Institute of Visualization and Interactive Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit Nr. 3

Depth-Driven Variational Methods for Stereo Reconstruction

Daniel Maurer

Course of Study: Informatik

Examiner: Prof. Dr-Ing. Andrés Bruhn

Supervisor: Prof. Dr-Ing. Andrés Bruhn

Commenced: 2014-04-22

Completed: 2014-10-22

CR-Classification: I.2.10, I.4.8

ABSTRACT

Stereo reconstruction belongs to the fundamental problems in computer vision, with the aim of reconstructing the depth of a static scene. In order to solve this problem the corresponding pixels in both views must be found. A common technique is to minimize an energy (cost) function. Therefore, most methods use a parameterization in form of a displacement information (disparity). In contrast, this thesis uses, extends and examines a depth parameterization. (i) First a basic depth-driven variational method is developed based on a recently presented method of Basha et al. [2]. (ii) After that, several possible extensions are presented, in order to improve the developed method. These extensions include advanced smoothness terms that incorporate image information and enable an anisotropic smoothing behavior. Further advanced data terms are considered, which use modified constraints to allow a more accurate estimation in different situations. (iii) Finally, all extensions are compared with each other and with a disparity-driven counterpart.

ZUSAMMENFASSUNG

Die Stereorekonstruktion zählt zu den grundlegenden Problemen des Maschinensehens (Computer Vision), mit dem Ziel die Tiefeninformation einer statischen Szene zu rekonstruieren. Um dieses Problem zu lösen, müssen die zueinandergehörigen Bildpunkte in beiden Ansichten identifiziert werden (Korrespondenzfindung). Ein häufig verwendeter Ansatz beinhaltet die Minimierung eines Energiefunktionals (Kostenfunktional). Dabei verwenden die meisten Ansätze eine Parametrisierung in Form von Verschiebungsinformation (Disparität). Im Unterschied dazu verwendet, erweitert und untersucht diese Arbeit eine Tiefen-Parametrisierung. (i) Zuerst wird ein tiefengetriebener Variationsansatz auf Basis einer kürzlich vorgestellten Arbeit von Basha et al. [2] entwickelt. (ii) Danach werden mehrere mögliche Erweiterungen präsentiert, um das entwickelte Verfahren zu verbessern. Diese Erweiterungen beinhalten fortgeschrittene Glattheitsterme, die Bildinformationen miteinbeziehen und ein anisotropes Glättungsverhalten ermöglichen. Zudem werden fortgeschrittene Datenterme, mit veränderten Annahmen, betrachtet, um eine genauere Schätzung zu ermöglichen. (iii) Zum Schluss werden alle Erweiterungen miteinander und mit einem disparitätsgetriebenen Äquivalent verglichen.

ACKNOWLEDGMENTS

At this point I would like to thank everyone who has supported me in writing this thesis.

First and foremost I would like to thank Professor Andrés Bruhn for his constant support and valuable suggestions. I also thank Sebastian Volz for providing me with example code and explanations with it. I thank Yong Chul Ju for being part of some helpful discussions. Another thanks goes to my fellow student Kai Mindermann for proofreading.

Further I would like to thank my girlfriend Damaris Rasch for her help, love and delicious food. Last but not least, I want to thank my family for their permanent help and support that made my studies possible.

Thanks!

CONTENTS

1 INTRODUCTION

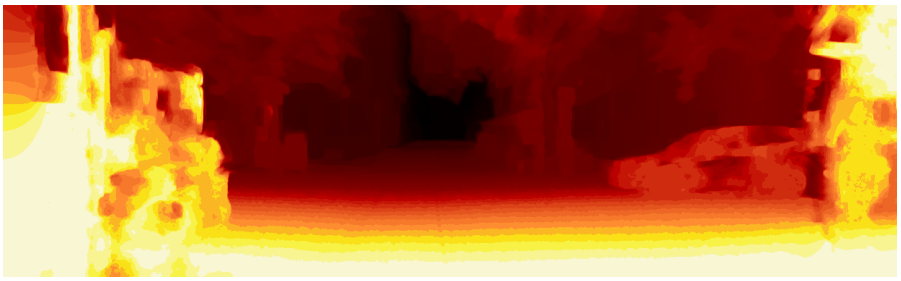
1.1 MOTIVATION

A classical and central problem in computer vision is the reconstruction of depth in a static scene. Given two images of a scene captured from two different viewpoints this problem is known as stereo reconstruction. Stereo reconstruction makes use of the fact that the offset of objects in both views is related to the depth. It can be divided in two main steps. First, corresponding pixels in both views must be identified. In a second step the depth is computed by triangulation of the correspondences. As the triangulation is simple, with known camera calibration and corresponding pixels, the main challenge remains the first step - solving the correspondence problem.

In order to compute a very precise solution especially global optimization approaches have been established in literature. These methods determine all corresponding pixels simultaneously, as the minimizer of a suitable energy functional. Therefore, most of the methods use a parameterization in the form of displacement information, known as the disparity. Even though such approaches have the advantage of being very similar to motion estimation methods and therefore facilitate similar concepts, they also have a number of disadvantages. In particular the computation is performed in the 2-D image space and not directly in the 3-D space, consequently it is difficult to make use of prior knowledge. Moreover disparity-driven methods cannot be easily extended to a setting that uses more than two cameras, due to the fact that for arbitrary number of cameras the geometric relations become arbitrarily complicated. This more-general problem is known as multi-view stereo. Depth-driven approaches provide an elegant solution to this problem. They model the problem in 3-D space and calculate the depth directly.

FIELDS OF APPLICATION. Stereo reconstruction and the obtained 3-D structure allows to accomplish a broad variety of tasks. For example, it allows to determine the distances to objects, which is a very useful information, that allows to find a safe path through an unknown environment or to localize the own position in a known map. It offers the foundation for fully automatically *robot navigation* [18] or the design of *driver assistance systems* [14], which support the driver in critical situations. Figure 1 shows a scene taken in a pedestrian area and the reconstructed color-coded depth information.

(a) One image of the stereo pair showing the scene.



(b) Reconstructed color-coded depth information.

Figure 1: *A possible field of application.* Showing a reconstruction of a scene from the KITTI benchmark dataset [10].

Another application is found in the post-processing of modern *Camera-Arrays* [26]. Here the knowledge of the depth is needed to correct for parallax and allow to produce correct images.

1.2 RELATED WORK

First attempts of depth-driven approaches go back to the work of Robert and Deriche [15]. They were the first to use such a parameterization, but without mentioning details of the challenging minimization. New approaches such as the method of Basha et al. [2] use such a parameterization with an additional motion estimation (scene flow). Furthermore, they offer insights into a suitable minimization, which is based on the well-known nested fixed-point iteration method of Brox et al. [4].
Stühmer et al. [21] combine such an approach with recent camera tracking algorithms to achieve a dense reconstruction in real-time from a single handheld camera, where the easy extension to multiple images allows to increase the robustness of the estimated depth.

1.3 AIM OF THIS THESIS

The aim of this thesis is to develop a simple depth-driven stereo approach using two cameras based on the method of Basha et al. [2]. Further the developed stereo approach shall be improved by the use of advanced data and smoothness terms. In addition, the thesis aims

for a comparison between the developed variational depth-driven method and a variational disparity-driven approach.

1.4 ORGANIZATION

This thesis is organized as follows: Chapter 2 introduces basic concepts and notations used throughout this thesis, which allow to build the basic method in the following chapter (Chapter 3). After that, several extensions are presented in Chapter 4, in order to improve the developed method. Finally, the method and the extensions are evaluated and compared in Chapter 5 and a conclusion is given in the last chapter (Chapter 6).

2 BACKGROUND

This chapter introduces basic concepts and notations that are used throughout this thesis. Most of the following geometry-related concepts and examples can be found with more extensive explanations in [11].

Due to the fact that the developed depth-driven approach assumes that the imaging systems follows the basic pinhole model, it forms the main part of this chapter. In addition, a short introduction to homogeneous coordinates, the epipolar geometry and the calculus of variations is given.

2.1 GEOMETRY-RELATED BACKGROUND

2.1.1 *Homogeneous Coordinates*

Homogeneous coordinates were invented by Möbius (1827) and Plücker (1830) [19].

Homogeneous coordinates are a nice and simple extension to the Euclidean space $\mathbb{R}^n$, which allow to perform affine transformations through a simple matrix multiplication. An example for such an affine transformation will be shown in the following section. To transform Euclidean coordinates to homogeneous coordinates, vectors $\mathbf{x} \in \mathbb{R}^n$ are expanded to vectors $\tilde{\mathbf{x}} \in \mathbb{R}^{n+1}$ by appending a one, so the forward transformation reads

$$\begin{aligned} H : \mathbb{R}^n &\to \mathbb{R}^{n+1} \\ \mathbf{x} = (x_1, \ldots, x_n)^\top &\mapsto \tilde{\mathbf{x}} = (x_1, \ldots, x_n, 1)^\top \end{aligned} \tag{1}$$

The back transformation is almost as simple. The first n entries are divided by the last entry:

$$\begin{aligned} H^{-1} : \mathbb{R}^{n+1} &\to \mathbb{R}^n \\ \tilde{\mathbf{x}} = (\tilde{x}_1, \ldots, \tilde{x}_{n+1})^\top &\mapsto \mathbf{x} = \left(\frac{\tilde{x}_1}{\tilde{x}_{n+1}}, \ldots, \frac{\tilde{x}_n}{\tilde{x}_{n+1}} \right)^\top \end{aligned} \tag{2}$$

Taking a close look at the back transformation shows that a point in Euclidean coordinates can have several counterparts in homogeneous coordinates. This means that

$$\begin{aligned} H^{-1}(\tilde{\mathbf{x}}) &= H^{-1}(\lambda \cdot \tilde{\mathbf{x}}) \\ H^{-1}\left((\tilde{x}_1, \ldots, \tilde{x}_{n+1})^\top \right) &= H^{-1}\left((\lambda \cdot \tilde{x}_1, \ldots, \lambda \cdot \tilde{x}_{n+1})^\top \right) \end{aligned}$$

denote the same point (for $\lambda \neq 0$). Further there are points in homogeneous coordinates that have no Euclidean counterpart. These are the

homogeneous coordinates holding a zero as last entry, which would produce a division by zero using the back transformation.

$$\tilde{\mathbf{x}} = (\tilde{x}_1, \ldots, \tilde{x}_n, 0)^\top$$

This can be understood as a point at infinity and would mean that a division by zero would result in infinity.

2.1.2 *The Basic Pinhole Model*

"The Worldwide Pinhole Photography Day is held each year the last Sunday of April." pinholeday.org

After introducing the homogeneous coordinates the basic pinhole model is explained. It describes how the Camera C projects a point $\mathbf{P} \in \mathbb{R}^3$ on the image plane $\Omega \subset \mathbb{R}^2$ resulting into the projected point $\mathbf{p} \in \Omega$. The geometry of the basic pinhole model is illustrated in Figure 2.

To keep the model simple, the camera center $\mathbf{C}$ is placed at the origin of the Euclidean 3-space $\mathbb{R}^3$. The image plane is then placed in front of the camera center at the plane $Z = f$, with f being the focal length. The Z axis, in which the camera is pointing, is known as principle axis and the intersection point with the image plane c is named principle point.

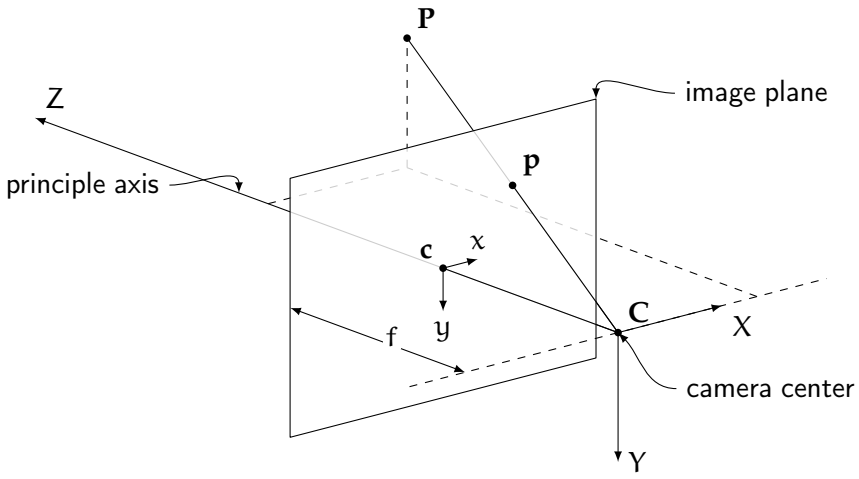


Figure 2: The geometry of the basic pinhole camera.

The projection $\pi : \mathbb{R}^3 \to \Omega, \mathbf{P} \mapsto \mathbf{p}$ can be understood as an intersection of a ray, starting at $\mathbf{P}$ going through $\mathbf{C}$, and the image plane. With the intercept theorem in mind and a close look at Figure 3 follows:

$$\begin{pmatrix} P_X \\ P_Y \\ P_Z \end{pmatrix} \mapsto \begin{pmatrix} f \cdot P_X / P_Z \\ f \cdot P_Y / P_Z \end{pmatrix} \tag{3}$$

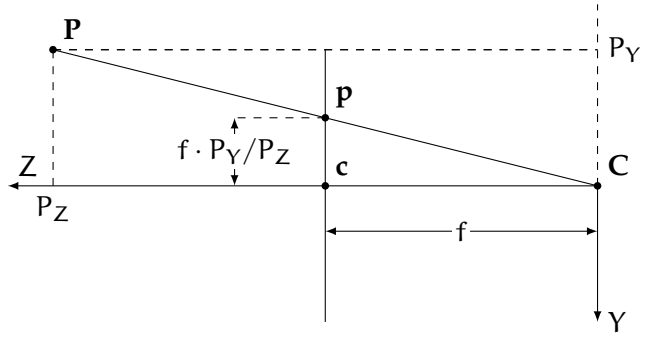


Figure 3: The geometry of the basic pinhole camera.

By making use of homogeneous coordinates this projection can be expressed as a linear mapping. This is why Equation 3 may be written as a matrix multiplication:

$$\begin{pmatrix} P_X \\ P_Y \\ P_Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f \cdot P_X \\ f \cdot P_Y \\ P_Z \end{pmatrix} = \underbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{M} \begin{pmatrix} P_X \\ P_Y \\ P_Z \\ 1 \end{pmatrix} \tag{4}$$

The matrix M used here is called the camera projection matrix. It allows the description of the camera projection in a single 3×4 matrix using the homogeneous coordinates given by $\tilde{\mathbf{p}} = M\tilde{\mathbf{P}}$.

As a quick reminder: $\tilde{\mathbf{p}}$ is a homogeneous equivalent to $\mathbf{p}$.

2.1.3 *Two Cameras in one World*

The previous section introduced a complete model of the pinhole camera. Unfortunately, the made assumptions do not allow to have several pinhole cameras within the same world coordinate frame. Apart from that, the model only allows a very specific camera. In order to overcome these limitations, the following assumptions must be discarded:

A. The world and camera coordinate frames are identical.

B. The origin of the image plane coordinates is located at the principle point.

C. The image plane coordinates have equal scales in x and y direction.

The first assumption to be canceled is assumption A which leads to the extrinsic camera parameters.

EXTRINSIC CAMERA PARAMETERS. The relation between two coordinate frames can be expressed by a rotation and a translation. Homogeneous coordinates allow to write this relation in a homogeneous

4×4 matrix. Let $T_{world \to cam}$ denote the forward transformation from the world coordinate frame to the camera coordinate frame, of the form

In this case it is clear that 0 is not a scalar.

$$T_{world \to cam} = \begin{bmatrix} \mathcal{R} & t \\ 0 & 1 \end{bmatrix}$$

Here $\mathcal{R}$ is a 3×3 rotation matrix and t a 3×1 translation vector. A sketch of this is shown in Figure 4. The same matrix also describes the backward coordinate transformation of a point. Given a point $\mathbf{P}^{world}$ relative to the world coordinate frame it is related to the camera coordinate frame by

$$\tilde{\mathbf{P}}^{world} = T_{world \to cam} \tilde{\mathbf{P}}^{cam}$$

Using the inverse transform $T_{cam \to world}$ of $T_{world \to cam}$ which is given by

$$T_{cam \to world} = \begin{bmatrix} \mathcal{R}^{-1} & -\mathcal{R}^{-1} t \\ 0 & 1 \end{bmatrix} \tag{5}$$

allows to set up a more general camera projection matrix. Therefore, the 3-D point $\mathbf{P}^{world}$ in the world coordinate frame is expressed in camera coordinates using the backward coordinate transformation, which leads to

$$M = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathcal{R}^{-1} & -\mathcal{R}^{-1} t \\ 0 & 1 \end{bmatrix} \tag{6}$$

In literature the parameters $R = \mathcal{R}^{-1}$ and $t = -\mathcal{R}^{-1} t$ are called extrinsic or external camera parameters.

After canceling assumption A, assumptions B and C are in line. This results in the intrinsic camera parameters.

INTRINSIC CAMERA PARAMETERS. In practice the origin of the image plane coordinates is not at the principle point. By taking this offset, seen in Figure 5, into acccount the new camera projection matrix reads

$$M = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{7}$$

where $c = (c_x, c_y)^\top$ are the coordinates of the principle point. By introducing the matrix

$$K = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{8}$$

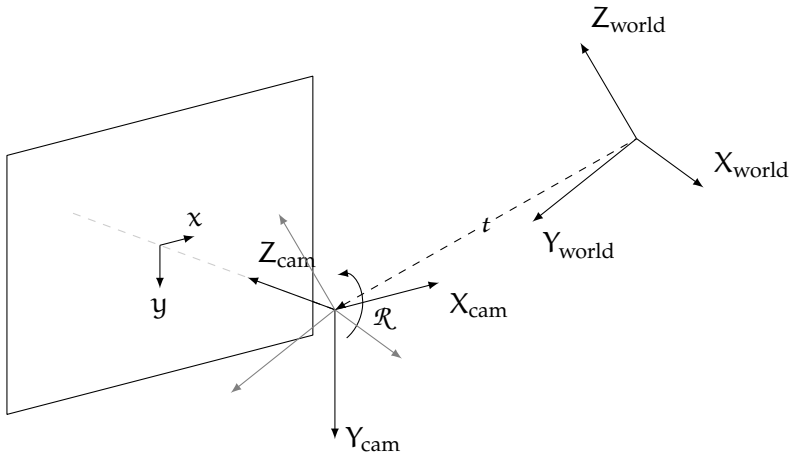


Figure 4: The translation t and rotation $\mathcal{R}$ of the camera coordinate frame relative to the world coordinate frame.

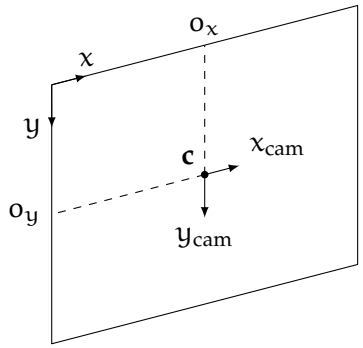


Figure 5: Relation between image and camera coordinate system.

which is called camera calibration matrix, the camera projection matrix is written as

$$M = \begin{bmatrix} K & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{9}$$

The next step is canceling the assumption C, saying that the image plane coordinates have equal scales in x and y direction. Hence a scale factor for each direction is introduced. These scale factors n_x and n_y define the number of pixels per unit distance in image coordinates. Applying this scale to the camera calibration matrix leads to

$$K = \begin{bmatrix} n_x & 0 & 0 \\ 0 & n_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

Now $s_x = f \cdot n_x$ and $s_x = f \cdot n_x$ denote the scaled focal lengths in terms of pixel dimensions and $o = (o_x, o_y)^\top$ the principle point offset in terms of pixel dimensions.

For even more generality a skew parameter can be added, but it is zero for most of the cameras and is not considered in this thesis. The final camera projection matrix M is now a 3×4 matrix of rank 3 and offers 10 degrees of freedom (11 including the not introduced skew factor), defined up to an arbitrary scale. Knowing the projection matrix, this means knowing the intrinsic and the extrinsic parameters, the camera is said to be fully calibrated.

2.1.4 *The Epipolar Geometry*

Sturm names Hauck as the first person to uncover the Epipolar geometry in 1883 [22].

Being capable of describing an arbitrary number of cameras in one world coordinate frame, a first stereo camera setup can be introduced. This will be the so-called converging camera setup, seen in Figure 6 and on the left side of Figure 7. In this arrangement the two cameras are translated along the x-axis and are rotated somewhat towards each other.

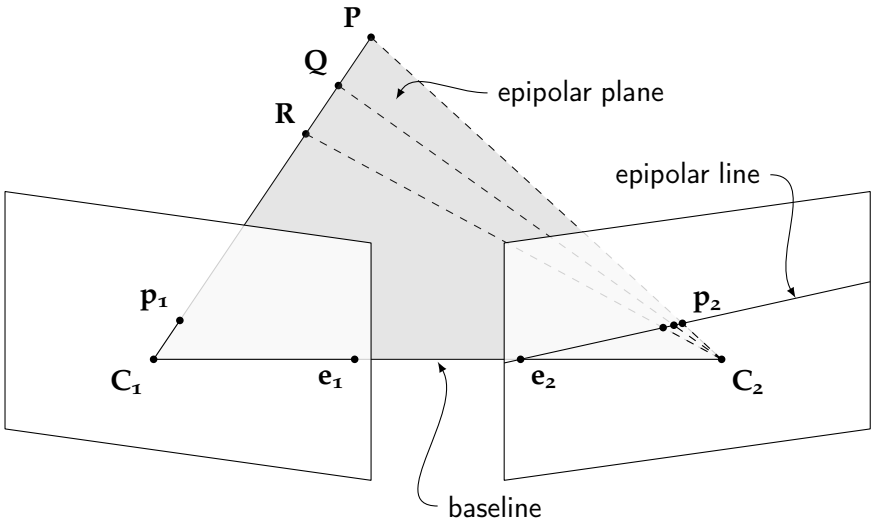


Figure 6: The geometry of a stereo image pair (converging camera setup).

Trying to solve the correspondence problem for the point $\mathbf{p_1}$, taken by the first camera, will lead to the epipolar geometry. The corresponding 3-D point **P** must lie on the optical ray going through the camera center $\mathbf{C_1}$ and the projected point $\mathbf{p_1}$. In consequence a second optical ray going through the second camera center $\mathbf{C_2}$ and the corresponding point $\mathbf{p_2}$ must intersect the first optical ray. Not knowing **P** or $\mathbf{p_2}$ this could be any point on the first optical ray, e.g. **P**, **Q** or **R**. This means the point $\mathbf{p_2}$ must lie on the projection of the first optical ray, which is known as the epipolar line. Obviously there also is a corresponding epipolar line in the first image plane.

