherwise, the disturbances might dominate the post-fitting process and lead to a deterioration of the depth estimation. Here approaches that incorporate model assumptions into the correspondence search are advantageous because they can improve the matching quality itself by increasing the matching area.

The second example is an overlay of structures from different depths. This may occur at mirroring surfaces like a wet street or at transparent surfaces. In both cases the correspondence search gets ambiguous because there are two seemingly correct correspondences that overlay. This can only be resolved by giving a bias to one of the two competing depths or by using a probability distribution for representing the depth and shifting the decision problem to another processing stage.

The third example shows the problem of ambiguity that arises from repeating patterns. This is one of the hardest problems for local matching because the level of ambiguity is not constrained. Even by incorporating model assumptions this problem can be quite severe. Again approaches that incorporate the model in the correspondence search are advantageous over post-fitting methods because the disparity map carries already the wrong information. However, in most cases additional temporal integration or scene knowledge might be necessary for a coherent and robust solution of this problem.

4.2. Related Work

As explained above, there are two main approaches for model-based depth estimation. On the one hand models are fit into pre-calculated disparity or 3-D point cloud data. The major technique used here is RANdom SAmple Consensus (RANSAC) [Fischler and Bolles, 1981]. On the other hand models are integrated directly into the correspondence search. This means that the model assumption constrains the correspondence search to be coherent with a certain model. Almost all approaches of this category are based on the homography mapping which limits these to a planar surface model.

4.2.1. RANSAC

The most straightforward way of fitting a model into stereo disparity or 3-D point cloud data is to apply a least squared error (LSE) fitting. It is well known, however, that LSE fitting is very sensitive to outliers. Hence, it is necessary to remove outliers prior to the actual fitting. The by far most frequently used technique for this purpose is the RANSAC [Fischler and Bolles, 1981] method, which comprises three basic steps:

1. Randomly select just enough data points for a direct model parameter calculation, e.g. three data points for a planar model.

2. Analytically determine the model parameters from the selected data points, e.g. calculate the orientation and anchor point of a plane from three data points that are not co-linear.

3. Calculate the census set. This set consists of all data points whose distance from the estimated model is below a threshold δ. Sometimes only the cardinality of the set is calculated.

The above steps are repeated for a certain number of times. Afterwards the largest census set is used for a final fitting of the model, e.g. by means of LSE fitting. Since the census set consists only of points that were close to a sampled model instance, it contains no outliers. Sometimes the final fitting is omitted and the parameters of the largest census set are used.

4.2.2. Homography-constrained Depth Estimation

Fitting models into pre-processed disparity data mainly improves the depth accuracy by averaging the disparity estimations of a set of pixels according to the selected surface model. In contrast, incorporating surface models directly into the correspondence search improves the depth estimation itself. The reason is that this allows for much larger patch sizes to be matched. A common way of doing so is to incorporate a homography transformation H [Hartley and Zisserman, 2004] into the correspondence search. A homography describes the perspective view-transformation of a planar surface between arbitrarily placed cameras (the planar surface must be visible in the corresponding views).

One advantage of the homography transformation is that it can be written in matrix notation which allows for an elegant way of incorporating it into a least squared error optimization. The main idea is to estimate the homography parameters that best describe the observed view transformation of an image region S. For two camera images I

and J, this estimation is described by the minimization

$$\min_{\mathbf{p}} \sum_{\mathbf{x} \in S} \big(I(\mathbf{x}) - J(H(\mathbf{x}, \mathbf{p})) \big)^2 , \tag{4.1}$$

where $H(\mathbf{x}, \mathbf{p})$ is the homography mapping of the image coordinates $\mathbf{x}$ with the parameters $\mathbf{p}$. Deriving (4.1) for the homography parameters $\mathbf{p}$ leads to an iterative gradient descent which is very similar to the image registration proposed by Lucas and Kanade [Lucas and Kanade, 1981]. For example Habbecke and Kobbelt [Habbecke and Kobbelt, 2005] elaborated on this idea using a Gauss-Newton style matching and approximated partial image derivatives. They also managed to extend the minimization to take more than two images into account leading to very accurate and robust 3D estimations in multi-camera setups.

The main disadvantage of a minimization based on the homography mapping is its restriction to a planar surface model. Unfortunately, there is no straightforward way of extending this approach to other surface models. Furthermore, the minimization in a squared error fashion is prone to fail for illumination differences between the different images, a case which is quite common in real-world applications.

4.2.3. Affine-constrained Depth Estimation

For rectified stereo images the homography transformation breaks down to an affine transformation consisting of translation, shear and scaling. Compared to the homography-based estimation this reduces the number of parameters from eight to three which greatly improves the robustness and speed. The minimization is simplified to

$$\min_{\mathbf{A}, \mathbf{d}_a} \sum_{\mathbf{x} \in S} \big(I^L(\mathbf{x}) - I^R(\mathbf{A}\mathbf{x} + \mathbf{d}_a) \big)^2 . \tag{4.2}$$

Here $\mathbf{A}$ describes the scaling and the shear and $\mathbf{d}_a$ describes the translation. Since in rectified stereo images the y-position of corresponding pixels is the same, only the scale and shear in x-direction is of interest. Thus, the three parameters of interest $\mathbf{p} = (p_1, p_2, p_3)$ are the scale in x-direction p_1, the shear in x-direction p_2 and the translation in x-direction p_3

$$\mathbf{A} = \begin{pmatrix} p_1 & p_2 \\ 0 & 1 \end{pmatrix} \ , \ \ \mathbf{d}_a = (p_3, 0)^T \ . \tag{4.3}$$

Apart from the simplification, the affine-based estimation is equivalent to the homography-based estimation in a rectified stereo setup. In particular, it shares the disadvantage of supporting only planar surfaces.

4.3. Direct Surface Fitting

This section introduces a novel approach called *direct surface fitting* which overcomes the planar limitation of a homography-constrained depth estimation. The main idea is to use a direct search method [Lewis et al., 2000] for optimization (hence the name "direct" surface fitting) instead of a gradient descent approach. The reason for this is that direct search methods work without explicit gradient information. This enables an easy way of optimizing for parameters of non-linear surfaces like spheres or cylinders which otherwise would be very difficult or infeasible. In this work the Hooke-Jeeves optimization [Hooke and Jeeves, 1961] is used because of its simple but robust nature.

Similar to the gradient-based approaches, the objective of the direct optimization is to minimize the error between the original left view and a transformed right view

$$\min_{\mathbf{p}} \sum_{\mathbf{x} \in S} C\big(I^L(\mathbf{x}) - I^R(T_M(\mathbf{x}, \mathbf{p}))\big) \ , \tag{4.4}$$

where $C(I, J)$ is an (error) cost function measuring the difference between two images I and J, and T_M is an image transformation that warps a right camera image into the view of the left camera image under the constraint of a surface model M. The question is, how to realize the image transformation T_M for models others than planes. In this work the image transformation formulas are derived by replacing the depth z_L in the basic stereoscopic mapping equation (2.19), which was discussed in section 2.2.3, with the parametric description of a geometric model.

The next three sections will sketch the derivations for planes, spheres and cylinders. The full derivations can be found in the appendices (A.1, A.2, A.3). Although only three surface models are discussed here the method is applicable in an analogous way to any other parametric surface. Please note that the following derivations are based on a parallel stereo setup with rectified images because these are the underlying assumptions of the basic mapping equation (2.19). However, as the derivations will reveal, the approach itself is not constrained to such a setting and could easily be extended to cameras in general position by further taking the essential matrix **E** from section 2.2.2 into account.

4.3.1. Planar Model

The first surface model that is discussed in this work is a planar model. In order to get a transformation that describes the perspective view-changes of a parametric surface, its mathematical description needs to be rearranged for z_L and substituted in the basic mapping equation (2.19). For describing a planar surface model the normal equation of a plane could be used. However, a description based on rotational angles is used here because such parameters are more intuitive to understand and evaluate. Moreover, it is very easy to restrict the equations to only three free parameters which simplifies the parameter search process. Please note that the following derivation is only a sketch of the detailed derivation in appendix A.1.

Figure 4.2 shows a schematic top view of a planar surface which projects to the two camera chips of a parallel stereo camera. This planar surface (*target plane*) is described relative to a *virtual plane* which is parallel to the camera sensor chips. The two planes differ only by a rotation about the x- and y-axis and share a common anchor point $\mathbf{x}_a$. The orientation of the *target plane* is specified via rotation angles about the x-axis (α_x) and y-axis (α_y). Note that these two rotations suffice to describe any possible plane orientation. From analytical geometry, it can be derived that points $\mathbf{x}'$ from the *virtual plane* are transformed into points $\mathbf{x}$ on the rotated *target plane* by applying the transformation matrix

$$\mathbf{T} = \begin{pmatrix} \cos\alpha_y & \sin\alpha_x \sin\alpha_y & \cos\alpha_x \sin\alpha_y \\ 0 & \cos\alpha_x & -\sin\alpha_x \\ -\sin\alpha_y & \sin\alpha_x \cos\alpha_y & \cos\alpha_x \cos\alpha_y \end{pmatrix}, \tag{4.5}$$

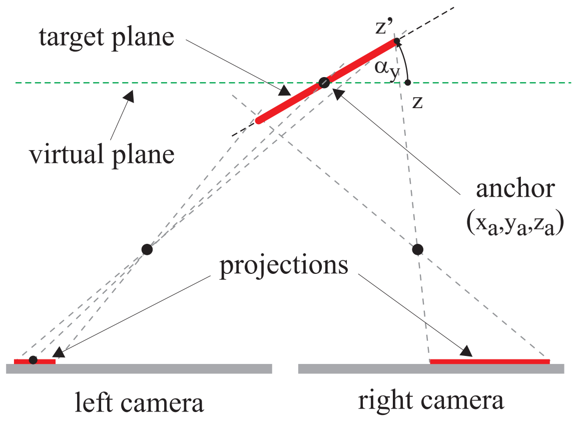


Figure 4.2.: Top-view of the projection of a planar surface to the image planes of a parallel stereo camera. The planar surface (target plane) can be described by an anchor point and the orientation relative to a virtual plane. This virtual plane is parallel to the camera sensor chips, i.e. the relative orientation is also relative to the image planes.

leading to the following transformation formula for the left camera (L) coordinate system

$$\mathbf{x}_L = \mathbf{T}\left[\mathbf{x}'_L - \mathbf{x}_a\right] + \mathbf{x}_a \ . \tag{4.6}$$

As the *virtual plane* is parallel to the image planes of the cameras, the z-coordinate for points on this fronto-parallel plane is always equal to the z-coordinate of the anchor point, $z'_L = z_a$. Using this, the transformation equation above can be rewritten to

$$\begin{pmatrix} x_L \\ y_L \\ z_L \end{pmatrix} = \mathbf{T} \begin{pmatrix} x'_L - x_a \\ y'_L - y_a \\ 0 \end{pmatrix} + \begin{pmatrix} x_a \\ y_a \\ z_a \end{pmatrix} . \tag{4.7}$$

With this, the depth z_L on the *target plane*, given the anchor point and rotation angles, reads as

$$z_L = (y'_L - y_a)\sin\alpha_x \cos\alpha_y - (x'_L - x_a)\sin\alpha_y + z_a, \tag{4.8}$$

where $(x'_L - x_a)$ and $(y'_L - y_a)$ can also be expressed with their counterparts on the rotated *target plane*. Expressions for both terms can be derived by rearranging the transformation equations (4.7)

$$x'_L - x_a = \frac{x_L - x_a - (y'_L - y_a)\sin\alpha_x \sin\alpha_y}{\cos\alpha_y} \tag{4.9}$$

$$y'_L - y_a = \frac{y_L - y_a}{\cos\alpha_x} \,. \tag{4.10}$$

Applying these two equations to the depth formula (4.8) and replacing the 3-D world coordinates with their 2-D chip projections (using the projection equations (2.17) and (2.18)) finally leads to

$$z_L = f\frac{x_a \sin\alpha_y - y_a \tan\alpha_x + z_a \cos\alpha_y}{u_{Lx}\sin\alpha_y - u_{Ly}\tan\alpha_x + f\cos\alpha_y} \,. \tag{4.11}$$

This equation describes z_L in terms of the parameters of a planar model. Substituting z_L in the basic mapping equation (2.19) leads to

$$u_{Rx} = u_{Lx} - b\frac{u_{Lx}\sin\alpha_y - u_{Ly}\tan\alpha_x + f\cos\alpha_y}{x_a \sin\alpha_y - y_a \tan\alpha_x + z_a \cos\alpha_y}\begin{pmatrix} 1 \\ 0 \end{pmatrix} \,. \tag{4.12}$$

Using the above mapping equation, the view of a plane in the left camera can be mapped to the view of that plane in the right camera by means of the planar parameters (z_a, α_x and α_y). The values for x_a and y_a can be chosen arbitrarily. They just define at which position the depth z_a of the planar model is estimated. Please note that the planar mapping equations correspond to the well-known homography transformation [Hartley and Zisserman, 2004]. This derivation was done in order to ease the understanding of the derivation of the other (non-linear) models that will follow in the next two sections.

4.3.2. Spherical Model

In this section the formulas for mapping a spherical surface are derived. This demonstrates the advantage of the proposed framework which is not restricted to a certain surface model like the homography transform. As in section 4.3.1, the target is to formulate z_L as a function of a parametric model. A sphere in the three-dimensional space with radius

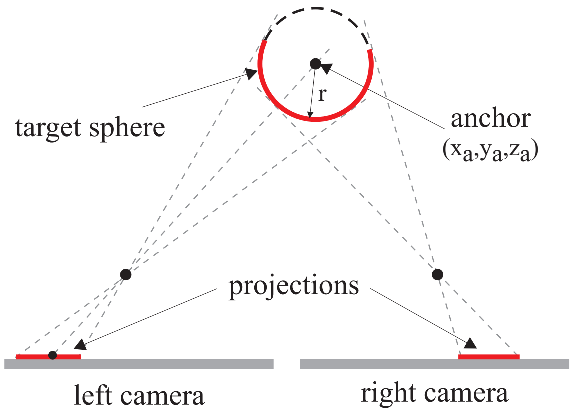


Figure 4.3.: Top-view of the projection of a spherical or cylindrical surface to the image planes of a parallel stereo camera. The spherical model is described by its center point. For a cylindrical model also the orientation of the symmetry axis with respect to the image planes has to be taken into account.

r can be described by

$$r^2 = (x - x_a)^2 + (y - y_a)^2 + (z - z_a)^2 \quad , \tag{4.13}$$

where (x_a, y_a, z_a) is the anchor point (center) of the sphere. For a graphical explanation see Fig. 4.3.

Similar to what has been done with the planar equations in section 4.3.1, the 3-D world points are replaced with their projections on the image planes using the projection equations (2.17) and (2.18). As the replacement is straightforward it is omitted for brevity, please see appendix A.2 for a full derivation. The resulting formula rearranged for z_L reads as

$$z_{L1,2} = \frac{\mu \pm \sqrt{\mu^2 - \nu\lambda}}{\lambda} \quad , \tag{4.14}$$

with

$$\lambda = 1 + \frac{u_{Lx}^2 + u_{Ly}^2}{f^2} \tag{4.15}$$

$$\mu = z_a + \frac{u_{Lx} x_a + u_{Ly} y_a}{f} \tag{4.16}$$

$$\nu = x_a^2 + y_a^2 + z_a^2 - r^2 \ . \tag{4.17}$$

At a first glance having two solutions in the spherical depth equation (4.14) looks puzzling. In fact, a closer look at Fig. 4.3 reveals that using the "–" in the spherical depth equation (4.14) means mapping a sphere (convex structure) and using the "+" means mapping a bowl (concave structure). Therefore, substituting z_L in the basic mapping equation (2.19) with the spherical depth equation (4.14) leads to two transformation equations. The first equation is for transforming the view of a sphere

$$\mathbf{u}_R = \mathbf{u}_L - \frac{bf\lambda}{\mu - \sqrt{\mu^2 - \nu\lambda}} \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \tag{4.18}$$

and the second for transforming the view of a bowl

$$\mathbf{u}_R = \mathbf{u}_L - \frac{bf\lambda}{\mu + \sqrt{\mu^2 - \nu\lambda}} \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \tag{4.19}$$

These equations allow for a mapping of the view of a sphere or a bowl from the left camera to the right camera by means of the spherical model parameters (z_a, x_a, y_a, r).

4.3.3. Cylindrical Model

The derivation of the formulas for the cylindrical model follows the same scheme like for the planar and spherical model. Since the formulas get a bit lengthy, the following derivation is just a very brief sketch. A detailed derivation is discussed in the appendix A.3.

The setup of the cylindrical model is very similar to that of the sphere (see Fig. 4.3) and is described mathematically by

$$r^2 = (x - x_a)^2 + (z - z_a)^2 \ . \tag{4.20}$$

This means the applied cylindrical model is infinite in the y-direction, i.e. along its symmetrical axis. In contrast to the spherical model, it is necessary to incorporate a rotation matrix like for the planar model

$$\mathbf{T} = \begin{pmatrix} \cos\alpha_z & -\sin\alpha_z & 0 \\ \cos\alpha_x \sin\alpha_z & \cos\alpha_x \cos\alpha_z & -\sin\alpha_x \\ \sin\alpha_x \sin\alpha_z & \sin\alpha_x \cos\alpha_z & \cos\alpha_x \end{pmatrix} . \tag{4.21}$$

For the cylindrical model, the rotation is only about the x-axis and the z-axis because a rotation about the y-axis would have no effect on the cylindric model because of symmetry. This leads to six parameters for the model of the cylinder with an anchor point (a_x, a_y, a_z), rotation angles (α_x, α_z) and radius r. But actually the model has only five parameters because the y-position for the infinitely expanded cylinder can be fixed to an arbitrary value. The remainder of the derivation proceeds in a way analogous to the plane and the sphere (see appendix A.3 for a full detailed derivation). The resulting depth formula has a structure similar to that of the sphere

$$z_{L1,2} = \frac{\tau \pm \sqrt{\tau^2 - \eta\kappa}}{\kappa} \ , \tag{4.22}$$

with

$$\kappa = u_{Lx}^2 \frac{A}{f^2} + u_{Ly}^2 \frac{B}{f^2} + 2u_{Lx}u_{Ly}\frac{C}{f^2} + \frac{2}{f}(u_{Ly}D + u_{Lx}E) + F \tag{4.23}$$

$$\eta = y_a^2 A + x_a^2 B + 2x_a y_a C + 2z_a(y_a D + x_a E) + z_a^2 F - r^2 \tag{4.24}$$

$$\begin{aligned} \tau = {} & u_{Ly} y_a \frac{A}{f} + u_{Lx} x_a \frac{B}{f} + (u_{Ly} x_a + u_{Lx} y_a)\frac{C}{f} + \\ & \frac{z_a}{f}(u_{Ly}D + u_{Lx}E) + y_a D + x_a E + z_a F \ , \end{aligned} \tag{4.25}$$

where

$$A = \sin\alpha_x \sin\alpha_z \cos\alpha_z \tag{4.26}$$
$$B = 1 - \cos^2\alpha_x \cos^2\alpha_z \tag{4.27}$$
$$C = \cos^2\alpha_z \tag{4.28}$$
$$D = 1 - \sin^2\alpha_x \cos^2\alpha_z \tag{4.29}$$
$$E = -\sin\alpha_x \cos\alpha_x \cos^2\alpha_z \tag{4.30}$$
$$F = \sin\alpha_x \sin\alpha_z \cos\alpha_z \ . \tag{4.31}$$

Substituting z_L of the basic mapping equation (2.19) with the cylindrical depth equation (4.22) leads to two transformation equations

$$\mathbf{u}_R = \mathbf{u}_L - \frac{bf\kappa}{\tau \pm \sqrt{\tau^2 - \eta\kappa}} \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \tag{4.32}$$

These equations allow for a mapping of the view of a cylindrical shape from the left camera to the right camera by means of the cylindrical model parameters (x_a, y_a, z_a, α_x, α_z, r). As was pointed out in section 4.3.2, "$-$" again corresponds to mapping convex structures and "$+$" corresponds to mapping concave structures.

4.3.4. Model Parameter Estimation

The mapping formulas that were derived in the above sections can now be used to adapt the formulation of the minimization (4.4) to a specific model. For optimization the direct search method of Hooke-Jeeves [Hooke and Jeeves, 1961] is used.

Hooke-Jeeves Parameters

```
(1) $s$: step size
(2) $s_{\min}$: minimal step size
(3) $P = \{p_1, ...p_n\}$: initial parameter set
(4) $f$: fitness function to minimize
```

Hooke-Jeeves

```
(1) $e_{\min} = f(p_1, ..., p_n)$
(2) while $s > s_{\min}$
```

```
        (2.1) $\forall i : e_i^+ = f(p_1, ..., p_i + s, ..., p_n)$
        (2.2) $\forall i : e_i^- = f(p_1, ..., p_i - s, ..., p_n)$
        (2.3) $e_{\text{new}} = \min\{e_1^+, e_1^-, ..., e_n^+, e_n^-\}$
        (2.4) $P_{\text{new}}$ = New $P$ according to $e_{\text{new}}$
        (2.5) *if* $e_{\text{new}} < e_{\min}$
                (2.5.1) $e_{\min} = e_{\text{new}}$
                (2.5.2) $P = P_{\text{new}}$
        (2.6) *else*
                (2.6.1) $s = s/2$
        (2.7) Optional: Use the last two $P$ to perform an interpolation step
    (3) return $P$
```

The working scheme of Hooke-Jeeves is simple. Starting from an initial parameter set P, an iterative refinement is conducted by sampling alternative parameter sets around the currently best solution using the step size s. From these alternative sets the best one is selected. If no better solution is found, the step size is reduced. This is repeated until a minimal step size $s_{\min}$ has been reached. For direct surface fitting a sampling step (Fig. 4.4) comprises the back-warping of the right stereo image into the view of the left stereo image by means of the sampling parameters of a parametric surface and the calculation of a fitness value by means of a cost function that evaluates the similarity (or difference) between the original left image and the back-warped right image.

Since direct search methods like Hooke-Jeeves optimize a fitness functions by means of sampling, they do not require an explicit formulation of the fitness's gradient. This leads to some advantages over gradient-based approaches. First, it is very easy to swap the model assumption in the minimization (4.4). In contrast to this, the formal description of the fitness gradient which is necessary for any kind of gradient descent, gets very complex for non-linear surface models. Second, the fitness gradients are eventually based on the image gradients. Unfortunately, these can only be approximated locally, e.g. by means of a Taylor expansion [Habbecke and Kobbelt, 2005, Lucas and Kanade, 1981]. Because of this, gradient-based approaches usually need to rely on a resolution pyramid. There is no such necessity when using a direct search method, because it searches the parameter space by means of a simulated-annealing-like sampling.

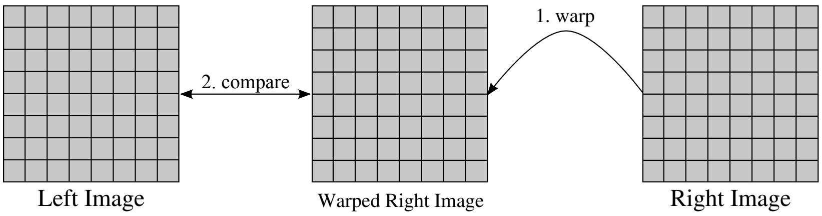


Figure 4.4.: A sampling step for Hooke-Jeeves in the direct surface fitting algorithm. By using the sampling parameters of a parametric surface the right stereo image is back-warped to the view of the left stereo images. Afterwards the original left image is compared to the warped image by means of a cost function that evaluates the similarity (or difference).

A third advantage of direct search methods is that they allow for arbitrary cost functions C. In gradient-based approaches the matching error between the original left image of a surface and the transformed right image is usually the sum of the pixel-wise squared error (SSD). In contrast, a direct search method allows for non-linear and non-continuous cost functions like SAD, NCC or SNCC which usually are infeasible for gradient descent. In particular, the cross-correlation measures are of interest in this work because their invariance against changes in the illumination makes the overall fitting more robust in real-world conditions. Last but not least, direct search methods are numerically very stable. For the fitting method presented in this work only simple arithmetic and trigonometric functions are used for warping the image and the calculation of the costs requires only some additions and multiplications.