Further the line joining the camera centers is called baseline and intersection points with the image plane are named epipoles $\mathbf{e_1}$ an $\mathbf{e_2}$. The plane containing the camera centers and an arbitrary 3-D point is called epipolar plane, the intersection of the epipolar plane and the image planes although results in the epipolar lines.

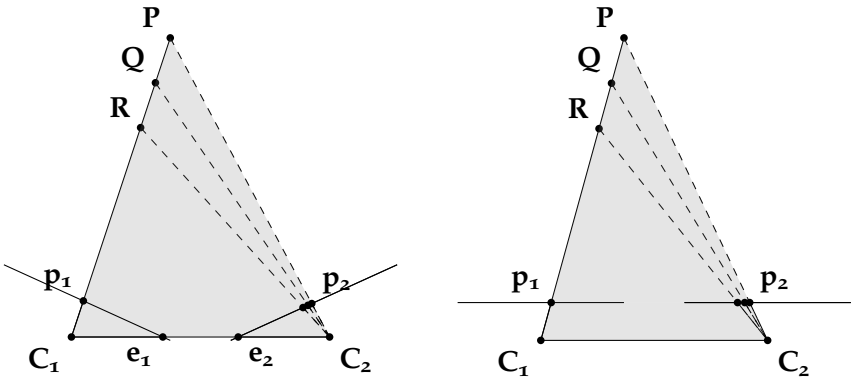


Figure 7: A top view of the converging camera setup (left) and the ortho-parallel camera setup (right).

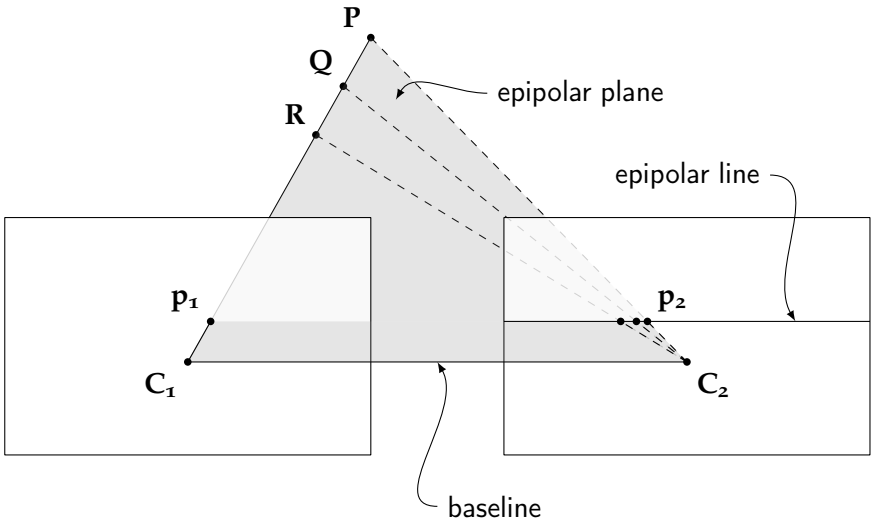


Figure 8: The geometry of a stereo image pair (ortho-parallel camera setup).

Another common used camera setup is the ortho-parallel camera setup, shown in Figure 8 and on the right side of Figure 7. In this setup the cameras have parallel optical axes pointing in the same direction. Which is why the relation can be described with a single translation and without rotation. This setup comes with the advantage that the epipolar lines are horizontal, this means that displacement is only along the X-axis. Due to the fact that the 1-D search space fits perfect to the pixel discretization, there are algorithms that trans-

form images taken with other cameras setups to match the epipolar lines. This procedure is known as image rectification [9].

2.2 CALCULUS OF VARIATIONS

Historically, the calculus of variations originated from concrete problems of geometry and physics. [12]

As aforementioned in the introduction, the correspondence problem is solved as the minimizer of a suitable energy functional. In order to do so, the use of calculus of variations will help. The object of calculus of variations is to find extrema of functionals [12]. A functional is a quantity that depends on a number of functions. Briefly, a functional is a function of functions. A necessary condition for the existence of an extremum is the vanishing of the first variation, which leads to the Euler-Lagrange equation.

2.2.1 *The Euler-Lagrange Equation*

A simple problem is determining the minimizer of the following functional

$$E(y) = \int_a^b F(x, y, y_x)\, dx \tag{11}$$

$C^1[a,b]$ are all continuously differentiable functions defined on the closed interval $[a,b] \in \mathbb{R}$

where $F(x, y, y_x)$ is called the Lagrange-Function and $E(y)$ is defined on $D \subset C^1[a,b]$. Let the function $y(x) \in C^1[a,b]$ be a local minimizer of the functional (11) and the Lagrange-Function $F : [a,b] \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ continuous and continuously differentiable with respect to y and y_x. Then holds

$$F_{y_x}(x, y, y_x) \in C^1[a,b] \tag{12}$$

$$0 = F_y(x, y, y_x) - \frac{d}{dx} F_{y_x}(x, y, y_x) \tag{13}$$

with natural boundary conditions

$$F_{y_x}(a, y(a), y_x(a)) = 0$$
$$F_{y_x}(b, y(b), y_x(b)) = 0$$

A proof to this theorem can be found in [12]. The Equation 13 is known as the Euler-Lagrange-Equation, which each local minimizer y must satisfy. If $E(y)$ is strictly convex, a local minimizer y is the unique minimizer.

2.2.2 *The Case of Several Variables*

The previous simple problem is in 1-D, the problems later in this thesis will be in a 2-D space which is why an extension is needed. According to [7] this generalization is straightforward. For a functional

$$E(u) = \int_\Omega F(x, y, u, u_x, u_y)\, dx dy \tag{14}$$

over a given region Ω the Euler-Lagrange equation reads

$$0 = F_u - \frac{\partial}{\partial x} F_{u_x} - \frac{\partial}{\partial y} F_{u_y} \tag{15}$$

with the natural boundary condition

$$0 = n^\top \begin{pmatrix} F_{u_x} \\ F_{u_y} \end{pmatrix} \tag{16}$$

on the image boundary of $\partial\Omega$ with the normal vector n.

2.2.3 *Functionals with Higher-Order Derivatives*

Another extension of the Euler-Lagrange equation is needed if the Lagrange-Function contains higher order derivatives. This extension is although a straightforward extension as in the 2-D case. These are energy functionals of the following form

$$E(u) = \int_\Omega F\left(x, y, u, u_x, u_y, u_{xx}, u_{xy}, u_{yx}, u_{yy}\right) dx dy \tag{17}$$

The Euler-Lagrange equation reads

$$\begin{aligned} 0 = F_u - \frac{\partial}{\partial x} F_{u_x} - \frac{\partial}{\partial y} F_{u_y} \\ + \frac{\partial^2}{\partial x^2} F_{u_{xx}} + \frac{\partial^2}{\partial x \partial y} F_{u_{xy}} + \frac{\partial^2}{\partial y \partial x} F_{u_{yx}} + \frac{\partial^2}{\partial y y} F_{u_{yy}} \end{aligned} \tag{18}$$

with the natural boundary conditions

$$0 = n^\top \begin{pmatrix} F_{u_x} - \frac{\partial}{\partial x} F_{u_{xx}} - \frac{\partial}{\partial y} F_{u_{xy}} \\ F_{u_y} - \frac{\partial}{\partial x} F_{u_{yx}} - \frac{\partial}{\partial y} F_{u_{yy}} \end{pmatrix} \tag{19}$$

$$0 = n^\top \begin{pmatrix} F_{u_{xx}} \\ F_{u_{xy}} \end{pmatrix}, \quad 0 = n^\top \begin{pmatrix} F_{u_{yx}} \\ F_{u_{yy}} \end{pmatrix} \tag{20}$$

on the image boundary of $\partial\Omega$ with the normal vector n. A derivation of these natural boundary conditions is given in Section A.4.

3

THE METHOD

This chapter will describe the modeling of the simple depth-driven stereo approach using two cameras based on the method of Basha et al. [2]. It is split up in several parts starting with an explanation of the parameterization, followed by the introduction of a variational model. Then a suitable minimization strategy and discretization is given for the proposed energy functional.

3.1 THE PARAMETERIZATION

The depth-driven stereo method uses a 3-D point cloud parameterization, with respect to a reference view. This means the surface is described as a finite size point set $\mathcal{P} \subset \mathbb{R}^3$. Therefore, each pixel in the reference view $\mathbf{p}_0 \in \Omega_0 \subset \mathbb{R}^2$ has a corresponding 3-D unknown $\mathbf{P} \in \mathcal{P}$. The projection of a 3-D point $\mathbf{P}$ onto the image taken by the camera C_i is denoted by $\mathbf{p}_i \in \Omega_i$. Furthermore it is assumed that the cameras follow the previously introduced pinhole model of Section 2.1.2 and Section 2.1.3. For a better understanding a sketch showing a simple scene containing a cube is given in Figure 9.

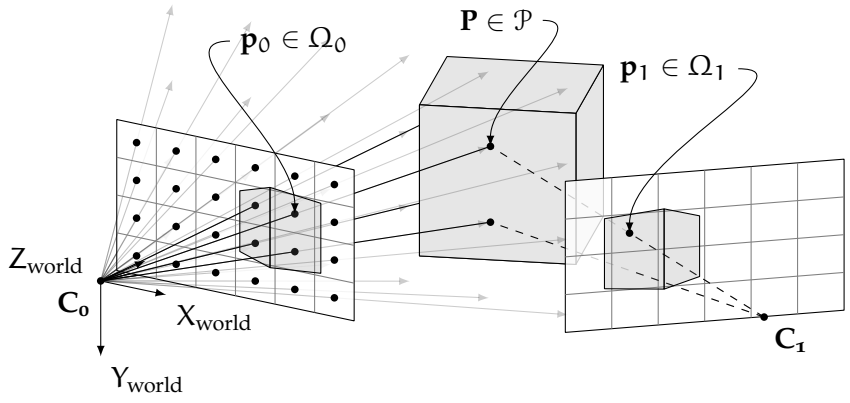


Figure 9: The parameterization of the surface as 3-D point cloud. Each 3-D point lies on an optical ray of the reference camera C_0. Which is aligned with the world coordinate system.

The reference camera coordinate system is assumed to be aligned with the world coordinate system. In this case the X- and Y-coordinate of a point $\mathbf{P}$ can be expressed as a function of Z, since $\mathbf{P}$ must lie on the optical ray through the known camera center and $\mathbf{p}_0$. To find these functions the next step will be a close look at the projection

π_0 of the reference camera. Using the corresponding homogeneous coordinates $\tilde{\mathbf{p}}_0$ and $\tilde{\mathbf{P}}$ it reads

$$\tilde{\mathbf{p}}_0 = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} K & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = M^0 \tilde{\mathbf{P}} \tag{21}$$

Since the reference camera coordinate system is aligned with the world coordinate system the matrix holding the extrinsic parameters becomes the identity. It remains

$$\tilde{\mathbf{p}}_0 = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} s_x & 0 & o_x & 0 \\ 0 & s_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = M^0 \tilde{\mathbf{P}} \tag{22}$$

performing the matrix multiplication results in

$$\tilde{\mathbf{p}}_0 = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} s_x \cdot X + o_x \cdot Z \\ s_y \cdot Y + o_y \cdot Z \\ Z \end{pmatrix} = M^0 \tilde{\mathbf{P}} \tag{23}$$

a back transform to Euclidean coordinates and rearranging the equation leads to

$$\begin{pmatrix} X \\ Y \end{pmatrix} = Z \begin{pmatrix} x/s_x \\ y/s_y \end{pmatrix} - Z \begin{pmatrix} o_x/s_x \\ o_y/s_y \end{pmatrix} \tag{24}$$

Knowing this relationship, Z remains the only unknown of the problem, since X and Y can be determined by Equation 24 as functions of Z and $\mathbf{p}_0$. This even holds if the number of used cameras is increased, which allows an easy extension to additional views. Finally, each 3-D point $\mathbf{P} \in \mathcal{P}$ is given by

$$\mathbf{P} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = Z(x, y) \begin{pmatrix} x/s_x - o_x/s_x \\ y/s_y - o_y/s_y \\ 1 \end{pmatrix} \tag{25}$$

The projection $\mathbf{p}_i = (x_i, y_i)^\top$ of a point $\mathbf{P}$ onto the camera C_i is then computed by applying the camera projection matrix in homogeneous coordinates and back transform these to Euclidean coordinates. This can be written as

$$\mathbf{p}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} = \frac{\left[M^i\right]_{1,2} \tilde{\mathbf{P}}}{\left[M^i\right]_3 \tilde{\mathbf{P}}} \tag{26}$$

where $\left[M^i\right]_{1,2}$ contains the first and the second row of the camera projection matrix of C_i, which is a 2×4 matrix, and $\left[M^i\right]_3$ the third row, which is a 1×4 matrix. Therefore, x_i and y_i are given by

$$\begin{aligned}
x_i &= \frac{[M^i]_1 \tilde{\mathbf{P}}}{[M^i]_3 \tilde{\mathbf{P}}} = \frac{M^i_{11} \cdot X + M^i_{12} \cdot Y + M^i_{13} \cdot Z + M^i_{14} \cdot 1}{M^i_{31} \cdot X + M^i_{32} \cdot Y + M^i_{33} \cdot Z + M^i_{34} \cdot 1} \\
&= \frac{Z \overbrace{\left(M^i_{11} \cdot (x/s_x - o_x/s_x) + M^i_{12} \cdot (y/s_y - o_y/s_y) + M^i_{13}\right)}^{a} + \overbrace{M^i_{14}}^{b}}{Z \underbrace{\left(M^i_{31} \cdot (x/s_x - o_x/s_x) + M^i_{32} \cdot (y/s_y - o_y/s_y) + M^i_{33}\right)}_{c} + \underbrace{M^i_{34}}_{d}} \\
&= \frac{a \cdot Z + b}{c \cdot Z + d} \\
y_i &= \frac{[M^i]_2 \tilde{\mathbf{P}}}{[M^i]_3 \tilde{\mathbf{P}}} = \frac{M^i_{21} \cdot X + M^i_{22} \cdot Y + M^i_{23} \cdot Z + M^i_{24} \cdot 1}{M^i_{31} \cdot X + M^i_{32} \cdot Y + M^i_{33} \cdot Z + M^i_{34} \cdot 1} \\
&= \frac{Z \overbrace{\left(M^i_{21} \cdot (x/s_x - o_x/s_x) + M^i_{22} \cdot (y/s_y - o_y/s_y) + M^i_{23}\right)}^{\check{a}} + \overbrace{M^i_{24}}^{\check{b}}}{Z \underbrace{\left(M^i_{31} \cdot (x/s_x - o_x/s_x) + M^i_{32} \cdot (y/s_y - o_y/s_y) + M^i_{33}\right)}_{c} + \underbrace{M^i_{34}}_{d}} \\
&= \frac{\check{a} \cdot Z + \check{b}}{c \cdot Z + d}
\end{aligned} \tag{27}$$

3.2 THE VARIATIONAL MODEL

Now that the parameterization is clear the next step will be to set up the variational model. Therefore, a energy functional is formulated, that allows to compute the unknown depth $Z : \Omega_0 \to \mathbb{R}$ as a minimizing function. The proposed energy functional has the following form

$$E(Z) = E_{\text{Data}}(Z) + \alpha \, E_{\text{Smooth}}(Z) \tag{28}$$

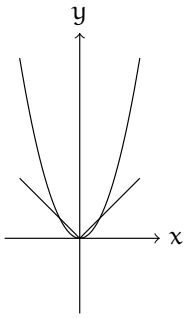
It consists of a data term $E_{\text{Data}}(Z)$ and a smoothness term $E_{\text{Smooth}}(Z)$. The data term models the assumption that the projection of the reconstructed surface $\mathcal{P}$ fits to both views. Unfortunately, the usage of the data term alone is an ill-posed problem due to ambiguities of the projection. Hence the smoothness term enforces Z to be smoothly varying in space. In addition, this allows to obtain solutions for occluded pixels. Besides these two terms a regularization parameter $\alpha > 0$ is introduced to steer their relative impact.

In order to use this energy functional, constancy assumptions on image features have to be made. Corresponding image features have to remain unchanged between both views with the aim of defining a quantity, that allows to tell how well corresponding pixels match. By assuming a scene of Lambertian objects, the brightness constancy

assumption is a favorable choice, since regardless of the viewing direction objects appear equally bright. Thus the data term reads

$$E_{Data}(Z) = \int_{\Omega_0} \left(|I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})|^2\right) dxdy \tag{29}$$

Where $I_i(x, y) : \Omega_i \subset \mathbb{R}^2 \to \mathbb{R}$ denotes the image taken by camera C_i. As a result of the quadratic penalizer the influence of outliers is very high. This can be reduced by using a non-quadratic penalizer. As in [2, 4] the regularized L_1 norm will be used



The increasing influence of a quadratic penalizer vs. the regularized L_1 norm.

$$\Psi(s^2) = \sqrt{s^2 + \epsilon^2} \tag{30}$$

with a small $\epsilon > 0$ to ensure differentiability and strict convexity. Applying the function to the current data term leads to the robust data term

$$E_{Data}(Z) = \int_{\Omega_0} \Psi\left(|I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})|^2\right) dxdy \tag{31}$$

Further one should note that this non-linearized formulation allows the estimation of large displacements. The linearization is postponed to the numerical scheme.

After having discussed the data term a concrete smoothness term is needed. Therefore, the gradient of Z will be penalized with respect to the reference view to obtain a smoothly varying surface. This can be expressed as

$$E_{Smooth}(Z) = \int_{\Omega_0} |\nabla Z|^2 dxdy \tag{32}$$

As before, the quadratic penalizer will not produce a desirable solution, because depth discontinuities are smoothed. Hence the same subquadratic function $\Psi(s^2)$ will be applied as in case of the data term. This will lead to an isotropic depth-driven smoothing behavior and that allows to preserve discontinuities.

$$E_{Smooth}(Z) = \int_{\Omega_0} \Psi\left(|\nabla Z|^2\right) dxdy \tag{33}$$

Finally, the complete energy functional reads

$$E(Z) = \int_{\Omega_0} \Psi\left(|I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})|^2\right) + \alpha\, \Psi\left(|\nabla Z|^2\right) dxdy \tag{34}$$

3.3 MINIMIZING THE ENERGY FUNCTIONAL

Having the energy functional at hand, it remains the question how to find the minimizing function. To achieve this the Euler-Lagrange

equation introduced in Section 2.2 will help. For the energy functional (Equation 34) the corresponding Euler-Lagrange equation requires the following partial derivatives

$$\begin{aligned} F_Z &= 2 \cdot \Psi' \left(|I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})|^2 \right) \cdot (I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})) \\ &\qquad \cdot (I_0(\mathbf{p_0}) - I_1(\mathbf{p_1}))_Z \\ F_{Z_x} &= 2 \cdot \Psi' \left(|\nabla Z(x,y)|^2 \right) \cdot Z(x,y)_x \cdot \alpha \\ F_{Z_y} &= 2 \cdot \Psi' \left(|\nabla Z(x,y)|^2 \right) \cdot Z(x,y)_y \cdot \alpha \end{aligned}$$

plugging these partial derivatives in Equation 15 the corresponding Euler-Lagrange equation reads

$$\begin{aligned} 0 = \Psi' \left(|I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})|^2 \right) \cdot (I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})) \cdot (I_0(\mathbf{p_0}) - I_1(\mathbf{p_1}))_Z \\ -\alpha \operatorname{div} \left(\Psi' \left(|\nabla Z|^2 \right) \cdot \nabla Z \right) \end{aligned} \tag{35}$$

with the natural boundary condition

$$0 = n^\top \nabla Z = \partial_n Z \tag{36}$$

Introducing the following abbreviation $\Delta = I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})$ allows a short notation of Equation 35.

$$0 = \Psi' \left(|\Delta|^2 \right) \cdot \Delta \cdot \Delta_Z - \alpha \operatorname{div} \left(\Psi' \left(|\nabla Z|^2 \right) \cdot \nabla Z \right) \tag{37}$$

Unfortunately, the energy functional (Equation 34) is non-linear and non-convex, because of the perspective projection, the non-linearized data term and the non-quadratic penalizers. In particular the non-convexity poses a problem, since this is why the energy functional has multiple local minimizer. Consequently, a suitable minimization strategy is needed. To handle these challenges the well-known nested fixed-point iteration method of Brox et al. [4] is used. With the aim of creating a linear equation the method consists of three major parts: a fixed point iteration, a incremental computation, and a coarse-to-fine strategy.

FIXED POINT ITERATION. The first step is to introduce a fixed point iteration step on Z, to overcome the non-linearity in the argument. Let $I_i(\mathbf{p_i}^k)$ denote the image value of the projected point $\mathbf{P}^k$ which corresponds to the depth $Z^k(x,y)$ and let Δ^k denote the abbreviation $\Delta^k = I_0(\mathbf{p_0}^k) - I_1(\mathbf{p_1}^k)$, where k denotes the iteration index. Using a semi-implicit scheme in the data term and an implicit scheme in the smoothness term, the solution of Z^{k+1} can be obtained as a solution of

$$0 = \Psi' \left(\left| \Delta^{k+1} \right|^2 \right) \cdot \Delta^{k+1} \cdot \Delta_Z^k - \alpha \operatorname{div} \left(\Psi' \left(|\nabla Z^{k+1}|^2 \right) \cdot \nabla Z^{k+1} \right) \tag{38}$$

INCREMENTAL COMPUTATION. The Equation 38 still remains non-linear due to the non-linear Ψ' functions and the Δ^{k+1} term. To remove the second non-linearity an incremental computation is introduced by the splitting the unknown depth in the known depth Z^k from the old time step and the unknown depth increment dZ^k from the new time step

$$Z^{k+1} = Z^k + dZ^k \tag{39}$$

Using a first order Taylor expansion the linearization reads

$$I_i(\mathbf{p_i}^{k+1}) \approx I_i(\mathbf{p_i}^k) + \partial_Z I_i(\mathbf{p_i}^k) \cdot dZ^k \tag{40}$$

and it follows that

$$\Delta^{k+1} \approx \Delta^k + \Delta_Z^k \cdot dZ^k \tag{41}$$

In order to keep up a compact notation the Ψ' terms are abbreviated as

$$\begin{aligned} \left(\Psi'\right)_{\text{Data}}^k &= \Psi'\Big(\left|\Delta^k + \Delta_Z^k \cdot dZ^k\right|^2 \Big) \\ \left(\Psi'\right)_{\text{Smooth}}^k &= \Psi'\Big(|\nabla \left(Z^k + dZ^k\right)|^2\Big) \end{aligned} \tag{42}$$

Now, Equation 38 can be rewritten as

$$\begin{aligned} 0 = &\left(\Psi'\right)_{\text{Data}}^k \cdot \left(\Delta^k + \Delta_Z^k \cdot dZ^k\right) \cdot \Delta_Z^k \\ &-\alpha \operatorname{div}\Big(\left(\Psi'\right)_{\text{Smooth}}^k \cdot \nabla \left(Z^k + dZ^k\right)\Big) \end{aligned} \tag{43}$$

The remaining non-linearity of this new equation is because of the Ψ' functions, but since the chosen Ψ is a strict convex function, Equation 43 that has to be solved at each fixed point step corresponds to a a convex problem with a unique minimizer. The next step is to remove the remaining non-linearity. This is achieved by a second fixed point iteration, which is referred to as inner fixed point iteration. The iteration step is denoted by an index l. Further the abbreviations $\left(\Psi'\right)_{\text{Data}}^{k,l}$ and $\left(\Psi'\right)_{\text{Smooth}}^{k,l}$ use the depth at iteration k and the depth increment $dZ^{k,l}$ at iteration k, l. Finally, the following equation is obtained at each fixed point iteration, that is linear in $dZ^{k,l+1}$ reads

$$\begin{aligned} 0 = &\left(\Psi'\right)_{\text{Data}}^{k,l} \cdot \left(\Delta^k + \Delta_Z^k \cdot dZ^{k,l+1}\right) \cdot \Delta_Z^k \\ &-\alpha \operatorname{div}\Big(\left(\Psi'\right)_{\text{Smooth}}^{k,l} \cdot \nabla \left(Z^k + dZ^{k,l+1}\right)\Big) \end{aligned} \tag{44}$$

COARSE-TO-FINE STRATEGY. The original energy functional is non-convex, so to approximate a good local minimizer or even to obtain the global minimizer the outer fixed point iteration is embedded into a coarse-to-fine scheme. Hence, a refinement factor $\eta \in (0, 1)$ is introduced, to build an image pyramid. Starting at the coarsest level $k = 0$ up to the original problem size, where the coarse result is used as initialization of the next finer level. Figure 10 shows by the example of

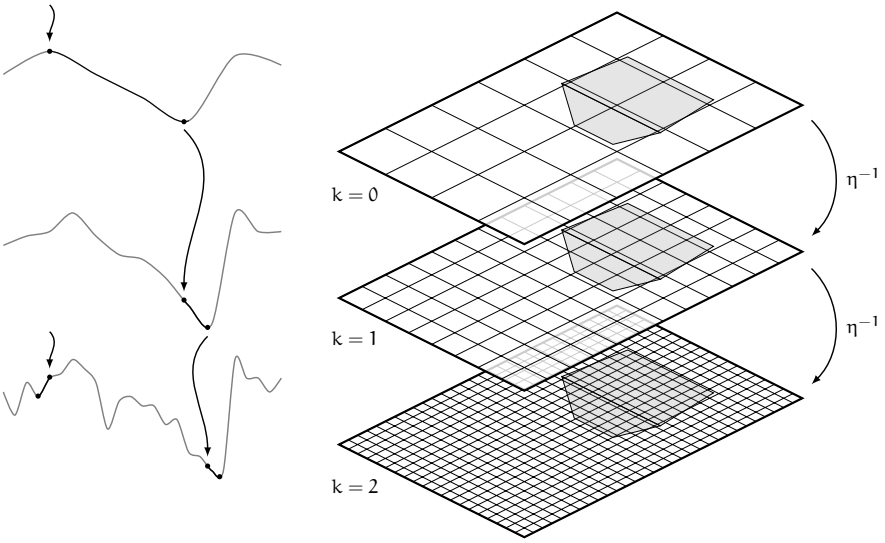


Figure 10: The coarse-to-fine strategy: while starting on the coarsest level leads to a good local minimizer or even the global minimizer, a non-hierarchical approach may lead to a poor local minimizer.

an energy landscape how the coarse-to-fine approach helps to avoid poor local minimizers, while a non-hierarchical approach may lead to a poor local minimizer.

This reduced image size also requires an adaption of the camera projection matrices, strictly speaking the camera calibration matrices, in order to suit each level. This can be done in a way that keeps the magnitude of the depth the same on each level. For this reason the pixel dimension is adopted according to η. Equation 10 shows how the pixel dimension influences the camera calibration matrix and a scaled version for a coarser level reads

$$K_\eta = \begin{bmatrix} \eta \cdot n_x & 0 & 0 \\ 0 & \eta \cdot n_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \eta \cdot s_x & 0 & \eta \cdot o_x \\ 0 & \eta \cdot s_y & \eta \cdot o_y \\ 0 & 0 & 1 \end{bmatrix} \quad (45)$$

To use this strategy a few questions are still unanswered. The first question is how to initialize Z^0 and $dZ^{k,0}$. The second question is how to compute expressions like $I_i(\mathbf{p_i}^k)$.

BACK PROJECTION. The second question is answered with a back projection towards the reference view. In order to do so the already computed depth Z^k is used to determine the 3-D points $\mathcal{P}^k$. Those points are then projected towards the second view, which results in the point $\mathbf{p_1}^k$. In most cases $\mathbf{p_1}^k$ will lie between the image pixels and an approximation can be computed using a bilinear interpolation.

This value can then be written in a back projected image $\mathcal{I}_1^k(x, y)$. That is

$$\mathcal{I}_1^k(x, y) = I_1(\mathbf{p_1}^k) \tag{46}$$

Figure 11 illustrates this procedure.

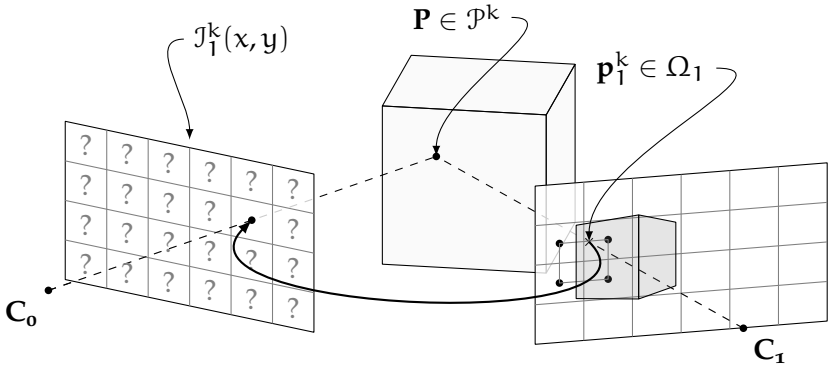


Figure 11: Back projection towards the reference view. This can be understood as a compensation of the already computed depth Z^k.

THE INITIALIZATION. For the depth increment a trivial initialization $dZ^{k,0} = 0$ is chosen. But such an initialization is not possible for Z^0, since a trivial initialization with $Z^0 = 0$ would produce a set of points $\mathcal{P}^0$ lying entirely in the reference camera origin. This would result in a very poor or even unusable back projected image $\mathcal{I}_1^0(x, y)$. To avoid this problem a plane sweeping approach is used.

To this end, a constant Z is assumed and for a finite number of planes (e.g. $Z^0 = 0, 1, \ldots$) a normalized data term is evaluated. Normalized means here normalized with respect to the number of 3-D points projected on the second view. The plain with the smallest energy is then used as initialization. For the quadratically penalized data term this reads

$$E_{\text{Init}}(Z) = w_1^{-1} \cdot \int_{\Omega_0} \left(|I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})|^2 \right) dx dy \tag{47}$$

with w_1 being the number of projected 3-D points that lie on the second image taken by the camera C_1. A sketch is given in Figure 12. The next question is which planes should be evaluated, depending on the cameras intrinsic parameters and the relative camera pose a reasonable depth range can vary strongly. To answer this question an ortho-parallel or a converging camera setup is assumed and it is assumed that the point $\mathbf{P}$ lying on the optical axis of C_1 is seen by both cameras. In addition, it is assumed that the epipoles of the cameras do not lie in the image. With these assumptions a reasonable depth range can be computed as follows:

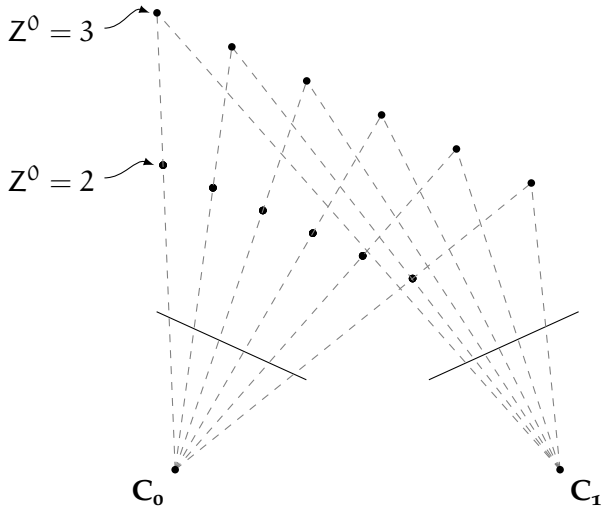


Figure 12: Plain sweeping to compute a constant initial depth.

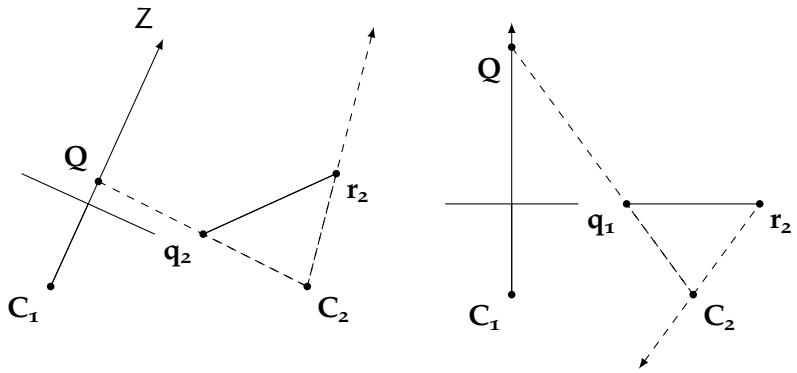


Figure 13: Depth range computation.

1. Compute the epipolar line for the second view of the optical axis of the camera C_0.

2. Compute the intersection points $\mathbf{q}_1, \mathbf{r}_1$ of the computed epipolar line with the image boundary.

3. Compute the intersection points $\mathbf{Q}, \mathbf{R}$ of the optical axis of the camera C_0 and the optical ray going through camera center $\mathbf{C_1}$ and the previously computed points $\mathbf{q}_1, \mathbf{r}_1$. This is illustrated in Figure 13.

4. The smaller Z-coordinate of both points with $Z > 0$ is used as starting value. If the larger Z-coordinate is non-negative it can be used as stopping value, otherwise a multiple of the starting value is used.

After answering the open questions on the initialization the next step is to discretize the derivatives. But before doing this a handy notation is introduced.

3.4 FOLLOWING THE MOTION TENSOR NOTATION

To further simplify the notation this section introduces a depth tensor notation, which follows the idea of the motion tensor notation in optical flow estimation [5]. Therefore, a depth increment vector $\mathbf{dZ}^{k,l} := (dZ^{k,l}, 1)^\top$, a vector $\mathbf{S}_\nabla := (\Delta_Z^k, \Delta^k)^\top$ and a depth tensor T^k are introduced. The depth tensor is a symmetric 2×2 matrix defined as

$$T^k := \mathbf{S}_\nabla \mathbf{S}_\nabla^\top = \begin{pmatrix} (\Delta_Z^k)^2 & \Delta_Z^k \Delta^k \\ \Delta_Z^k \Delta^k & (\Delta^k)^2 \end{pmatrix} \tag{48}$$

This allows to rewrite the fixed point iteration from Equation 44 as

$$\begin{aligned} 0 = {} & \left(\Psi'\right)_{\text{Data}}^{k,l} \cdot \left(T_{11}^k \cdot dZ^{k,l} + T_{12}^k\right) \\ & -\alpha \operatorname{div}\left(\left(\Psi'\right)_{\text{Smooth}}^{k,l} \cdot \nabla\left(Z^k + dZ^{k,l+1}\right)\right) \end{aligned} \tag{49}$$

and the abbreviations of the Ψ terms as

$$\begin{aligned} \left(\Psi'\right)_{\text{Data}}^{k,l} &:= \Psi'\left(\mathbf{dZ}^{k,l\top}\, T^k\, \mathbf{dZ}^{k,l}\right) \\ \left(\Psi'\right)_{\text{Smooth}}^{k,l} &:= \Psi'\left(|\nabla\left(Z^k + dZ^k\right)|^2\right) \end{aligned} \tag{50}$$

3.5 DISCRETIZATION

In this section an important step towards the numerical solution of the fixed point iteration (49) is taken, speaking of the discretization. Since each 3-D point $\mathbf{P}^k \in \mathcal{P}^k$ corresponds to a pixel $\mathbf{p}_0^k \in \Omega_0$ in the reference view the unknown function $dZ^k(x, y)$ can be considered as a rectangular pixel grid with the grid size $h_x^k \times h_y^k$. Then, the approximation to dZ^k at a pixel (i, j) is given by $dZ_{i,j}^k$. Analogous approximations are $Z_{i,j}^k$, $T_{i,j}^k$, $\left(\Psi'\right)_{\text{Data } i,j}^{k,l}$ and $\left(\Psi'\right)_{\text{Smooth } i,j}^{k,l}$. The discrete version of Equation 49 is given by

$$\begin{aligned} 0 = {} & \left(\Psi'\right)_{\text{Data } i,j}^{k,l} \cdot \left(\left(T_{11}^k\right)_{i,j} \cdot dZ_{i,j}^{k,l} + \left(T_{12}^k\right)_{i,j}\right) \\ & -\alpha \operatorname{div}\left(\left(\Psi'\right)_{\text{Smooth } i,j}^{k,l} \cdot \nabla\left(Z_{i,j}^k + dZ_{i,j}^{k,l+1}\right)\right) \end{aligned} \tag{51}$$

In order to make use of it the contained derivatives must be calculated. This is done by using finite difference approximations.

DISCRETE DEPTH TENSOR. Starting with the depth tensor T^k, the approximation of $\Delta_Z^k = I_0(\mathbf{p_0}^k)_Z - I_1(\mathbf{p_1}^k)_Z$ is required. Hence, the

derivative of $I_i(\mathbf{p_i}^k)$ with respect to Z is required. Applying the chain rule it follows

In this paragraph the superscript k is dropped for x_i^k, y_i^k and Z^k, so $\mathbf{p_i}^k := (x_i, y_i)^\top$ and $Z := Z^k$.

$$\begin{aligned}\partial_Z I_i(\mathbf{p_i}^k) &= \left(\nabla_i I_i(\mathbf{p_i}^k)\right)^\top \cdot \partial_Z \mathbf{p_i}^k \\ &= \partial_{x_i} I_i \cdot \partial_Z x_i + \partial_{y_i} I_i \cdot \partial_Z y_i \end{aligned} \tag{52}$$

where $\nabla_i = (\partial_{x_i}, \partial_{y_i})^\top$. Using the coefficients defined in (27) the derivatives $\partial_Z x_i$ and $\partial_Z y_i$ read

$$\partial_Z x_i = \frac{ad - bc}{(c \cdot Z + d)^2} \qquad \partial_Z y_i = \frac{\check{a}d - \check{b}c}{(c \cdot Z + d)^2} \tag{53}$$

To compute the derivative $(\nabla_i I_i)^\top$ Basha et al. [2] used the back projected image and the following relation of the gradient of the back projected image $(\nabla \mathcal{I}_i)^\top$ with gradient of the original image,

$$(\nabla_i I_i(x_i, y_i))^\top = (\nabla \mathcal{I}_i(x, y))^\top \cdot J^{-1} \tag{54}$$

where J denotes the Jacobian matrix. The derivation and the entries of the Jacobian can be found in Section A.2. However an alternative approach is to first compute the gradient of the original image and perform a back projection of the gradient, i.e. this is

$$(\nabla_i I_i(x_i, y_i))^\top = (\nabla_i \mathcal{I}_i(x, y))^\top \tag{55}$$

Later on an experiment (Section 5.3) will help to choose the favored approach. The needed gradients are approximated using a central fourth order finite difference scheme, which reads

$$\begin{aligned}\partial_x I_{i\ i,j} &= \frac{-I_{i\ i+2,j} + 8I_{i\ i+1,j} - 8I_{i\ i-1,j} + I_{i\ i-2,j}}{12h_x^k} \\ \partial_y I_{i\ i,j} &= \frac{-I_{i\ i,j+2} + 8I_{i\ i,j+1} - 8I_{i\ i,j-1} + I_{i\ i,j-2}}{12h_y^k} \end{aligned} \tag{56}$$

Putting all this together allows to compute the parts of the fixed point iteration related to the data term.

SMOOTHNESS TERM. To approximate the smoothness term related part, speaking of the divergence expression, a nested central second order finite difference scheme is used. A derivation can be found in Section A.1. Using the abbreviation

$$(\Psi')^{k,l}_{\text{Smooth } i,j} := (\Psi')^{k,l}_{S\ i,j}$$

this reads

$$\begin{aligned} &-\alpha \operatorname{div}\left((\Psi')^{k,l}_{S\ i,j} \cdot \nabla\left(Z^k_{i,j} + dZ^{k,l+1}_{i,j}\right)\right) \\ &= -\alpha \sum_{n\in\{x,y\}} \sum_{(\tilde{i},\tilde{j})\in\mathcal{N}_n(i,j)} w^{k,l}_{i,j,\tilde{i},\tilde{j},n}\left(Z^{k,l}_{\tilde{i},\tilde{j}} + dZ^{k,l+1}_{\tilde{i},\tilde{j}} - Z^{k,l}_{i,j} - dZ^{k,l+1}_{i,j}\right) \end{aligned} \tag{57}$$

where

$$w^{k,l}_{i,j,\tilde{i},\tilde{j},n} = \alpha \cdot \frac{(\Psi')^{k,l}_{S\ \tilde{i},\tilde{j}} + (\Psi')^{k,l}_{S\ i,j}}{2 \cdot h_n^2} \tag{58}$$

and $\mathcal{N}_n(i,j)$ denotes the set of neighbors of the pixel (i,j) in direction n. In this case these are the four direct neighbor pixels $(i+1,j)$, $(i-1,j)$, $(i,j+1)$ and $(i,j-1)$. The value of $(\Psi')^{k,l}_{S\ i,j}$ is given by

$$\Psi'(s^2) = \frac{1}{2 \cdot \sqrt{s^2+\epsilon^2}} \tag{59}$$

where the argument is computed by using a central fourth order finite difference scheme which as before.

$$\begin{aligned} \partial_x Z_{i,j} &= \frac{-Z_{i+2,j} + 8Z_{i+1,j} - 8Z_{i-1,j} + Z_{i-2,j}}{12h_x^k} \\ \partial_y Z_{i,j} &= \frac{-Z_{i,j+2} + 8Z_{i,j+1} - 8Z_{i,j-1} + Z_{i,j-2}}{12h_y^k} \end{aligned} \tag{60}$$

A stencil is given in Section B.1.

3.6 SOLVING THE LINEAR EQUATION

Finally, the method comes down to solve a linear discrete equation at each pixel that reads

$$\begin{aligned} 0 = &(\Psi')^{k,l}_{\text{Data } i,j} \cdot \left(\left(T^k_{11}\right)_{i,j} \cdot dZ^{k,l}_{i,j} + \left(T^k_{12}\right)_{i,j} \right) \\ &-\alpha \sum_{n\in\{x,y\}} \sum_{(\tilde{i},\tilde{j})\in\mathcal{N}_n(i,j)} w^{k,l}_{i,j,\tilde{i},\tilde{j},n} \left(Z^{k,l}_{\tilde{i},\tilde{j}} + dZ^{k,l+1}_{\tilde{i},\tilde{j}} - Z^{k,l}_{i,j} - dZ^{k,l+1}_{i,j} \right) \end{aligned}$$

This can be expressed as a linear equation system $A\mathbf{x} = \mathbf{b}$, where A is a sparse matrix. Therefore, an iterative solver is a suitable and simple choice in order to approximate the solution. Such an iterative solver is the Successive Overrelaxation Method (SOR) [28]. Hence, at each inner loop l several iterations are performed. The corresponding iteration step s with the overrelaxation parameter ω reads

$$\begin{aligned} &dZ^{k,l+1,s+1}_{i,j} \\ =&(1-\omega) \cdot dZ^{k,l+1,s}_{i,j} + \omega \cdot \Bigg((\Psi')^{k,l}_{\text{Data } i,j} \left(T^k_{12}\right)_{i,j} \\ &- \sum_{n\in\{x,y\}} \sum_{(\tilde{i},\tilde{j})\in\mathcal{N}^-_n(i,j)} w^{k,l}_{i,j,\tilde{i},\tilde{j},n} \left(Z^{k,l}_{\tilde{i},\tilde{j}} + dZ^{k,l+1,s+1}_{\tilde{i},\tilde{j}} - Z^{k,l}_{i,j} \right) \\ &- \sum_{n\in\{x,y\}} \sum_{(\tilde{i},\tilde{j})\in\mathcal{N}^+_n(i,j)} w^{k,l}_{i,j,\tilde{i},\tilde{j},n} \left(Z^{k,l}_{\tilde{i},\tilde{j}} + dZ^{k,l+1,s}_{\tilde{i},\tilde{j}} - Z^{k,l}_{i,j} \right) \Bigg) \\ &\cdot \left(-\left(T^k_{11}\right)_{i,j} \cdot (\Psi')^{k,l}_{\text{Data } i,j} - \sum_{n\in\{x,y\}} \sum_{(\tilde{i},\tilde{j})\in\mathcal{N}_n(i,j)} w^{k,l}_{i,j,\tilde{i},\tilde{j},n} \right)^{-1} \end{aligned}$$

Here $\mathcal{N}_n^-(i,j)$ denotes the set of neighbors of the pixel (i,j) in direction n that have already been computed in the iteration step s and $\mathcal{N}_n^+(i,j)$ denotes the set of neighbors of the pixel (i,j) in direction n that have not been computed yet in the iteration step s.

3.7 PRESMOOTHING

The last point to add is a small preprocessing step, it consists of a convolution of the input image with a Gaussian kernel of standard deviation σ_{pre}. This step allows to remove high-frequency noise of the images and guarantees that the image can be seen as a function of C^∞. It reads

$$I_i = K_{\sigma_{pre}} * I_i^{orig} \tag{61}$$

Here, $K_{\sigma_{pre}}$ denotes the Gaussian kernel and $*$ the convolution operator. Moreover, I_i^{orig} denotes the unsmoothed image and I_i the resulting smoothed image.