Runtime

Notwithstanding its advantages, Hooke-Jeeves is rarely used as it is considered inefficient. Compared to gradient based approaches Hooke-Jeeves needs much more iterations until it converges. However, the overall speed depends on the function to optimize. Especially, using gradient based approaches on images is quite expensive because for calculating the local gradients the images have to be filtered in each iteration. This filtering is avoided when using a direct search method like Hooke-Jeeves. In [Habbecke and Kobbelt, 2005] a very efficient gradient method for plane estimation was proposed which is about a factor of two to three faster than the Levenberg-Marquardt minimization. Their implementation needs roughly 15 iterations. On an AMD Athlon 64 3500+ they need around 0.2ms for one iteration of a patch of 1000 pixels, i.e. the overall computation time is 3ms. In terms of iterations the Hooke-Jeeves implementation is quite expensive as it needs on average 175 iterations for planar surfaces. However, on a comparable system (one core of an Intel Xeon X5355) the overall computation time for a patch of 1000 pixels is 6.8ms. This demonstrates that Hooke-Jeeves can compete with state-of-the-art gradient based optimization when it comes to plane fitting.

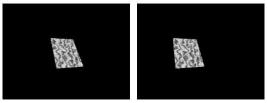
4.4. Evaluation

This section analyzes the direct surface fitting that has been proposed. To this end some proof-of-concept evaluations are performed on artificially rendered scenes, on standard stereo benchmark scenes and on real-world images. In a second part, the direct surface fitting is compared to RANSAC fitting of disparity data and the affine-constrained gradient descent approach.

4.4.1. Proof-Of-Concept

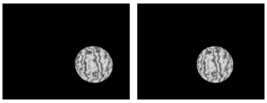
In order to prove the concept of using Hooke-Jeeves for vision-based model fitting and to evaluate the accuracy of the parameter estimation the approach was first applied to some virtual scenes. To this end, camera images were rendered by means of POVRay (`http://www.povray.org/`), a free ray-tracing program. The artificial camera images were rendered according to a standard parallel stereo camera setup with a baseline of 7cm. The objects were placed in a distance of 50cm in front of the virtual stereo camera. Figure 4.5 depicts the rendered images and the results achieved by the proposed approach.

Comparing the ground truth values of the parameters with the estimated parameter values shows that the Hooke-Jeeves optimization is able to estimate the model parameters very precisely. Although the objects cover only image regions of about 100×100 pixels, angles are estimated up to a half degree for the plane and up to two degrees for



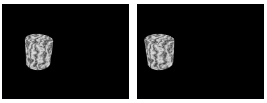
	Measured	Estimated
α_x	$+37°$	$+37.00°$
α_y	$-23°$	$-22.81°$
z_a	500mm	499.70mm

(a) Plane



	Measured	Estimated
x_a	150mm	149.3mm
y_a	-70mm	-69.1mm
z_a	500mm	500.3mm
r	100mm	100.1mm

(b) Sphere



	Measured	Estimated
x_a	-150mm	-150.8mm
y_a	0mm	0.0mm
z_a	500mm	500.2mm
α_x	$-31°$	$-31.6°$
α_z	$-13°$	$-11.1°$
r	70mm	69.6mm

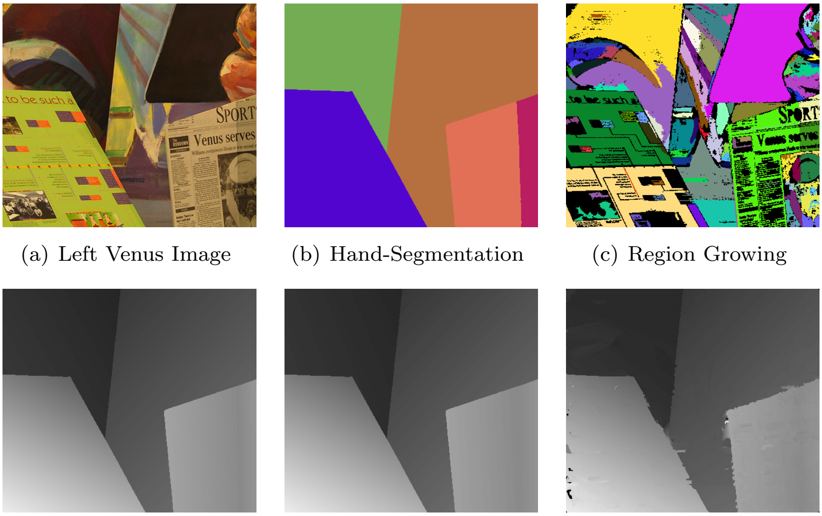
(c) Cylinder

Figure 4.5.: Results of direct surface fitting applied to three different rendered objects (a) Plane (b) Sphere and (c) Cylinder. The images at the top show the left and right camera image of the different objects. The tables below the images show the ground truth parameters of the objects and the parameters estimated with the proposed approach.

the cylinder, and positions and radii up to one mm.

For evaluating the precision of the approach under more realistic conditions, it was applied to the Venus scene from the Middlebury data set [Scharstein and Szeliski, 2003]. This scene consists of five planar surfaces. The five planar segments of this scene (shown in Fig. 4.6(b)) were segmented by hand. For each of these segments the planar parameters were estimated independently. Note that a segmentation is necessary only for left image, as the search process warps the right image into the left image for comparison. Afterwards a disparity map is computed from the estimated parameters. The results are shown in Fig. 4.6(e).

Comparing the ground truth disparity map in Fig. 4.6(d) and the estimated disparity map in Fig. 4.6(e) reveals almost no errors. The percentage of bad pixels (see equation 3.16) with an accuracy of 0.5 pixels is 0.00%, i.e. no erroneous estimations. Please note that the percentage of bad pixels is the common error measure used to compare results on the Middlebury data set which is described in [Scharstein and Szeliski, 2003] as well as in section 3.3.2. However, the image was segmented by hand. A computer calculated segmentation may produce a lot more segments of poorer quality. Hence, it is important to test the performance of model estimation together with a computer generated segmentation. One common assumption in the field of computer vision



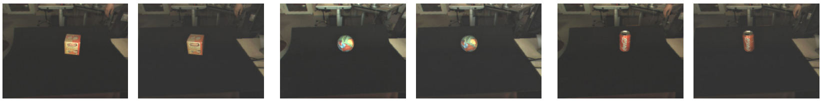
(a) Left Venus Image (b) Hand-Segmentation (c) Region Growing

(d) Ground Truth (e) Result for (b) (f) Result for (c)

Figure 4.6.: Results on the Venus scene from the Middlebury data set. (a) Left camera image, (d) ground truth disparity, (b) segmentation of the image by hand and (c) image segmentation into homogeneous regions using region growing (note that regions smaller than 100 pixels are shown in black). (e) and (f) show the disparity maps produced by direct surface fitting. Here the planar parameter estimation was applied to each segmented region separately. The disparity values were calculated from the estimated parameters.

is that homogeneous regions are likely to be planes. A segmentation into homogeneous regions can be calculated by a simple region growing algorithm. Figure 4.6(c) shows that such a segmentation leads to a large number of regions of different size and quality. Note that regions smaller than 100 pixels are displayed in black. Although this automated pre-processing constitutes quite a challenge, the model fitting based on Hooke-Jeeves is still able to produce a good estimation. The percentage of bad pixels (accuracy 0.5 pixels) is 1.39%. This shows that the proposed algorithm is able to estimate model parameters for imperfect and even very small segments as long as the model assumption holds.

The next two experiments investigate the performance for real stereo camera images acquiring real-world objects under real-world conditions.



	Top Face	Front Face
α_x	$+48.48°$	$-36.92°$
α_y	$+8.44°$	$+15.11°$
z_a	496.52mm	498.76mm

(a) Box

	Measured	Estimated
r	49.30mm	53.98mm
z_a	$\sim$ 525mm	542.19mm

(b) Ball

	Measured	Estimated
r	35.00mm	35.40mm
z_a	$\sim$ 540mm	542.84mm
α_x	—	$-23.64°$
α_z	—	$+0.30°$

(c) Can

Figure 4.7.: Results of direct surface fitting applied to three different real-world objects (a) Box (b) Ball and (c) Can. The images at the top show the left and right camera image of the different objects. The tables below the Ball and the Can show the ground truth radius compared to the estimated radius. For the Box the result for the two visible faces are shown, the estimations show that the angle between them is close to 90°.

Unfortunately, only partial ground truth data is available here. Figure 4.7 shows the stereo images of a Box, a Ball and a Can. Below the images of the Ball and the Can the estimated radius is compared to the radius measured by hand. As can be seen the estimation is quite accurate despite of the fact that the objects are really small in size. Comparing the rotation angle α_x of the front face with that of the top face of the Box shows that the faces differ approximately by 85°. This is very close to the 90° the faces should differ and is a strong indicator that the estimation was correct.

In order to get an impression of how the approach works with imperfect objects and cluttered scenes, a cluttered scene with an Apple, a Bottle and a Tea Box was arranged. Figure 4.8 shows that scene and the estimated disparities of the model fitting approach compared to disparities extracted using a standard block matching stereo approach with SNCC. For better visibility, disparity maps were zoomed and deprived of the background using the object masks. Comparing the resulting disparity maps of the proposed direct surface fitting to block-matching stereo with SNCC shows that direct surface fitting is able to produce reasonable results in cluttered environments. However, due to the underlying model the disparity maps of direct surface fitting are much smoother and free of outliers. Furthermore, this experiment

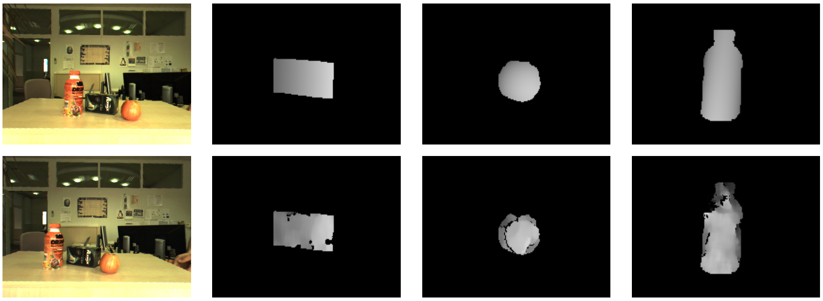

(a) Office scene (b) Tea Box (c) Apple (d) Bottle

Figure 4.8.: Results of direct surface fitting compared to a standard block-matching stereo using SNCC. (a) Top and bottom image show the left and right stereo image, respectively. The gray-level images on the right side show close-ups of the disparities for the three objects (b) Tea-Box, (c) Apple and (d) Bottle. The top row shows the disparity maps generated by direct surface fitting and the bottom row the results of block-matching stereo with SNCC.

shows that direct surface fitting is robust against imperfect model assumptions. Although the Apple and the Bottle do not have the exact shape of a sphere and a cylinder the algorithm is able to find a good fit of the models which approximates the shape of the real-world objects very closely.

4.4.2. Comparison

The experiments so far have proven the feasibility of matching surface models into stereo camera images by means of the Hooke-Jeeves optimization. In contrast, the experiments in this section compare the proposed direct surface fitting approach with two other methods: RANSAC (see section 4.2.1) and affine-constrained gradient descent (see section 4.2.3). Since the affine transformation (as well as the homography) is only able to describe the mapping of planar surfaces, all three approaches are only compared based on a planar model. For the comparison, the Venus, Bull, Sawtooth and Poster scenes (see Fig. 4.9) from the Middlebury stereo evaluation data sets [Scharstein and Szeliski, 2002] are used. These scenes consist solely of planar surfaces which makes them well suited for the analysis of planar fitting. As

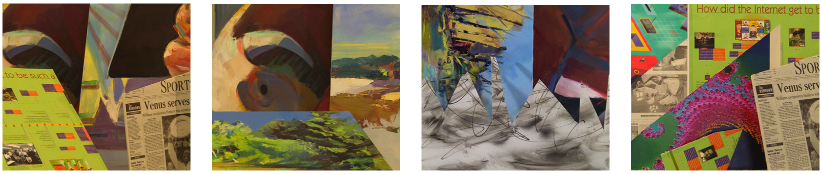
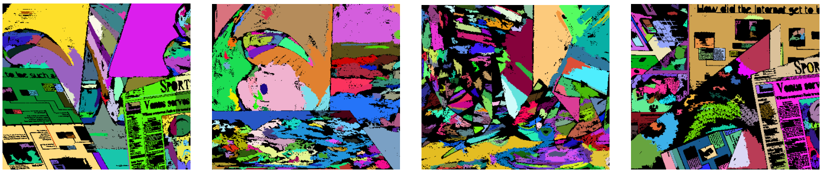


Figure 4.9.: Left images of the Venus, Bull, Sawtooth and Poster scene.



(a) Venus (b) Bull (c) Sawtooth (d) Poster

Figure 4.10.: Segmentation of the four test scenes Venus, Bull, Sawtooth and Poster into homogeneous regions by means of region growing. The regions are illustrated in pseudo-colors. Regions smaller than 100 pixels are discarded and shown in black.

was pointed out earlier one cannot guarantee a perfect segmentation. Hence, again region growing was used to segment the left camera images instead of segmenting the images by hand. From these regions only the ones with more than 100 pixels were selected (see Fig. 4.10). The other regions are ignored in the following evaluations because small regions are prone to unpredictable errors which might distort the results.

In this first setup the three method are assessed with respect to their general depth estimation performance and how well they can cope with partial occlusions. Note that the methods are compared on the basis of the disparity error and not the planar parameter error because this is more expressive for the actual accuracy of the depth estimation.

The second setup consists of stereo camera images taken from within a car while driving. The task for the three plane estimation methods is to estimate the position and orientation of the street. These images are very challenging because they depict a wet street which exhibits many distracting reflections.

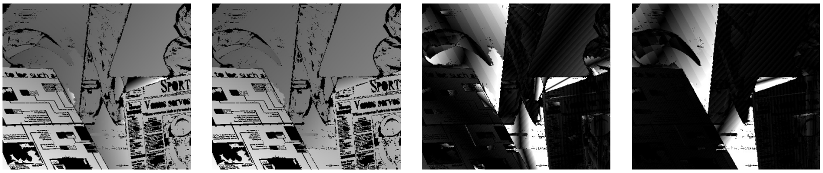
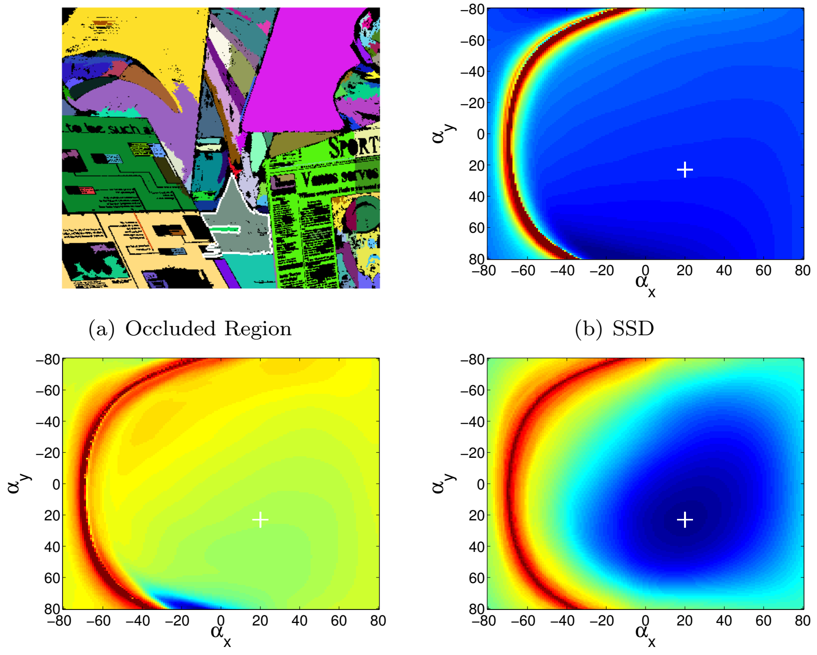


Figure 4.11.: The first two images show the disparity maps of the Venus scene for planar image fitting using (a) affine-constrained gradient descent and (b) direct surface fitting with the SSD cost function. The planar fitting was applied to each region shown in Fig. 4.10(a). For better visualization (c) and (d) show the disparity error maps (pixel-wise absolute difference to ground truth). The error is encoded by intensity, large errors (more than 2 pixels disparity) are shown in white and small errors in black. These results were gained using the plain gray images. They highlight the usual tendency of image fitting methods to fail for partially occluded regions.

General Performance and Occlusions

Already at the beginning of the evaluation it became apparent that approaches which fit models into images are prone to fail for regions that are partially occluded in the other image. For example Fig. 4.11 shows the resulting disparity maps and disparity error maps for the Venus scene for affine-constrained gradient descent and direct surface fitting with SSD. It can be observed that both methods tend to wrongly estimate the orientations of background regions near occluding objects. This effect is most striking for the region above the newspaper (brownish, triangular region in Fig. 4.10(a)).

An analysis of the problematic regions revealed that the main reason for the bad performance are the usually large intensity differences between a background region and its occluder. As parts of an occluded region are only visible in one camera image, a correct warping from one camera image into the view of the other camera image will lead to an overlay of completely different intensity values. This in turn will lead to large errors in the squared error distance of the minimization equations (4.2) and (4.4). It strikes that the characteristics of this problem are very similar to the fattening problem of NCC in standard block-matching stereo which has been discussed in section 3.2.1. Thus, for one of the problematic regions (the one with the white outline in



(a) Occluded Region (b) SSD

(c) NCC (d) SNCC

Figure 4.12.: Matching error landscape for the partially occluded plane with the white outline in (a) using direct surface fitting with cost functions (b) SSD, (c) NCC, (d) SNCC. The plots were generated by fixing the correct depth of the plane and varying the angles α_x and α_y in one degree steps from $-80°$ to $+80°$. The white cross marks the correct angles of the planar surface. The matching error is encoded by color, large errors are dark red and small errors are dark blue.

Fig. 4.12(a)), direct surface fitting was executed with the three cost functions SSD, NCC and SNCC. While the fitting failed for SSD and NCC it succeeded for SNCC.

Figure 4.12 shows 2-D plots of the matching error landscape for fitting this planar region with direct surface fitting using SSD, NCC and SNCC. For generating these plots the correct depth of the plane was given and fixed, and the angular orientations of the plane α_x and α_y were sampled in one degree steps for $-80°$ to $+80°$. It can be observed that the matching error landscape of SSD and NCC do not have their minimum at the correct place (white cross). In contrast, the matching error landscape of SNCC is able to capture the 3-D structure of the plane in a smooth transition to a correct minimum. Unfortunately, SNCC can not be used for gradient-based approaches because it is not straightforward to calculate the necessary derivatives. However, it turns out that for larger image regions it is possible to approximate the SNCC cost function.

As was discussed above the main problem of incorrect matching values for partially occluded surface regions is the fact that when using the correct warping, pixels from the occluded region are compared to pixels of the occluding region. Since these two regions can differ dramatically in pixel intensity this leads to a large matching error even for a small overlap of the two regions. This does not happen when using SNCC because its local NCC calculation in the first stage with a very small filter size removes the intensity differences. From this, one can conclude that normalizing the gray images by locally removing mean and variance should reduce the problem of intensity difference to a level were it does not disturb the surface matching. In particular, the normalization of the images is done by

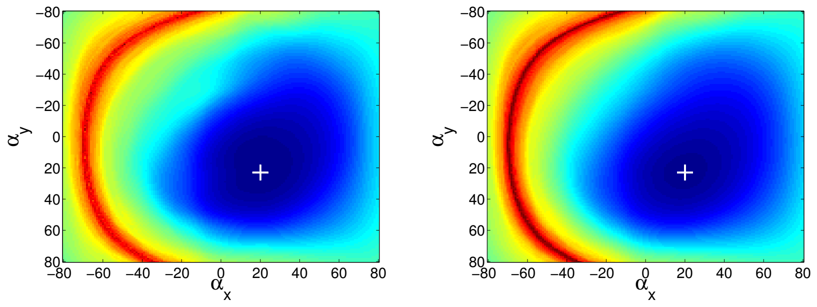
$$I_x^{\text{norm}} = \frac{I_x - \mu_x}{\sigma_x} \ , \tag{4.33}$$

with

$$\mu_x = \frac{1}{|N(x)|} \sum_{x' \in N(x)} I_{x'} \ , \quad \sigma_x = \sqrt{\frac{1}{|N(x)|} \sum_{x' \in N(x)} (I_{x'} - \mu_x)^2} \ , \tag{4.34}$$

where $N(x)$ is the local neighborhood of pixel x. The normalization can be computed efficiently using box filters for calculating local mean and standard deviation. Similar to the observations in section 3.2.1, the smaller the neighborhood used for normalization the more the occlusion effect is reduced. On the other hand there does not seem to be a large drawback of using small neighborhood sizes except for camera noise. In the following experiments a neighborhood of 3x3 pixels is used for Middlebury scenes and 5x5 pixels for real-world car scenes.

In order to validate the argumentation above, Fig. 4.13 shows the



(a) Normalized SSD (b) SNCC

Figure 4.13.: Side-by-side view of the matching error landscape of direct surface fitting with SSD on normalized gray images and SNCC on the plain gray images. This demonstrates that using normalized gray images together with the SSD measure is an approximation to the SNCC measure. The plots were generated in exactly the same way as the plots in Fig. 4.12.

matching error landscape of SSD on normalized gray images in a side-by-side view with the matching error landscape of SNCC for the same surface region as in Fig. 4.12. It can clearly be observed that using normalized gray images together with the SSD cost function leads to a matching error landscape that is very similar to the one of SNCC on the plain gray images. Indeed the minimum of the normalized SSD is not as nicely peaked at the correct position as for SNCC but the landscape has the same smooth transition to the minimum. Hence, in the following experiments both affine-constrained gradient descent and direct surface fitting are applied to normalized gray images using the cost function SSD. However, in real applications of direct surface fitting it is better to use SNCC because of the more nicely peaked error minima.

This brings us back to the experiment (Fig. 4.11) from the beginning of this section. The whole experiment was redone but this time using the introduced image normalization technique. The resulting disparity and disparity error maps are shown in Fig. 4.14. Comparing these error maps of the affine-constrained gradient descent and direct surface fitting to the ones without image normalization (Fig. 4.11(c) and 4.11(d)) shows a clear improvement. Only a few minor wrong estimations re-

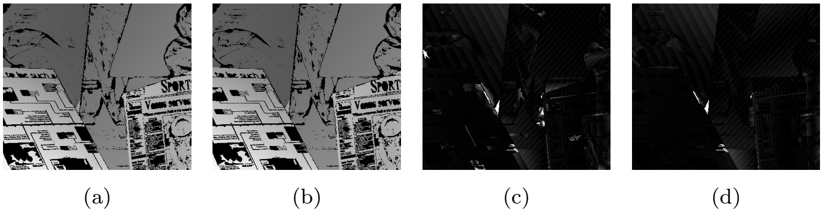


Figure 4.14.: Similarly to Fig. 4.11 the disparity and disparity error maps for (a), (c) affine-constrained gradient descent and (b), (d) direct surface fitting are shown. In contrast to Fig. 4.11, mean and variance normalized images were used for matching. The normalization was done using a filter size of 3x3. It demonstrates that image normalization (with very small filters) is able to dramatically reduce the occlusion problem for image fitting methods.

main and these are in most cases due to an imprecise segmentation. Hence, one can conclude that image mean and variance normalization dramatically reduces the occlusion problem for image fitting methods because it is an approximation of the SNCC cost function.

It is important to stress here, that the improvement cannot be explained by a reduced difference of corresponding pixels but only by a reduced difference of non-corresponding pixels (occluded pixel matched with an occluding pixel). There are three main indicators for this. Firstly, the Venus scene was taken under ideal conditions, i.e. corresponding pixels have almost the same intensity. Secondly, the improvement vanishes if larger neighborhoods are used for mean and variance calculation. Last but not least, wrongly estimated planar fittings would not populate around depth discontinuities but rather distribute equally all over the image. Hence, the actual reason for the improvement can only be that the normalization makes pixels within one image more equal which reduces the difference between non-correlating pixels. Thus, the influence of strong contrasting structures, like the strong contrast edge between the bright newspaper and the dark background, is reduced.

Now that the occlusion problem is resolved a reasonable comparison of the image fitting methods to disparity fitting based on RANSAC is feasible. Table 4.1 compares the quality of direct surface fitting, affine-constrained gradient descent and RANSAC for the four Middlebury test scenes (Fig. 4.9). Like in section 3.3.3 the quality is assessed by

Table 4.1.: Quality of the three plane fitting methods RANSAC, affine-constrained gradient descent (ACGD) and direct surface fitting (DSF). The quality is measured by the percentage of bad pixels, i.e. pixels whose disparity differs more than 0.5 from ground truth.

scene	RANSAC	ACGD	DSF
Venus	1.42%	1.65%	**0.37**%
Bull	**1.37**%	1.68%	1.68%
Sawtooth	8.24%	7.55%	**7.05**%
Poster	**5.20**%	8.49%	5.44%

means of the percentage of bad pixels [Scharstein and Szeliski, 2002] for an error threshold of 0.5. The results show that the three approaches have a similar quality but that the affine-constrained gradient descent method is slightly worse than the other two approaches. In particular, it seems to struggle with the Poster scene. One reason for this might be the occlusions which are more frequent in this scene as compared to the other three. Nevertheless, all three approaches show a comparably good performance in these well-natured conditions. For a more application relevant evaluation, the next section compares the three approaches with respect to their robustness in difficult real-world scenes.

Robustness against Heavy Distortions

The general claim of image fitting approaches is that by integrating over larger image areas for fitting, the robustness is improved and the aperture problem reduced. In order to test this hypothesis, a short street stream (320 frames) has been recorded with a stereo camera system mounted on a car. The stream was recorded in a rainy weather condition. The goal for all three fitting methods is to estimate the planar parameters of the street in front of the car.

Figure 4.15(a) shows exemplarily one frame of this rain stream. As can be seen in Fig. 4.15(b) the traditional correlation-based stereo processing struggles to find correct correspondences. The reason for this is the superposition of the street structure with the image of the surrounding because of the mirror-like state of the street. Due to the bad quality of the correlation-based stereo, the RANSAC fitting is prone to

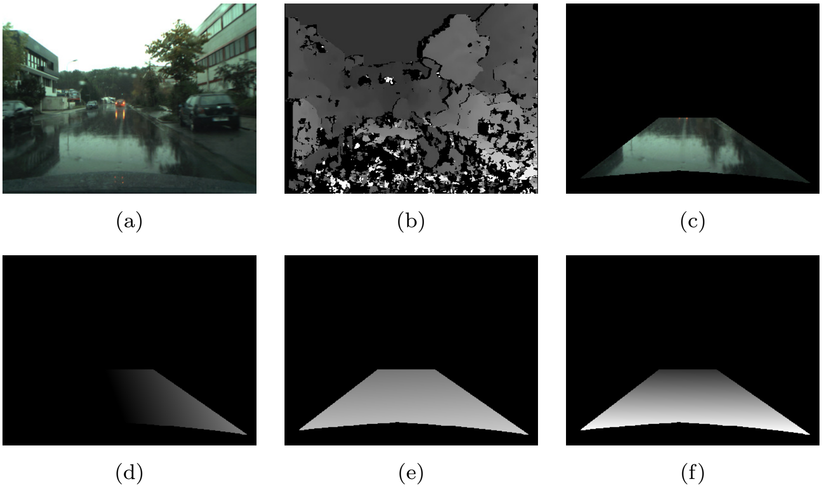
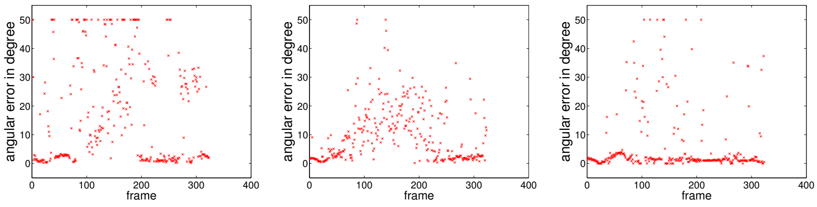


Figure 4.15.: (a) Example of a difficult scene showing a wet street with a mirror-like state. (b) The disparity map of block-matching stereo. It is obvious that the block-matching stereo fails for a large part of the street. (c) Image region for which the planar fitting is done. Resulting disparity maps: (d) RANSAC, (e) affine-constrained gradient descent and (f) direct surface fitting.

fail to find a reasonable estimation of the street. Figure 4.15(d) shows that the fitted plane is very skewed. In contrast to this the result of the affine-constrained gradient descent (see Fig. 4.15(e)) and direct surface fitting (see Fig. 4.15(f)) are much better. However, in this frame the affine-constrained gradient descent was not able to find the correct inclination of the street.

All three methods were applied to the 320 frames (32 seconds) of the rain stream. In order to assess the robustness of the plane fitting under these challenging conditions, the so-called pitch angle is evaluated. The pitch angle describes the relative orientation between the car and the street. It changes when the car accelerates, decelerates or due to small irregularities on the street's surface. Since the pitch angle changes over time it is an important information for intelligent vehicle systems to guide image processing. Here, no ground truth information of the pitch angle is available, but it is know that it closely varies around 90° because the angle between camera and street is roughly 90° when the



(a) RANSAC (b) Gradient Descent (c) Direct Surface Fitting

Figure 4.16.: These plots show the angular error for estimating the pitch angle in a video stream recorded with a stereo camera mounted in a car. The stream was recorded in a rainy weather condition (see Fig. 4.15) which leads to strong image distortions. While RANSAC and the affine-constrained gradient descent fail for a large part of the stream, direct surface fitting is able to produce good results for the whole stream.

car is standing still. Furthermore, when the scene was recorded there was hardly any acceleration and the street surface was flat. By means of this knowledge, an angular error can be calculated that should be close to zero but is allowed to have small deviations from this point as long as it is a smooth change. Figure 4.16 shows the angular error of the pitch angle for all three methods. For a better contrast, angular errors larger than 50° are clipped.

The first thing that strikes is that RANSAC and the affine-constrained gradient descent fail to estimate the pitch angle between frame 80 and 200. This is the most difficult part of the stream with heavy reflections as seen in Fig. 4.15(a). In contrast the estimation with direct surface fitting produces good results with only a few outliers for the whole stream. For the frames before 80 and after 200 the three approaches have a similar performance. This again shows that in principle all three approaches have a similar performance but under difficult conditions the direct surface fitting is more robust than RANSAC or gradient descent estimation. However, the results have to be seen with some caution. The performance of the affine-constrained gradient descent and direct surface fitting depend on the initialization. In order to be fair with respect to RANSAC the parameter search for gradient descent and direct surface fitting was initialized at 70°, i.e. with an initial error of approximately 20°. This is in general close enough for gradient descent but not so close that the estimation becomes too easy with respect to the RANSAC estimation. One can argue that RANSAC is in disadvantage here because the other two methods started from a too good initialization. On the other hand a processing on image streams allows for a temporal integration or a tracking of the planar parameters over time [Corso et al., 2003] which strongly reduces the problem of finding a good initialization. For example it is very easy to just use the parameters from the last frame as an initialization for the next frame. This would result in a much closer initialization which would lead to even better results for the affine-constrained gradient descent and direct surface fitting.

4.5. Sparse Direct Surface Fitting

Unfortunately, direct surface fitting can be become computationally expensive for large image regions. The main cost comes from warping the right image, which has a runtime that is linear in the number of pixels that are warped, i.e. doubling the image or region size leads to double the time for warping. For large regions this cost will dominate all other processing costs. Therefore, it is necessary to consider some improvements for real-time processing. First of all, it is quite obvious that multiple planes can easily be processed in parallel (e.g. multi-threaded) because their estimations do not depend on each other. Thus, it is easy to get a linear gain in runtime for multiple threads when dealing with multiple planes. However, in some applications like street surface estimation the regions can become so large that the processing time for a single region can already take way more than 100ms. Unfortunately, improving the runtime for a single estimation is not straightforward. One could think of using multiple threads for the sampling steps in the Hooke-Jeeves optimization but the gain would be rather low because of the numerous synchronizations that would be necessary. So another solution is required.

One strong advantage of direct surface fitting compared to block-matching stereo is the increased robustness which results from the larger image regions that are used for estimation. But this improvement saturates at a certain region size, i.e. at a certain point there is only a negligible increase in robustness when increasing the size of an image region. The reason for this is that for some region size the con-

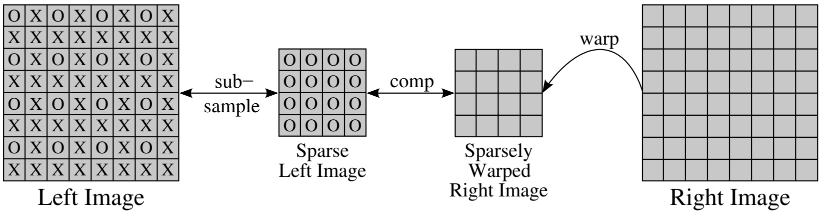


Figure 4.17.: Sparse direct surface fitting. Instead of using the full region in the left image and performing in accordance a backward-warping from the right image for the whole region in the left image, only a subset of the pixels are used. The left image is subsampled and only for the subsampled pixels in the left image a backward-warping from the right image is performed.

tained information is enough for a highly robust estimation so that a further increase of the region size gives only very little additional information for the estimation problem. This means that large regions can be processed in a sparse fashion without loosing robustness or accuracy. Usually, within a sample step of Hooke-Jeeves for the whole region in the left image a backward-warping from the right image is done (see Fig. 4.4). Then the warped region is compared to the original left region by calculating a fitness value according to the cost measure. Now, instead of conducting a backward-warping for the full region the warping is only done for some sparsely selected image points.

One approach that efficiently uses the characteristics of a computers memory is to equally sample every n^{th} pixel in x- and y-direction. In this work, taking every n^{th} pixel will be referred to as a fitting with sparseness n, e.g. a sparseness of 4 means taking every 4^{th} pixel in x- and y-direction which means that only 1/16 of all region pixels are warped. A schematic plot of this sparse warping is shown in Fig. 4.17. First, the left image is subsampled (not resized!) according to the sparseness factor thereby memorizing the original pixel positions. These original positions are used to conduct the backward-warping from the right image which results in a sparsely warped image. Then the sparsely warped right image and the subsampled left image are compared according to the cost function. Since only a small subset of the full region is warped, the processing time of the warping is dramatically reduced.

In Table 4.2 an analysis of this sparse direct surface fitting is shown

Table 4.2.: Comparison of matching quality and speed for direct surface fitting and sparse direct surface fitting. For this analysis the Venus scene was used together with the segmentation in Fig. 4.10(a) but only regions larger than 1000 pixels where used. These are 24 regions which cover about 67% of the image. A sparseness of n refers to taking only every n^{th} pixel in x- and y-direction for performing the surface fitting.

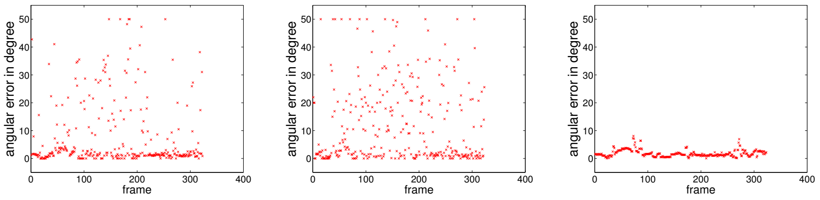
sparseness	1	2	3	4	5	6	7	8
bad pixels (%)	0.12	0.12	0.30	2.50	2.98	6.40	7.22	9.28
runtime (ms)	1296	564	369	293	287	271	219	229
speedup	1	2.3	3.5	4.4	4.5	4.8	5.9	5.7

Table 4.3.: Matching quality and speed analysis of the sparse direct surface fitting for the large, dark green region to the left of Fig. 4.10(a) (Venus benchmark scene). The accuracy of the estimation stays very high for a wide range of sparseness factors because there is no measurable amount of bad pixels for a threshold of 0.5. The runtime saturates at 10ms.

sparseness	1	2	3	4	5	6	7	8
bad pixels (%)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
#pixels	13684	3421	1520	855	547	380	279	214
runtime (ms)	109.0	32.6	18.4	17.7	10.6	9.3	9.6	10.2
speedup	1	3.3	5.9	6.2	10.3	11.7	11.4	10.7

for an Intel Core i5-650. The results show that up to a sparseness of 3 there is no large penalization in matching performance while having a speedup of factor 3.5. From sparseness 4 on, some partly occluded regions are estimated wrongly. But one should also consider that most regions are 1000-2000 pixels large, i.e. for a sparseness of 4 they have already decreased to 63-125 pixels. It strikes that the speedup starts to saturate for a sparseness of 5-6. The reason is that the sparse processing mainly accelerates the warping and the matching cost computation. At one point the runtime of the remaining code will dominate the overall runtime leading to the observed saturation (a phenomenon also known as Amdahl's law [Amdahl, 1967]). Hence, it is important to select the sparseness with respect to the region size because larger regions benefit stronger than smaller regions.

In Table 4.3 the fitting performance of the large, dark green region to the left of Fig. 4.10(a) is shown. This regions has a size of 13684 pixels. The results highlight that for large image regions the gain in speed can



(a) Sparseness 2 (b) Sparseness 4 (c) Sparseness 4 TI

Figure 4.18.: Sparse direct surface fitting on the challenging rain stream shown in Fig. 4.15. The plots are generated like the evaluation for the non-sparse fitting in Fig. 4.16(c). While fitting with sparseness 2 has a performance similar to the non-sparse fitting, a sparseness of 4 results in a high noise level. On the other hand, applying just a simple temporal integration (TI) of using a frame's estimated parameter as a starting value for the next frame, dramatically reduces the noise and increases the performance.

easily be a factor of 10 without having a relevant loss in performance. It also strikes here that the speedup saturates for a sparseness of 5-6. A general observation is that for planar fitting the runtime saturates at about 10ms. This can also be seen in Table 4.2 where the overall runtime for the 24 regions saturates at about 220ms. Thus, one can conclude that the non-sparse (non-pixel dependent) parts of the sparse direct surface fitting algorithm have a runtime of about 10ms.

For the real-world application of street surface estimation on 400x300 images this means that the runtime of direct surface fitting with SNCC for a region size of 10000 pixels reduces from 170ms to 9.3ms for a sparseness of 6, a speedup of factor 18. However, difficult conditions might require a less sparse fitting. For example Fig. 4.18 shows the performance of direct surface fitting with SNCC on the rain test set from Fig. 4.15 for a sparseness of 2 and 4. It can be seen that fitting with sparseness 2 is close to the performance of the non-sparse fitting (Fig. 4.16(c)). In contrast, using a sparseness of 4 increases the noise level considerably. On the other hand these estimations are without any temporal integration, i.e. the surface estimation is run for each frame separately. Figure 4.18(c) demonstrates that already a simple reusage of a frame's parameters as a starting point for the next frame dramatically reduces the noise and increases the estimation performance. This shows that sparse direct surface fitting can be applied in real-world scenarios with super real-time speed. Furthermore, it is reasonable to assume that using more elaborated temporal integration like Kalman filtering [Kalman, 1960] can further improve the estimation performance which allows for even less sparse fitting in challenging situations.

4.6. Summary

This chapter introduced *direct surface fitting*, a novel technique for stereoscopic depth estimations which can be used to complement the results of block-matching stereo in difficult situations. It is inspired by the homography-constrained gradient descent on stereo images which is frequently used to estimate the orientation and depth of planar surfaces. Since this allows for a matching of large image patches the aperture problem is reduced which leads to a higher robustness. Unfortunately, a homography describes only planar surfaces and the gradient-descent-based optimization is limited to a simple squared error distance. In contrast direct surface fitting can fit different parametric surface models and allows for arbitrary matching cost functions. This becomes possible because the traditional gradient descent optimization is replaced with the direct search method of Hooke-Jeeves [Hooke and Jeeves, 1961].

A comparison of direct surface fitting, the affine-constrained (homography-constrained) gradient descent and the RANSAC estimation for planar surfaces showed that the three methods are comparable in depth accuracy. However, when it comes to scenes with heavy distortions direct surface fitting is much more robust than the other two methods. RANSAC suffers from the fact that its estimation is based on disparity maps and thus inherits the problems of the underlying stereo method. The affine-constrained gradient descent gets stuck in local minima very easily because the gradients have to be approximated linearly. In contrast, direct surface fitting is based on sampling in a simulated-annealing-like manner which leads to a good robustness against local minima.

One drawback of the underlying Hooke-Jeeves optimization is the large number of iterations compared to gradient descent. However, it has been shown here that the overall runtime for estimating planar parameters is comparable to state-of-the-art gradient-based approaches.

The main reason is that gradient-based approaches need to calculate local image gradients in each iteration. As Hooke-Jeeves does not need these gradients an iteration is much faster. Still the estimation of a large region can take half a second in reasonable scenarios. To solve this problem a sparse variant of direct surface fitting has been proposed. The main idea is to use only a subset of the pixels of a large region for surface fitting. It has been demonstrated in this work that this can bring down the estimation time from hundreds of milliseconds to about 10ms without a major loss in depth accuracy. This makes the proposed method very suitable for real-time applications.

5. Application to Vision Systems

This chapter presents two vision systems that employ the *summed normalized cross-correlation* (SNCC) introduced in chapter 3 and *direct surface fitting*, a model-based depth estimation by means of Hooke-Jeeves, introduced in chapter 4.

The first system [Schmüdderich et al., 2010] is a car detection system which works offline on data streams recorded from a vehicle equipped with a stereo camera. For depth processing this system uses block-matching stereo with SNCC for a fast and general depth estimation and direct surface fitting for robustly locating the street in 3-D. The whole system is evaluated for a variety of conditions like normal weather, rain, snow and night. The steady performance demonstrates the robustness for real-world conditions of the techniques introduced in this thesis.

The second system [Einecke et al., 2011] is a robotic vision system that is working on a humanoid robot for real-time grasping of objects in unknown environments. This system also uses block-matching stereo with SNCC for a general depth map generation. Direct surface fitting is used to complement the local estimates from the block-matching stereo for difficult scene parts like homogeneous table tops. The swift and reliable grasping of the integrated robotic system demonstrates the real-time processing capabilities of the techniques introduced in this thesis.

5.1. Car Detection System

In the recent years, the research on camera-based driver assistance systems has intensified. The main reason for this is that the usually used RADAR and LIDAR sensors are good for estimating the distance to leading vehicles but bad for scene analysis. In contrast, cameras provide a much richer flow of information. This section discusses a camera-based car detection system [Schmüdderich et al., 2010] that includes the SNCC-based block-matching introduced in chapter 3 and the model-based direct surface fitting introduced in chapter 4.

5.1.1. System Target

The system discussed in the following sections relates to the area of intelligent vehicles, in particular to platforms that are equipped with a stereo camera. The target of the presented system is to detect all cars within each frame of an image sequence (video), i.e. the system is not meant to actively intervene in the control of the vehicle. Rather, the car detection is meant to act as a support module for driver assistance systems like collision warning or collision mitigation.

As cars are used in all seasons, it is important that the car detection works in different lighting and weather conditions. That poses a major challenge because it is difficult to make camera-based processing robust against variable conditions. In related work camera-based systems are often only demonstrated under well-defined conditions, e.g. only at day-time or normal weather conditions. In contrast, the system presented here is targeted to work in many different weather and lighting conditions. Furthermore, the system should work in different kinds of street scenarios like inner-city and highway. This means that the system can not use default assumptions like a certain number of lanes or the maximal speed of the vehicles.

5.1.2. System Structure

Figure 5.1 shows a reduced sketch of the system's architecture with a focus on the modules that were implemented in the course of this doctoral thesis and modules that use the results of these implemented modules as an input. A more detailed overview about the other system elements is given in [Schmüdderich et al., 2010].

The main source of information for the car detection system is the stereo camera of the ego-vehicle. The 3-D information calculated from the stereo images is used on the one hand to improve a car classifier by rejecting spatially implausible detections. On the other hand the 3-D

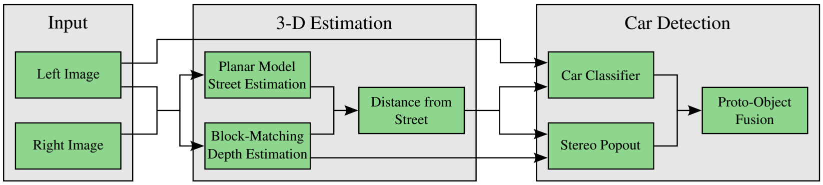


Figure 5.1.: Architecture of the car detection system. The stereo images are used to calculate an SNCC block-matching depth map and a separate estimation of the position and orientation of the street by direct surface fitting. This information is used by two different cues to detect cars: a car classifier and a generic obstacle detector based on stereo. Eventually, the detections of both cues are fused.

data is used for a depth map based obstacle detection referred to as stereo popout.

For representing the object hypothesis from the car classifier and the stereo popout the system uses the concept of proto-objects [Schmüdderich et al., 2008]. A proto-object can be regarded as a pointer to an object in the environment. As long as this pointer remains valid, all visual information of the object can be obtained by dereferencing this pointer and extracting the information from the image. The advantage of proto-objects is that they do not necessarily need the object to be classified. This allows for an easy fusion of the detection results of the stereo popout with the detections of the car classifier. In order to temporally stabilize the proto-objects they are tracked within the proto-object fusion module.

In Fig. 5.2(a) and 5.2(b) the proto-objects detected by the stereo popout and the car classifier are shown for a typical city night scene. These detections are fused into spatially and temporally coherent proto-objects as shown in Fig. 5.2(c). Stabilized detections are visualized via solid quadrilaterals, i.e. those detections are temporally coherent. The color of a quadrilateral indicates the tracking ID. Eventually, these detections can be passed to higher processing stages like a time-to-contact analysis or a scene interpretation for collision warnings.

Depth Estimation

As explained above the car platform used for recording is equipped with a stereo camera. From the images of this camera a depth map is

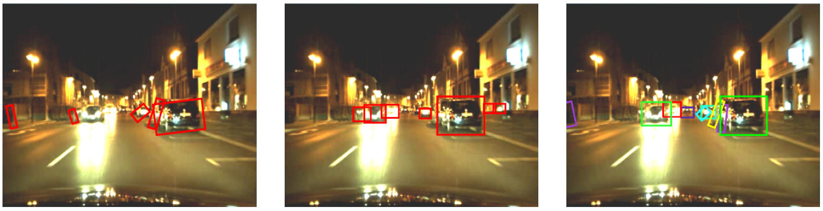

(a) Stereo Popout (b) Car Classifier (c) Proto-Object Fusion

Figure 5.2.: Results of the different sub-modules in the car detection module. Red boxes in (a) and (b) represent detections by the stereo popout and the car classifier, respectively. Quadrilaterals in (c) visualize tracked and fused detections where the colors indicate the tracking ID.

computed using the block-matching stereo with SNCC introduced in chapter 3. The depth map is used to locate the detected objects in the 3-D world and by that relative to the own vehicle.

A second module incorporates the direct surface fitting algorithm, introduced in chapter 4, to estimate the position and orientation of the street relative to the car. Here a planar model is used to approximate the street surface. Of course non-linear models like splines in principle approximate the street surface much more accurately than a planar model. However, cases in which the difference in approximation is significant are limited and non-linear models are usually less stable and more computationally expensive. This means that a little bit of accuracy is traded for a higher robustness and speed.

As explained in chapter 4, direct surface fitting needs a rough image mask of the surface to estimate. In this car detection system the mask is provided by an adaptive street area segmentation [Michalke et al., 2009] that is based on color, structure and depth cues. Each feature provides a binary region map and a pixel-based probability map of the street. These maps are combined into one probability map for the street which is transformed into a street mask by thresholding. Further robustness is achieved by a temporal integration of the segmentation results. One major advantage compared to other street segmentation algorithms is that all the parameters are self-adapted online, which makes the approach robust against changes of the scene characteristics. Fig. 5.3 shows an example of a street scene, the extracted street mask,

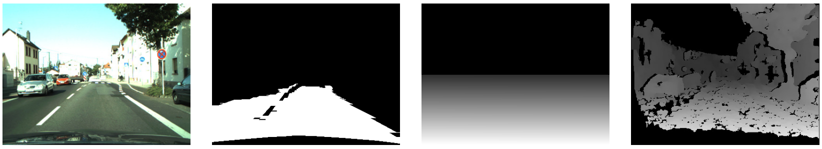


Figure 5.3.: The depth processing on the car is twofold. For the left stereo image (a) a street mask (b) is extracted via segmentation. This mask is used by direct surface fitting to locate the street plane in 3-D (c). Furthermore, the stereo images are used to calculate a depth map of the whole scene (d) via SNCC block-matching.