The parameter σ_{pre} is also referred to as noise scale [6]. It is important to note, that if σ_{pre} is chosen too large, small details are eliminated. Because the coarse-to-fine approach already includes a kind of smoothing, only a very small amount or even no presmoothing is needed.

4 EXTENSIONS

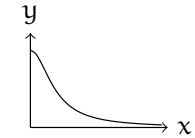
4.1 ADVANCED SMOOTHNESS TERMS

Up to now an isotropic depth-driven smoothness term was used. In this section more advanced smoothness terms are proposed.

4.1.1 *Isotropic Image- & Depth-Driven Regularization*

It often can be observed that discontinuities in the depth correlate with image edges, because object boundaries often coincide with image edges. Therefore, embedding this information can help to improve the depth field at discontinuities [1]. The desired behavior is to reduce the smoothing at image edges. In order to achieve such an smoothing-behavior an additional weighting function $g(s^2)$ is added. Here the Perona-Malik diffusivity [13] is used, which reads

$$g(s^2) = \frac{1}{1 + \frac{s^2}{\lambda^2}} \tag{62}$$



A weighting function $g(s^2)$.

where $\lambda > 0$ is the contrast parameter, which separates forward and backward diffusion. This is a positive and decreasing function. It is applied to the spatial image gradient $|\nabla I_0|^2$, which serves as a fuzzy edge detector. In case of RGB color images, the absolute value of the gradient of each color channel is summed up and squared. Because of g being positive and decreasing it will reduce the diffusion at image edges. The new smoothness term, which is a depth- and image-driven combination, reads

$$E_{\text{Smooth}}(Z) = \int_{\Omega_0} g(|\nabla I_0|^2) \cdot \Psi\left(|\nabla Z|^2\right) \mathrm{d}x\mathrm{d}y \tag{63}$$

and will be referred to as isotropic image- and depth-driven regularizer. The regularization parameter ϵ of Ψ and the contrast parameter λ of g allow to adjust the behavior to result in a good interaction between both parts. With this new smoothness term the following questions arise: How does it change the Euler-Lagrange equation and how is it minimized? Therefore, a generic form is considered in the following.

A GENERIC FORM. The smoothness term related part of the Euler-Lagrange equation can be understood as a diffusion-like process [27]. This part reads

$$\operatorname{div}(D \cdot \nabla Z) \tag{64}$$

with the diffusion tensor D, a positive semidefinite symmetric 2×2 matrix. The Diffusion tensor D steers the diffusion-like process. It is given by

$$D := \begin{pmatrix} a & b \\ b & c \end{pmatrix} \tag{65}$$

a, b and c are NOT the short notation of the projections.

Thus, the divergence expression can be written as

$$\begin{aligned} \operatorname{div}(D \cdot \nabla Z) &= \operatorname{div} \begin{pmatrix} aZ_x + bZ_y \\ bZ_x + cZ_y \end{pmatrix} \\ &= \partial_x (aZ_x) + \partial_x (bZ_y) + \partial_y (bZ_x) + \partial_y (cZ_y) \end{aligned}$$

The diffusion tensor to the isotropic depth-driven smoothness term, that has been used so far, reads

$$D = I \cdot \Psi'\left(|\nabla Z|^2\right) = \begin{pmatrix} \Psi'\left(|\nabla Z|^2\right) & 0 \\ 0 & \Psi'\left(|\nabla Z|^2\right) \end{pmatrix} \tag{66}$$

where I denotes the identity matrix. In this case $b = 0$ and the mixed terms $\partial_x (bZ_y)$ and $\partial_y (bZ_x)$ vanish. The diffusion tensor of the new combined image- and depth-driven smoothness term reads

$$D = \begin{pmatrix} g(|\nabla I_0|^2) \cdot \Psi'\left(|\nabla Z|^2\right) & 0 \\ 0 & g(|\nabla I_0|^2) \cdot \Psi'\left(|\nabla Z|^2\right) \end{pmatrix} \tag{67}$$

Here the mixed terms also vanish. Hence, the same discretization as in the pure depth-driven isotropic case can be used. The new weights for Equation 58 read

$$w^{k,l}_{i,j,\tilde{i},\tilde{j},n} = \alpha \cdot \frac{1}{h_n^2} \frac{g(|\nabla I_0|^2)^k_{\tilde{i},\tilde{j}} + g(|\nabla I_0|^2)^k_{i,j}}{2} \frac{(\Psi')^{k,l}_{S\ \tilde{i},\tilde{j}} + (\Psi')^{k,l}_{S\ i,j}}{2} \tag{68}$$

4.1.2 *Anisotropic Complementary Regularization*

Unfortunately, the use of image information has some shortcomings. In rich textured regions oversegmentation effects appear, because the discontinuities in depth do not correlate well with the image edges. Pure depth-driven smoothness terms as proposed in the basic method do not have this problem, but often result in less accurate edges. A more clever solution to this problem is a joint regularizer, as proposed by Sun et al. [23] and further improved by Zimmer et al. [30]. Such a smoothness term takes the directional information of the image structure and the magnitude of the evolving depth information. The directional information is gathered using the structure tensor [8]:

$$S_\rho(\nabla I_0) := K_\rho * \left(\nabla I_0 \nabla I_0^\top\right) \tag{69}$$

The convolution with a Gaussian kernel integrates neighborhood information. The directional information is extracted using the eigenvectors of S_ρ, which are two orthonormal vectors s_1 and s_2, due to the fact that S_ρ is a symmetric positive semidefinite 2×2 matrix. The corresponding eigenvalues $\mu_1 \geqslant \mu_2 \geqslant 0$ help to identify the direction. The eigenvector s_1, corresponding to the larger eigenvalue μ_1, points across the local image structure, consequently s_2 points along it. In addition, Zimmer et al. propose a single robust penalization, that means to use a subquadratic penalization in s_1-direction and a quadratic penalization in s_2-direction. The final smoothness term reads

$$E_{\text{Smooth}}(Z) = \Psi_P\left(\left(s_1^\top \nabla Z\right)^2\right) + \left(s_2^\top \nabla Z\right)^2 \tag{70}$$

This leads to the following diffusion tensor

$$D = (s_1, s_2)\begin{pmatrix} \Psi_P'\left(\left(s_1^\top \nabla Z\right)^2\right) & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} s_1^\top \\ s_2^\top \end{pmatrix} \tag{71}$$

Ψ_P denotes the Perona-Malik regulariser [13] given by

$$\Psi_P(s^2) := \lambda^2 \log\left(1 + \frac{s^2}{\lambda^2}\right) \tag{72}$$

and

$$\Psi_P'(s^2) := \frac{1}{1 + \frac{s^2}{\lambda^2}} \tag{73}$$

As before, $\lambda > 0$ is the contrast parameter, which separates forward and backward diffusion.

It appears that the new diffusion tensor has off-diagonal entries different from zero. Because of that the mixed partial derivatives $\partial_x(bZ_y)$ and $\partial_y(bZ_x)$ of the divergence expression do not vanish. A standard discretization as before will not guarantee the non-negativity requirement. Hence, a special discretization that was first used in [30] is applied. A stencil of the discretization is given in Section B.2.

Later, additional constraints are embedded in the data term, such as the gradient constancy and additional constraints for the RGB color representation. In this case the structure tensor is replaced by the regularization tensor

$$R_\rho := K_\rho * \sum_{c=1}^{3}\left(\nabla I_0^c \nabla I_0^{c\top} + \gamma\left(\nabla\left(I_0^c\right)_x \nabla\left(I_0^c\right)_x^\top + \nabla\left(I_0^c\right)_y \nabla\left(I_0^c\right)_y^\top\right)\right)$$

to gather the directional information using the constraint edges. Here $\mathbf{I}_i = \left(I_i^1, I_i^2, I_i^3\right)$ denotes a RGB color image.

4.1.3 *Anisotropic Depth-Driven Regularization*

A further possible extension to the isotropic depth-driven smoothness term is the step towards an anisotropic depth-driven smoothness term. Zimmer et al. [29] propose a way to achieve a true anisotropic depth-driven behavior. Therefore, the smoothness term is not modeled as a part of the energy functional, rather they directly modify the diffusion tensor that evokes the intended behavior. As before, the structure tensor is used to obtain directional information. However this time the evolving depth information is used instead of the image information. Hence, the corresponding structure tensor reads

$$S_\rho(\nabla Z_\sigma) := K_\rho * \left(\nabla Z_\sigma \nabla Z_\sigma^\top\right) \tag{74}$$

where an additional Gaussian convolution with the standard deviation σ is performed on the depth information to reduce the influence of noise and staircasing effects. And as before a Gaussian convolution with the standard deviation ρ is applied to integrate the information over a neighborhood. Hence, the proposed diffusion tensor reads

$$D = \Psi_P'\left(S_\rho(\nabla Z_\sigma)\right) = (s_1, s_2)\begin{pmatrix}\Psi_P'(\mu_1) & 0\\ 0 & \Psi_P'(\mu_2)\end{pmatrix}\begin{pmatrix}s_1^\top\\ s_2^\top\end{pmatrix} \tag{75}$$

in this case the matrix-valued function Ψ_P' denotes a function which applies the scalar-valued version to each eigenvalue and leaves the eigenvectors unchanged. This leads to a anisotropic smoothing behavior that adapts to the local structure. Mainly three different cases can arise

A. *Homogeneous regions*, where both eigenvalues vanish, this leads to a smoothing in both directions.
$\mu_1 \approx \mu_2 \approx 0 \Rightarrow \Psi_P'(\mu_1) \approx \Psi_P'(\mu_2) \approx 1$

B. *Straight edges*, where one eigenvalue vanishes which leads to a anisotropic smoothing along the edge but not across.
$\mu_1 \gg \mu_2 \approx 0 \Rightarrow \Psi_P'(\mu_1) \approx 0, \Psi_P'(\mu_2) \approx 1$

C. *Corners*, no eigenvalues vanishes which prevents smoothing.
$\mu_1 > \mu_2 \gg 0 \Rightarrow \Psi_P'(\mu_1) \approx \Psi_P'(\mu_2) \approx 0$

As in the case of the complementary regularizer the diffusion tensors has off-diagonal entries different from zero, which is why the same discretization is used. The stencil is given in Section B.2.

4.1.4 *Second Order Isotropic Regularization*

Up to now all the proposed smoothness terms are of first order. This has the disadvantage that surfaces parallel to the reference camera are

preferred, because the gradient would be zero in this case. To eliminate this shortcoming second order smoothness terms help: They treat slanted surfaces equally. A possible second order isotropic regularizer is the following

$$E_{\text{Smooth}}(Z) = \int_{\Omega_0} \Psi\left(\|\mathcal{H}_Z\|_F^2\right) dx dy \tag{76}$$

it penalizes the Frobenius Norm of the Hessian of Z which reads

$$\begin{aligned}\|\mathcal{H}_Z\|_F &= \sqrt{Z_{xx}^2 + Z_{xy}^2 + Z_{yx}^2 + Z_{yy}^2} \\ &= \sqrt{Z_{xx}^2 + 2 \cdot Z_{xy}^2 + Z_{yy}^2}\end{aligned}$$

Here again the use of the regularized L_1 norm allows the preservation of depth discontinuities. The new energy functional now contains second order terms and no first order terms anymore. Of course this changes the Euler-Lagrange equation. Plugging the following partial derivatives

$$\begin{aligned}F_Z &= 2 \cdot \Psi_D'\left(|\Delta|^2\right) \cdot \Delta \cdot \Delta_Z \\ F_{Z_x} &= 0 \\ F_{Z_y} &= 0 \\ F_{Z_{xx}} &= 2\alpha \cdot \Psi_S'\left(|\nabla Z|^2\right) \cdot Z_{xx} \\ F_{Z_{xy}} &= 2\alpha \cdot \Psi_S'\left(|\nabla Z|^2\right) \cdot Z_{xy} \\ F_{Z_{yx}} &= 2\alpha \cdot \Psi_S'\left(|\nabla Z|^2\right) \cdot Z_{yx} \\ F_{Z_{yy}} &= 2\alpha \cdot \Psi_S'\left(|\nabla Z|^2\right) \cdot Z_{yy}\end{aligned}$$

in Equation 18 the new Euler-Lagrange equation reads

$$0 = \Psi'\left(|\Delta|^2\right) \cdot \Delta \cdot \Delta_Z - \alpha \nabla^{*\top}\left(\Psi'\left(\|\mathcal{H}_Z\|_F^2\right) \cdot \nabla^* Z\right) \tag{77}$$

where $\nabla^* = (\partial_{xx}, \partial_{xy}, \partial_{yx}, \partial_{yy})^\top$ and the associated boundary conditions read

$$0 = n^\top \begin{pmatrix} -\partial_x Z_{xx} - \partial_y Z_{xy} \\ -\partial_x Z_{yx} - \partial_y Z_{yy} \end{pmatrix}, \quad 0 = n^\top \begin{pmatrix} F_{Z_{xx}} \\ F_{Z_{xy}} \end{pmatrix}, \quad 0 = n^\top \begin{pmatrix} F_{Z_{yx}} \\ F_{Z_{yy}} \end{pmatrix}$$

It is seen, that the divergence expression is replaced by a sort of second order divergence. The derivation of the discretization is shown in Section A.3 and the corresponding stencil in Section B.3.

To keep the relative impact of data and smoothness term the same as before, the regularization parameter α is adapted to the warping level k. This is achieved by setting $\alpha^k = \alpha/\eta^\ell$ where ℓ is the pyramid level ($\ell(k) = k^{\max} - k$) which results in increased values of α at coarser levels.

4.2 ADVANCED DATA TERMS

The previous section introduced several advanced smoothness terms. In contrast, this section will deal with improving the data term. The basic method already makes use of a robust data term, with a non-quadratic penalizer. Another possibility to improve the data term is the use of modified constraints to allow a more accurate estimation in different situations.

4.2.1 *Gradient Constancy*

In real world sequences slight brightness changes can appear between both views. This can have different kinds of causes, such as asynchronous exposure times of the cameras or images that were taken at different points in time, while in the meanwhile the illumination changed. Consequently, recent optic flow methods sometimes use a gradient constancy assumption, such as the one proposed by Brox et al. [4]. In a general setting, especially with a wide baseline the image gradients may vary, due to the change of viewpoint. However, the gradient of the back projected image stays the same. The gradient constancy assumption reads

$$\nabla I_0(\mathbf{p_0}) = \nabla I_1(\mathbf{p_1}) \tag{78}$$

As before the assumption is used in a non-linearized manner to allow large displacements. In addition, the robust penalizer function Ψ is applied separately to both assumptions. In order to steer the relative impact a weighting parameter γ is introduced. The new data term reads

$$E_{\text{Data}}(Z) = \int_{\Omega_0} \Psi\left(|\Delta|^2\right) + \gamma\,\Psi\left(|\nabla\Delta|^2\right) \mathrm{d}x\mathrm{d}y \tag{79}$$

and the corresponding Euler-Lagrange equation (using the smoothness term of the basic method)

$$\begin{aligned} 0 = {} & \Psi'\left(|\Delta|^2\right) \cdot \Delta \cdot \Delta_Z + \Psi'\left(|\nabla\Delta|^2\right) \cdot (\nabla\Delta)^\top \cdot (\nabla\Delta)_Z \\ & -\alpha \operatorname{div}(D \cdot \nabla Z) \end{aligned} \tag{80}$$

with the natural boundary condition

$$0 = n^\top \nabla Z = \partial_n Z \tag{81}$$

After repeating the steps of Section 3.3, an additional depth tensor is obtained which reads

$$T_{\nabla}^{k} := \begin{pmatrix} (\nabla\Delta^k)_Z^\top (\nabla\Delta^k)_Z & (\nabla\Delta^k)_Z^\top (\nabla\Delta^k) \\ (\nabla\Delta^k)_Z^\top (\nabla\Delta^k) & (\nabla\Delta^k)^\top (\nabla\Delta^k) \end{pmatrix} \tag{82}$$

$$:= \begin{pmatrix} ((\partial_x\Delta^k)_Z)^2 & (\partial_x\Delta^k)_Z (\partial_x\Delta^k) \\ (\partial_x\Delta^k)_Z (\partial_x\Delta^k) & (\partial_x\Delta^k)^2 \end{pmatrix} \tag{83}$$

$$+ \begin{pmatrix} ((\partial_y\Delta^k)_Z)^2 & (\partial_y\Delta^k)_Z (\partial_y\Delta^k) \\ (\partial_y\Delta^k)_Z (\partial_y\Delta^k) & (\partial_y\Delta^k)^2 \end{pmatrix} \tag{84}$$

and which allows to write the new fixed point iteration, equivalent to Equation 49, with the additional gradient constancy assumption as

$$\begin{aligned} 0 = \quad & (\Psi')^{k,l}_{\text{Data}} \cdot (T^k_{11} \cdot dZ^{k,l} + T^k_{12}) \\ & + (\Psi')^{k,l}_{\text{DataGradient}} \cdot (T^k_{\nabla 11} \cdot dZ^{k,l} + T^k_{\nabla 12}) \\ & - \alpha \operatorname{div}\left((\Psi')^{k,l}_{\text{Smooth}} \cdot \nabla \left(Z^k + dZ^{k,l+1} \right) \right) \end{aligned} \tag{85}$$

with the new abbreviation

$$(\Psi')^{k,l}_{\text{DataGradient}} := \Psi'\left(\mathbf{dZ}^{k,l\top}\, T^k_{\nabla}\, \mathbf{dZ}^{k,l} \right) \tag{86}$$

4.2.2 *Color Images*

Up to now only gray value images have been considered. Of course, it is desirable to exploit the additional color information, if color images are available. Hence, the data term must be extended. If RGB color images $\mathbf{I}_i = (I_i^1, I_i^2, I_i^3)$ are available, one possibility is to assume constancy of all color channels.

$$\begin{aligned} I_0^1(\mathbf{p_0}) - I_1^1(\mathbf{p_1}) &= 0 \\ I_0^2(\mathbf{p_0}) - I_1^2(\mathbf{p_1}) &= 0 \\ I_0^3(\mathbf{p_0}) - I_1^3(\mathbf{p_1}) &= 0 \end{aligned}$$

These new assumptions can be coupled and result in the new data term

$$E_{\text{Data}}(Z) = \int_{\Omega_0} \Psi\left(\sum_{c=1}^{3} |\Delta^c|^2 \right) dx\,dy \tag{87}$$

where $\Delta^c = I_0^c(\mathbf{p_0}) - I_1^c(\mathbf{p_1})$. After repeating the steps of Section 3.3 the new fixed point iteration is equivalent to Equation 49 with a new joint depth tensor that reads

$$T^{C,k} := \sum_{i=1}^{3} T^{c,k} = \sum_{i=1}^{3} \begin{pmatrix} (\Delta_Z^{c,k})^2 & \Delta_Z^{c,k}\Delta^{c,k} \\ \Delta_Z^{c,k}\Delta^{c,k} & (\Delta^{c,k})^2 \end{pmatrix}$$

Of course, this can be extended analogously to the additional gradient constancy assumption which gives the following data term

$$E_{\text{Data}}(Z) = \int_{\Omega_0} \Psi\left(\sum_{c=1}^{3} |\Delta^c|^2\right) + \gamma\,\Psi\left(\sum_{c=1}^{3} |\nabla\Delta^c|^2\right) \mathrm{d}x\mathrm{d}y \tag{88}$$

and results in the fixed point iteration of Equation 85 with the new joint gradient depth tensor

$$\begin{aligned} T_{\nabla}^{C,k} &:= \sum_{c=1}^{3} T_{\nabla}^{c,k} \\ &= \sum_{i=1}^{3} \begin{pmatrix} \left(\nabla\Delta^{c,k}\right)_Z \left(\nabla\Delta^{c,k}\right)_Z^\top & \left(\nabla\Delta^{c,k}\right)_Z \left(\nabla\Delta^{c,k}\right)^\top \\ \left(\nabla\Delta^{c,k}\right)_Z \left(\nabla\Delta^{c,k}\right)^\top & \left(\nabla\Delta^{c,k}\right) \left(\nabla\Delta^{c,k}\right)^\top \end{pmatrix} \end{aligned}$$

4.2.3 *Multiple Views*

As already mentioned, the used parameterization can be easily extended to use an arbitrary number of cameras. Therefore, the data term is modified to consider the information of all images and the extended version reads

$$E_{\text{Data}}(Z) = \int_{\Omega_0} \sum_i \Psi\left(|\Delta_i|^2\right) + \gamma\,\Psi\left(|\nabla\Delta_i|^2\right) \mathrm{d}x\mathrm{d}y \tag{89}$$

with $\Delta_i = I_0(\mathbf{p_0}) - I_i(\mathbf{p_i})$. The simplicity of this extension is the consequence of the parameterization with respect to the reference view. In contrast, disparity-driven approaches do not allow such an simple extension for arbitrary cases.