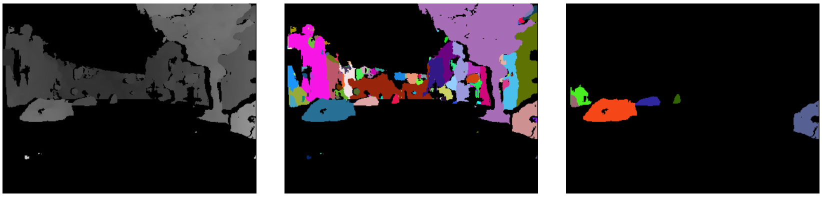
the estimated street location as a depth map and the depth map of the whole scene calculated by means of SNCC block-matching.

After the parameters of the street surface have been estimated they are combined with the scene depth map to calculate a height map. This height map holds the 3-D distance from the street's planar surface for each pixel. To calculate this height distance each pixel is transformed into its 3-D representation using the scene depth map. Then for each of the 3-D points that result from this transformation the distance to the planar surface of the street is calculated. For doing so the standard mathematical plane-point-distance equations are used, i.e. the distance is measured vertical to the planar surface. Eventually, the distance for each 3-D point is stored at its corresponding pixel position in the height map. This map is a very valuable information that allows for an easy rejection of improbable detections of the car classifier. Especially false detections within houses or trees are reliably suppressed. Furthermore, the height map is used for a generic obstacle detection called stereo popout.

Stereo Popout

The stereo popout is a cue that does not require specific knowledge about an object but identifies regions of contiguous depth that stand out from their surrounding. The approach consists of three steps:

First, the block-matching depth map is pruned of all pixels that belong to the ground (or street) as shown in Fig. 5.4(a) for the example from Fig. 5.3. The pruning is necessary as otherwise a depth-based segmentation is likely to fail because the ground connects different ob-



(a) Pruned Depth (b) Popout (c) Sieved

Figure 5.4.: Stereo popout detection of obstacles. Subtracting the street plane (Fig. 5.3(c)) from the depth map (Fig. 5.3(d)) results in the pruned depth map (a). A subsequent region growing leads to multiple object hypotheses (b). These are reduced by sieving out plausible hypotheses that are connected to the ground and have a proper size (c).

jects. In a second step, the pruned depth map is segmented using a seedless region growing. This results in a multitude of regions that are hypotheses of objects (see Fig. 5.4(b)). Third, all hypotheses that are not connected to the ground are removed (see Fig. 5.4(c)). The reason for this is that only objects on the ground are potential obstacles. Furthermore, also miss-estimations due to the aperture problem are reduced.

At this point of processing the stereo popout would be a too generic cue for the task of car detection. It would detect pedestrians, motorbikes, cyclists and cars but also speed bumps, delineators, trees and houses alike. Even though, all of these are important for crash detection the task of the current system is to concentrate on cars. Hence, the object hypotheses are further sieved by means of two height thresholds. On the one hand, the objects are required to be larger than 1m which removes small obstacles like speed bumps or delineators. On the other hand, the objects are required to be smaller than 2.6m which removes houses and trees as well as trucks and buses.

An advantage of the stereo popout processing is that it makes only a few assumptions of the scene. Thus it can be applied to a large variety of scenes. For example, in [Burschka et al., 2002] a very similar approach was successfully applied to detect obstacles in indoor environments for a subsequent path planning of a mobile robot.

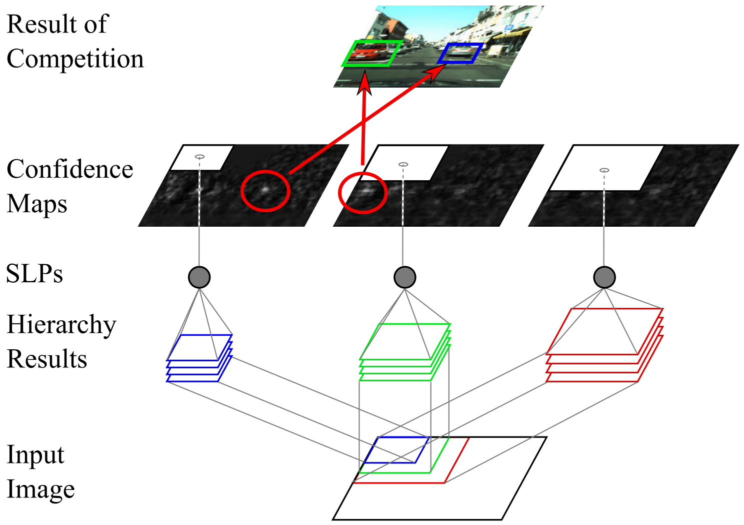


Figure 5.5.: Car detection by the appearance-based classifier. A visual hierarchy is computed for different scales. Then for each scale and each image-point an SLP classifies the area around the image point as containing a car or not. This results in a confidence-map for each scale. A local competition selects the maxima within these maps.

Appearance-Based Classifier

Please note that the classifier discussed in this section was not implemented in the course of this doctoral thesis. Nevertheless, it is explained here to give a clearer picture of the improvement of the classifier that is gained by combining it with the estimated depth maps.

The appearance-based classifier generates object hypotheses in three successive steps, which are visualized in Figure 5.5. First the output of a hierarchical feed-forward architecture as proposed in [Wersing and Körner, 2003] is computed at multiple scales, resulting in a set of feature maps for each scale and each image-point. Second, for each scale a Single Layer Perceptron (SLP) receives the feature maps around one image point and computes a confidence for this point depicting the center of a car, i.e. if the image-patch around an image point is likely to approximate the boundaries of a car, the confidence value for this point is high, otherwise it is low.

In a last step the confidence maps are fed into a competitive selection method that generates a given number of object hypotheses. For this, the local maxima across all scales are detected. Each maximum is directly associated with a Region of Interest (ROI) in the image, where the center of the ROI lies on the maximum and the size of the ROI depends on the scale associated to the corresponding map. The selection works in a greedy fashion. At first, the maximum with the highest confidence is chosen and used to create the proto-object $P_{C,1}$. With the corresponding ROI the confidence values in all other maps are suppressed. The remaining maxima are then processed in descending order to create the proto-objects $P_{C,2}$ to $P_{C,N}$. In the course of this descending processing, all maxima whose ROI are covered by more than 75% of the already inhibited area are rejected. This process stops when a maximal number of hypotheses is reached or no further maxima remain.

To control the number and quality of detections the confidence maps are thresholded by a value Θ_C. With the choice of the Θ_C a trade-off between false detections and missed cars is made. This trade-off can be significantly improved by incorporating additional scene knowledge. This is were the height map and the scene depth map come into play. These evaluate the classifier detections by checking the height of a detection above the ground-plane and by relating the image size of the detection with the 3-D size calculated from the depth map. All hypotheses with implausible heights or 3-D sizes are ignored during the local maxima search process.

For training the SLP, segments containing cars and segments containing non-cars are cropped from the training scenes and normalized in size. The SLP learns to generate high values for positive examples and low values for negative ones. The result is a so-called view-tuned-unit [Wersing and Körner, 2003] which responds robustly to car views of different viewing angle, delivering competitive performance in benchmarks like the UINC car detection [Wersing et al., 2008].

5.1.3. Related Work

The area of intelligent vehicle research has grown a lot in the recent years. The number of systems build is very large and can not be discussed exhaustively in this thesis. Hence, this section will mainly concentrate on the related work with respect to the 3-D depth estimation.

Ground Plane Estimation: In the domain of intelligent vehicles there are three major approaches for the street surface estimation. The most simple approach is the so-called v-disparity [Labayrade et al., 2002, Suganuma, 2009]. By summing up the disparities in a disparity-histogram for each image line, the disparity map is transformed into the v-disparity space. Here the street can easily be extracted via a line detection algorithm. Although this approach is very fast and simple, it lacks robustness because all the depth data is used without taking care of whether pixels belong to the street or not. Thus all non-street pixels will perturb the estimation. Another approach is to perform a gradient-descent on the homography transformation [Seki and Okutomi, 2006, Guo et al., 2009] between the stereo images. As has been shown in chapter 4, the homography-constrained gradient descent is very accurate but lacks the robustness of direct surface fitting. Yet another kind of approaches use non-linear street models like quadratic functions [Oniga et al., 2007] or splines [Wedel et al., 2008]. Although these are used regularly they have not yet been shown to work robustly in difficult weather conditions.

Dense Stereo: It is hard to tell which particular algorithms are generally used for dense stereo processing in intelligent vehicles because many authors do not explicitly state what stereo method they use. However, a lot of vision systems rely on local block-matching methods [Oniga et al., 2007, Michalke et al., 2009, Rodríguez et al., 2010, Perrollaz et al., 2010] mainly because they require real-time speed. Additionally in the recent years, also the stereo calculation by means of semi-global matching [Hirschmüller, 2005] has got an increasing attention but requires hardware-specific implementations on FPGA's [Gehrig et al., 2009] or GPU's [Haller et al., 2010] for real-time processing.

Stereo Popout: Most vision systems that detect traffic participants solely based on stereoscopic vision employ variants of the v-disparity space to cluster objects therein [Hu et al., 2005, Suganuma, 2009, Kormann et al., 2010]. Others transfer the depth image into an equally rasterized birds-eye view [Seki and Okutomi, 2006, Oniga et al., 2007] to segment objects like in an occupancy grid. Only very few systems apply a segmentation directly to the depth map [Kormann et al., 2010], like the stereo popout discussed in this work. The clustering of stereoscopic depth data is often preceded by a selection of a subset of pixels [Seki and Okutomi, 2006, Oniga et al., 2007, Suganuma, 2009] by means of the height of the pixels with respect to the street. In doing so, the street pixels are removed which increases the robustness of the depth segmentation. Unfortunately, all discussed publications except [Seki and Okutomi, 2006] give only qualitative results on the detected objects by showing some example images. A thorough, qualitative analysis as will be done in the next section is completely missing which limits a sound comparison. Furthermore, none of the discussed publications give information about the performance with respect to more difficult conditions like night, rain or snow.

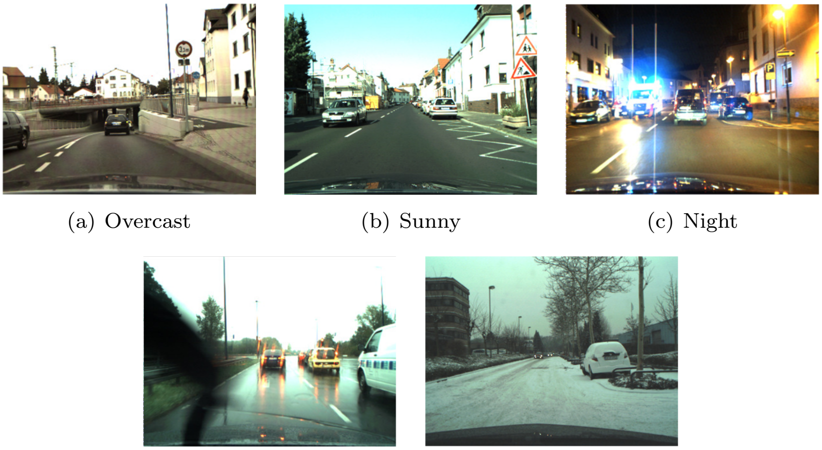
Car Classification: One of the most prominent and elaborated systems for detecting traffic participants is [Leibe et al., 2007]. In that work the authors apply a classifier previously used only for pedestrian detection [Leibe et al., 2005] and combine it with a ground plane estimation from optical flow and visual tracking. The performance of the whole system is assessed by means of two inner-city streams. Although this analysis highlights the system's performance in complex scenes an analysis with respect to different weather conditions is missing.

5.1.4. Experiments

The following experiments were conducted on image streams that have been acquired with a recording vehicle equipped with two mvBlueFox CCD color cameras from Matrix Vision. The cameras are attached behind the windshield with a baseline of about 34cm and deliver synchronized images with 800x600 pixels at a rate of 10Hz. For stereo processing the images are transformed to gray and scaled down to 400x300 pixels. The data from the cameras as well as internal car data from the CAN bus are transmitted via LAN to a storage system in the trunk. All the experiments discussed in the following were done offline after labeling the image streams for ground-truth comparison.

The performance of the system is assessed by means of ROC curves of the detected cars. There are two main aspects that are evaluated with the experiments:

1. Is there a quantitative gain in using depth estimation for rejecting implausible classifier detections?



(a) Overcast (b) Sunny (c) Night

(d) Rain (e) Snow

Figure 5.6.: These images are example frames of the five different weather conditions that were used for evaluating the system. For each condition a stream of roughly 10min length was recorded. Afterwards, all cars for one frame per second were hand-labeled to generate ground truth data for evaluating the car-detection system.

2. How well does the object-unspecific stereo popout cue perform when searching specifically for car objects?

For evaluating the robustness of the system, its performance under different weather conditions (overcast day, sunny day, night, rain, snow (see Figure 5.6)) and different scene types (inner city, highway, rural road, industrial area) is analyzed. To this end five video streams of roughly 10 minutes length each were recorded in different weather conditions. For all streams the same route was traversed, encompassing all of the aforementioned scene types. Hence, these streams allow for a detailed analysis of the questions raised.

In order to train the classifier, some parts of the overcast, sunny and rain stream were used. These parts were removed from the streams before evaluation. For a quantitative evaluation, one image per second has been annotated by hand to create ground truth data. The annotation constitutes a rectangular mask for each car in the scene, approximating the shape of the cars. The masks are *intrinsic*, i.e they approximate the actual object shape even if the object is not completely visible. Additionally, the amount of occlusion for each object is labeled.

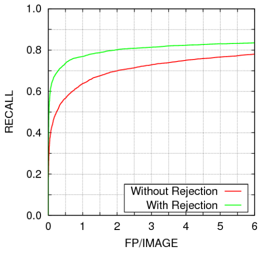
The performance of the single methods for the car detection task is assessed through ROC analysis. To investigate the trade-off between correct detections and false-positive detections of non-car scene elements, the false-positive rate per image frame (FP/IMAGE) is evaluated with respect to the so called RECALL R

$$R = \frac{TP}{TP + FN} \quad . \tag{5.1}$$

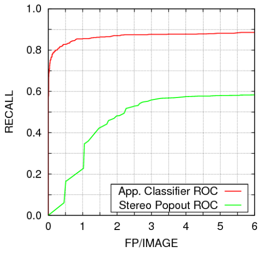
Here, TP stands for true-positives, the number of labeled cars which were detected by the system, and FN stands for false-negatives, the number of labeled cars which were not detected by the system. A labeled car is considered detected (true-positive) if the mutual overlap between the annotated and one detected region is at least 50%. Otherwise, it is considered a false-negative. Accordingly, detected regions are considered a false-positive if there is no annotated region which has at least 50% overlap with the detected region.

The advantage of using ROCs lies in analyzing the detection methods across their whole working range instead of using a single working point. For classifiers usually the confidence threshold Θ_C is varied. Unfortunately, the stereo popout cue has no confidence and thus no corresponding threshold. In contrast, the detections of the unspecific stereo cue are influenced by the height thresholds for selecting car-like obstacles. Since these are multiple thresholds this will not result in an ROC curve but rather in an ROC point cloud. To be comparable to the classifier cue, the Pareto front of the ROC point cloud of the stereo cue is calculated. This results in a curve which is also referred to as ROC curve in this work. When comparing two ROC curves in the following figures, the one ROC curve being above the other (i.e. higher in ordinate direction) is considered to be better because it shows continuously better performance. A complete review of the results would go beyond the scope of this thesis. Hence, only the most striking results are discussed.

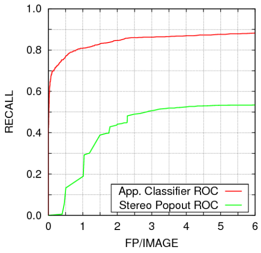
Firstly, the results show that using the stereoscopic depth for rejecting implausible classifier detections dramatically improves the performance of the classifier cue. The car classifier employed in this system



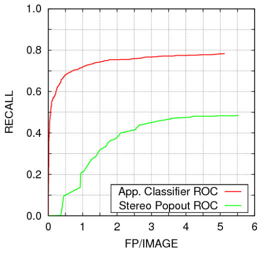
(a) Plausibility Check



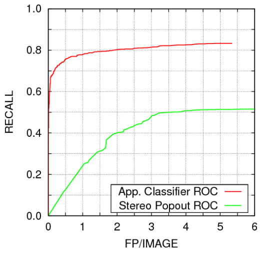
(b) Overcast Day



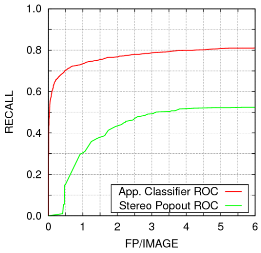
(c) Sunny Day



(d) Night



(e) Rain



(f) Snow

Figure 5.7.: (a) ROC comparison of the average (over all streams) classifier cue performance with and without using the stereoscopic depth estimation to reject implausible detections. (b-f) Comparison of the classifier cue and stereo popout cue performance for different weather conditions.

is appearance-based, i.e. its response depends on local image information. Due to this, the classifier itself is not able to revoke implausible detections according to their position or size. Indeed, a street scene has a defined structure and allows for several plausibility assumptions like cars being close to the ground plane, cars having a certain physical size or cars occurring only in certain areas of an image. As Fig. 5.7(a) shows, the average performance (over all streams) of the car classifier is significantly improved using the 3-D stereoscopic depth data to reject implausible detections.

A second observation from the results in Fig. 5.7 is that the detection performance of the stereo popout is much lower than the performance of the car classifier for all tested conditions. When having a more detailed look into the detection results of the object-unspecific stereo cue several

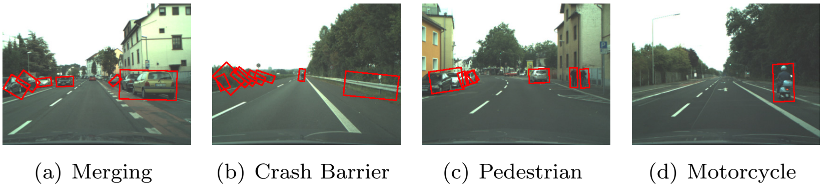

(a) Merging (b) Crash Barrier (c) Pedestrian (d) Motorcycle

Figure 5.8.: Examples of problematic situations which lead to a bad performance of the stereo popout cue for the car-detection evaluation. (a) Nearby obstacles like the two cars on the right can get merged into one detection. This will lead to an increase of false-negatives and false-positives because the mutual overlap is too small. (b-d) Crash barriers, pedestrians and motorcycles often pass the selection of car-like obstacles. This leads to an increase of false-positives.

shortcomings of the evaluation arise. First, the stereo cue tends to merge closely located objects (see Fig. 5.8(a)). This results in a larger obstacle region which is not considered a correct detection because the mutual overlap to a labeled car is less than 50%. In contrast, the detection will be considered as a false-positive. An evaluation in the appendix C with a less strong criterion based on the center point of a detection supports this argumentation. Second, the sieving of car detections from the unspecific obstacle detections is only a heuristic based on the depth data itself. This sieving is not working very well. For example pedestrians (Fig. 5.8(c)), motorcycles (Fig. 5.8(d)) and crash barriers (Fig. 5.8(b)) often remain after sieving for car obstacles. This in turn leads to additional false-positives because only cars were labeled and are considered a correct detection. Third, the stereo cue was only applied to 400x300 images which limits the maximal detection depth to roughly 50m. Altogether one can conclude that the detection performance of the stereo cue is very good in detecting obstacles but that extracting specific obstacles just from the depth information is very difficult. Thus a fair offline evaluation of the stereo cue can only be done by labeling all obstacles and not just cars. However, in [Schmüdderich et al., 2010] it could be shown that fusing the detected proto-objects from both the stereo and the classifier cue leads to an overall detection improvement, i.e. the stereo cue is able to complement the detections of the classifier cue.

A last but very important observation is that both the stereo and the classifier cue with implausibility check show a steady performance for the different weather and lighting conditions (Fig. 5.7(b-f)). This proves that both cues are robust against diverse conditions making them well-suited for real-world application. It also demonstrates that the depth estimation techniques introduced in chapter 3 and chapter 4 are working well in real-world conditions.

5.1.5. Discussion

In the above sections a car detection system was introduced that employs the techniques introduced in chapters 3 and 4 to boost the performance of a car classifier and to calculate a stereo popout. The latter combines the depth map of the SNCC block-matching stereoscopic depth estimation with a car-relative 3-D street estimation based on direct surface fitting to detect obstacles in a generic and unspecific way. In order to test the system and its cues, experiments in different weather and lighting conditions were conducted.

First, it has been shown that the performance of the car classifier can be improved significantly by using the calculated stereoscopic depth to reject implausible detections. Since the cars are detected in the physical world, this allows for some well-justified assumptions like that the cars are connected to the ground or have a certain physical size. By means of these assumptions many implausible classifier detections can be rejected.

Second, a comparison of the car classifier with the unspecific stereo popout revealed that the car detection performance of the latter is much worse. The main identified reason is that the object-unspecific stereo popout struggles to output only car obstacles. Obstacles like pedestrians or crash barriers are often hard to segregate from cars by using only depth information. Thus they pass the car-obstacle selection which deteriorates the detection result because these obstacles are considered a false-positive. From this, one can conclude that unspecific cues can not be evaluated in the same way as specific cues. It is rather necessary to take the unspecific nature into account, e.g. by labeling all obstacles for evaluation.

Nevertheless, both the classifier cue with plausibility check and the stereo popout showed a steady performance for all tested conditions. This demonstrates that the car classifier as well as the stereo popout are very robust with respect to environmental conditions. Furthermore, this proves that the introduced SNCC block-matching (see chapter 3) and the introduced direct surface fitting (see chapter 4) are well-suited for real-world applications because both the stereo popout and the classifier cue use the stereoscopic depth as an input. For block-matching stereo with SNCC it could also be shown that it significantly boosts the stereo popout (see appendix C) as compared to block-matching stereo with NCC. What remains to be shown, however, is the real-time capabilities of direct surface fitting and SNCC block-matching stereo. For this purpose, the next section discusses a robotic vision system which uses these two novel techniques. The vision system is implemented on a humanoid robot whose task is to grasp objects from a table to deliver them to a human tutor.

5.2. Robotic Vision System

Nowadays, humanoid robots have reached a technical state where they are not only capable of walking but also have the ability to grasp and manipulate objects. Furthermore, algorithms for planning the motion of these robots have matured and yield reliable and natural results. One still researched key element for autonomous robots that act in dynamic environments is a robust perception of the immediate surrounding. The following sections introduce a robotic vision system [Einecke et al., 2011] that makes use of the techniques introduced in chapter 3 and chapter 4 for a robust perception in dynamic environments.

5.2.1. System Target

The target is to enable a humanoid robot with stereo-vision to grasp and manipulate objects in an unconstrained environment. One drawback of current state-of-the-art robotic systems is that they either need a predefined layout of the environment and the 3-D shape of the objects, or special hardware like laser scanners for acquiring the shape of objects. Furthermore, the estimation of a 3-D scene can take several seconds. In contrast, the system discussed in this work should be able to keep track of changes of the 3-D scene layout in real-time by just using visual input. Another requirement is, that the error of the depth estimation has to be below 1cm for the robot to be able to grasp an object. The robot platform used in this setup poses an additional challenge to the depth estimation. Due to joint limits, the robot is not able to look down far enough to have a typical household table in its field of view when standing right in front of it. In order to grasp an object from a table, the robot must be able to estimate the object's (handle) position from a distance of at least 1.5m.

5.2.2. Related Work

There are several approaches that try to tackle the problem of scene recognition and object manipulation in dynamic environments. The authors of [Berenson et al., 2007] present a method for grasp planning in complex scenes. They show that with the use of motion capture data and known object geometries, a stable and collision-free grasping is possible in cluttered scenes. Similarly in [Huebner et al., 2009] object shapes are predefined and a modeling phase is included in which 3-D object meshes are designed, and later matched to the visual perception. By this means the robot is able to evaluate different grasp hypotheses based on the known geometries. In the work of [Mühlig et al., 2010] also predefined object shapes are used. These are matched to the stereo-vision input, which allows a robot to grasp objects on a table. All these approaches rely on predefined 3-D object models. This assumes a "closed world", which is not applicable to everyday environments as the amount of different object shapes is too vast.