As before, this modification comes down to adding additional depth tensors to the fixed point iteration.

$$\begin{aligned} 0 = \sum_i \Big(& \left(\Psi'\right)_{\text{Data}}^{i,k,l} \cdot \left(T_{11}^{i,k} \cdot \mathrm{d}Z^{k,l} + T_{12}^{i,k}\right) \\ & + \left(\Psi'\right)_{\text{DataGradient}}^{i,k,l} \cdot \left(T_{\nabla 11}^{i,k} \cdot \mathrm{d}Z^{k,l} + T_{\nabla 12}^{i,k}\right) \Big) \\ & - \alpha\,\mathrm{div}\left(\left(\Psi'\right)_{\text{Smooth}}^{k,l} \cdot \nabla\left(Z^k + \mathrm{d}Z^{k,l+1}\right)\right) \end{aligned} \tag{90}$$

with the new abbreviations

$$\left(\Psi'\right)_{\text{Data}}^{i,k,l} := \Psi'\left(\mathbf{dZ}^{k,l\top}\, T^{i,k}\, \mathbf{dZ}^{k,l}\right) \tag{91}$$

$$\left(\Psi'\right)_{\text{DataGradient}}^{i,k,l} := \Psi'\left(\mathbf{dZ}^{k,l\top}\, T_{\nabla}^{i,k}\, \mathbf{dZ}^{k,l}\right) \tag{92}$$

and

$$T^{i,k} = \begin{pmatrix} (\Delta_Z^{i,k})^2 & \Delta_Z^{i,k}\Delta^{i,k} \\ \Delta_Z^{i,k}\Delta^{i,k} & (\Delta^{i,k})^2 \end{pmatrix} \tag{93}$$

$$T_{\nabla}^{i,k} := \begin{pmatrix} \left(\nabla\Delta^{i,k}\right)_Z^\top \left(\nabla\Delta^{i,k}\right)_Z & \left(\nabla\Delta^{i,k}\right)_Z^\top \left(\nabla\Delta^{i,k}\right) \\ \left(\nabla\Delta^{i,k}\right)_Z^\top \left(\nabla\Delta^{i,k}\right) & \left(\nabla\Delta^{i,k}\right)^\top \left(\nabla\Delta^{i,k}\right) \end{pmatrix} \tag{94}$$

5

EVALUATION AND EXPERIMENTS

5.1 EVALUATION METHODOLOGY

In order to judge and compare the quality of the developed method and the proposed extensions some sort of quality metric is needed. A commonly used approach is to compute error statistics with respect to ground truth data. For this purpose already benchmark datasets exist [10, 16]. These benchmarks mainly use the estimated disparity to evaluate the performance, because most stereo algorithms are disparity-driven. Since this thesis presents a depth-driven approach that operates in the 3-D space an additional 3-D quality metric is used. The two quality metrics are defined as follows:

- *Mean absolute error* of the disparity (MAE_d)

$$MAE_d = \frac{1}{|\Omega_0|} \sum_{(i,j)\in\Omega_0} |d_{i,j}^{truth} - d_{i,j}^{estimate}| \tag{95}$$

- *Mean absolute error* of the Euclidean distance (MAE_{3D})

$$MAE_{3D} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{P}\in\mathcal{P}} |\mathbf{P}^{truth} - \mathbf{P}^{estimate}| \tag{96}$$

With the use of multiple views situations can arise in which objects that are visible in one view are hidden behind other objects in the second view. Such constellations are called occlusions. Since it is impossible to find the correct correspondence for occlusions the error will likely increase in occluded regions. In order to allow a more detailed evaluation the metrics will be once applied to all pixels and once only for the non-occluded pixels. To determine the occluded pixels a forward/backward-check is used. Therefore, the sum of forward disparity $d_{i,j}^f$ and backward disparity $d_{i,j}^b$ is computed, which should vanish in case of non-occluded pixels. Because it is likely that values are needed that lie between pixels an error arises, since these values will be interpolated using a linear interpolation. Consequently, a threshold T_{occ} is introduced and occluded pixels are defined by

$$|d_{i,j}^f - d_{i+d_{i,j}^f, j+d_{i,j}^f}^b| > T_{occ}$$

In the following work this is threshold is set to $T_{occ} = 0.5$.

Apart from the computed error statistics an error visualization will be used to allow a visual evaluation of the accuracy. Hence, the signed

error will be displayed using a color coding from red over green to blue, where green indicates a correct depth. An illustration is seen in Figure 14. This allows to identify in which regions the error occurs and interpret it.

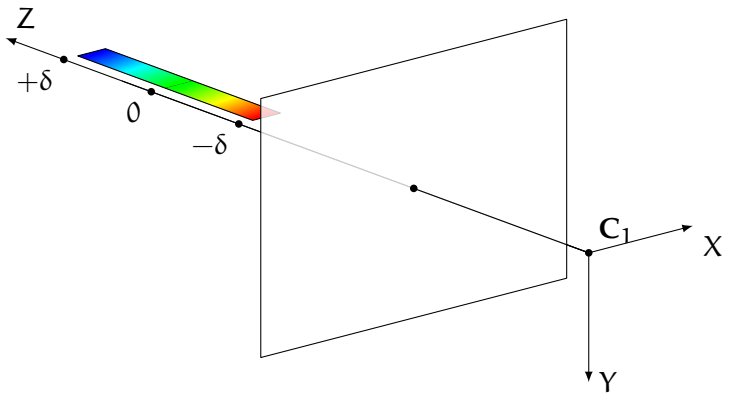


Figure 14: Color coding for the error visualization.

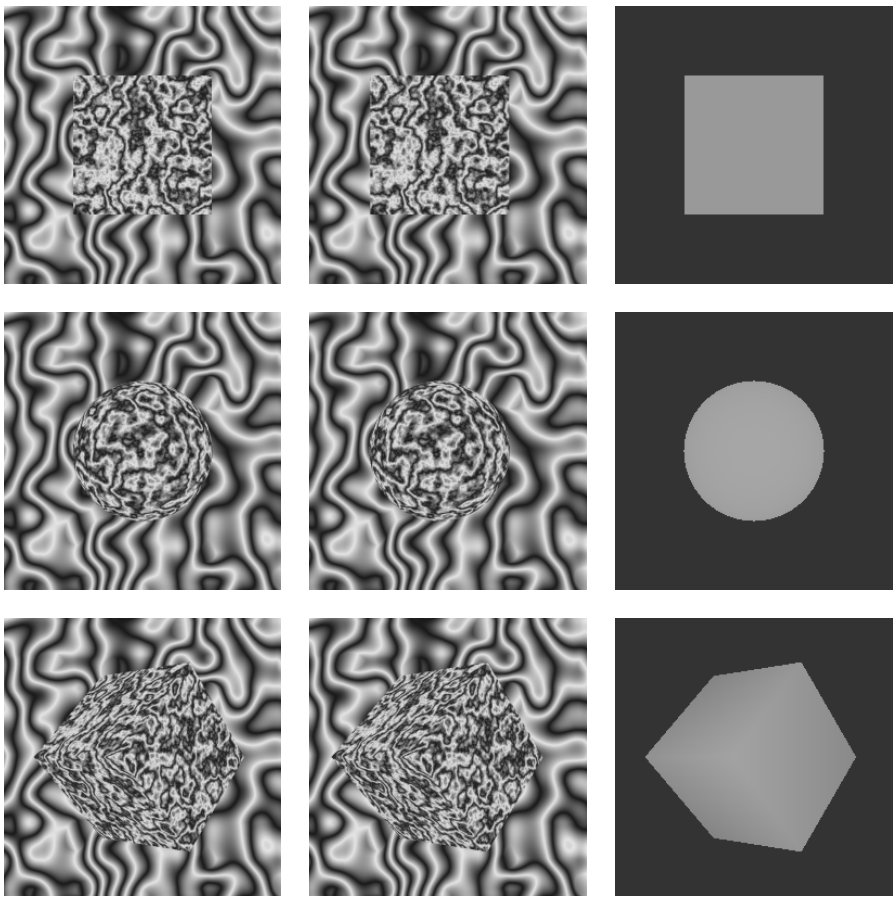
5.2 EVALUATION DATASETS

After introducing the metrics the last entity missing to perform the evaluation are test datasets. To start with simple scenes, without great complexity ray-traced images are used, which were created using the MegaPOV Ray-tracer [24] and the annotation patch [25]. The test scenes use a ortho-parallel camera setup and contain one object each, which is placed in front of a background parallel to the cameras. These objects are a cube, a sphere and a rotated cube. The test scenes are shown in Figure 16. In all scenes the background and the objects are well textured, what helps to minimize ambiguities.

As more complex test scenes the *Teddy* and *Cones* dataset, shown in Figure 15, of the Middlebury benchmark [17] are used. These datasets consist of rectified images with available ground truth disparities (quarter-pixel accuracy). Unfortunately, the benchmark does not provide the camera projection matrices, which are needed for the developed method. In order to overcome this problem, the reference camera is aligned with the origin and the extrinsic camera parameters of the second camera are set to a pure horizontal translation. This is possible because the images are rectified. The intrinsic camera parameters are set by choosing an arbitrary field of view angle, which determines the scaled focal length uniquely by the image size. In addition, the principle point is chosen to lie in the image center.

(a) Reference view (b) Second view (c) Ground truth

Figure 15: Evaluation datasets of the Middlebury benchmark [17]. *First row:* The *Teddy* dataset. *Second row:* The *Cones* dataset.



(a) Reference view (b) Second view (c) Ground truth

Figure 16: Simple test sequences created with the MegaPOV Ray-tracer. *First row:* The parallel cube dataset. *Second row:* The sphere dataset. *Third row:* The rotated cube dataset.

5.3 IMAGE DERIVATIVE

In Section 3.5 two possible strategies to compute the needed derivative $(\nabla_i I_i)^\top$ were presented

A. Relate the gradient of the back projected image with the gradient of the original image using the Jacobian.

B. Use the back projected gradient.

The previous introduced test scenes should help to decide the preferred method. Therefore, the proposed metrics are computed for all five datasets of Section 5.2, using the fixed parameters $\sigma_{pre} = 0.4$, $\eta = 0.98$, $k^{max} = 200$, $l^{max} = 4$, $\omega = 1.8$ and SOR-cycles $= 10$. The regularization parameter α has been optimized with respect to the MAE_{3D} of all pixels. Table 1 shows the computed metrics, which reveal that the method B gives slightly better results. On account of this experiment and to the fact that the approach B requires less computational effort, it is the preferred method and is used in the following experiments.

	all		non-occluded	
	MAE_{3D}	MAE_d	MAE_{3D}	MAE_d
PARALLEL CUBE				
A. ($\alpha = 0.14$)	0.019	0.035	0.009	0.018
B. ($\alpha = 0.15$)	0.018	0.032	0.008	0.017
SPHERE				
A. ($\alpha = 0.15$)	0.017	0.032	0.007	0.016
B. ($\alpha = 0.16$)	0.017	0.031	0.007	0.016
ROTATED CUBE				
A. ($\alpha = 0.15$)	0.021	0.038	0.012	0.024
B. ($\alpha = 0.16$)	0.020	0.035	0.011	0.022
TEDDY				
A. ($\alpha = 0.11$)	0.120	1.191	0.081	0.806
B. ($\alpha = 0.14$)	0.116	1.082	0.077	0.721
CONES				
A. ($\alpha = 0.27$)	0.083	1.220	0.049	0.671
B. ($\alpha = 0.34$)	0.073	1.108	0.045	0.624

Table 1: The computed metrics using the two presented approaches to compute the derivative $(\nabla_i I_i)^\top$.

5.4 COMPARISON OF THE SMOOTHNESS TERM

This section will compare the regularization strategies of the developed method, the proposed extensions (Section 4.1) and additionally homogeneous regularization. To implement homogeneous regularization the quadratic penalizer $\Psi(s^2) = s^2$ is used for the smoothness term. As data term the robust data term for gray value images of Section 3.2 is used, with a gray value version of the datasets. Again, the same parameters as in Section 5.3 are kept fixed with the same values and the regularization parameter α is optimized with respect to the MAE_{3D} of all pixels.
In Figure 18, Figure 19, Figure 20, Figure 21 (Figure 22) and Figure 23 (Figure 24) the datasets of Section 5.2 are used to compare the different type of regularizers.

- *Homogeneous Regularizers*. One can clearly see that the first order homogeneous regularizer (Figure 18b, Figure 19b, Figure 20b, Figure 21b and Figure 23b) gives oversmoothed results. The same holds for the second order homogeneous regularizer (Figure 18e, Figure 19e, Figure 20e, Figure 21e and Figure 23e).

- *Isotropic Depth-Driven Regularizers*. Using the isotropic depth-driven regularizer (Figure 18c, Figure 19c, Figure 20c, Figure 21c and Figure 23c), obvious sharper depth edges are achieved compared to the homogeneous regularizer. The error visualization shows as well, that the error shrinks at the depth edges.

  The second order isotropic regularizer (Figure 18f, Figure 19f, Figure 20f, Figure 21f and Figure 23f) improves the depth edges of it is homogeneous counterpart, but not as good as in the first oder case.

- *Isotropic Image- & Depth-Driven Regularizer*. Using the isotropic image- & depth-driven regularizer (Figure 18d, Figure 19d, Figure 20d, Figure 21d and Figure 23d) proposed in Section 4.1.1 does not show a noticeable visual improvement in these scenes. In the ray-traced images this is especially due to the fact that the texture is everywhere and does not indicate object boundaries.

- *Anisotropic Complementary Regularizer*. The anisotropic complementary regularizer (Figure 18g, Figure 19g, Figure 20g, Figure 21g and Figure 21g obtains sharp looking edges, but with aberrations. Especially the strong textured regions in the ray-traced images lead to lots of aberrations.

- *Anisotropic Depth-Driven Regularizer*. The anisotropic depth-driven regularizer (Figure 18h, Figure 19h, Figure 20h, Figure 21h and Figure 23h) seems to produce a very accurate depth edge. But a look at the error visualization shows, that the error appears more constant than in the isotropic depth-driven case.

The associated metrics are listed in Table 2. One may notice that the 3-D quality metric MAE_{3D} does not necessarily correlate with the 2-D quality metric MAE_d. To understand this non-correlation, a simple example is illustrated in Figure 17. It shows the 3-D point error and the corresponding disparity error.

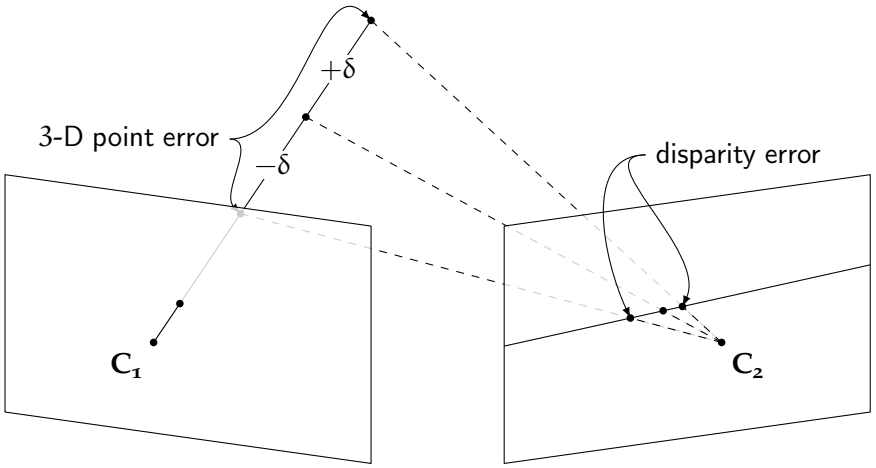


Figure 17: Illustration of the the relation between the 3-D error used for the MAE_{3D} and the 2-D disparity error used for the MAE_d.

As previously mentioned the homogeneous regularizers give over-smoothed depth edges this reflects also in the quality metrics. The anisotropic complementary regularizer does not perform to well on the ray-traced datasets, due to the misleading image boundary. In contrast it performs better on the *Teddy* and *Cones* datasets. The second order isotropic depth-driven regularizer appears to result in worse quality metrics as the first order isotropic depth-driven regularizer, due to less sharp depth edges. Even though the anisotropic depth-driven regularizer gives good looking visual results it does not achieve as good quality metrics as the isotropic depth-driven regularizer. Finally, the isotropic image- & depth-driven regularizer performs a little better on the *Teddy* and *Cones* datasets according to the quality metrics, but not on all ray-traced datasets. In case of the sphere dataset it deteriorates the quality metric.

Summarizing, the isotropic depth-driven regularizer gives the best overall results in the ray-traced scenes. Whereas, the Isotropic Image- & Depth-Driven Regularizer and the anisotropic complementary Regularizer give better results on the more realistic *Teddy* and *Cones* datasets.

	all		non-occluded	
	MAE_{3D}	MAE_d	MAE_{3D}	MAE_d
Parallel cube				
Homogeneous	0.079	0.120	0.067	0.102
Second order homogeneous	0.075	0.112	0.063	0.093
Anisotropic Complementary	0.056	0.097	0.045	0.081
Second order isotropic	0.038	0.059	0.026	0.040
Anisotropic depth-driven	0.019	0.032	0.008	0.015
Isotropic depth-driven	0.018	0.032	0.008	0.017
Isotropic image- & depth-driven	0.017	0.033	0.007	0.018
Sphere				
Homogeneous	0.071	0.110	0.060	0.093
Second order homogeneous	0.062	0.095	0.050	0.078
Isotropic image- & depth-driven	0.053	0.665	0.044	0.666
Anisotropic Complementary	0.048	0.093	0.039	0.080
Second order isotropic	0.030	0.049	0.018	0.032
Anisotropic depth-driven	0.024	0.043	0.015	0.029
Isotropic depth-driven	0.017	0.031	0.007	0.016
Rotated cube				
Homogeneous	0.091	0.129	0.083	0.117
Second order homogeneous	0.084	0.118	0.075	0.106
Anisotropic Complementary	0.038	0.060	0.031	0.050
Second order isotropic	0.034	0.053	0.025	0.039
Anisotropic depth-driven	0.025	0.041	0.016	0.027
Isotropic image- & depth-driven	0.021	0.042	0.013	0.031
Isotropic depth-driven	0.020	0.035	0.011	0.022
Teddy				
Second order homogeneous	0.193	1.559	0.160	1.179
Homogeneous	0.181	1.467	0.147	1.135
Anisotropic depth-driven	0.163	1.330	0.128	1.005
Second order isotropic	0.149	1.279	0.111	0.869
Isotropic depth-driven	0.116	1.068	0.076	0.710
Anisotropic Complementary	0.115	1.001	0.082	0.712
Isotropic image- & depth-driven	0.112	1.051	0.074	0.697
Cones				
Second order homogeneous	0.094	1.318	0.066	0.839
Homogeneous	0.090	1.284	0.063	0.829
Second order isotropic	0.084	1.216	0.058	0.775
Anisotropic depth-driven	0.074	1.122	0.047	0.638
Anisotropic Complementary	0.073	1.095	0.048	0.647
Isotropic depth-driven	0.073	1.108	0.045	0.624
Isotropic image- & depth-driven	0.072	1.111	0.044	0.611

Table 2: The quality metrics corresponding to Figure 18, Figure 19, Figure 20, Figure 21 and Figure 23.

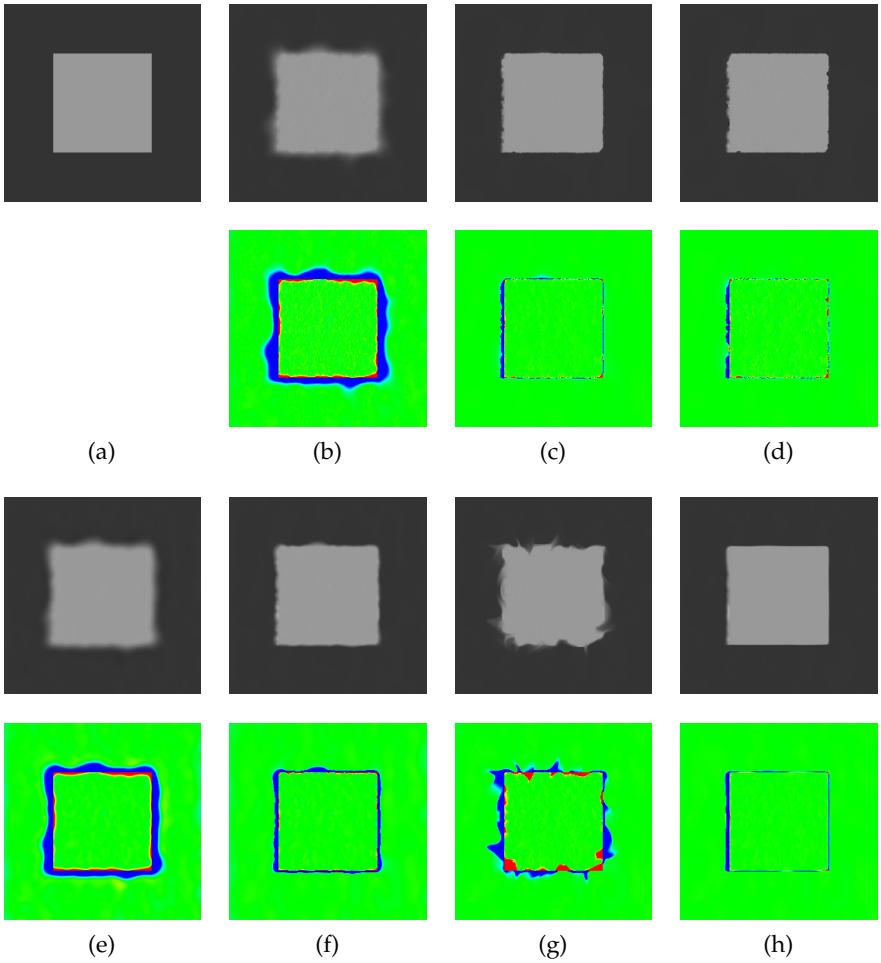


Figure 18: Parallel cube dataset with different smoothness terms. *First row, from left to right:* (a) Ground truth depth values. (b) Depth values and error visualization for the homogeneous regularization ($\alpha = 2.0$). (c) Isotropic depth-driven regularization ($\alpha = 0.15$). *Second row, from left to right:* (d) Isotropic image- and depth-driven regularization ($\alpha = 0.23$, $\lambda = 0.14$). (e) Second order homogeneous regularization ($\alpha = 20.0$). (f) Second order isotropic depth-driven regularization ($\alpha = 0.3$). *Second row, from left to right:* (g) Anisotropic complementary regularization ($\alpha = 9.5$, $\lambda = 0.01$, $\rho = 3$). (h) Anisotropic depth-driven regularization ($\alpha = 16.8$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$).

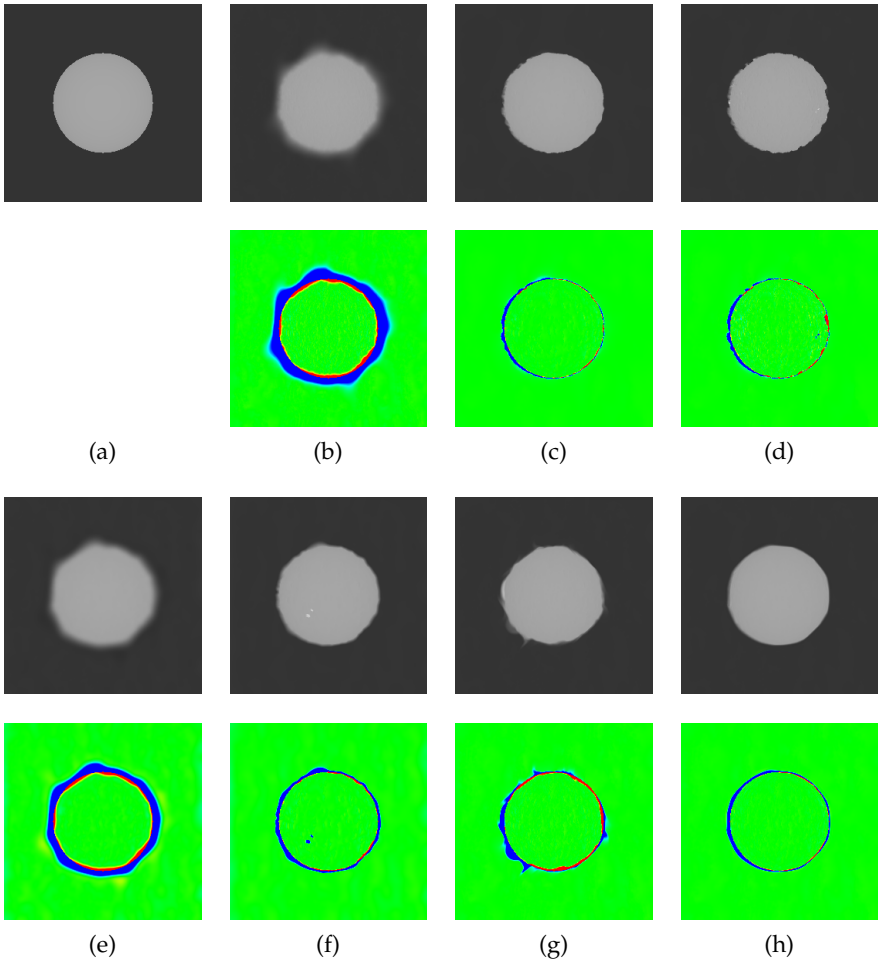


Figure 19: Sphere dataset with different smoothness terms. *First row, from left to right:* (a) Ground truth depth values. (b) Depth values and error visualization for the homogeneous regularization ($\alpha = 3.0$). (c) Isotropic depth-driven regularization ($\alpha = 0.17$). *Second row, from left to right:* (d) Isotropic image- and depth-driven regularization ($\alpha = 0.15$, $\lambda = 0.14$). (e) Second order homogeneous regularization ($\alpha = 25.0$). (f) Second order isotropic depth-driven regularization ($\alpha = 0.17$). *Second row, from left to right:* (g) Anisotropic complementary regularization ($\alpha = 9.5$, $\lambda = 0.01$, $\rho = 3$). (h) Anisotropic depth-driven regularization ($\alpha = 15.0$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$).

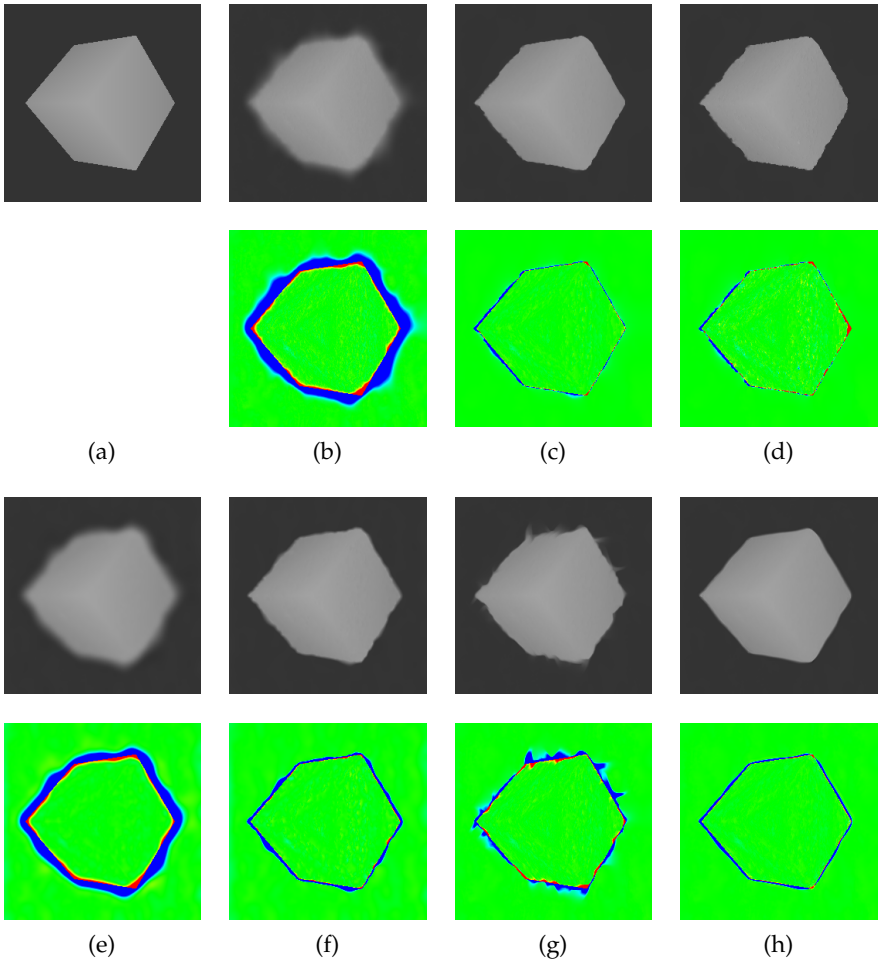


Figure 20: Rotated cube dataset with different smoothness terms. *First row, from left to right:* (a) Ground truth depth values. (b) Depth values and error visualization for the homogeneous regularization ($\alpha = 3.0$). (c) Isotropic depth-driven regularization. ($\alpha = 0.16$) *Second row, from left to right:* (d) Isotropic image- and depth-driven regularization ($\alpha = 0.2$, $\lambda = 0.14$). (e) Second order homogeneous regularization ($\alpha = 40.0$). (f) Second order isotropic depth-driven regularization ($\alpha = 0.13$). *Second row, from left to right:* (g) Anisotropic complementary regularization ($\alpha = 9.5$, $\lambda = 0.01$, $\rho = 3$). (h) Anisotropic depth-driven regularization ($\alpha = 13.0$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$).

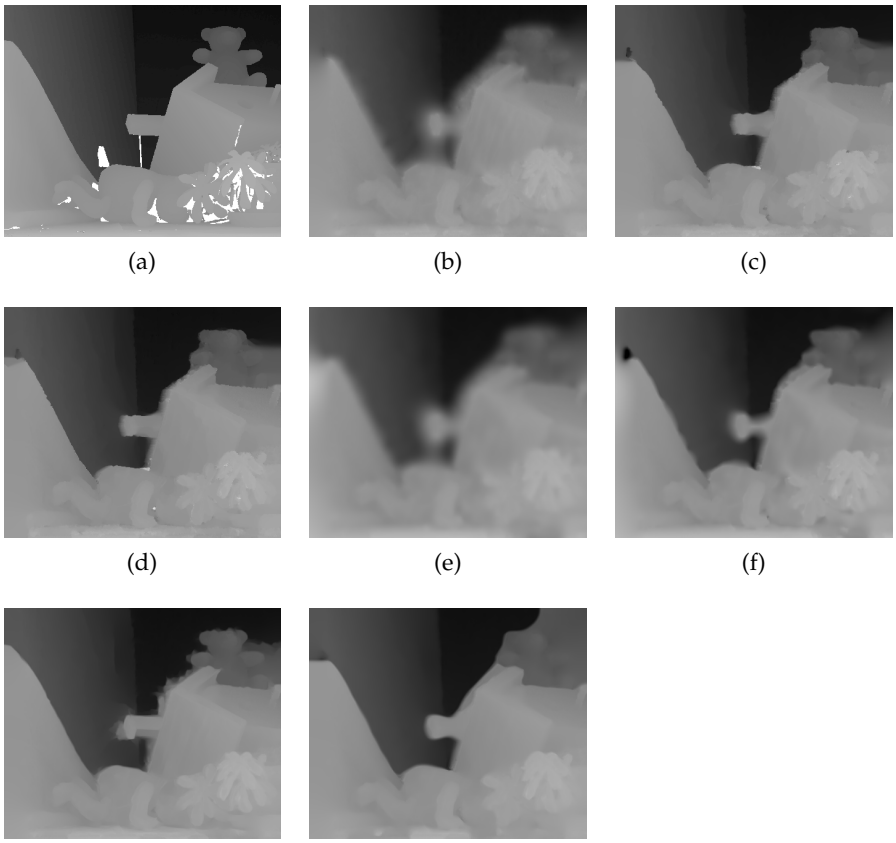


Figure 21: Results for the *Teddy* dataset of the Middlebury benchmark [17] with different smoothness terms. *First row, from left to right:* (a) Ground truth depth values, the white regions indicate that no ground truth values are available. (b) Depth values for the homogeneous regularization ($\alpha = 5.0$). (c) Isotropic depth-driven regularization ($\alpha = 0.14$). *Second row, from left to right:* (d) Isotropic image- and depth-driven regularization ($\alpha = 0.16$, $\lambda = 0.14$). (e) Second order homogeneous regularization ($\alpha = 120.0$). (f) Second order isotropic depth-driven regularization ($\alpha = 0.291$). *Second row, from left to right:* (g) Anisotropic complementary regularization ($\alpha = 13.21$, $\lambda = 0.01$, $\rho = 3$). (h) Anisotropic depth-driven regularization ($\alpha = 24.6$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$).

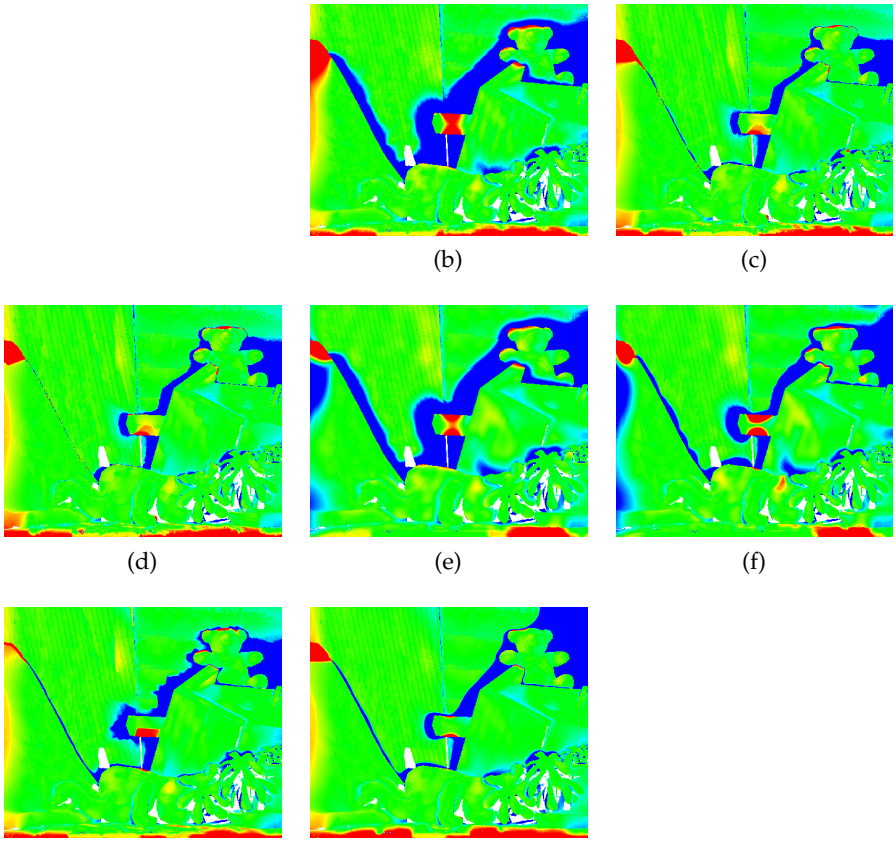
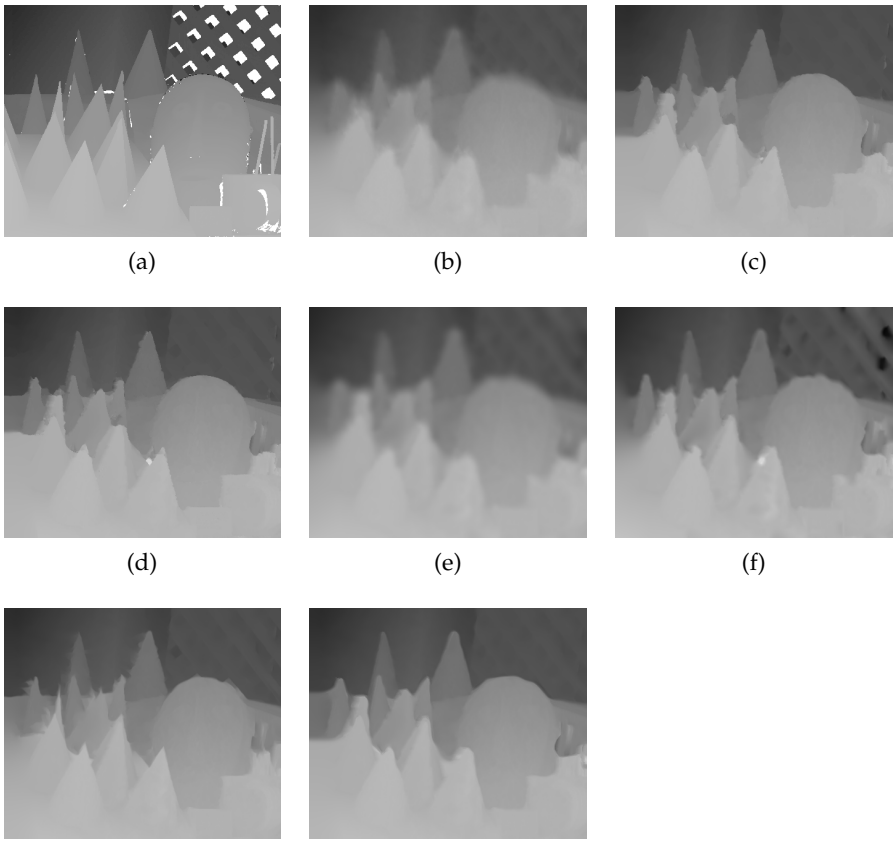


Figure 22: Error visualization for the *Teddy* dataset of the Middlebury benchmark [17] with different smoothness terms. *First row, from left to right:* (b) Error visualization for the homogeneous regularization. ($\alpha = 5.0$). (c) Isotropic depth-driven regularization ($\alpha = 0.14$). *Second row, from left to right:* (d) Isotropic image- and depth-driven regularization ($\alpha = 0.16$, $\lambda = 0.14$). (e) Second order homogeneous regularization ($\alpha = 120.0$). (f) Second order isotropic depth-driven regularization ($\alpha = 0.291$). *Second row, from left to right:* (g) Anisotropic complementary regularization ($\alpha = 13.21$, $\lambda = 0.01$, $\rho = 3$). (h) Anisotropic depth-driven regularization ($\alpha = 24.6$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$).



(a) (b) (c)

(d) (e) (f)

(g) (h)

Figure 23: Results for the *Cones* dataset of the Middlebury benchmark [17] with different smoothness terms. *First row, from left to right:* (a) Ground truth depth values, the white regions indicate that no ground truth values are available. (b) Depth values for the homogeneous regularization ($\alpha = 13.3$). (c) Isotropic depth-driven regularization ($\alpha = 0.34$). *Second row, from left to right:* (d) Isotropic image- and depth-driven regularization ($\alpha = 0.43$, $\lambda = 0.1$). (e) Second order homogeneous regularization ($\alpha = 200.0$). (f) Second order isotropic depth-driven regularization ($\alpha = 0.44$). *Second row, from left to right:* (g) Anisotropic complementary regularization ($\alpha = 26.4$, $\lambda = 0.01$, $\rho = 3$). (h) Anisotropic depth-driven regularization ($\alpha = 28.1$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$).

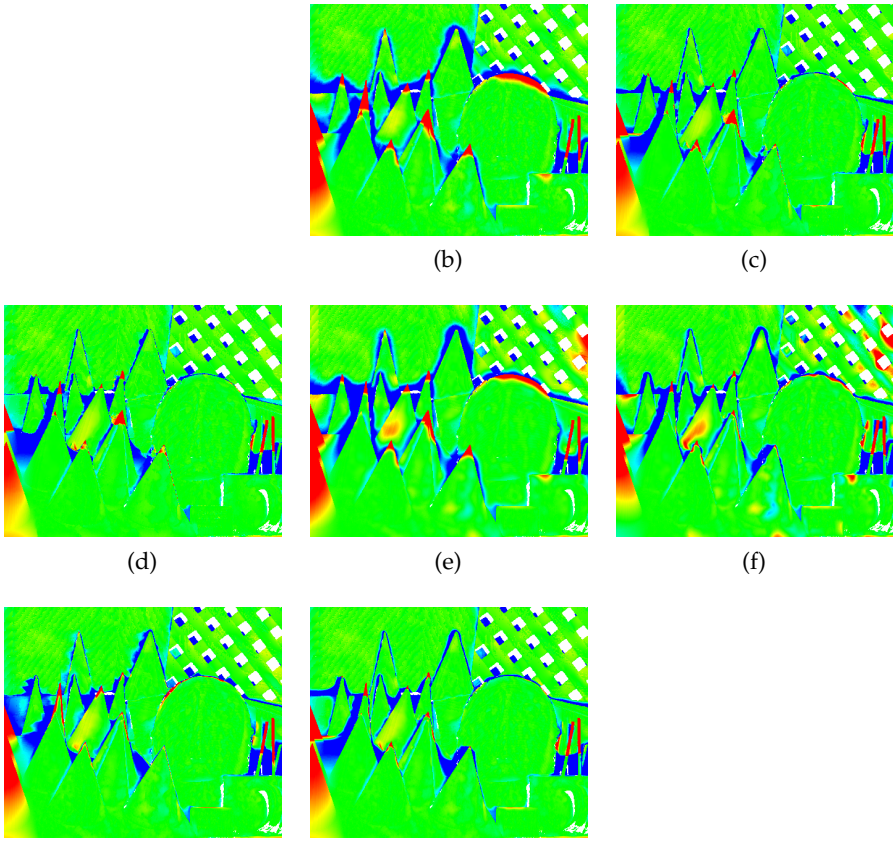
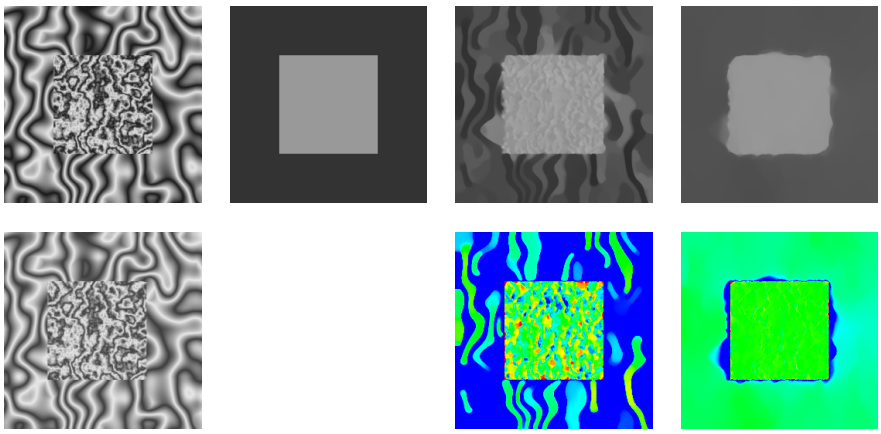


Figure 24: Results for the *Cones* dataset of the Middlebury benchmark [17] with different smoothness terms. *First row, from left to right:* (b) Error visualization for the homogeneous regularization ($\alpha = 13.3$). (c) Isotropic depth-driven regularization ($\alpha = 0.34$). *Second row, from left to right:* (d) Isotropic image- and depth-driven regularization ($\alpha = 0.43$, $\lambda = 0.1$). (e) Second order homogeneous regularization ($\alpha = 200.0$). (f) Second order isotropic depth-driven regularization ($\alpha = 0.44$). *Second row, from left to right:* (g) Anisotropic complementary regularization ($\alpha = 26.4$, $\lambda = 0.01$, $\rho = 3$). (h) Anisotropic depth-driven regularization ($\alpha = 28.1$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$).

5.5 DATA TERM COMPARISON

The previous section compared and examined the presented smoothness terms. In this section some experiments will help to do the same with the proposed data terms. As smoothness term the isotropic depth-driven regularizer is used.

BRIGHTNESS ENHANCEMENT. In order to demonstrate the usefulness of the gradient constancy, the cube dataset is modified and evaluated. The modification consists of a global additive brightness enhancement in the second view. Figure 25a shows the modified views. Now the depth values are computed once, using only the brightness constancy assumption (Figure 25c) and with a additional gradient constancy assumption (Figure 25d). Comparing both results shows that the additional gradient constancy allows to compute a satisfiable result, while the brightness constancy alone leads to a poor result.



(a) (b) (c) (d)

Figure 25: Modified cube dataset, where the brightness is increased in the second view. *First row, from left to right:* (a) Unmodified reference view and second view (increased brightness). (b) Ground truth depth values. (c) Depth values and error visualization using only the brightness constancy assumption ($\alpha = 0.15$). (d) Brightness constancy assumption and the gradient constancy assumption ($\gamma = 5.0$, $\alpha = 0.6$).

After showing the benefits of the gradient constancy assumption in case of brightness changes, the previously evaluated *Teddy* and *Cones* datasets are evaluated once more using gray value images with additional gradient constancy, RGB value images and RGB value images with additional gradient constancy. The output is shown in Figure 26 and Figure 27.

- *Additional Gradient constancy with gray value images*. In case of the *Teddy* dataset an improvement of the depth values at the

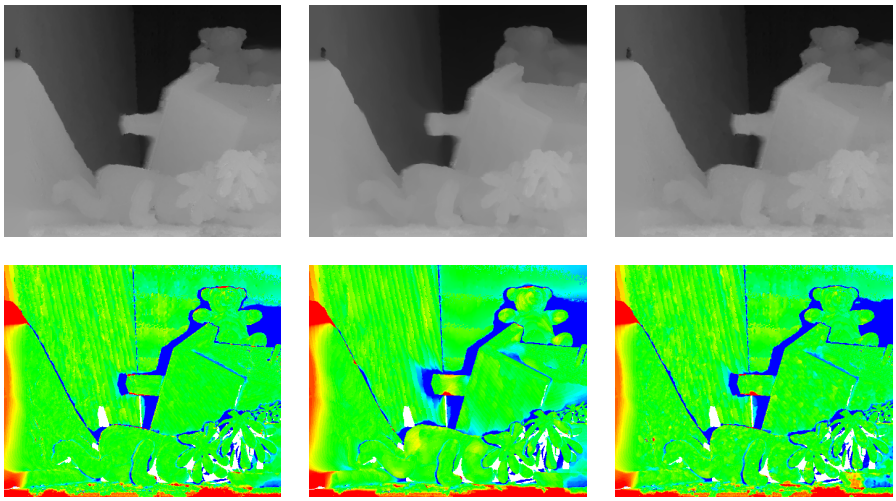
chimney and stuffed toy up front. In the *Cones* dataset no clear visual improvement is noticeable.

- *RGB value images images*. In case of the *Teddy* dataset no clear visual improvement is noticeable. In the *Cones* dataset an improvement of the depth values of the cones can be observed.
- *Additional Gradient constancy with RGB value images*. In case of the *Teddy* dataset some the error visualization reveals additional errors compared to the run with the gray value images. In the *Cones* dataset a further improvement is observed.