Without predefining object shapes, one needs to rely on high-quality sensor information about the environment. In [Rusu et al., 2009] the ability to extract a 3-D scene representation by using a continuously tilting laser rangefinder is shown. This representation is good enough to grasp box-like and cylindrical objects that are not known before. The authors of [Rasolzadeh et al., 2010] impressively show that in static scenes purely vision-based input from a robotic head with 4 cameras can be enough to allow for a model-free grasping of objects. A grasping plane is matched to the 3-D point cloud of a segmented object to achieve a top-grasp with a 6-DOF robotic arm. In [Saxena et al., 2007] a model-free approach is presented that incorporates a learning algorithm to infer 3-D grasp points on objects given their image. In combination with an obstacle map generated from stereo vision, a robotic arm is able to unload objects from a dish washer.

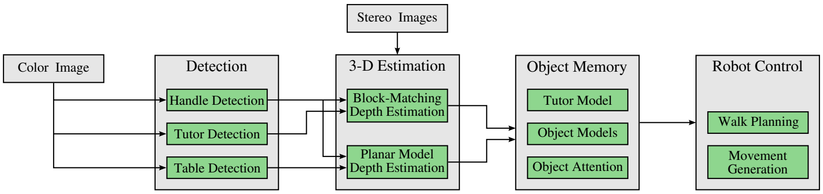


Figure 5.9.: Overview of the robotic system. The processing pipeline consists of four main modules. The first detects relevant objects while the second enriches the detected objects with 3-D information to locate them in space. Module three converts the detected objects into the robot's internal representation and module four controls the robot's movement.

In conclusion there are several shortcomings in current state-of-the-art systems. Either, the robot's internal model of the environment is predefined, the scene has to be static and analyzed thoroughly, or additional external sensory data (e.g. from a motion capturing system) is necessary to enrich the robot's internal model. There are two reasons for this: First, the visual perception of the environment is still an open issue. Second, the evaluation of motion planning algorithms is more straightforward if errors in the model generation of the environment can be neglected.

5.2.3. System Structure

In the robotic system presented here, the target is to have a real-time, vision-based generation of internal scene representations for unconstrained environments. For this purpose, the block-matching stereoscopic depth estimation with SNCC introduced in chapter 3 is combined with the model-based direct surface fitting introduced in chapter 4, a color-based object segmentation and a Hough line detection.

In Fig. 5.9 a systematic overview of the visual processing is depicted. As can be seen, the processing pipeline consists of four main modules. The first is a detection stage which locates interesting objects in the input images. In the subsequent module the detections from the first module are enriched with 3-D information acquired from stereo camera images. The third module converts the detected objects into the internal object memory of the robot. Based on this memory the last module performs the robotic control of walk path planning and movement generation.

In order to make this system independent of predefined 3-D object shapes the system tries to detect handles instead of full objects because these are generic parts that are shared among a plurality of objects. This way the robot can interact with many objects that have neither been seen in advance nor been predefined by hand. The handles of the objects are detected in the 2-D images by means of a Hough line detection stage [Schmüdderich, 2010, sec. 5.2.2] and are enriched with 3-D data from both stereo processing sub-modules. The detection of the planar table tops follows a similar argumentation. It is not wise to learn the position, shape and orientation of tables or desktops, it is better to detect planar surfaces and estimate their 3-D location as this is more generic.

Furthermore, the robot has an inbuilt ability to recognize a human tutor. The recognized tutor is combined with the depth information provided by the block-matching stereo to feed a human kinematic model. By doing so, the robot can estimate the pose of the human tutor which allows for a more natural interaction with the robot by means of simple gestures. The idea of the human tutor is that he can show all important objects to the robot that the robot did not recognize by himself. In order to ease this training, all objects are homogeneously colored, i.e. when the tutor presents an object to the robot, it will just learn the color.

Object Detection and Attention

In the presented system the robot detects relevant objects based on a color-segmentation of the visual input. The robot learns this color in interaction with the human tutor. The method described in this section allows for an easy and natural interaction by taking an object into both hands.

In the first step the hands of the tutor are detected by skin-color segmentation and stored in a 2-D binary map. Then the convex hull around both hands is calculated. The hand pixels are subtracted from the convex hull, resulting in a 2-D binary map of the area between the tutor's hands. Within this area all pixels from the original input image are selected which fulfill the following conditions:

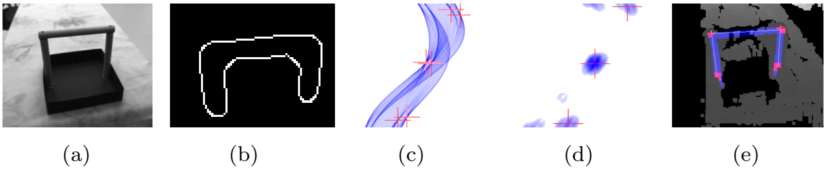


Figure 5.10.: The process of estimating 3-D lines using Hough transformation. For the object shown in (a) a cut-out is extracted and converted into an edge map (b). The edge image is transformed to the Hough space (c), where a DoG-filter is applied (d) to extract lines in the center of the object's handle bars. The end-points of each line are combined with depth-information (e) to obtain a 3-D line representation.

- They have approximately the same depth as the hands.
- They are within a reasonable lighting range in order to counteract overexposure (white parts) and camera noise (dark parts).
- They contain color information (saturation threshold).

The color of the object is extracted by averaging the hue of all selected pixels. In order to increase the robustness, a simple recursive low-pass filter is applied and the values are only updated if the amount of selected pixels surpasses a given threshold. The resulting color is used to detect objects in a similar way as in [Bolder et al., 2007]. Furthermore, the knowledge about an object's color can be used to generate masks for the direct surface fitting explained in chapter 4.

Handle Estimation

After an object has been identified the 3-D orientation and position of its handle need to be estimated in order to grasp the corresponding object. For locating handles in the input images a Hough transformation based approach [Schmüdderich, 2010, sec. 5.2.2] is used. This handle estimation consists of five steps.

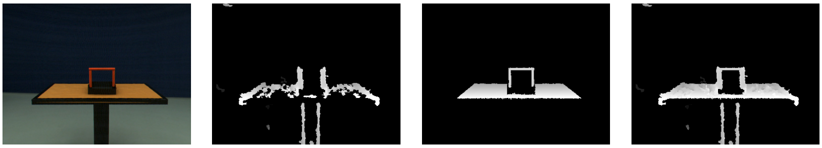
In the first step a rectangular area around the object is cut out. This reduces the computational cost as only a sub-part of the image is processed. Within the cut-out area the Canny-Edge-Detector [Canny, 1986] is applied to extract candidate pixels of the handle. Figure 5.10(b) shows an example edge map of the object shown in Fig. 5.10(a). The third step is to transform to the edge map into the Hough space [Hough, 1962] for line detection [Duda and Hart, 1972]. In the Hough space, straight lines can be detected by finding the maxima. Figure 5.10(c) shows the Hough transformation of the edge map in Fig. 5.10(b). The red crosses mark the six identified maxima. These six maxima represent the six straight lines the handle is composed of. The fourth step is a DoG filtering of the Hough space. This filtering groups pairs of parallel straight lines by merging them into one straight line (see Fig. 5.10(d)) that is located in the center between both lines. Due to the merging, the maxima now represent straight lines that go through the middle of the straight handle parts instead of representing the outer edges. This is important for the last step. The maxima are transformed back into the image space as can be seen exemplarily in Fig. 5.10(e). Then the lines are transformed into 3-D lines by using the 3-D information from the depth map.

Depth Estimation

The 3-D estimation employed in this robotic system uses the block-matching stereo with SNCC (see chapter 3) for a generic scene depth map and direct surface fitting (see chapter 4) with a planar model (see section 4.3.1) for dedicated single measurements.

The feed-forward depth map computed by means of block-matching stereo is the main source of depth information for the robot system. The 3-D position of the head and the hands of the tutor are solely based on this depth map. Also the handle's 3-D positioning is mainly based on the depth information from the block-matching map. However, the handles are very homogeneous and might be aligned with the epipolar lines (see section 2.2.3) which makes it very difficult for block-matching stereo to calculate a depth value at all. Figure 5.11(a) shows a typical scene with a basket on a table. As can be seen the horizontal upper part of the handle is in alignment with the horizontal epipolar lines. This alignment leads to an aperture problem which impedes the estimation of depth for the horizontal handle part. This problem is clearly visible in the depth map in Fig. 5.11(b). Hence, it is necessary to use a different depth estimation approach for such critical but important scene elements.

As discussed in chapter 4 the fitting of a geometrical model directly to the stereo images is much more robust than block-matching approaches



(a) Scene (b) BM Stereo (c) Planar Stereo (d) Fused Stereo

Figure 5.11.: Example of aperture problems. (a) Image of a horizontal handle and a weakly textured table top. (b) Standard block-matching stereo processing fails to estimate depth for the horizontal part of the basket handle and the table's surface. (c) Result of direct surface fitting using a planar model for the basket handle and the table. (d) Fused result of standard stereo and the planar stereo. The depth maps are coded in gray, near pixels are bright and far pixels are dark.

in such situations. In order to improve the depth perception the Hooke-Jeeves based direct surface fitting introduced in chapter 4 is used. As explained this fitting needs a rough mask of the area to fit. From the object memory (explained in the next section) the robot knows the color of the object's handle. He uses this top-down information to generate a binary mask of the handles by means of a simple color segmentation which is then fed into the direct surface fitting algorithm. Since handles are usually not skewed, they can be regarded as lying within a plane. Thus a planar model is sufficient for estimating the depth of handles. In the same manner, an additional depth estimation for homogeneous table tops is calculated.

Figure 5.11(c) shows the exemplary application of direct surface fitting with a planar model for estimating the handle of the basket and the surface of the weakly textured table top in Fig. 5.11(a). Unlike the block-matching approach a robust and dense estimation is achieved. The result of this planar estimation is fused (see Fig. 5.11(d)) with the result of the block-matching stereo by using the planar estimation to fill the holes in the depth map of the block-matching stereo. This way the depth map acts as a kind of black board, which all algorithms and processing stages can easily access and process. Of course it would also be possible to use the estimated parameters of the planar model. However, this makes the system less flexible because it increases the dependencies between the different processing modules.

Object Memory and Representation

In order to increase the stability of the robots perception of the real world all perceived objects are stored in a *Persistent Object Memory* (POM) [Gienger et al., 2010a, Mühlig et al., 2010]. The POM can be viewed as the working memory of the robot which accumulates all sensory data. Stabilization of the perception is realized by means of a mixture of low-pass, median and model-based filters. One important aspect of the POM is that each object is associated with a confidence value that relates to the quality of the perception. If objects are occluded or have not been observed for a certain time, their confidence value starts to decay.

Another important aspect of the POM is that it maintains a model of the human tutor. As explained above the human tutor is mainly detected by means of skin-color and enriched with 3-D data from the block-matching stereo. Within the POM the tutor model is subject to an inverse kinematic control scheme. As discussed in [Mühlig et al., 2010] this strongly improves the perception of the human tutor because the kinematic control does not allow for unnaturally fast movements. Furthermore, the single parts of the tutor (head, left hand and right hand) are coupled into a coherent model which limits the relative positions to natural configurations. The model also allows for a straightforward evaluation of the tutors pose, e.g. detect if a hand is raised, which can be used for a simple interaction with the robot. Additionally, the attention mechanism of the POM shifts the attention of the robot to objects that are touched or moved by the tutor.

The internal representation of the POM is based on a kinematic tree, which comprises the robot's links as well as all perceived entities including their geometrical properties. This allows to easily define controllers for the robot that operate directly on observed objects [Gienger et al., 2010a].

Robot Control

For movement generation, the robotic system uses the whole body control algorithm described in [Gienger et al., 2010b]. One key feature of this approach is that a primary control objective x can be tracked precisely in the task space, while the redundant null space can be exploited to satisfy secondary objectives, e.g. an avoidance of joint limits. This enables the robot to perform smooth movements which also look very natural. Furthermore, the whole body controller is coupled with a walking and balancing controller [Hirose et al., 2001], which stabilizes the motion.

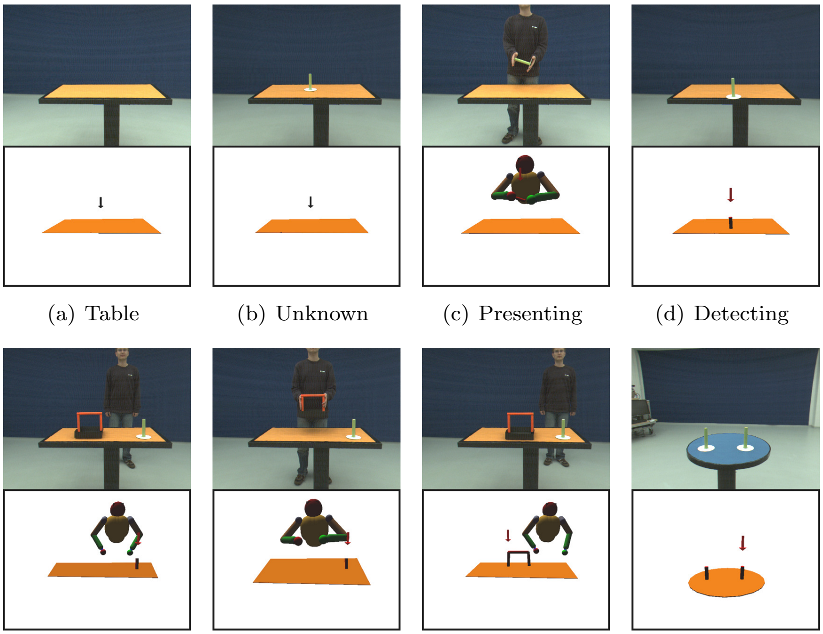
For reaching and grasping objects, it is very important to select a good target stance position of the robot with respect to the object. Thus the stance position is optimized under the constraints of reaching and grasping the object. This optimal stance position is then used to compute a walking path from the current position to the object by means of the movement optimization explained in [Toussaint et al., 2007].

5.2.4. Experiments

This section discusses two experiments that were conducted with the introduced robotic system. The first experiment is a pure perception experiment which demonstrates the system's ability to recognize the 3-D position and orientation of objects and tables. Here different objects are presented to the robot by the human tutor. Furthermore, the objects are put on different table tops. In a second experiment, the robot is asked to deliver a newly presented object from a table. This experiment shows the overall system performance and flexibility because the robot has to grasp an object and avoid obstacles in a scene setup it sees for the first time. The successful grasping of the object also shows that the depth estimation is very precise as the depth estimation is done from a distance of roughly 1.5m.

Scene Perception

As explained in section 5.2.3, the robotic system uses the SNCC block-matching stereo from chapter 3 and the direct surface fitting from chapter 4 together with a set of other vision algorithms to perceive its environment. The gathered information is used to dynamically construct an internal representation in real-time which the robot can use for planning grasping and walk-movement trajectories. The whole perception runs with a frequency of about 5-7 Hz on a Pentium Quad Core using the non-threaded version of the block-matching stereo with SNCC and


(a) Table (b) Unknown (c) Presenting (d) Detecting

(e) Unknown 2 (f) Presenting 2 (g) Detecting 2 (h) Round Table

Figure 5.12.: Internal scene representation. (a) A table is recognized. (b) Unknown object on the table is not recognized. (c) Human tutor is presenting the unknown object, shifting the robot's attention to it. (d) Robot detects the now known object. (e) Another unknown object. (f) Human tutor presents the new object. (g) Robot has shifted its attention to the new object. (h) Recognition of a round table with two recognized objects.

the non-sparse version of direct surface fitting. Figure 5.12 shows a sequence of the robot's camera views together with the corresponding internal representation.

First, a table is placed into a previously empty scene (see Fig. 5.12(a)). The robot notices the table by means of its color, estimates the table's position and orientation and adds the table to its internal representation. As long as the table is visible the robot will keep track of the table's positional parameters. Next, an object is placed on top of the table (see Fig. 5.12(b)). As the object is unknown to the robot it will ignore this object. In a next step, the tutor takes the object into both hands (see Fig. 5.12(c)). The robot recognizes the tutor and that the tutor has something in his hands. This is achieved by the mechanism explained in section 5.2.3 which compares the depth of the tutor's hands with the depth of the area between them. If the depth is similar then it is very likely that the tutor is carrying an object. Now the robot recognizes the object as being important and shifts its attention to it. The robot extracts the object's color in order to be able to detect it without tutor interaction (see Fig. 5.12(d)). If a new object with a different color is placed on the table the robot will ignore this object because it is unknown to him (see Fig. 5.12(e)). Again the tutor can present the new object to the robot (see Fig. 5.12(f)) which will make the robot shift its attention from the old object to the new one. After the robot has extracted the new object's color it will recognize this new object without the tutor's interaction (see Fig. 5.12(g)). For completion, Fig. 5.12(h) shows a different scene setup with a round table and two identical objects. This demonstrates that the system is not tailored to a specific type of table and that it does not expect only one instance of the object it is currently focused on. The attention of the robot (indicated by the down-pointing arrow) will be on the object that was last touched or moved by the human tutor.

For an assessment of the accuracy of the visual perception the mean and variance of the visual estimation were recorded over time. This recording consists of 350 image frames of the scene in Fig. 5.12(g) and 200 image frames of the scene in Fig. 5.12(h). Table 5.1 shows the results for the position, orientation and the size estimation of the objects in these scenes. Please note that the handle of the basket in Fig. 5.12(g) is split into three straight parts due to the Hough line extraction. The positions of the estimated objects are the (x, y, z) coordinates. For these no ground truth values are available. However, the standard deviation of the estimation is in most cases below one centimeter, i.e. the estimation is very stable. The z-vector in Table 5.1 denotes the plane normal for the table and the symmetry axis of the straight handle parts. Here the standard deviation is between one and four degrees.

For the size of the objects Table 5.1 prints the ground truth sizes together with mean and standard deviation of the estimations. The size for the rectangular table denotes its width and depth elongation, for the round table it denotes its radius and for the straight handle parts

Table 5.1.: Accuracy of the visual perception over 350 frames of the scene in Fig. 5.12(g) and for 200 frames of the scene in Fig. 5.12(h). Sizes are in meters, position and size deviations are in mm and the vector deviation is in degree. The size of the rectangular table is its width and depth elongation, the size of the handle elements is their length and the size of the round table is its radius. Note that the basket handle is split up into its three straight parts.

object	σ position (mm)	σ z-vector (deg)	μ size (m)	real size (m)	σ size (mm)
rect. table	(3.3, 0.9, 0.9)	1.02	(0.46, 1.02)	(0.5 1.0)	(9.6, 2.9)
basket left	(8.6, 2.0, 1.8)	4.68	0.12	0.15	2.8
basket up	(7.0, 1.1, 1.6)	0.63	0.18	0.20	2.7
basket right	(6.3, 1.4, 1.4)	2.62	0.11	0.15	2.4
round table	(16.7, 13.1, 3.1)	4.29	0.30	0.30	12.0
left object	(25.9, 2.9, 7.3)	1.95	0.15	0.14	3.5
right object	(33.4, 3.6, 9.0)	1.96	0.14	0.14	3.2

their length. One can see that all sizes are estimated quite accurately. In many cases the accuracy is about one centimeter with a worst estimation error of four centimeters, e.g. for the rectangular table. As for the position of the objects, also the size of the objects is subject to only very little variations. Considering the fact that these errors are a combination of the errors of the color segmentation, the 2-D Hough line extraction and the depth estimation, these results demonstrate the high accuracy of the proposed scene perception. Furthermore, the following delivering experiment shows that the estimation is accurate enough for a reliable grasping and collision avoidance.

Delivering Experiment

In the last section, it has been shown how the robot dynamically constructs an internal representation of its environment. As the representation is extracted from the current visual input it is generic and allows the robot to interact and move in unknown scenes. This unconstrained interacting is demonstrated by means of the following delivering task which involves the unconstrained 3-D position and orientation estimation of an previously unknown object, the grasping of the object, and the delivering of the object to the human tutor. Figure 5.13 shows an internal view sequence of the robot while performing such a task.

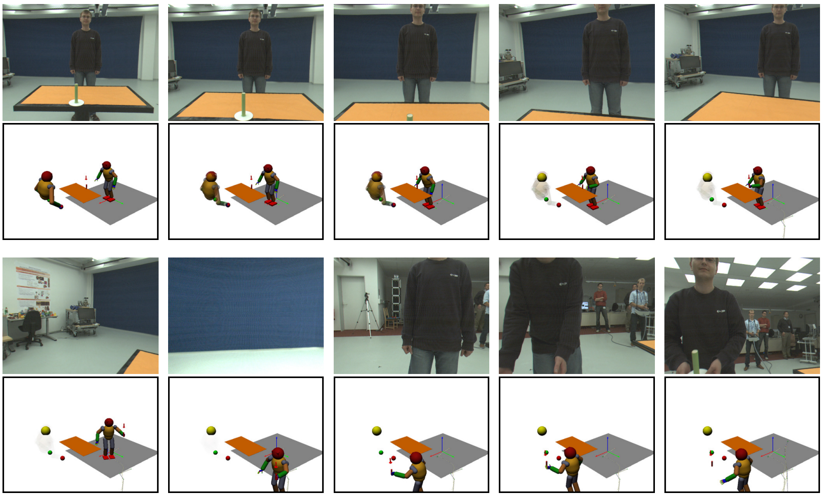


Figure 5.13.: Image sequence of the robot's camera view and its internal representation for an object delivering scenario. First the human tutor indicates an object that the robot shall deliver. Then the robot approaches the table and grasps the object using the depth estimation for judging the 3-D position and orientation of the object. Afterwards the robot walks to the tutor avoiding the table obstacle and hands over the object.

In the first image the robot looks at the scene using the frustrum-based gaze behavior described in [Einecke et al., 2011]. This way the robot tries to keep the relevant objects within its field of view. The next two frames show the robot approaching the table for grasping the object. The walk path is determined as described in section 5.2.3 and depicted by a thin dotted line on the floor. Frames four and five show the robot grasping the object. In frames six to eight the robot walks towards the tutor thereby avoiding to collide with the table. Again the planned path is depicted by the dotted line. In the last two frames the robot hands over the object to the human tutor.

It is important to note that neither the position and orientation of the object nor the position, orientation and physical extent of the table are known to the robot in advance. The robot estimates all this data from its visual input at a pace of roughly 5Hz. Here the estimation of the table is very crucial because its physical presence is important for planning the grasping (not trying to reach through the table) and for planning the movement towards the human tutor (not colliding with the table). As the sequence in Fig. 5.13 shows, the robot successfully estimates its surrounding and successfully plans and executes the grasping of the object and the movement to the human tutor for delivering the demanded object.

5.2.5. Discussion

The presented robotic system couples a visual 3-D scene perception, a persistent object memory and a task-level whole-body control for interacting in previously unknown scenes with unknown objects. For a stable and robust visual 3-D perception, the block-matching stereo with SNCC introduced in chapter 3 has been coupled with the model-based direct surface fitting introduced in chapter 4. In order to allow for a generic object manipulation, a Hough line approach [Schmüdderich, 2010, sec. 5.2.2] is used to detect object handles independent of the object identity. The robot uses this visual information to dynamically construct a representation of the current scene in real-time. In contrast to other approaches no a priori object or scene knowledge is used. By means of the integration with a dynamic movement generation the robot is able to use its online 3-D scene estimation to plan and successfully execute grasping and delivering of objects, thereby avoiding detected obstacles.

The conducted experiments demonstrate that the 3-D estimation is accurate and robust. Even though the position and orientation of the object handles are estimated at a distance of roughly 1.5m, the robot is able to safely grasp objects on a table. In addition the estimation is fast enough to be carried out in real-time. This allows for a fast update of the 3-D structure of the immediate surrounding. In summary this system proves the applicability and benefit of the algorithms introduced in chapters 3 and 4. In particular, it demonstrates the real-time capabilities of the SNCC block-matching stereo and the planar direct surface fitting because the robot is able to perceive its 3-D environment with an update rate of 5Hz. Additionally, the good performance for the low-textured scene parts highlights the advantage of the model-based direct surface fitting for challenging conditions in which block-matching methods are prone to fail. Furthermore, the system integration shows that the need for a mask for the model-based fitting can easily be fulfilled by using object knowledge in a top-down manner.

Although the performance of the presented robotic system is promising it still has some limitations. First, the current system learns and detects objects mainly based on their color. Using a dedicated object classifier like [Wersing et al., 2007] for detecting and recognizing the objects would make the whole system much more generic as compared to a simple color segmentation. In addition an unspecific object detection similar to the stereo popout presented in section 5.1.2 could be used to detect yet unknown objects which might be learned in an interaction with the human tutor. Second, the robot can hold only one object in its attentional buffer, i.e. it will forget about the last object's properties as soon as the attention shifts to a new object. By integrating a memory architecture like [Rebhan et al., 2008] with a short-term and a long-term memory, the robot could store and remember multiple objects even between different scenes. Third, the obstacle avoidance of the presented system makes use only of the objects in the POM. Hence, it will ignore all obstacles that it can not transfer to the object memory. It is very important to complement this with navigation techniques based on generic obstacles like [Burschka et al., 2002] or occupancy grids [Moravec and Elfes, 1985, Elfes, 1989].

6. Summary

The goal of this thesis was to develop algorithms for stereoscopic depth estimation which are suited for real-time, real-world applications. To reach this goal two novel methods have been proposed. First, the *summed normalized cross-correlation* has been introduced, a new cost function for block-matching stereo processing. Second, the *direct surface fitting* algorithm has been proposed which allows to fit parametric surface models directly to the stereo images.

The proposed *summed normalized cross-correlation* (SNCC) cost function is an improvement of the normalized cross-correlation (NCC) which does not suffer from the fattening effect. Currently, the two main existing explanations for the fattening effect of NCC are the perspective view changes between the cameras and variance differences between neighboring image regions. In contrast it has been revealed in this work, that the fattening effect of NCC arises from strong contrasts within a matching block. Thus, instead of calculating the correlation-coefficient for a full matching block, SNCC calculates the correlation-coefficient for small overlapping sub-blocks and sums these up for averaging. This two-stage approach of a small NCC filter and a larger summation filter strongly reduces the fattening effect without increasing the noise. Evaluations on the Middlebury stereo benchmark have shown that using SNCC for standard block-matching leads to a highly accurate and dense stereo algorithm that is almost comparable to state-of-the-art algorithms based on global optimizations like graph cut or belief propagation. However, it has also been shown that by using efficient box-filters the SNCC calculations are at least one magnitude faster than any global optimization. Hence, block-matching stereo with SNCC provides a much better balance between speed and accuracy than any existing method. Furthermore, a parallel processing scheme of block-matching stereo for multiple CPU cores has been introduced. It has been shown that this scheme allows for a linear gain in speed with respect to the number of CPU cores used. For eight cores the runtime comes already close to the runtime of comparable implementations on high-end GPUs.

The second major result of this thesis is the proposed *direct surface fitting* algorithm. It is inspired by the homography-constrained gradient descent fitting that is frequently used in state-of-the-art work to estimate the orientation and depth of planar surfaces. The major drawback of this approach is its restriction to a planar model. The newly proposed *direct surface fitting* overcomes this limitation by replacing the gradient descent search with the direct search method of Hooke-Jeeves. This has been shown for the surface models of spheres and cylinders. The general belief in the inefficiency of Hooke-Jeeves with respect to gradient descent could be disproved. In fact, a comparison showed that the overall runtime is very similar while *direct surface fitting* is much more robust than homography-constrained gradient descent. Unfortunately, *direct surface fitting* shares the characteristic of the homography fitting that for large image regions the estimation can become computationally very demanding. In order to tackle this problem a sparse fitting based on an efficient sub-sampling of large image regions has been proposed which leads to a large speedup while losing only little accuracy. Due to this, *direct surface fitting* becomes well-suited for real-time applications as even the estimation of a large street surface can be performed in 10-20ms on a single CPU core.

To validate the real-world applicability of the two proposed methods, they were integrated into a car-detection system and the vision system of a humanoid robot. In the car-detection system the block-matching stereo with SNCC is combined with the *direct surface fitting* to segment obstacles on the street which are then refined to car hypotheses. Evaluations in five different weather and lighting conditions have shown that the depth-based car-detection has a robust and stable performance. However, in comparison with a specialized car-classifier the performance is much lower. The main reason for this is that the refinement to only car hypotheses is very difficult. On the other hand it could be shown that the performance of the specialized car-classifier can be significantly improved by using the calculated depth information to reject implausible detections. In the robotic vision system the two proposed stereoscopic methods are used to acquire robust and highly accurate estimations of table surfaces and handles. The experiments have demonstrated that the combined depth estimations allow for a collision-free grasping from a table and a collision-free walking around a table. Altogether the results achieved with the two applications demonstrate the real-world capabilities, the accuracy and robustness as well as the real-time capabilities of the proposed depth estimation methods.

To conclude, in this work two novel methods for stereoscopic depth estimation have been proposed which improve over state-of-the-art approaches in the balance between speed and quality as well as in robustness. An integration into two vision systems has demonstrated the real-world applicability of both methods. Furthermore, a multi-core processing scheme of the block-matching stereo and a sparse matching scheme for *direct surface fitting* allow for super real-time speed, leaving sufficient time for subsequent processing modules to execute.

Outlook

Although the proposed methods mainly comply the goal of this thesis to develop real-time and real-world capable stereoscopic depth estimation algorithms, there are still some issues that could be tackled and investigated in future work.

First, the fast and simple disparity fill-in mechanism which is used in the post-processing of the block-matching stereo algorithm could be improved. The experimental results for the Middlebury benchmark have revealed some shortcomings. For example disparity gaps at the image borders are likely to be filled with wrong disparity values. The reason for this is that the simple fill-in mechanism uses only the depth information. Here a more sophisticated method taking for example color information into account could improve the benchmark results. The challenge would be to keep up with the fast processing of the simple mechanism in order not to hamper the high overall speed of the depth estimation. On the other hand, the real-world applications have shown that often it is not wise to force a stereo algorithm to output a completely dense depth map. Especially for large invalidated regions it is usually much better to keep these as they are so that it is clear that the depth measurement failed at certain image regions instead of having interpolated values of unknown reliability. Hence, for real-world applications it might be of greater interest to develop fill-in mechanisms that reliably fill only the small gaps that are caused by image noise and low-texture.

Second, future work should investigate to which extend the *summed* *normalized cross-correlation* (SNCC) cost function is also beneficial for block-matching optical flow. Since stereoscopic depth can be regarded as a special case of optical flow it is likely that a block-matching SNCC leads also to a major improvement in optical flow calculations. One thing that needs to be adapted is the multi-core parallel processing scheme. Due to the vertical search directions in optical flow the necessarily larger overlap for horizontal striping would lead to a significant decrease of the speedup. Here a distribution of the search range to different threads could be more beneficial although this will require a more complicated merging of the single thread results.

Third, the *direct surface fitting* algorithm introduced in this thesis could be extended to several, arbitrarily positioned cameras allowing for multi-camera surface fitting. Furthermore, such an extension would allow for a more sophisticated temporal integration of the stereoscopic model fitting. For doing so the stereo camera images from two successive time steps could be warped into one reference camera at one time step thereby combining the matching value of four camera views (two camera views from each time step). This would lead to a larger robustness against the aperture problem due to additional constraints.

A. Derivation of Model-Based Mapping

This is a full-fledged derivation of the mapping formulas of the plane, sphere and cylinder used for the direct surface fitting algorithm introduced in chapter 4.

A.1. Planar Model

In order to derive a formula z_L that depends on planar model parameters, we describe a planar image region (*target plane*) relative to a *virtual plane* that is parallel to the camera sensor chips. The planes differ by a rotation at a certain anchor point about the x- and y-axis. Fig. A.1 shows a schematic top view. The anchor point is specified in world coordinates and denoted with $\mathbf{x}_a$. The orientation is specified via rotation angles about the x-axis (α_x) and y-axis (α_y). Note that these two rotations suffice to describe any possible plane orientation.

From analytical geometry, it can be derived that points $\mathbf{x}'$ from the *virtual plane* are transformed into points $\mathbf{x}$ on the rotated *target plane* by applying the transformation matrix

$$\mathbf{T} = \begin{pmatrix} \cos\alpha_y & \sin\alpha_x \sin\alpha_y & \cos\alpha_x \sin\alpha_y \\ 0 & \cos\alpha_x & -\sin\alpha_x \\ -\sin\alpha_y & \sin\alpha_x \cos\alpha_y & \cos\alpha_x \cos\alpha_y \end{pmatrix}, \tag{A.1}$$

leading to the following transformation formula

$$\mathbf{x}_L = \mathbf{T}\left[\mathbf{x}'_L - \mathbf{x}_a\right] + \mathbf{x}_a \ . \tag{A.2}$$

As the *virtual plane* is parallel to the sensor chips of the cameras, the z-coordinate for points on this fronto-parallel plane is always equal to the z-coordinate of the anchor point, i.e. $z' = z_a$. Using this, we can

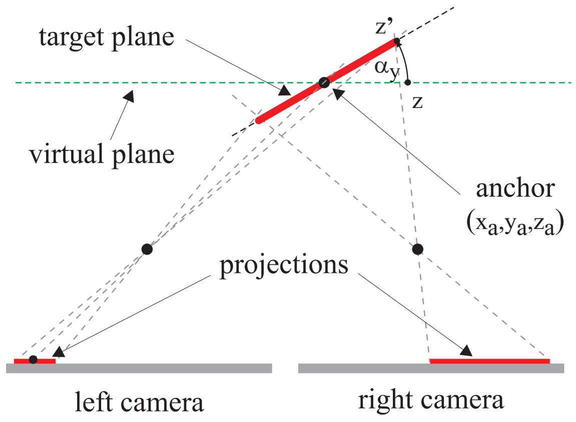


Figure A.1.: Top-view of the projection of a planar surface to the image planes of a parallel stereo camera. The planar surface (target plane) can be described by an anchor point and the orientation relative to a virtual plane. This virtual plane is parallel to the camera sensor chips, i.e. the relative orientation is also relative to the image planes.

rewrite the transformation equation above to

$$\begin{pmatrix} x_L \\ y_L \\ z_L \end{pmatrix} = \mathbf{T} \begin{pmatrix} x'_L - x_a \\ y'_L - y_a \\ 0 \end{pmatrix} + \begin{pmatrix} x_a \\ y_a \\ z_a \end{pmatrix} . \tag{A.3}$$

It is straightforward to extract an equation for the depth z_L on the *target plane* from the equation above

$$z_L = -(x'_L - x_a) \sin \alpha_y + (y'_L - y_a) \sin \alpha_x \cos \alpha_y + z_a \ , \tag{A.4}$$

where $(x'_L - x_a)$ and $(y'_L - y_a)$ can also be described by their counterparts on the rotated *target plane*. Expressions for both terms can be derived by rearranging the transformation equations (A.3)

$$x'_L - x_a = \frac{x_L - x_a - (y'_L - y_a) \sin \alpha_x \sin \alpha_y}{\cos \alpha_y} \tag{A.5}$$

$$y'_L - y_a = \frac{y_L - y_a}{\cos \alpha_x} \ . \tag{A.6}$$

Furthermore, we can substitute $(y'_L - y_a)$ in equation (A.5) with equation (A.6)

$$x'_L - x_a = \frac{x_L - x_a - (y_L - y_a)\tan\alpha_x \sin\alpha_y}{\cos\alpha_y} \ . \tag{A.7}$$

Applying the substitutions (A.7) and (A.6) for $(x'_L - x_a)$ and $(y'_L - y_a)$ to the depth formula (A.4) leads to

$$\begin{aligned} z_L - z_a &= -(x_L - x_a)\tan\alpha_y + (y_L - y_a)\left[\frac{\tan\alpha_x \sin^2\alpha_y}{\cos\alpha_y} + \tan\alpha_x \cos\alpha_y\right] \\ &= -(x_L - x_a)\tan\alpha_y + (y_L - y_a)\left[\sin^2\alpha_y + \cos^2\alpha_y\right]\frac{\tan\alpha_x}{\cos\alpha_y} \\ &= -(x_L - x_a)\tan\alpha_y + (y_L - y_a)\frac{\tan\alpha_x}{\cos\alpha_y} \ . \end{aligned} \tag{A.8}$$

Now we need to get rid of the world coordinates by replacing the 3-D world coordinates x_L and y_L with their 2-D chip projections using the projection equations (2.17) and (2.18)

$$\begin{aligned} z_L &= -\left(\frac{z_L u_{Lx}}{f} - x_a\right)\tan\alpha_y + \left(\frac{z_L u_{Ly}}{f} - y_a\right)\frac{\tan\alpha_x}{\cos\alpha_y} + z_a \\ &= z_L\left(\frac{u_{Ly}}{f}\frac{\tan\alpha_x}{\cos\alpha_y} - \frac{u_{Lx}}{f}\tan\alpha_y\right) + x_a\tan\alpha_y - y_a\frac{\tan\alpha_x}{\cos\alpha_y} + z_a \end{aligned}$$

This is rearranged by bringing all z_L terms to the left side of the equation

$$z_L\left(\frac{u_{Lx}}{f}\tan\alpha_y - \frac{u_{Ly}}{f}\frac{\tan\alpha_x}{\cos\alpha_y} + 1\right) = x_a\tan\alpha_y - y_a\frac{\tan\alpha_x}{\cos\alpha_y} + z_a \ .$$

Then both sides are multiplied by $\cos\alpha_y$

$$\begin{aligned} &z_L\left(\frac{u_{Lx}}{f}\sin\alpha_y - \frac{u_{Ly}}{f}\tan\alpha_x + \cos\alpha_y\right) = \\ &\qquad\qquad x_a\sin\alpha_y - y_a\tan\alpha_x + z_a\cos\alpha_y \ , \end{aligned}$$

and multiplied by f

$$z_L \left(u_{Lx} \sin \alpha_y - u_{Ly} \tan \alpha_x + f \cos \alpha_y\right) = \\ f \left(x_a \sin \alpha_y - y_a \tan \alpha_x + z_a \cos \alpha_y\right) \ .$$

Rearranging the equation above finally leads to

$$z_L = f \frac{x_a \sin \alpha_y - y_a \tan \alpha_x + z_a \cos \alpha_y}{u_{Lx} \sin \alpha_y - u_{Ly} \tan \alpha_x + f \cos \alpha_y} \ . \tag{A.9}$$

With this we have an equation that describes z_L in terms of the parameters of a planar model. Substituting z_L in the basic mapping equation (2.19) leads to

$$u_{Rx} = u_{Lx} - b \frac{u_{Lx} \sin \alpha_y - u_{Ly} \tan \alpha_x + f \cos \alpha_y}{x_a \sin \alpha_y - y_a \tan \alpha_x + z_a \cos \alpha_y} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \ . \tag{A.10}$$

These equations allow for a mapping of the view of a plane from the left camera to the right camera by means of the planar parameters (z_a, α_x and α_y). The values for x_a and y_a can be chosen arbitrarily. They just define at which position the depth z_a of the planar model is estimated. Please note that the mapping equation (A.10) for the planar model correspond to the well-known homography transformation.

A.2. Spherical Model

As in section A.1, we need to formulate z_L as a function of the parametric model. A sphere in the three dimensional space can be formulated as

$$r^2 = \left(x_L - x_a\right)^2 + \left(y_L - y_a\right)^2 + \left(z_L - z_a\right)^2 \ , \tag{A.11}$$

where (x_a, y_a, z_a) is the anchor point (center) of the sphere and r its radius. For a better understanding see Fig. A.2. First, we need to reorganize the sphere formulation above

$$\left(z_L - z_a\right)^2 = r^2 - \left(x_L - x_a\right)^2 - \left(y_L - y_a\right)^2 \ . \tag{A.12}$$

A closer look reveals that with the equation above one can only calculate the depth for points with $r^2 \geq \left(x - x_a\right)^2 + \left(y - y_a\right)^2$, which is only

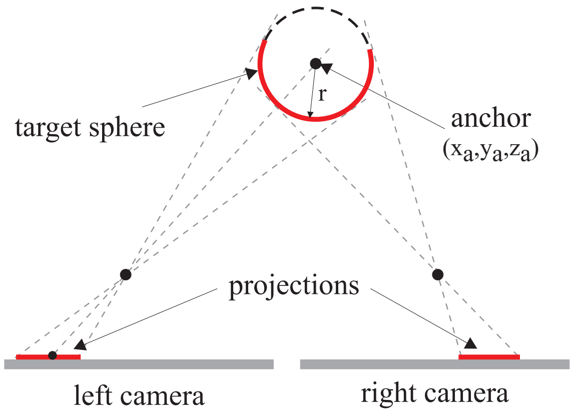


Figure A.2.: Top-view of the projection of a spherical or cylindrical surface to the image planes of a parallel stereo camera. The spherical model is described by its center point. For a cylindrical model also the orientation with respect to the image planes have to be taken into account.

true for points that lie within the sphere or on its surface. For all other points the value for z_L gets complex, so we later neglect those pixels during the mapping process.

Before we can proceed, we need to get rid of the world coordinates in the sphere formulation (A.12) by replacing the 3-D world coordinates of x_L and y_L with their 2-D chip projections using the projection equations (2.17) and (2.18)

$$(z_L - z_a)^2 = r^2 - \left(\frac{z_L u_{Lx}}{f} - x_a\right)^2 - \left(\frac{z_L u_{Ly}}{f} - y_a\right)^2 \tag{A.13}$$

As we need to rearrange for z_L, we have to expand the formula above

$$z_L^2 - 2z_L z_a + z_a^2 = r^2 - \left(\frac{u_{Lx}^2 z_L^2}{f^2} - 2\frac{u_{Lx} z_L}{f} x_a + x_a^2\right) - \left(\frac{u_{Ly}^2 z_L^2}{f^2} - 2\frac{u_{Ly} z_L}{f} y_L + y_a^2\right)$$

$$\underbrace{z_L^2}_{1} \underbrace{-2z_Lz_a}_{2} \underbrace{+z_a^2}_{3} = \underbrace{r^2}_{3} \underbrace{-\frac{u_{Lx}^2 z_L^2}{f^2}}_{1} \underbrace{+2\frac{u_{Lx}z_L}{f}x_a}_{2} \underbrace{-x_a^2}_{3}$$
$$\underbrace{-\frac{u_{Ly}^2 z_L^2}{f^2}}_{1} \underbrace{+2\frac{u_{Ly}z_L}{f}y_L}_{2} \underbrace{-y_a^2}_{3} .$$

As we are dealing with a quadratic formulation, we have to group the terms into z_L^2, z_L and the constants, which has already been indicated by the curly braces 1, 2 and 3

$$\begin{aligned} 0 &= \underbrace{z_L^2\left(1+\frac{u_{Lx}^2}{f^2}+\frac{u_{Ly}^2}{f^2}\right)}_{1} - \\ &\quad \underbrace{2z_L\left(z_a+\frac{u_{Lx}}{f}x_a+\frac{u_{Ly}}{f}y_L\right)}_{2} + \\ &\quad \underbrace{\left(z_a^2-r^2+x_a^2+y_a^2\right)}_{3} . \end{aligned}$$

In order to solve the quadratic equation above we have to bring it to the normal form. For convenience the substitutions λ, μ and ν are used

$$\begin{aligned} 0 &= z_L^2\underbrace{\left(1+\frac{u_{Lx}^2+u_{Ly}^2}{f^2}\right)}_{\lambda} - \\ &\quad 2z_L\underbrace{\left(z_a+\frac{u_{Lx}x_a+u_{Ly}y_L}{f}\right)}_{\mu} + \\ &\quad \underbrace{\left(x_a^2+y_a^2+z_a^2-r^2\right)}_{\nu} \end{aligned}$$

with

$$\lambda = 1 + \frac{u_{Lx}^2 + u_{Ly}^2}{f^2} \tag{A.14}$$

$$\mu = z_a + \frac{u_{Lx}x_a + u_{Ly}y_L}{f} \tag{A.15}$$

$$\nu = x_a^2 + y_a^2 + z_a^2 - r^2 \ . \tag{A.16}$$

This leads to a compact form which reads as

$$0 = z_L^2\lambda - 2z_L\mu + \nu \tag{A.17}$$

$$0 = z_L^2 - 2z_L\frac{\mu}{\lambda} + \frac{\nu}{\lambda} \tag{A.18}$$

Having the normal form and the substitutions we can now determine z_L

$$z_{L1,2} = -\frac{1}{2}\frac{-2\mu}{\lambda} \pm \sqrt{\frac{1}{4}\left(\frac{-2\mu}{\lambda}\right)^2 - \frac{\nu}{\lambda}}$$

$$z_{L1,2} = \frac{\mu}{\lambda} \pm \sqrt{\frac{\mu^2 - \nu\lambda}{\lambda^2}}$$

$$z_{L1,2} = \frac{\mu \pm \sqrt{\mu^2 - \nu\lambda}}{\lambda} \ . \tag{A.19}$$

At a first glance having two solutions in the spherical depth equation (A.19) looks puzzling. In fact, a closer look at Fig. A.2 reveals that using the "−" in the spherical depth equation (A.19) means mapping a sphere (convex structure) and using the "+" means mapping a bowl (concave structure). Therefore, substituting z_L in the basic mapping equation (2.19) with the spherical depth equation (A.19) leads to two transformation equations. The first is the equation for transforming the view of a sphere

$$\mathbf{u}_R = \mathbf{u}_L - \frac{bf\lambda}{\mu - \sqrt{\mu^2 - \nu\lambda}} \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \tag{A.20}$$

and the second for transforming the view of a bowl

$$\mathbf{u}_R = \mathbf{u}_L - \frac{bf\lambda}{\mu + \sqrt{\mu^2 - \nu\lambda}} \begin{pmatrix} 1 \\ 0 \end{pmatrix} . \tag{A.21}$$

These equations allow for a mapping of the view of a sphere or a bowl from the left camera to the right camera by means of the spherical model parameters $(z_a,\, x_a,\, y_a,\, r)$.

A.3. Cylindric Model

For the derivation of the formulas for the cylindrical model we follow the same scheme like for the planar and spherical model. The setup of the cylindrical model is very similar to that of the sphere (see Fig. A.2). In this work, the cylindrical model is described by

$$r^2 = (x' - x_a)^2 + (z' - z_a)^2 \ , \tag{A.22}$$

This means the cylindrical model is infinite in the y-direction. In contrast to the spherical model, it is necessary to incorporate a rotation matrix as it has been done for the planar model

$$\mathbf{T} = \begin{pmatrix} \cos\alpha_z & -\sin\alpha_z & 0 \\ \cos\alpha_x \sin\alpha_z & \cos\alpha_x \cos\alpha_z & -\sin\alpha_x \\ \sin\alpha_x \sin\alpha_z & \sin\alpha_x \cos\alpha_z & \cos\alpha_x \end{pmatrix} . \tag{A.23}$$

This means that like the planar model parameters the cylindrical angular parameters are with respect to a fronto-parallel cylinder. For the cylindrical model, there is only a rotation about the x-axis and the z-axis because a rotation about the y-axis would have no effect on the cylindric model. This leads to six parameters for the model of the cylinder with anchor point $(x_a,\, y_a,\, z_a)$, rotation angles $(\alpha_x,\, \alpha_z)$ and radius r. But actually the model has only five parameters because the y-position for the infinitely expanded cylinder can be fixed.