After a look at the visual results the related metrics, given in Table 3, are analyzed. They reveal that all extensions lead to an improvement. In case of the *Teddy* dataset the gray value images with additional gradient constancy lead to the best quality metrics. For the *Cones* dataset the RGB images with additional gradient constancy lead to the best quality metrics. In both cases the combination of RGB images and additional gradient constancy improves the quality metrics compared to the gray value images without gradient constancy. Hence the following experiments will use the combination of RGB images and additional gradient constancy assumption as data term.

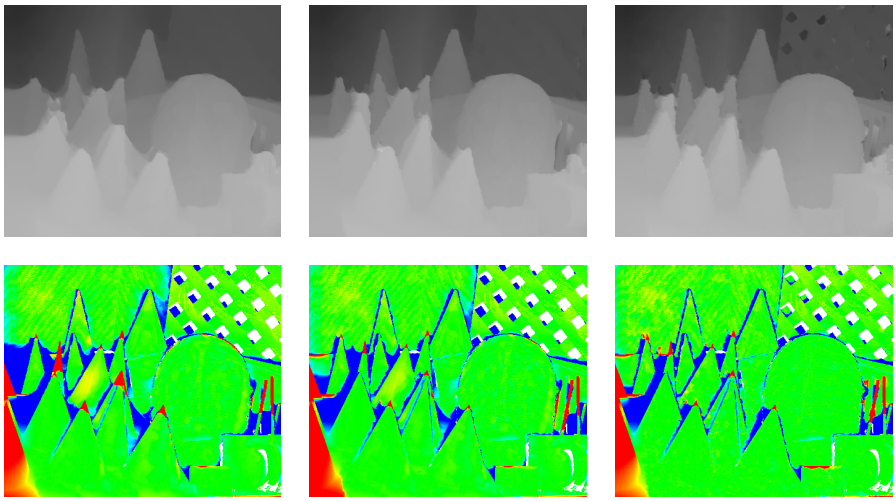
	all		non-occluded	
	MAE_{3D}	MAE_d	MAE_{3D}	MAE_d
Teddy				
Gray value constancy	0.116	1.068	0.076	0.710
RGB value constancy	0.115	1.061	0.075	0.692
Gray & gradient constancy	0.100	0.900	0.059	0.531
RGB & gradient constancy	0.107	0.958	0.067	0.596
Cones				
Gray value constancy	0.073	1.108	0.045	0.624
RGB value constancy	0.071	1.081	0.040	0.536
Gray & gradient constancy	0.067	1.017	0.040	0.533
RGB & gradient constancy	0.067	1.006	0.037	0.490

Table 3: The quality metrics corresponding to Figure 26 and Figure 27.



(a) (b) (c)

Figure 26: Results for the *Teddy* dataset of the Middlebury benchmark [17] with different data terms using the isotropic depth-driven smoothness term. *First row, from left to right:* (a) Depth values and error visualization using gray value images and additional gradient constancy assumption ($\gamma = 5.0$, $\alpha = 0.31$). (b) RGB color images ($\alpha = 0.28$). (c) RGB color images and additional gradient constancy assumption ($\gamma = 5.0$, $\alpha = 0.54$).



(a) (b) (c)

Figure 27: Results for the *Cones* dataset of the Middlebury benchmark [17] with different data terms using the isotropic depth-driven smoothness term. *First row, from left to right:* (a) Depth values and error visualization using gray value images and additional gradient constancy assumption ($\gamma = 5.0$, $\alpha = 0.7$). (b) RGB color images ($\alpha = 0.62$). (c) RGB color images and additional gradient constancy assumption ($\gamma = 5.0$, $\alpha = 1.01$).

5.6 A DISPARITY-DRIVEN APPROACH

As already mentioned in the introduction, this thesis aims at a comparison between the developed variational depth-driven method and a variational disparity-driven approach. This section introduces the variational disparity-driven approach, which will then be compared with developed method.

As basis the method of Zimmer et al.[29] is used which assumes an ortho-parallel camera setup. This results in a pure horizontal displacement denoted by u (disparity). As for the developed depth-driven method the used energy functional of the disparity-driven method has the following form

$$E(u) = E_{\text{Data}}(u) + \alpha\, E_{\text{Smooth}}(u) \tag{97}$$

The data term $E_{\text{Data}}(u)$ and the smoothness term $E_{\text{Smooth}}(u)$ are chosen accordingly to the depth-driven method, in order to have equivalent assumptions in both approaches. For example the basic method

$$E(Z) = \int_{\Omega_0} \Psi\left(|I_0(\mathbf{p_0}) - I_1(\mathbf{p_1})|^2\right) + \alpha\, \Psi\left(|\nabla Z|^2\right) dxdy$$

is compared to

$$E(u) = \int_{\Omega_0} \Psi\left(|I_0(x,y) - I_1(x+u,y)|^2\right) + \alpha\, \Psi\left(|\nabla u|^2\right) dxdy$$

The minimization is achieved similarly using the fixed-point iteration of Brox et al. [4] and will be not further explained here.

The next experiment uses the *Teddy* and the *Cones* dataset of Section 5.2 to compare the depth-driven parameterization and the disparity-driven parameterization. As data term the brightness and gradient constancy assumption with the RGB valued version of the datasets are used.

- *Isotropic Depth/Disparity-Driven*. In the *Teddy* dataset (Figure 28a and Figure 29a) only slight differences are noticeable. The depth-driven approach seems to perform a bit better in the background, whereas the disparity-driven approach performs better in the foreground. In the *Cones* dataset (Figure 30a and Figure 31a) the depth-driven approach is slightly better at the location between the front middle cone and the cone next to it.

- *Isotropic Image- & Depth/Disparity-Driven Regularizer*. In the *Teddy* dataset (Figure 28b and Figure 29b) the disparity-driven approach resolves the depth of the blanket (on the left side) better, while the depth-driven approach resolves the chimney better. The *Cones* dataset (Figure 30b and Figure 31b) shows almost no noticeable differences.

- *Second Order Isotropic Depth/Disparity-Driven*. In the *Teddy* dataset (Figure 28c and Figure 29c) the depth-driven approach resolves the background better, but the disparity-driven approach performs better at the left image boundary and at the right side of the teddy. In the *Cones* dataset (Figure 30b and Figure 31b) only the left bottom corner is resolved better by the depth-driven approach.

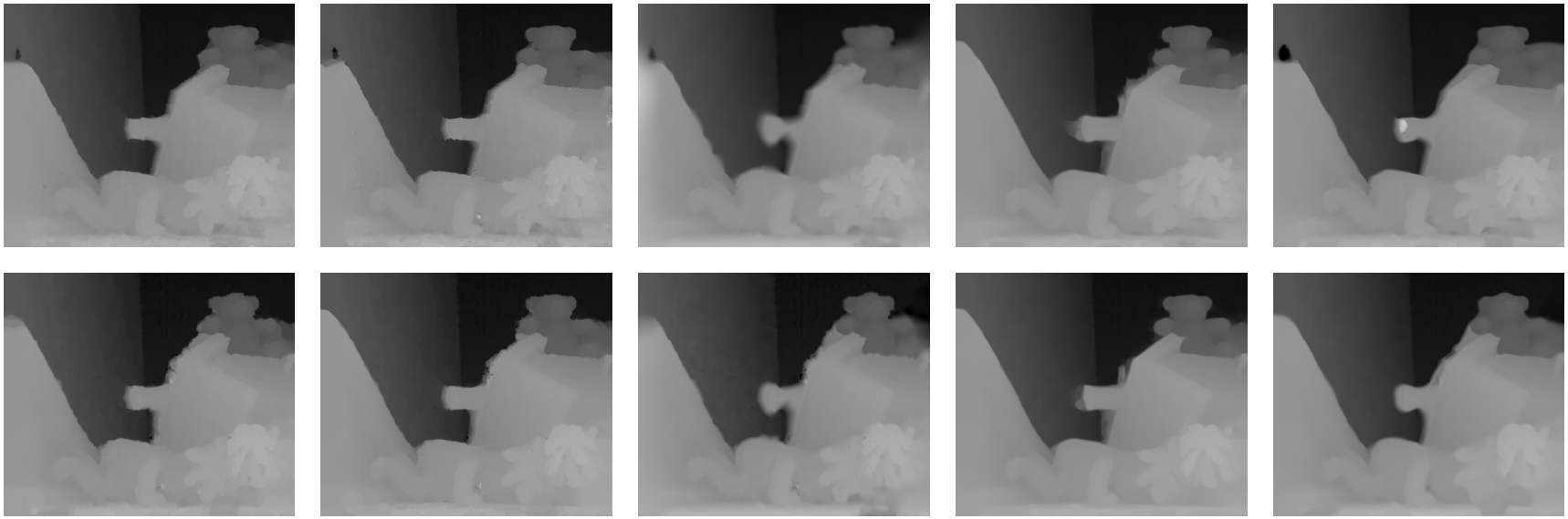
- *Anisotropic Complementary Regularizer*. In both datasets the disparity-driven approach leads to less aberrations at the depth edges, than the depth-driven approach.

- *Anisotropic Depth/Disparity-Driven Regularizer*. In the *Teddy* dataset (Figure 28c and Figure 29c) the depth-driven approach has errors on the chimney and at the top edge of the blanket (on the left side), in contrast to the disparity-driven approach. In the *Cones* dataset (Figure 30b and Figure 31b) almost no noticeable differences can be observed.

The associated metrics are listed in Table 4. It shows that in the *Teddy* dataset disparity-driven approach achieves in most cases better metrics. In contrast, the depth-driven approach obtains better metrics in most cases of the *Cones* dataset. The error difference is smaller in the non-occluded regions and again does not necessary correlate with the disparity error. In both scenes the *Anisotropic Complementary Regularizer* achieves the best results.

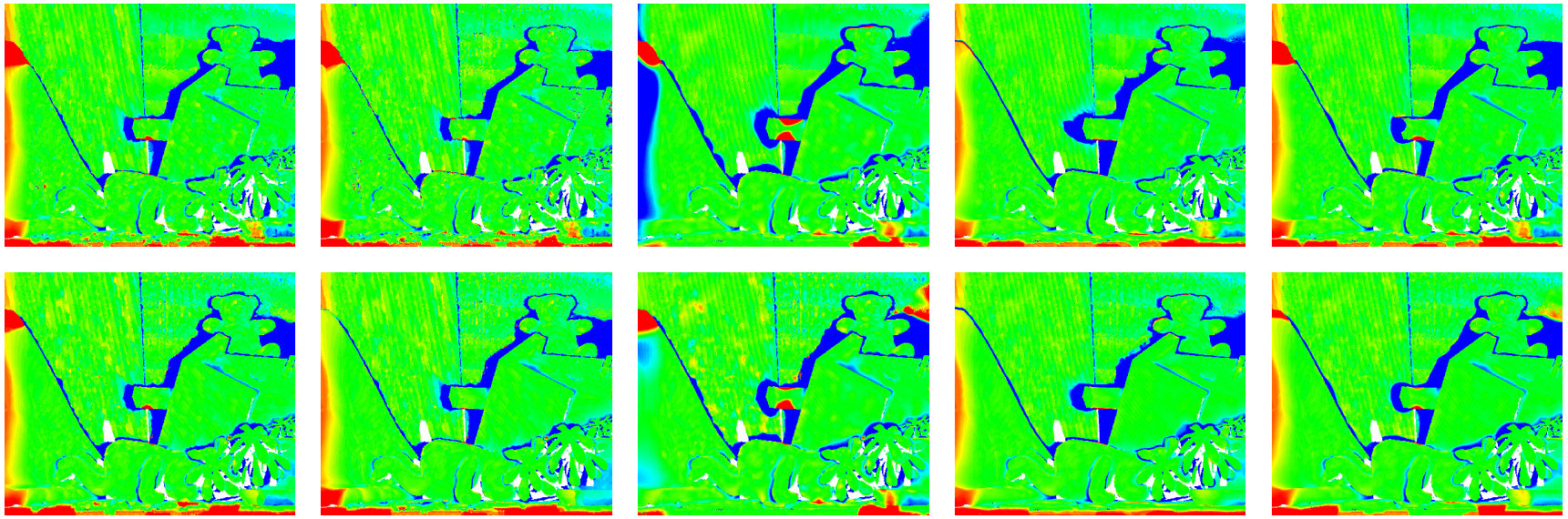
	all		non-occluded	
	MAE_{3D}	MAE_d	MAE_{3D}	MAE_d
Teddy (depth-driven parameterization)				
Isotropic depth-driven	0.107	0.958	0.067	0.596
Isotropic image-& depth-driven	0.102	0.936	0.060	0.553
Second order isotropic	0.122	1.118	0.079	0.600
Anisotropic complementary	0.100	0.867	0.067	0.577
Anisotropic depth-driven	0.118	1.023	0.073	0.639
Teddy (disparity-driven parameterization)				
Isotropic disparity-driven	0.104	0.956	0.066	0.613
Isotropic image-& disp.-driven	0.098	0.915	0.064	0.603
Second order isotropic	0.107	0.872	0.073	0.587
Anisotropic complementary	0.086	0.776	0.061	0.541
Anisotropic disparity-driven	0.106	0.933	0.070	0.608
Cones (depth-driven parameterization)				
Isotropic depth-driven	0.067	1.006	0.037	0.490
Isotropic image-& depth-driven	0.065	0.973	0.037	0.483
Second order isotropic	0.069	0.980	0.039	0.498
Anisotropic complementary	0.066	1.007	0.040	0.527
Anisotropic depth-driven	0.067	1.011	0.038	0.502
Cones (disparity-driven parameterization)				
Isotropic disparity-driven	0.069	1.036	0.040	0.526
Isotropic image-& disp.-driven	0.067	1.000	0.040	0.516
Second order isotropic	0.070	1.022	0.041	0.522
Anisotropic complementary	0.061	0.902	0.039	0.493
Anisotropic disparity-driven	0.073	1.035	0.044	0.527

Table 4: The quality metrics corresponding to Figure 28 (Figure 29) and Figure 30 (Figure 31).



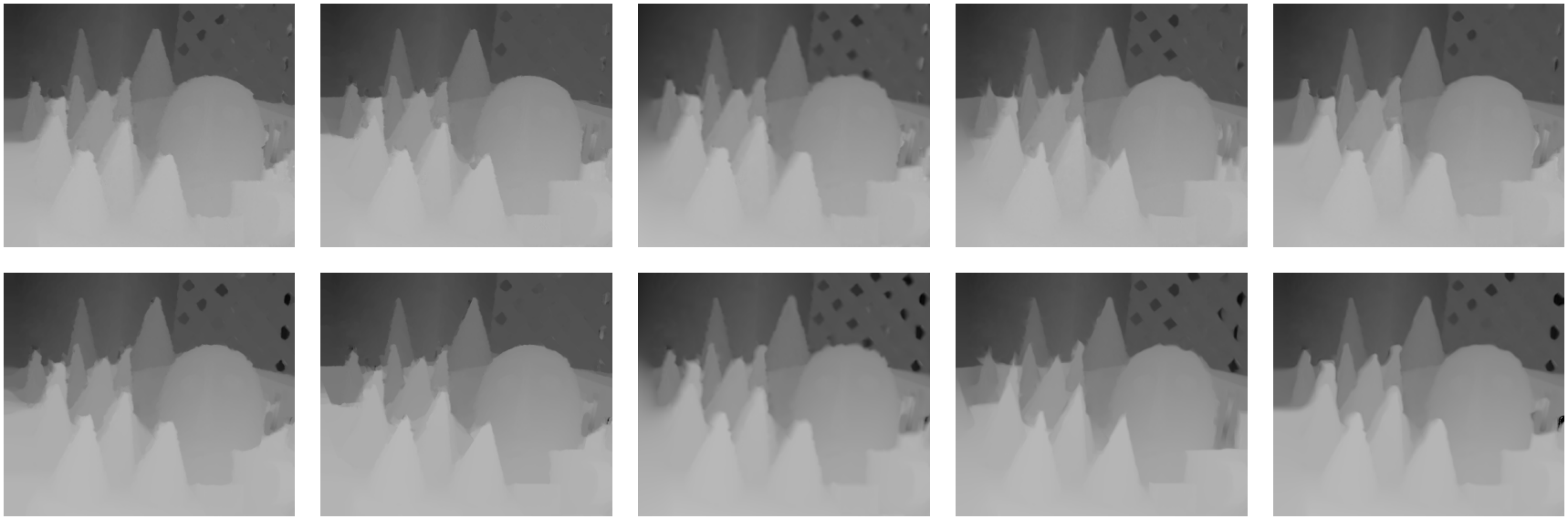
(a) (b) (c) (d) (e)

Figure 28: Results for the *Teddy* dataset of the Middlebury benchmark [17] with a brightness and gradient constancy data with RGB values and different smoothness terms. *First row, from left to right:* (a) Depth values for the isotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 0.54$, *Bottom :* disparity-driven $\alpha = 0.05$). (b) Isotropic image- and depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 0.56$, $\lambda = 0.14$, *Bottom :* disparity-driven $\alpha = 0.08$, $\lambda = 0.14$). (c) Second order isotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 1.5$, *Bottom :* disparity-driven $\alpha = 0.06$). (d) Anisotropic complementary regularization (*Top :* depth-driven $\alpha = 64.1$, $\lambda = 0.01$, $\rho = 3$, *Bottom :* disparity-driven $\alpha = 0.9$, $\lambda = 0.1$, $\rho = 3$). (e) Anisotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 56.0$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$, *Bottom :* disparity-driven $\alpha = 0.83$, $\lambda = 0.1$, $\rho = 3$, $\sigma = 1.5$).



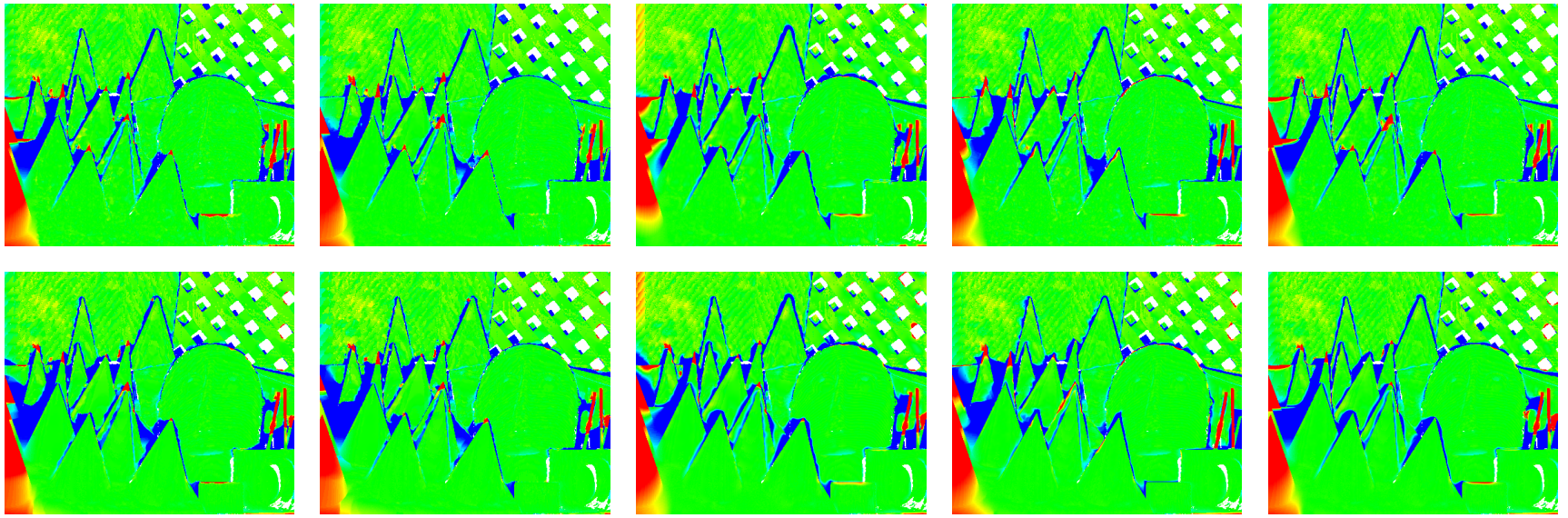
(a) (b) (c) (d) (e)

Figure 29: Results for the *Teddy* dataset of the Middlebury benchmark [17] with a brightness and gradient constancy data with RGB values and different smoothness terms. *First row, from left to right:* (a) Error visualization for the isotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 0.54$, *Bottom :* disparity-driven $\alpha = 0.05$). (b) Isotropic image- and depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 0.56$, $\lambda = 0.14$, *Bottom :* disparity-driven $\alpha = 0.08$, $\lambda = 0.14$). (c) Second order isotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 1.5$, *Bottom :* disparity-driven $\alpha = 0.06$). (d) Anisotropic complementary regularization (*Top :* depth-driven $\alpha = 64.1$, $\lambda = 0.01$, $\rho = 3$, *Bottom :* disparity-driven $\alpha = 0.9$, $\lambda = 0.1$, $\rho = 3$). (e) Anisotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 56.0$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$, *Bottom :* disparity-driven $\alpha = 0.83$, $\lambda = 0.1$, $\rho = 3$, $\sigma = 1.5$).



(a) (b) (c) (d) (e)

Figure 30: Results for the *Cones* dataset of the Middlebury benchmark [17] with a brightness and gradient constancy data with RGB values and different smoothness terms. *First row, from left to right:* (a) Depth values with isotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 1.01$, *Bottom :* disparity-driven $\alpha = 0.14$). (b) Depth values for the isotropic image- and depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 2.28$, $\lambda = 0.14$, *Bottom :* disparity-driven $\alpha = 0.17$, $\lambda = 0.14$). (c) Second order isotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 1.7$, *Bottom :* disparity-driven $\alpha = 0.16$). (d) Anisotropic complementary regularization (*Top :* depth-driven $\alpha = 64.0$, $\lambda = 0.01$, $\rho = 3$, *Bottom :* disparity-driven $\alpha = 1.33$, $\lambda = 0.1$, $\rho = 3$). (e) Anisotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 85.1$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$, *Bottom :* disparity-driven $\alpha = 1.37$, $\lambda = 0.1$, $\rho = 3$, $\sigma = 1.5$).