Using the rotation matrix $\mathbf{T}$, points $\mathbf{x}'_L$ from the fronto-parallel cylinder are transformed to points $\mathbf{x}_L$ on the rotated cylinder by

$$\mathbf{x} = \mathbf{T}\left[\mathbf{x}' - \mathbf{x}_a\right] - \mathbf{x}_a \ . \tag{A.24}$$

Please note, that for convenience the index L is skipped for the rest of the derivation. Substituting the rotation matrix $\mathbf{T}$ in the transformation equation (A.24) leads to

$$x = (x' - x_a) * \cos\alpha_z - (y' - y_a) * \sin\alpha_z + x_a \tag{A.25}$$

$$\begin{aligned} y = {} & (x' - x_a) * \cos\alpha_x \sin\alpha_z + (y' - y_a) * \cos\alpha_x \cos\alpha_z - \\ & (z' - z_a) * \sin\alpha_x + y_a \end{aligned} \tag{A.26}$$

$$\begin{aligned} z = {} & (x' - x_a) * \sin\alpha_x \sin\alpha_z + (y' - y_a) * \sin\alpha_x \cos\alpha_z + \\ & (z' - z_a) * \cos\alpha_x + z_a \end{aligned} \tag{A.27}$$

In a next step, we have to get rid of the points x' from the fronto-parallel cylinder. Thereto, (A.26) is multiplied with $\frac{\sin\alpha_x}{\cos\alpha_x}$ and subtracted from (A.27),

$$\begin{aligned} z - y\frac{\sin\alpha_x}{\cos\alpha_x} &= (z' - z_a)\left[\cos\alpha_x + \frac{\sin^2\alpha_x}{\cos\alpha_x}\right] + \\ & \quad z_a - y_a\frac{\sin\alpha_x}{\cos\alpha_x} \qquad \Big| \cdot \cos\alpha_x \\ z\cos\alpha_x - y\sin\alpha_x &= (z' - z_a)\left[\cos^2\alpha_x + \sin^2\alpha_x\right] + \\ & \quad z_a\cos\alpha_x - y_a\sin\alpha_x \ . \end{aligned} \tag{A.28}$$

Rearranging for $(z' - z_a)$ leads to

$$(z' - z_a) = (z - z_a)\cos\alpha_x - (y - y_a)\sin\alpha_x \ . \tag{A.29}$$

Furthermore, (A.25) is multiplied with $\frac{\cos\alpha_x \cos\alpha_z}{\sin\alpha_z}$ and added to (A.26)

$$\begin{aligned} y + x\frac{\cos\alpha_x\cos\alpha_z}{\sin\alpha_z} = {} & (x' - x_a)\cos\alpha_x\sin\alpha_z + \\ & (x' - x_a)\frac{\cos\alpha_x\cos^2\alpha_z}{\sin\alpha_z} - \\ & (z' - z_a)\sin\alpha_x + \qquad \Big| \cdot \frac{\sin\alpha_z}{\cos\alpha_x} \\ & x_a\frac{\cos\alpha_x\cos\alpha_z}{\sin\alpha_z} + \\ & y_a \end{aligned}$$

$$\begin{aligned} y\frac{\sin\alpha_z}{\cos\alpha_x} + x\cos\alpha_z = (x' - x_a)\left[\sin^2\alpha_z + \cos^2\alpha_z\right] - \\ (z' - z_a)\frac{\sin\alpha_x \sin\alpha_z}{\cos\alpha_x} + \\ x_a\cos\alpha_z + y_a\frac{\sin\alpha_z}{\cos\alpha_x} \;\; . \end{aligned} \tag{A.30}$$

Rearranging for $(x' - x_a)$ leads to

$$\begin{aligned} (x' - x_a) = (x - x_a)\cos\alpha_z + (y - y_a)\frac{\sin\alpha_z}{\cos\alpha_x} + \\ (z' - z_a)\frac{\sin\alpha_x \sin\alpha_z}{\cos\alpha_x} \;\; . \end{aligned} \tag{A.31}$$

Now we have three equations that describe the cylindrical model (A.22) and its rotation (A.29), (A.31). These three equations have to be merged in order to get rid of the points $\mathbf{x}'$ from the fronto-parallel cylinder. First, $(z' - z_a)$ in (A.31) is substituted using (A.29)

$$\begin{aligned} (x' - x_a) = (x - x_a)\cos\alpha_z + (y - y_a)\frac{\sin\alpha_z}{\cos\alpha_x} + \\ \left[(z - z_a)\cos\alpha_x - (y - y_a)\sin\alpha_x\right]\frac{\sin\alpha_x \sin\alpha_z}{\cos\alpha_x} \end{aligned} \tag{A.32}$$

$$\begin{aligned} (x' - x_a) = (x - x_a)\cos\alpha_z + (y - y_a)\frac{\sin\alpha_z}{\cos\alpha_x} + \\ (z - z_a)\sin\alpha_x \sin\alpha_z - (y - y_a)\frac{\sin^2\alpha_x \sin\alpha_z}{\cos\alpha_x} \end{aligned} \tag{A.33}$$

$$\begin{aligned} (x' - x_a) = (x - x_a)\cos\alpha_z + (y - y_a)\frac{\sin\alpha_z}{\cos\alpha_x}\left[1 - \sin^2\alpha_x\right] + \\ (z - z_a)\sin\alpha_x \sin\alpha_z \end{aligned} \tag{A.34}$$

$$\begin{aligned} (x' - x_a) = (x - x_a)\cos\alpha_z + (y - y_a)\sin\alpha_z\cos\alpha_x + \\ (z - z_a)\sin\alpha_x \sin\alpha_z \;\; . \end{aligned} \tag{A.35}$$

Before we can proceed with the substitution the equations for $(x' - x_a)$ (A.35) and $(z' - z_a)$ (A.29) have to be squared. For $(z' - z_a)$ this results in

$$\begin{aligned}(z' - z_a)^2 &= [(z - z_a)\cos\alpha_x - (y - y_a)\sin\alpha_x]^2 &\text{(A.36)}\\ &= (z - z_a)^2\cos^2\alpha_x - \\ &\quad 2(z - z_a)(y - y_a)\sin\alpha_x\cos\alpha_x + &\text{(A.37)}\\ &\quad (y - y_a)^2\sin^2\alpha_x\ ,\end{aligned}$$

and for $(x' - x_a)$ in

$$\begin{aligned}(x' - x_a)^2 &= [(x - x_a)\cos\alpha_z + (y - y_a)\sin\alpha_z\cos\alpha_x + \\ &\qquad (z - z_a)\sin\alpha_x\sin\alpha_z]^2 &\text{(A.38)}\\ &= (x - x_a)^2\cos^2\alpha_z + (y - y_a)^2\sin^2\alpha_z\cos^2\alpha_x + \\ &\quad (z - z_a)^2\sin^2\alpha_x\sin^2\alpha_z + \\ &\quad 2(x - x_a)(z - z_a)\sin\alpha_x\sin\alpha_z\cos\alpha_z + &\text{(A.39)}\\ &\quad 2(x - x_a)(y - y_a)\sin\alpha_z\cos\alpha_x\cos\alpha_z + \\ &\quad 2(y - y_a)(z - z_a)\sin\alpha_x\sin^2\alpha_z\cos\alpha_x\ .\end{aligned}$$

If the reader is still with me (s)he should sit down, because now the ugly part begins. We have to substitute the squared equations for $(x' - x_a)^2$ and $(z' - z_a)^2$ into the model equation (A.22). But before let us rearrange (A.22) a bit

$$r^2 = (x' - x_a)^2 + (z' - z_a)^2 \tag{A.40}$$

$$0 = (x' - x_a)^2 + (z' - z_a)^2 - r^2 \tag{A.41}$$

Substituting the equations for $(x' - x_a)^2$ and $(z' - z_a)^2$ brings us to

$$\begin{aligned}
0 = {} & (z - z_a)^2 \cos^2 \alpha_x - \\
& 2(z - z_a)(y - y_a) \sin \alpha_x \cos \alpha_x + \\
& (y - y_a)^2 \sin^2 \alpha_x + \\
& (x - x_a)^2 \cos^2 \alpha_z + \\
& (y - y_a)^2 \sin^2 \alpha_z \cos^2 \alpha_x + \\
& (z - z_a)^2 \sin^2 \alpha_x \sin^2 \alpha_z + \\
& 2(x - x_a)(z - z_a) \sin \alpha_x \sin \alpha_z \cos \alpha_z + \\
& 2(x - x_a)(y - y_a) \sin \alpha_z \cos \alpha_x \cos \alpha_z + \\
& 2(y - y_a)(z - z_a) \sin \alpha_x \sin^2 \alpha_z \cos \alpha_x - \\
& r^2
\end{aligned} \tag{A.42}$$

$$\begin{aligned}
0 = {} & (x - x_a)^2 \cos^2 \alpha_z + \\
& (y - y_a)^2 \left[\sin^2 \alpha_x + \sin^2 \alpha_z \cos^2 \alpha_x\right] + \\
& (z - z_a)^2 \left[\cos^2 \alpha_x + \sin^2 \alpha_x \sin^2 \alpha_z\right] + \\
& 2(x - x_a)(y - y_a) \sin \alpha_z \cos \alpha_x \cos \alpha_z + \\
& 2(z - z_a)(x - x_a) \sin \alpha_x \sin \alpha_z \cos \alpha_z + \\
& 2(z - z_a)(y - y_a) \left[\sin \alpha_x \cos \alpha_x \sin^2 \alpha_z - \sin \alpha_x \cos \alpha_x\right] - \\
& r^2 \ .
\end{aligned} \tag{A.43}$$

In order to ease the understanding of the upcoming steps, the following substitutions are introduced

$$A = \cos^2 \alpha_z \tag{A.44}$$

$$B = \sin^2 \alpha_x + \sin^2 \alpha_z \cos^2 \alpha_x = 1 - \cos^2 \alpha_x \cos^2 \alpha_z \tag{A.45}$$

$$C = \sin \alpha_z \cos \alpha_x \cos \alpha_z \tag{A.46}$$

$$D = \sin \alpha_x \sin \alpha_z \cos \alpha_z \tag{A.47}$$

$$\begin{aligned}
E &= \sin \alpha_x \cos \alpha_x \sin^2 \alpha_z - \sin \alpha_x \cos \alpha_x \\
&= - \sin \alpha_x \cos \alpha_x \cos^2 \alpha_z
\end{aligned} \tag{A.48}$$

$$F = \cos^2 \alpha_x + \sin^2 \alpha_x \sin^2 \alpha_z = 1 - \sin^2 \alpha_x \cos^2 \alpha_z \ . \tag{A.49}$$

This way the large blown-up cylindric model equation (A.43) becomes

$$\begin{aligned}0 = {} & (x-x_a)^2 A + \rightarrow ① \\ & (y-y_a)^2 B + \rightarrow ② \\ & (z-z_a)^2 F + \rightarrow ③ \\ & 2(y-y_a)(z-z_a)C + \rightarrow ④ \\ & 2(z-z_a)(x-x_a)D + \rightarrow ⑤ \\ & 2(z-z_a)(y-y_a)E - \rightarrow ⑥ \\ & r^2 \ .\end{aligned} \tag{A.50}$$

As it has been done for the planar and spherical model, we now have to remove the world coordinates in the cylinder formulation (A.50) by replacing them with their 2-D chip projections using the projection equations (2.17) and (2.18). Furthermore, we have to multiply out the brackets so that we can rearrange for z. As the equation (A.50) is still quite large, each term is handled separately (indicated by the circled numbers).

$$① \quad (x-x_a)^2 = x^2 - 2xx_a + x_a^2 = \frac{u_{Lx}^2 z^2}{f^2} - 2\frac{u_{Lx} z x_a}{f} + x_a^2 \tag{A.51}$$

$$② \quad (y-y_a)^2 = y^2 - 2yy_a + y_a^2 = \frac{u_{Ly}^2 z^2}{f^2} - 2\frac{u_{Ly} z y_a}{f} + y_a^2 \tag{A.52}$$

$$③ \quad (z-z_a)^2 = z^2 - 2zz_a + z_a^2 \tag{A.53}$$

$$④ \quad 2(x-x_a)(y-y_a) = 2\left(xy - xy_a - yx_a + x_a y_a\right) \tag{A.54}$$

$$= 2\left(\frac{u_{Lx} u_{Ly} z^2}{f^2} - \frac{u_{Lx} z y_a}{f} - \frac{u_{Ly} z x_a}{f} + x_a y_a\right) \tag{A.55}$$

⑤
$$2(z-z_a)(x-x_a) = 2\left(zx - zx_a - xz_a + z_ax_a\right) \tag{A.56}$$
$$= 2\left(\frac{u_{Lx}z^2}{f} - zx_a - \frac{u_{Lx}zz_a}{f} + z_ax_a\right) \tag{A.57}$$

⑥
$$2(z-z_a)(y-y_a) = 2\left(zy - zy_a - yz_a + z_ay_a\right) \tag{A.58}$$
$$= 2\left(\frac{u_{Ly}z^2}{f} - zy_a - \frac{u_{Ly}zz_a}{f} + z_ay_a\right) \tag{A.59}$$

As we will have to solve a quadratic equation in normal form, we first have to sort the terms ①-⑥ for z^2, z^1 and z^0.

$$\begin{aligned} z^2 : \quad & \frac{u_{Lx}^2z^2}{f^2}A + \frac{u_{Ly}^2z^2}{f^2}B + z^2F + \\ & \frac{2u_{Lx}u_{Ly}z^2}{f^2}C + \frac{2u_{Lx}z^2}{f}D + \frac{2u_{Ly}z^2}{f}E \end{aligned} \tag{A.60}$$

$$\begin{aligned} z^1 : \quad & -2\frac{u_{Lx}zx_a}{f}A - 2\frac{u_{Ly}zy_a}{f}B - 2zz_aF - \\ & \frac{2u_{Lx}zy_a}{f}C - \frac{2u_{Ly}zx_a}{f}C - 2zx_aD - \frac{2u_{Lx}zz_a}{f}D - \\ & 2zy_aE - \frac{2u_{Ly}zz_a}{f}E \end{aligned} \tag{A.61}$$

$$z^0 : \quad x_a^2A + y_a^2B + z_a^2F + 2x_ay_aC + 2z_ax_aD + 2z_ay_aE - r^2 \; . \tag{A.62}$$

For these sorted terms the following substitutes are introduced

$$\kappa = u_{Lx}^2\frac{A}{f^2} + u_{Ly}^2\frac{B}{f^2} + 2u_{Lx}u_{Ly}\frac{C}{f^2} + \frac{2}{f}(u_{Lx}D + u_{Ly}E) + F \tag{A.63}$$

$$\begin{aligned} \tau = {} & u_{Lx}x_a\frac{A}{f} + u_{Ly}y_a\frac{B}{f} + (u_{Lx}y_a + u_{Ly}x_a)\frac{C}{f} + \\ & \frac{z_a}{f}(u_{Lx}D + u_{Ly}E) + x_aD + y_aE + z_aF \end{aligned} \tag{A.64}$$

$$\eta = x_a^2A + y_a^2B + 2x_ay_aC + 2z_a(x_aD + y_aE) + z_a^2F - r^2 \; . \tag{A.65}$$

Altogether this leads to

$$0 = \kappa z^2 - 2\tau z + \eta \tag{A.66}$$

$$= z^2 - 2\frac{\tau}{\kappa} z + \frac{\eta}{\kappa} \; . \tag{A.67}$$

The resulting depth formula has a characteristic similar to that of the sphere

$$z_{L1,2} = \frac{\tau}{\kappa} \pm \sqrt{\frac{\tau^2}{\kappa^2} - \frac{\eta}{\kappa}} \tag{A.68}$$

$$= \frac{\tau}{\kappa} \pm \sqrt{\frac{\tau^2 - \eta\kappa}{\kappa^2}} \tag{A.69}$$

$$= \frac{\tau \pm \sqrt{\tau^2 - \eta\kappa}}{\kappa} \; . \tag{A.70}$$

As it was the case for the sphere, substituting z_L in the basic mapping equation (2.19) with the cylindrical depth equation (A.70) leads to two transformation equations

$$\mathbf{u}_R = \mathbf{u}_L - \frac{bf\kappa}{\tau \pm \sqrt{\tau^2 - \eta\kappa}} \begin{pmatrix} 1 \\ 0 \end{pmatrix} . \tag{A.71}$$

These equations allow for a mapping of the view of a cylindrical shape from the left camera to the right camera by means of the cylindrical model parameters (x_a, y_a, z_a, α_x, α_z, r). As was pointed out in section A.2, using "−" corresponds to map convex structures and using "+" corresponds to map concave structures.

B. Results on Common Stereo Scenes

In the following, disparity maps calculated with SNCC block-matching stereo (introduced in chapter 3) on common benchmark stereo images are shown. In all cases the algorithm explained in section 3.3.1 is used. Furthermore, for all images the parameters are the same except for the search range. The parameters are a filter size of 3x3 for the first stage (NCC), a filter size of 11x11 for the second stage (summation) and all disparity regions smaller than 200 pixels are regarded as invalid and filled in the subsequent fill-in step.

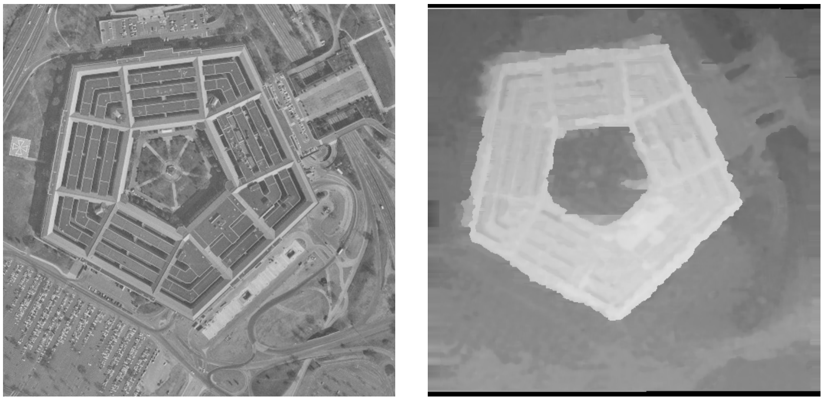


Figure B.1.: Resulting disparity map of block-matching stereo with SNCC on the standard test scene Pentagon.

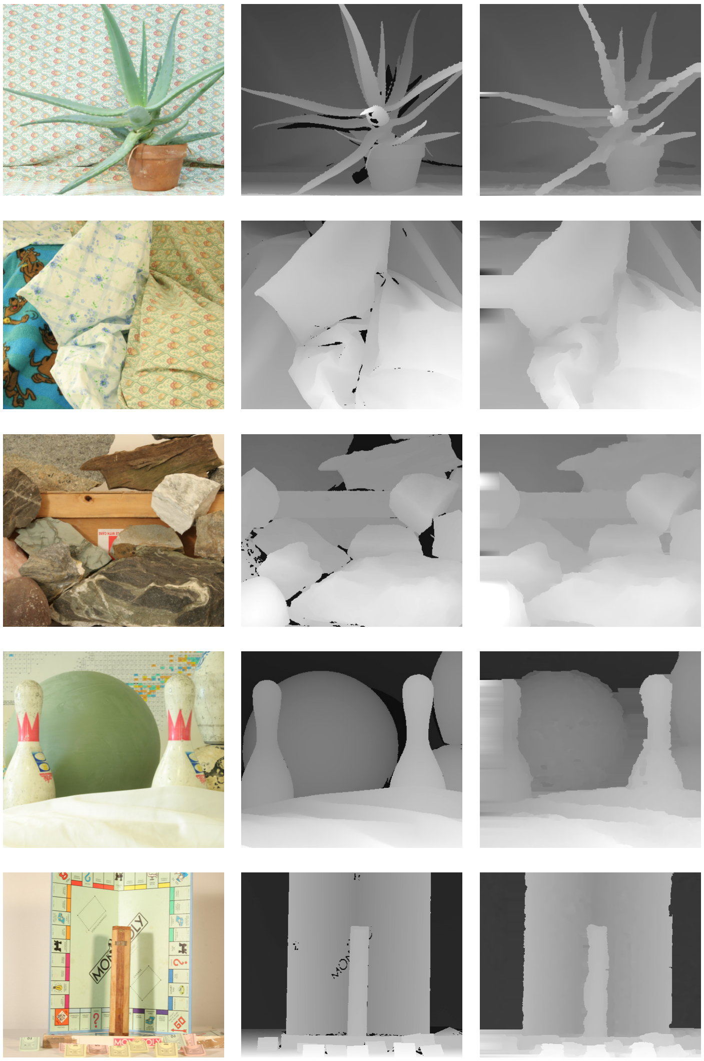


Figure B.2.: Results of SNCC block-matching stereo on the 2006 Middlebury scenes Aloe, Cloth2, Rocks1, Bowling1 and Monopoly. The left column shows the left stereo images, the middle column the ground truth disparity maps and the right column the results of SNCC. The disparities are encoded by intensity. Near pixels are bright and far pixels are dark.

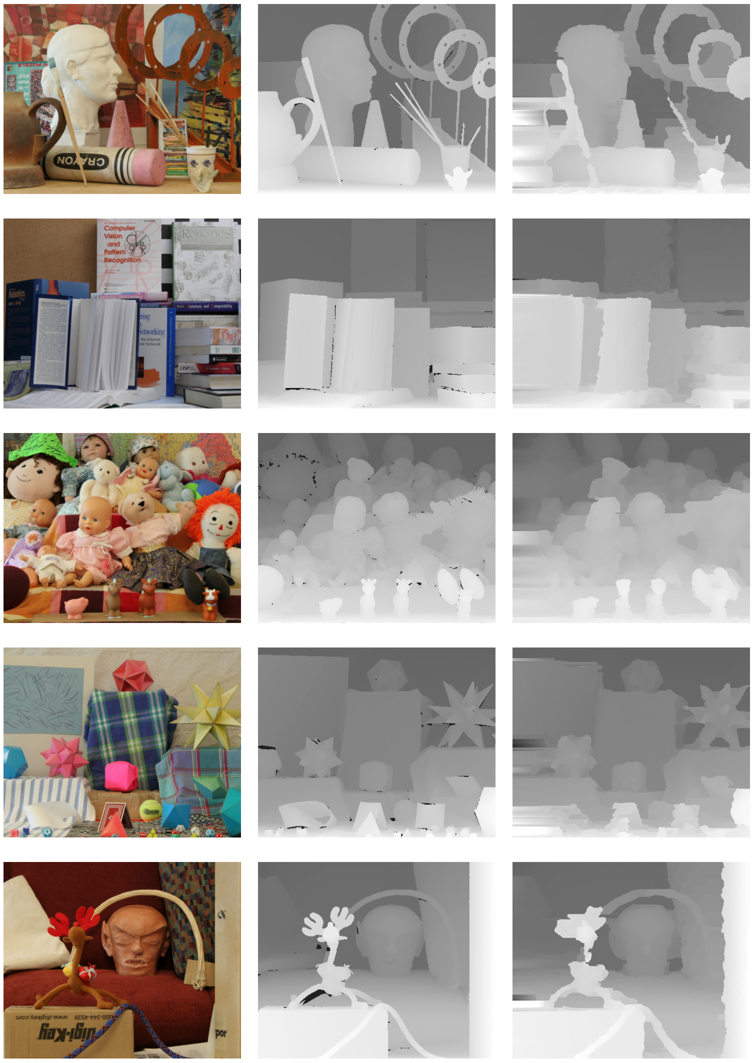


Figure B.3.: Results of SNCC block-matching stereo on the 2005 Middlebury scenes Art, Books, Dolls, Moebius and Reindeer. The left column shows the left stereo images, the middle column the ground truth disparity maps and the right column the results of SNCC. The disparities are encoded by intensity. Near pixels are bright and far pixels are dark.

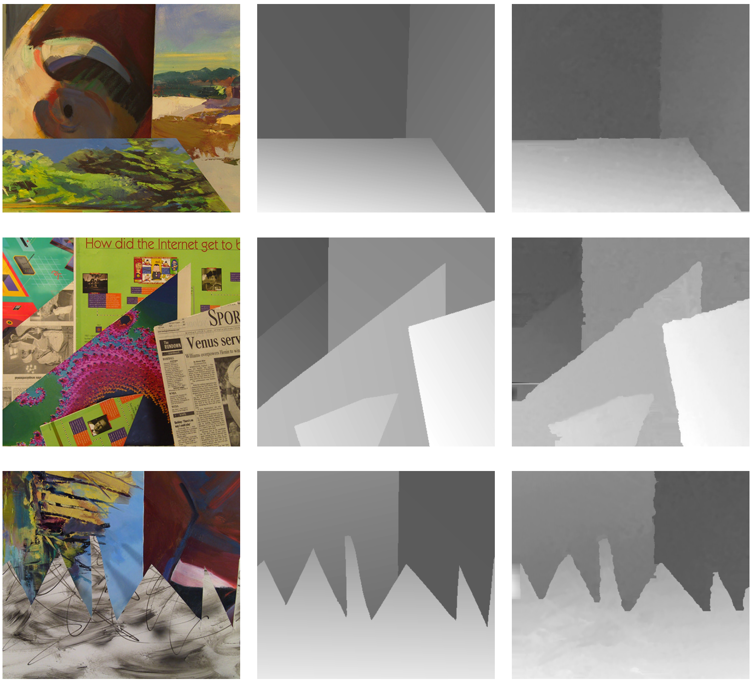
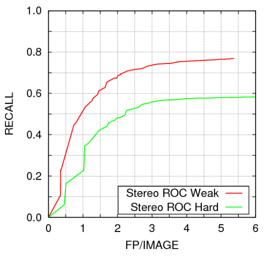


Figure B.4.: Results of SNCC block-matching stereo on the 2001 Middlebury scenes Bull, Poster and Sawtooth. The left column shows the left stereo images, the middle column the ground truth disparity maps and the right column the results of SNCC. The disparity values are encoded by means of intensity. Near pixels are bright and far pixels are dark.

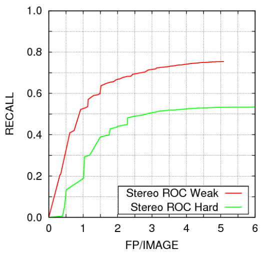
C. Further Analyses of Stereoscopic Car Detection

In section 5.1 of chapter 5 an analysis of the detection performance of the introduced stereo popout and car classifier is conducted. This analysis is done by means of ROC curves which compare the false positive rate per image against the recall. The detection rates are calculated by comparing the detections of the cues with labels that were generated by hand. The criterion for a correct match is the mutual overlap between the detector responses and the labeled rectangles of cars. For a true positive the mutual (areal) overlap between a detection and a label has to be 50%. Otherwise, the detection counts as a false positive or the label as a false negative. It has been discussed that this criterion is very difficult for the stereo popout because the stereo popout is a generic cue and thus has problems to give clearly separated detections of cars only. In particular, the stereo popout tends to merge horizontally adjacent cars into one detection, leading to an increase of both false positive and false negative rate. On the other hand, such merged detections are sufficient for several tasks like collision avoidance or scene analysis.

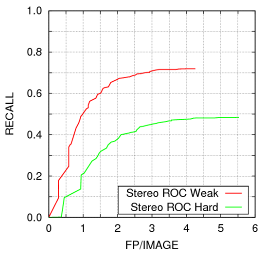
In order to show the potential of the stereo popout Fig. C.1 shows the detection performance of the stereo popout using the strong mutual overlap criterion and a weaker criterion based on center points. Here a detection is considered correct when the center point of the detected area lies within the area of a labeled car region. Thus, a merged detection is not longer considered a false positive. However, in most cases one of the two labeled regions related to the merged detection is still considered as a false negative because the center point of a merged detection usually only falls into one of the two regions. Comparing the results with the weak criterion to the results with the strong criterion shows that the performance of the stereo popout is dramatically increased in the weak case. This demonstrates that the stereo popout in principle detects much more cars than suggested by the strict criterion



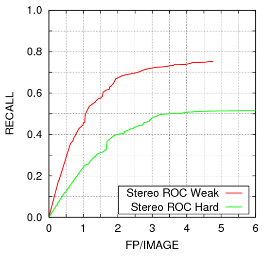
(a) Overcast Day



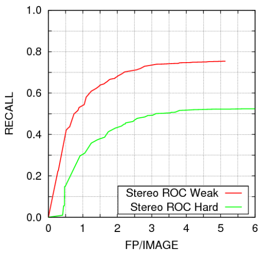
(b) Sunny Day



(c) Night



(d) Rain

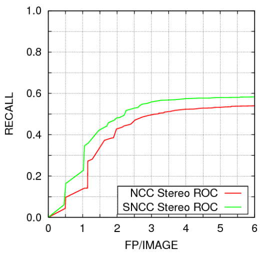


(e) Snow

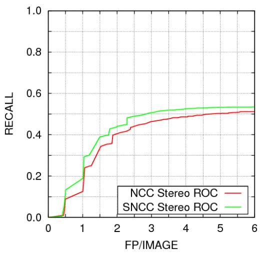
Figure C.1.: Car detection performance of the stereo popout cue using the hard mutual overlap criterion from section 5.1 and a weaker criterion which demands only the center point of a detection to be within a labeled car region. The weaker criterion demonstrates that the stereo popout in principle is able to find much more cars than what is suggested by the strong criterion but that the stereo popout often fails in delivering a clear segmentation or separation of the detection results.

but it fails in clearly separating them. However, the weak criterion does not account for false positives caused by the detection of other traffic participants. This can only be resolved by extending the labeled car regions with labeled regions of the other traffic participants.

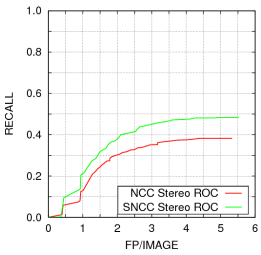
Another thing that is worth noting here, is that the stereo popout shows a consistently higher performance when the novel cost function *summed normalized cross-correlation* (SNCC) (see chapter 3) is used for the block-matching stereo calculations as compared to the standard normalized cross-correlation (NCC). To highlight this Fig. C.2 compares the car detection performance of the stereo popout for the two cases of the block-matching stereo module using SNCC or NCC. For this comparison the hard mutual overlap criterion is used. The results



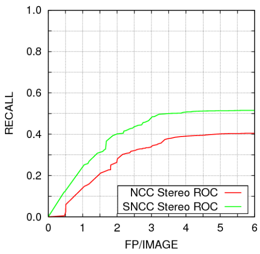
(a) Overcast Day



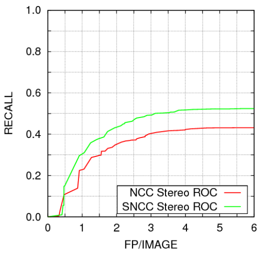
(b) Sunny Day



(c) Night



(d) Rain



(e) Snow

Figure C.2.: Comparison of the car detection performance of the stereo popout using SNCC and NCC block-matching stereo for computing the disparity maps. It is clearly observable that for all tested situations SNCC is superior to NCC. This highlights that the novel cost function SNCC, introduced in chapter 3, makes block-matching stereo also more accurate and robust in real-world applications. For this comparison the hard mutual overlap criterion was used.

show that in most situations SNCC increases the recall rate of the stereo popout by 0.1 as compared to NCC. This proves that the higher performance of SNCC observed on the benchmark scenes in chapter 3 transfers well to the real-world domain. Moreover, the good performance in the night and rain streams shows the high robustness of the SNCC cost function.

Bibliography

Amdahl, G. M. (1967). Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18-20, 1967, spring joint computer conference*, pages 483–485. ACM.

Anzai, A. and DeAngelis, G. C. (2010). Neural computations underlying depth perception. *Current Opinion in Neurobiology*, 20(3):367–375.

Banno, A. and Ikeuchi, K. (2009). Disparity map refinement and 3D surface smoothing via directed anisotropic diffusion. In *Proceedings of the 2009 IEEE International Workshop on 3-D Digital Imaging and Modeling (3DIM)*, pages 1870–1877.

Barnard, S. T. and Fischler, M. A. (1982). Computational stereo. *ACM Computing Surveys*, 14(4):553–572.

Barry, S. R. (2009). *Fixing My Gaze: A Scientist's Journey into Seeing in Three Dimensions.* Basic Books.

Bellman, R. (1952). On the theory of dynamic programming. In *Proceedings of the National Academy of Sciences*, volume 38, pages 716–719.

Berenson, D., Diankov, R., Nishiwaki, K., Kagami, S., and Kuffner, J. (2007). Grasp planning in complex scenes. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots (Humanoids).*

Blanchet, G., Buades, A., Coll, B., Morel, J., and Rouge, B. (2011). Fattening free block matching. *Journal of Mathematical Imaging and Vision*, 41(1):109–121.

Bleyer, M. and Chambon, S. (2010). Does color really help in dense stereo matching? In *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, pages 1–8.

Bolder, B., Dunn, M., Gienger, M., Janssen, H., Sugiura, H., and Goerick, C. (2007). Visually guided whole body interaction. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3054–3061.

Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137.

Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239.

Brown, M. Z., Burschka, D., and Hager, G. D. (2003). Advances in computational stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):993–1008.

Burschka, D., Lee, S., and Hager, G. (2002). Stereo-based obstacle avoidance in indoor environments with active sensor recalibration. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2066–2072.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698.

Cormen, T. H., Rivest, R. L., Leiserson, C. E., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press, 3rd edition.

Corso, J. J., Burschka, D., and Hager, G. D. (2003). Direct plane tracking in stereo images for mobile navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 875–880.

Crow, F. (1984). Summed-area tables for texture mapping. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 207–212.

Duda, R. O. and Hart, P. E. (1972). Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15.

Einecke, N. and Eggert, J. (2010a). Evaluation of direct plane fitting for depth and parameter estimation. In *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 492–497.

Einecke, N. and Eggert, J. (2010b). A two-stage correlation method for stereoscopic depth estimation. In *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 227–234.

Einecke, N., Mühlig, M., Schmüdderich, J., and Gienger, M. (2011). "Bring it to me" - Generation of behavior-relevant scene elements for interactive robot scenarios. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3415–3422.

Einecke, N., Rebhan, S., Willert, V., and Eggert, J. (2010). Direct surface fitting. In *Proceedings of the Fifth International Conference on Computer Vision Theory and Applications (VISAPP)*, pages 125–133.

Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57.

Ess, A., Leibe, B., Schindler, K., and Gool, L. V. (2008). A mobile vision system for robust multi-person tracking. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Falkenhagen, L. (2001). *Blockbasierte Disparitätsschätzung unter Berücksichtigung statistischer Abhängigkeiten der Disparitäten*. Number 657 in Fortschritt-Berichte VDI, Reihe 10: Informatik / Kommunikation. VDI Verlag GmbH.

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

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Forstmann, S., Kanou, Y., Ohya, J., Thuering, S., and Schmitt, A. (2004). Real-time stereo by using dynamic programming. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop*, pages 29–36.

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

Gehrig, S. K., Eberli, F., and Meyer, T. (2009). A real-time low-power stereo vision engine using semi-global matching. In *Proceedings of the 7th International Conference on Computer Vision Systems (ICVS)*, pages 134–143.

Georgieva, S., Peeters, R., Kolster, H., Todd, J. T., and Orban, G. A. (2009). The processing of three-dimensional shape from disparity in the human brain. *Journal of Neuroscience*, 29(3):727–742.

Gienger, M., Mühlig, M., and Steil, J. J. (2010a). Imitating object movement skills with robots - a task-level approach exploiting generalization and invariance. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1262–1269.

Gienger, M., Toussaint, M., and Goerick, C. (2010b). Whole body motion planning - building blocks for intelligent systems. In *Motion Planning for Humanoid Robots*. Springer, first edition.

Gong, M. and Yang, Y.-H. (2005). Near real-time reliable stereo matching using programmable graphics hardware. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 924–931.

Groß, H.-M., Böhme, H., Schröter, C., Müller, S., König, A., Einhorn, E., Martin, C., Merten, M., and Bley, A. (2009). TOOMAS: interactive shopping guide robots in everyday use - final implementation and experiences from long-term field trials. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2005–2012.

Guo, C., Mita, S., and McAllester, D. (2009). Drivable road region detection using homography estimation and efficient belief propagation with coordinate descent optimization. In *Proceedings of the 2009 IEEE Intelligent Vehicles Symposium (IV)*, pages 317–323.

Habbecke, M. and Kobbelt, L. (2005). Iterative multi-view plane fitting. In *Proceedings of the 11th International Fall Workshop Vision, Modeling, Visualization (VMV)*, pages 73–80.

Haller, I., Pantilie, C., Oniga, F., and Nedevschi, S. (2010). Real-time semi-global dense stereo solution with improved sub-pixel accuracy. In *Proceedings of the 2010 IEEE Intelligent Vehicles Symposium (IV)*, pages 369–376.

Hartley, R. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition.

Heo, Y. S., Lee, K. M., and Lee, S. U. (2008). Illumination and camera invariant stereo matching. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Hinkle, D. A. and Connor, C. E. (2002). Three-dimensional orientation tuning in macaque area V4. *Nature Neuroscience*, 5(7):665–670.

Hirose, M., Haikawa, Y., Takenaka, T., and Hirai, K. (2001). Development of humanoid robot Asimo. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) – Workshop 2*.

Hirschmüller, H. (2005). Accurate and efficient stereo processing by semi-global matching and mutual information. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages II: 807–814.

Hirschmüller, H. and Scharstein, D. (2009). Evaluation of stereo matching costs on images with radiometric differences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1582–1599.

Hirschmüller, H. and Scharstein, D. (2007). Evaluation of cost functions for stereo matching. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Hooke, R. and Jeeves, T. A. (1961). "Direct Search" Solution of Numerical and Statistical Problems. *Journal of the Association for Computing Machinery*, 8(2):212–229.

Hough, P. V. C. (1962). Method and means for recognizing complex patterns. U.S. Patent 3 069 654.

Howard, T. and Tappin, S. (2008). Three-dimensional reconstruction of two solar coronal mass ejections using the STEREO spacecraft. *Solar Physics*, 252:373–383.

Hu, Z., Lamosa, F., and Uchimura, K. (2005). A complete u-v-disparity study for stereovision based 3d driving environment analysis. In *Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM)*, pages 204–211.

Huebner, K., Welke, K., Przybylski, M., Vahrenkamp, N., Asfour, T., Kragic, D., and Dillmann, R. (2009). Grasping known objects with humanoid robots: A box-based approach. In *Proceedings of the International Conference on Advanced Robotics (ICAR)*, pages 1–6.

Humenberger, M., Zinner, C., Weber, M., Kubinger, W., and Vincze, M. (2010). A fast stereo matching algorithm suitable for embedded real-time systems. *Computer Vision and Image Understanding*, 114(11):1180–1202.

Janssen, P., Vogels, R., and Orban, G. A. (2000). Three-dimensional shape coding in inferior temporal cortex. *Neuron*, 27(2):385–397.

Kaiser, M., Kucera, T., Davila, J., Cyr, O. S., Guhathakurta, M., and Christian, E. (2008). The STEREO mission: An introduction. *Space Science Reviews*, 136:5–16.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME – Journal of Basic Engineering*, 82(1):35–45.

Kolmogorov, V. and Zabih, R. (2001). Computing visual correspondence with occlusions using graph cuts. In *Proceedings ot the International Conference on Computer Vision (ICCV)*, pages 508–515.

Kormann, B., Neve, A., Klinker, G., and Stechele, W. (2010). Stereo vision based vehicle detection. In *Proceedings of the Fifth International Conference on Computer Vision Theory and Applications (VISAPP)*, pages 431–438.

Labayrade, R., Aubert, D., and Tarel, J.-P. (2002). Real time obstacle detection on non flat road geometry through 'v-disparity' representation. In *Proceedings of the IEEE Intelligent Vehicle Symposium (IV)*, pages 646–651.

Leibe, B., Cornelis, N., Cornelis, K., and Gool, L. V. (2007). Dynamic 3d scene analysis from a moving vehicle. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Leibe, B., Seemann, E., and Schiele, B. (2005). Pedestrian detection in crowded scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 878–885.

Lewis, R. M., Torczon, V., and Trosset, M. W. (2000). Direct search methods: Then and now. *Journal of Computational and Applied Mathematics*, 124:191–207.

Longuet-Higgins, H. C. (1981). A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135.

Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679.

Ma, Y., Soatto, S., Kosecka, J., and Sastry, S. S. (2003). *An Invitation to 3-D Vision: From Images to Geometric Models.* Springer Verlag.

Marr, D. and Poggio, T. (1976). Cooperative computation of stereo disparity. *Science*, 194(4262):283–287.

Mattoccia, S., Giardino, S., and Gambini, A. (2009). Accurate and efficient cost aggregation strategy for stereo correspondence based on approximated joint bilateral filtering. In *Proceedings of the 9th Asian Conference on Computer Vision (ACCV)*, pages II: 371–380.

McDonnell, M. J. (1981). Box-filtering techniques. *Computer Graphics and Image Processing*, 17(1):65–70.

Michalke, T., Kastner, R., Herbert, M., Fritsch, J., and Goerick, C. (2009). Adaptive multi-cue fusion for robust detection of unmarked inner-city streets. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 1–8.

Miyazaki, D., Matsushita, Y., and Ikeuchi, K. (2009). Interactive shadow removal from a single image using hierarchical graph cut. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 234–245.

Moravec, H. and Elfes, A. (1985). High resolution maps from wide angle sonar. In *Proceedings of the 1985 IEEE International Conference on Robotics and Automation (ICRA)*, pages 116–121.

Mühlig, M., , Gienger, M., and Steil, J. J. (2010). Human-robot interaction for learning and adaptation of object movements. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4901–4907.

Oniga, F., Nedevschi, S., Meinecke, M. M., and To, T. B. (2007). Road surface and obstacle detection based on elevation maps from dense stereo. In *Proceedings of the 10th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 859–865.

Orban, G. A., Janssen, P., and Vogels, R. (2006). Extracting 3D structure from disparity. *TRENDS in Neuroscience*, 29(8):466–473.

Perrollaz, M., Spalanzani, A., and Aubert, D. (2010). Probabilistic representation of the uncertainty of stereo-vision and application to obstacle detection. In *Proceedings of the 2010 IEEE Intelligent Vehicles Symposium (IV)*, pages 313–318.

Porikli, F. (2005). Integral histogram: A fast way to extract histograms in cartesian spaces. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) - Volume 1*, pages 829–836.

Porikli, F. (2008). Constant time O(1) bilateral filtering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8.

Rasolzadeh, B., Björkman, M., Huebner, K., and Kragic, D. (2010). An active vision system for detecting, fixating and manipulating objects in the real world. *International Journal of Robotics Research*, 29(2-3):133–154.

Rebhan, S., Einecke, N., and Eggert, J. (2009). Consistent modeling of functional dependencies along with world knowledge. In *Proceedings of the International Conference on Cognitive Information System Engineering (ICCISE)*.

Rebhan, S., Röhrbein, F., Eggert, J., and Körner, E. (2008). Attention modulation using short- and long-term knowledge. In *Proceedings of the 6th International Conference on Computer Vision Systems (ICVS)*, volume 5008, pages 151–160.

Rodríguez, S. A., Frémont, V., Bonnifait, P., and Cherfaoui, V. (2010). Visual confirmation of mobile objects tracked by a multi-layer lidar. In *Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 849–854.

Roy, S. and Cox, I. J. (1998). A maximum-flow formulation of the n-camera stereo correspondence problem. In *Proceedings of the Sixth International Conference on Computer Vision (ICCV)*, pages 492–499.

Rusu, R. B., Sucan, I. A., Gerkey, B. P., Chitta, S., Beetz, M., and Kavraki, L. E. (2009). Real-time perception-guided motion planning for a personal robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4245–4252.

Salmen, J., Schlipsing, M., Edelbrunner, J., Hegemann, S., and Lüke, S. (2009). Real-time stereo vision: Making more out of dynamic programming. In *Proceedings of the 13th International Conference on Computer Analysis of Images and Patterns (CAIP)*, pages 1096–1103.

Sandini, G., Metta, G., and Vernon, D. (2007). The iCub cognitive humanoid robot: an open-system research platform for enactive cognition. In *50 Years of Artificial Intelligence*, pages 358–369. Springer-Verlag.

Saxena, A., Wong, L., Quigley, M., and Ng., A. Y. (2007). A vision-based system for grasping novel objects in cluttered environments. In *Prceedings ot the International Symposium of Robotics Research (ISRR).*

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42.

Scharstein, D. and Szeliski, R. (2003). High-accuracy stereo depth maps using structured light. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 195–202, Madison, WI,.

Schmüdderich, J., Brandl, H., Bolder, B., Heracles, M., Janssen, H., Mikhailova, I., and Goerick, C. (2008). Organizing multimodal perception for autonomous learning and interactive systems. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 312–319.

Schmüdderich, J. (2010). *Multimodal Learning of Grounded Concepts in Embodied Systems.* Berichte aus der Robotik. Shaker Verlag GmbH, Germany.

Schmüdderich, J., Einecke, N., Hasler, S., Gepperth, A., Bolder, B., Kastner, R., Franzius, M., Rebhan, S., Dittes, B., Wersing, H., Eggert, J., Fritsch, J., and Goerick, C. (2010). System approach for multi-purpose representations of traffic scene elements. In *Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1677–1684.

Seki, A. and Okutomi, M. (2006). Robust obstacle detection in general road environment based on road extraction and pose estimation. In *Proceedings of the 2006 IEEE Intelligent Vehicles Symposium (IV)*, pages 437–444.

Shibata, T., Vijayakumar, S., Conradt, J., and Schaal, S. (2001). Humanoid oculomotor control based on concepts of computational neuroscience. In *Proceedings of the Second IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 278–285.

Srivastava, S., Orban, G. A., DeMazière, P. A., and Janssen, P. (2009). A distinct representation of three-dimensional shape in macaque anterior intraparietal area: Fast, metric, and coarse. *Neuroscience*, 29(34):10613–10626.

Suganuma, N. (2009). Clustering and tracking of obstacles from virtual disparity image. In *Proceedings of the 2009 IEEE Intelligent Vehicles Symposium (IV)*, pages 111–116.

Sun, C. (1997). A fast stereo matching method. In *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 95–100.

Tomasi, C. and Manduchi, R. (1998). Bilateral filtering for gray and color images. In *Proceedings of the Sixth International Conference on Computer Vision (ICCV)*, pages 839–846.

Toussaint, M., Gienger, M., and Goerick, C. (2007). Optimization of sequential attractor-based movement for compact movement representation. In *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots (Humanoids)*, pages 122 – 129.

Tsai, D.-M. and Lin, C.-T. (2003). Fast normalized cross correlation for defect detection. *Pattern Recognition Letters*, 24(15):2625–2631.

Viola, P. A. and Jones, M. J. (2004). Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154.

Wang, L., Liao, M., Gong, M., Yang, R., and Nister, D. (2006). High-quality real-time stereo using adaptive cost aggregation and dynamic programming. In *Proceedings of the Third International Symposium* *on 3D Data Processing, Visualization, and Transmission (3DPVT)*, pages 798–805.

Wedel, A., Franke, U., Badino, H., and Cremers, D. (2008). B-spline modeling of road surfaces for freespace estimation. In *Proceedings ot the 2008 IEEE Intelligent Vehicles Symposium (IV)*, pages 828–833.

Wersing, H., Kirstein, S., Goetting, M., Brandl, H., Dunn, M., Mikhailova, I., Goerick, C., Steil, J. J., Ritter, H., and Körner, E. (2007). Online learning of objects in a biologically motivated visual architecture. *International Journal of Neural Systems*, 17(4):219–230.

Wersing, H., Kirstein, S., Schneiders, B., Bauer-Wersing, U., and Körner, E. (2008). Online learning for bootstrapping of object recognition and localization in a biologically motivated architecture. In *Proceedings of the IEEE International Conference on Computer Vision Systems (ICVS)*, pages 383–392.

Wersing, H. and Körner, E. (2003). Learning optimized features for hierarchical models of invariant object recognition. *Neural Computation*, 15(2):1559–1588.

Yang, Q., Tan, K.-H., and Ahuja, N. (2009). Real-time O(1) bilateral filtering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 557–564.

Yang, Q., Wang, L., Yang, R., Wang, S., Liao, M., and Nistér, D. (2006). Real-time global stereo matching using hierarchical belief propagation. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 989–998.

Yoon, K.-J. and Kweon, I. S. (2006). Adaptive support-weight approach for correspondence search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:650–656.

Zabih, R. and Woodfill, J. (1994). Non-parametric local transforms for computing visual correspondence. In *Proceedings of Third European Conference on Computer Vision (ECCV)*, pages 151–158.

Zhang, A. X. and Tay, A. L. P. (2006). Vergence control of 2 DOF pan-tilt binocular cameras using a log-polar representation of the visual cortex. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 4277–4283.

Zhang, K., Lu, J. B., Lafruit, G., Lauwereins, R., and Gool, L. V. (2009). Real-time accurate stereo with bitwise fast voting on CUDA. In *Proceedings of the 12th IEEE International Conference on Computer Vision (ICCV) Workshops*, pages 794–800.