(a) (b) (c) (d) (e)

Figure 31: Results for the *Teddy* dataset of the Middlebury benchmark [17] with a brightness and gradient constancy data with RGB values and different smoothness terms. *First row, from left to right:* (a) Error visualization for the isotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 1.01$, *Bottom :* disparity-driven $\alpha = 0.14$). (b) Isotropic image- and depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 2.28$, $\lambda = 0.14$, *Bottom :* disparity-driven $\alpha = 0.17$, $\lambda = 0.14$). (c) Second order isotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 1.7$, *Bottom :* disparity-driven $\alpha = 0.16$). (d) Anisotropic complementary regularization (*Top :* depth-driven $\alpha = 64.0$, $\lambda = 0.01$, $\rho = 3$, *Bottom :* disparity-driven $\alpha = 1.33$, $\lambda = 0.1$, $\rho = 3$). (e) Anisotropic depth/disparity-driven regularization (*Top :* depth-driven $\alpha = 85.1$, $\lambda = 0.01$, $\rho = 3$, $\sigma = 1.5$, *Bottom :* disparity-driven $\alpha = 1.37$, $\lambda = 0.1$, $\rho = 3$, $\sigma = 1.5$).

6 CONCLUSION

In this thesis a depth-driven variational stereo method has been presented, with a detailed step by step explanation, based on the recently presented method of Basha et al. [2]. Moreover, several extensions have been proposed to improve the developed method.

First the basic concepts and notations have been introduced in Chapter 2. With the fundamental knowledge ready to use, the depth-driven method was formed in Chapter 3. In Chapter 4 several extensions were presented, starting with extensions of the smoothness term, where among others information of the image edges was embedded and an anisotropic adaption of the smoothing behavior was incorporated. Secondly, the data term was modified to allow the use of more advanced constraints, such as the gradient constancy, the consideration of images with a RGB color representation and the use of additional views.

In the final part, Chapter 5, an evaluation methodology and test datasets were presented, which where used to compare all extensions with each other. In addition, a comparison with a disparity-driven method was performed.

In contrast to the 2-D disparity parameterization the 3-D point cloud parameterization brings several advantages, such as the simple extension to multiple views and the possibility to easily incorporate additional prior knowledge on the depth. Despite the fact that the point cloud parameterization yields a non-linear relation between the image coordinates and the unknowns (unlike the disparity parameterization) that lead to non-trivial computations, it did not show convergence problems and in some scenes even outperforms the disparity parameterization.

The evaluation shows that the isotropic depth-driven regularizer used in the basic developed method performs very well, almost best in all used ray-traced datasets. In contrast, the isotropic image- & depth-driven regularizer and the anisotropic complementary regularizer, which both embed image information, perform better on the more realistic Middlebury datasets. By the use of advanced data terms, such as the brightness and gradient constancy assumption with the RGB valued version of the datasets, this difference in performance remains and even slightly increases. Especially, the additional gradient constancy assumption of the proposed data terms lead to an improvement of the quality metrics, as well as a more robust version. While comparing the performance one must keep in mind that the results

depend on the applied metrics, because different metrics may lead to different results, as seen in case of the used 2-D and 3-D quality metric.

6.1 FUTURE WORK

REFLECTION MODEL. In real world sequences not all objects exhibit the Lambertian reflectance, hence these objects violate the brightness constancy assumption under certain circumstances. One possibility to handle this problem is to consider more complex reflection models.

OCCLUSION HANDLING. Most errors appear in occluded regions, therefore an incorporation of an occlusion handling would be desirable. This especially applies in the case of multiple views, because the number of occlusions increases.

LARGE DISPLACEMENTS. In stereo settings small objects or details in the foreground underlie a large displacement. The coarse-to-fine strategy considered in this thesis is not enough to address this problem, since at coarse levels, where the displacement is sufficiently small to be estimated, these objects or details disappear. To handle such large displacements feature matches can be integrated in the computation, similar as in the case of optical flow [3, 20].

NORMALIZED REGULARIZER. The presented regularizers use the gradient of Z computed on the 2-D pixel grid and therefore do not consider the increasing distance between the 3-D points by increasing depth. In order to deal with this problem a normalization may be desirable that takes the depth values into account.

A

DERIVATIONS

A.1 SMOOTHNESS TERM RELATED DERIVATIVES

The smoothness term related derivatives are approximated using a nested central second order finite difference scheme. The derivation reads

$$
\begin{aligned}
& \operatorname{div}\left(\left(\Psi'\right)_{i,j} \cdot \nabla\left(Z_{i,j} + dZ_{i,j}\right)\right) \\
\approx\ & \left(\left(\Psi'\right)_{i,j} \cdot \left(Z_{i,j} + dZ_{i,j}\right)_x\right)_x \\
& + \left(\left(\Psi'\right)_{i,j} \cdot \left(Z_{i,j} + dZ_{i,j}\right)_y\right)_y \\
=\ & \frac{\left(\Psi'\right)_{i+\frac{1}{2},j} \cdot \left(Z_{i+\frac{1}{2},j} + dZ_{i+\frac{1}{2},j}\right)_x - \left(\Psi'\right)_{i-\frac{1}{2},j} \cdot \left(Z_{i-\frac{1}{2},j} + dZ_{i-\frac{1}{2},j}\right)_x}{2 \cdot \left(\frac{1}{2} h_x\right)} \\
& + \frac{\left(\Psi'\right)_{i,j+\frac{1}{2}} \cdot \left(Z_{i,j+\frac{1}{2}} + dZ_{i,j+\frac{1}{2}}\right)_y - \left(\Psi'\right)_{i,j-\frac{1}{2}} \cdot \left(Z_{i,j-\frac{1}{2}} + dZ_{i,j-\frac{1}{2}}\right)_y}{2 \cdot \left(\frac{1}{2} h_y\right)} \\
=\ & \frac{\frac{\left(\Psi'\right)_{i+1,j} + \left(\Psi'\right)_{i,j}}{2} \cdot \frac{\left(Z_{i+1,j} + dZ_{i+1,j} - Z_{i,j} - dZ_{i,j}\right)}{2 \cdot \left(\frac{1}{2} h_x\right)}}{2 \cdot \left(\frac{1}{2} h_x\right)} \\
& - \frac{\frac{\left(\Psi'\right)_{i,j} + \left(\Psi'\right)_{i-1,j}}{2} \cdot \frac{\left(Z_{i,j} + dZ_{i,j} - Z_{i-1,j} - dZ_{i-1,j}\right)}{2 \cdot \left(\frac{1}{2} h_x\right)}}{2 \cdot \left(\frac{1}{2} h_x\right)} \\
& + \frac{\frac{\left(\Psi'\right)_{i,j+1} + \left(\Psi'\right)_{i,j}}{2} \cdot \frac{\left(Z_{i,j+1} + dZ_{i,j+1} - Z_{i,j} + dZ_{i,j}\right)}{2 \cdot \left(\frac{1}{2} h_y\right)}}{2 \cdot \left(\frac{1}{2} h_y\right)} \\
& - \frac{\frac{\left(\Psi'\right)_{i,j} + \left(\Psi'\right)_{i,j-1}}{2} \cdot \frac{\left(Z_{i,j} + dZ_{i,j} - Z_{i,j-1} + dZ_{i,j-1}\right)}{2 \cdot \left(\frac{1}{2} h_y\right)}}{2 \cdot \left(\frac{1}{2} h_y\right)} \\
=\ & \frac{\left(\Psi'\right)_{i+1,j} + \left(\Psi'\right)_{i,j}}{2h_x^2} \cdot \left(Z_{i+1,j} + dZ_{i+1,j} - Z_{i,j} - dZ_{i,j}\right) \\
& - \frac{\left(\Psi'\right)_{i,j} + \left(\Psi'\right)_{i-1,j}}{2h_x^2} \cdot \left(Z_{i,j} + dZ_{i,j} - Z_{i-1,j} - dZ_{i-1,j}\right) \\
& + \frac{\left(\Psi'\right)_{i,j+1} + \left(\Psi'\right)_{i,j}}{2h_y^2} \cdot \left(Z_{i,j+1} + dZ_{i,j+1} - Z_{i,j} + dZ_{i,j}\right) \\
& - \frac{\left(\Psi'\right)_{i,j} + \left(\Psi'\right)_{i,j-1}}{2h_y^2} \cdot \left(Z_{i,j} + dZ_{i,j} - Z_{i,j-1} + dZ_{i,j-1}\right)
\end{aligned}
$$

A.2 THE JACOBIAN

In Section 3.5 the Equation 54 relates the gradient of the back projected image with the original image gradient. It appears that this relation leads to the Jacobian matrix. The derivation of this relation can be derived as follows

$$
\begin{aligned}
(\nabla \mathcal{I}_i(x,y))^\top &= \begin{pmatrix} \partial_x \mathcal{I}_i(x,y) \\ \partial_y \mathcal{I}_i(x,y) \end{pmatrix}^\top && (98)\\
&= \begin{pmatrix} \partial_x (I_i(x_i,y_i)) \\ \partial_y (I_i(x_i,y_i)) \end{pmatrix}^\top && (99)\\
&= \begin{pmatrix} \partial_{x_i} I_i(x_i,y_i)\cdot \partial_x x_i + \partial_{y_i} I_1 \cdot \partial_x y_i \\ \partial_{x_i} I_1(x_i,y_i)\cdot \partial_y x_i + \partial_{y_i} I_1 \cdot \partial_y y_i \end{pmatrix}^\top && (100)\\
&= \begin{pmatrix} (\nabla_i I_i(x_i,y_i))^\top \cdot \begin{pmatrix} \partial_x x_i \\ \partial_x y_i \end{pmatrix} \\ (\nabla_i I_i(x_i,y_i))^\top \cdot \begin{pmatrix} \partial_y x_i \\ \partial_y y_i \end{pmatrix} \end{pmatrix}^\top && (101)\\
&= (\nabla_i I_i(x_i,y_i))^\top \cdot \underbrace{\begin{pmatrix} \frac{\partial x_i}{\partial x} & \frac{\partial x_i}{\partial y} \\ \frac{\partial y_i}{\partial x} & \frac{\partial y_i}{\partial y} \end{pmatrix}}_{J} && (102)
\end{aligned}
$$

The entries of the Jacobian matrix are given by

$$
\begin{aligned}
\frac{\partial x_i}{\partial x} &= \frac{-(a\cdot Z+b)\left(\frac{M_{31}}{s_x}Z + c\cdot \partial_x Z\right)}{(c\cdot Z+d)^2} + \frac{\left(\frac{M_{11}}{s_x}Z + a\cdot \partial_x Z\right)}{c\cdot Z+d} \\
\frac{\partial x_i}{\partial y} &= \frac{-(a\cdot Z+b)\left(\frac{M_{32}}{s_y}Z + c\cdot \partial_y Z\right)}{(c\cdot Z+d)^2} + \frac{\left(\frac{M_{12}}{s_y}Z + a\cdot \partial_y Z\right)}{c\cdot Z+d} \\
\frac{\partial y_i}{\partial x} &= \frac{-(\check{a}\cdot Z+\check{b})\left(\frac{M_{31}}{s_x}Z + c\cdot \partial_x Z\right)}{(c\cdot Z+d)^2} + \frac{\left(\frac{M_{21}}{s_x}Z + \check{a}\cdot \partial_x Z\right)}{c\cdot Z+d} \\
\frac{\partial y_i}{\partial y} &= \frac{-(\check{a}\cdot Z+\check{b})\left(\frac{M_{32}}{s_y}Z + c\cdot \partial_y Z\right)}{(c\cdot Z+d)^2} + \frac{\left(\frac{M_{22}}{s_y}Z + \check{a}\cdot \partial_y Z\right)}{c\cdot Z+d}
\end{aligned} \quad (103)
$$

The needed inverse Jacobian J^{-1} can be computed as follows

$$
\begin{aligned}
J^{-1} &= \frac{1}{\det(J)} \begin{pmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{pmatrix} \\
&= \frac{1}{J_{11}J_{22} - J_{12}J_{21}} \begin{pmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{pmatrix} && (104)
\end{aligned}
$$

As one can see in (103) the partial derivatives $\partial_x Z$ and $\partial_y Z$ are used. Which are approximated using a central fourth order finite difference scheme.

$$\begin{aligned}\partial_x Z_{i,j} &= \frac{-Z_{i+2,j} + 8Z_{i+1,j} - 8Z_{i-1,j} + Z_{i-2,j}}{12h_x^k} \\ \partial_y Z_{i,j} &= \frac{-Z_{i,j+2} + 8Z_{i,j+1} - 8Z_{i,j-1} + Z_{i,j-2}}{12h_y^k}\end{aligned} \tag{105}$$

A.3 SECOND ORDER ISOTROPIC SMOOTHNESS TERM

In this derivation the short notation $\Psi' := \Psi'\left(\|\mathcal{H}_Z\|_F^2\right)$ is used.

$$\begin{aligned}
&\nabla^{*\top}\left(\Psi' \cdot \nabla^* Z\right) \\
&\approx \left(\Psi' \cdot Z_{xx}\right)_{xx} + 2\left(\Psi' \cdot Z_{xy}\right)_{xy} + \left(\Psi' \cdot Z_{yy}\right)_{yy} \\
&= \frac{1}{h_x^2}\left(\left(\Psi' \cdot Z_{xx}\right)_{i+1,j} - 2\left(\Psi' \cdot Z_{xx}\right)_{i,j} + \left(\Psi' \cdot Z_{xx}\right)_{i-1,j}\right) \\
&\quad + \frac{1}{h_y^2}\left(\left(\Psi' \cdot Z_{yy}\right)_{i,j+1} - 2\left(\Psi' \cdot Z_{yy}\right)_{i,j} + \left(\Psi' \cdot Z_{yy}\right)_{i,j-1}\right) \\
&\quad + \frac{2}{4 \cdot h_x h_y}\Big(\left(\Psi' \cdot Z_{xy}\right)_{i+1,j+1} - \left(\Psi' \cdot Z_{xy}\right)_{i+1,j-1} \\
&\qquad\qquad - \left(\Psi' \cdot Z_{xy}\right)_{i-1,j+1} + \left(\Psi' \cdot Z_{xy}\right)_{i-1,j-1}\Big) \\
&= \frac{1}{h_x^2} \\
&\quad \Big(h_x^{-2} \cdot \left({\Psi'}_{i+1,j} \cdot Z_{i+2,j} - 2{\Psi'}_{i+1,j} \cdot Z_{i+1,j} + {\Psi'}_{i+1,j} \cdot Z_{i,j}\right) \\
&\qquad - 2h_x^{-2} \cdot \left({\Psi'}_{i,j} \cdot Z_{i+1,j} - 2{\Psi'}_{i,j} \cdot Z_{i,j} + {\Psi'}_{i,j} \cdot Z_{i-1,j}\right) \\
&\qquad + h_x^{-2} \cdot \left({\Psi'}_{i-1,j} \cdot Z_{i,j} - 2{\Psi'}_{i-1,j} \cdot Z_{i-1,j} + {\Psi'}_{i-1,j} \cdot Z_{i-2,j}\right)\Big) \\
&\quad + \frac{2}{4 \cdot h_x h_y} \\
&\quad \Big(\left(4 \cdot h_x h_y\right)^{-1}\left({\Psi'}_{i+1,j+1} \cdot Z_{i+2,j+2} - {\Psi'}_{i+1,j+1} \cdot Z_{i+2,j}\right. \\
&\qquad\qquad \left. - {\Psi'}_{i+1,j+1} \cdot Z_{i,j+2} + {\Psi'}_{i+1,j+1} \cdot Z_{i,j}\right) \\
&\quad - \left(4 \cdot h_x h_y\right)^{-1}\left({\Psi'}_{i+1,j-1} \cdot Z_{i+2,j} - {\Psi'}_{i+1,j-1} \cdot Z_{i+2,j-2}\right. \\
&\qquad\qquad \left. - {\Psi'}_{i+1,j-1} \cdot Z_{i,j} + {\Psi'}_{i+1,j-1} \cdot Z_{i,j-2}\right) \\
&\quad - \left(4 \cdot h_x h_y\right)^{-1}\left({\Psi'}_{i-1,j+1} \cdot Z_{i,j+2} - {\Psi'}_{i-1,j+1} \cdot Z_{i,j}\right. \\
&\qquad\qquad \left. - {\Psi'}_{i-1,j+1} \cdot Z_{i-2,j+2} + {\Psi'}_{i-1,j+1} \cdot Z_{i-2,j}\right)
\end{aligned}$$

$$
\begin{aligned}
&+\left(4\cdot h_x h_y\right)^{-1}\Big(\Psi'_{i-1,j-1}\cdot Z_{i,j}-\Psi'_{i-1,j-1}\cdot Z_{i,j-2}\\
&\qquad\qquad -\Psi'_{i-1,j-1}\cdot Z_{i-2,j}+\Psi'_{i-1,j-1}\cdot Z_{i-2,j-2}\Big)\Big)\\
&+\frac{1}{h_y^2}\\
&\Big(\quad h_y^{-2}\cdot\left(\Psi'_{i,j+1}\cdot Z_{i,j+2}-2\Psi'_{i,j+1}\cdot Z_{i,j+1}+\Psi'_{i,j+1}\cdot Z_{i,j}\right)\\
&\quad -2h_y^{-2}\cdot\left(\Psi'_{i,j}\cdot Z_{i,j+1}-2\Psi'_{i,j}\cdot Z_{i,j}+\Psi'_{i,j}\cdot Z_{i,j-1}\right)\\
&\quad +\ \ h_y^{-2}\cdot\left(\Psi'_{i,j-1}\cdot Z_{i,j}-2\Psi'_{i,j-1}\cdot Z_{i,j-1}+\Psi'_{i,j-1}\cdot Z_{i,j-2}\right)\Big)
\end{aligned}
$$

A.4 NATURAL BOUNDARY CONDITIONS

In this section the natural boundary conditions of a functional of the following kind

$$E(u)=\int_\Omega F\left(x,y,u,u_x,u_y,u_{xx},u_{xy},u_{yx},u_{yy}\right)\,dxdy \tag{106}$$

are derived. Here Ω is a rectangular subset of $\mathbb{R}^2$ with piecewise smooth boundary $\partial\Omega$. Further it is assumed that $u(x,y)$ is a sufficiently often differentiable minimizer of E, which is embedded into $u(x,y,e):=u(x,y)+\epsilon\eta(x,y)$ with an arbitrary perturbation function $\eta(x,y)$ in the region Ω. Because $u(x,y)$ minimizes $E(u)$ the scalar function $\Phi(\epsilon):=E(u+\epsilon\eta)$ has a minimum at $\epsilon=0$. This leads to

$$0=\Phi'(0)=\frac{d}{d\epsilon}E(v+\epsilon\eta)\bigg|_{\epsilon=0}=\delta E \tag{107}$$

where δE is known as the first variation of E at u in the direction of η Applying the chain rule it follows that

$$
\begin{aligned}
\delta E=\int_\Omega &(F_u\eta+F_{u_x}\eta_x+F_{u_y}\eta_y\\
&+F_{u_{xx}}\eta_{xx}+F_{u_{xy}}\eta_{xy}+F_{u_{yx}}\eta_{yx}+F_{u_{yy}}\eta_{yy})dxdy
\end{aligned}
\tag{108}
$$

Integration by parts of all the terms containing derivatives of η yields

$$\begin{aligned}
\delta E = & \int_\Omega (F_u \eta) dx dy \\
& + \int_{\partial\Omega} (F_{u_x} \eta n_x)\, dx dy \quad - \int_\Omega \left(\frac{\partial}{\partial x} F_{u_x} \eta \right) dx dy \\
& + \int_{\partial\Omega} \left(F_{u_y} \eta n_y \right) dx dy \quad - \int_\Omega \left(\frac{\partial}{\partial y} F_{u_y} \eta \right) dx dy \\
& + \int_{\partial\Omega} (F_{u_{xx}} \eta_x n_x)\, dx dy - \int_\Omega \left(\frac{\partial}{\partial x} F_{u_{xx}} \eta_x \right) dx dy \\
& + \int_{\partial\Omega} \left(F_{u_{xy}} \eta_x n_y \right) dx dy - \int_\Omega \left(\frac{\partial}{\partial y} F_{u_{xy}} \eta_x \right) dx dy \\
& + \int_{\partial\Omega} \left(F_{u_{yx}} \eta_y n_x \right) dx dy - \int_\Omega \left(\frac{\partial}{\partial x} F_{u_{yx}} \eta_y \right) dx dy \\
& + \int_{\partial\Omega} \left(F_{u_{yy}} \eta_y n_y \right) dx dy - \int_\Omega \left(\frac{\partial}{\partial y} F_{u_{yy}} \eta_y \right) dx dy
\end{aligned}$$

where for $(x, y) \in \partial\Omega$ the outer normal is denoted by $n(x, y) = (n_x, n_y)^\top$. A further integration by parts of all the integrals over Ω containing derivatives of η yields

$$\begin{aligned}
\delta E = & \int_\Omega (F_u \eta) dx dy \\
& + \int_{\partial\Omega} (F_{u_x} \eta n_x)\, dx dy \quad - \int_\Omega \left(\frac{\partial}{\partial x} F_{u_x} \eta \right) dx dy \\
& + \int_{\partial\Omega} \left(F_{u_y} \eta n_y \right) dx dy \quad - \int_\Omega \left(\frac{\partial}{\partial y} F_{u_y} \eta \right) dx dy \\
& + \int_{\partial\Omega} (F_{u_{xx}} \eta_x n_x)\, dx dy - \int_{\partial\Omega} \left(\frac{\partial}{\partial x} F_{u_{xx}} \eta n_x \right) dx dy \\
& \qquad\qquad + \int_\Omega \left(\frac{\partial^2}{\partial x^2} F_{u_{xx}} \eta \right) dx dy \\
& + \int_{\partial\Omega} \left(F_{u_{xy}} \eta_x n_y \right) dx dy - \int_{\partial\Omega} \left(\frac{\partial}{\partial y} F_{u_{xy}} \eta n_x \right) dx dy \\
& \qquad\qquad + \int_\Omega \left(\frac{\partial^2}{\partial x \partial y} F_{u_{xy}} \eta \right) dx dy \\
& + \int_{\partial\Omega} \left(F_{u_{yx}} \eta_y n_x \right) dx dy - \int_{\partial\Omega} \left(\frac{\partial}{\partial x} F_{u_{yx}} \eta n_y \right) dx dy \\
& \qquad\qquad + \int_\Omega \left(\frac{\partial^2}{\partial y \partial x} F_{u_{yx}} \eta \right) dx dy \\
& + \int_{\partial\Omega} \left(F_{u_{yy}} \eta_y n_y \right) dx dy - \int_{\partial\Omega} \left(\frac{\partial}{\partial y} F_{u_{yy}} \eta n_y \right) dx dy \\
& \qquad\qquad + \int_\Omega \left(\frac{\partial^2}{\partial y^2} F_{u_{yy}} \eta \right) dx dy
\end{aligned}$$

After rearranging, Equation 109 results in

$$\begin{aligned}\delta E = \quad & \int_\Omega \Big(F_u - \frac{\partial}{\partial_x} F_{u_x} - \frac{\partial}{\partial_y} F_{u_y} \\ & \qquad + \frac{\partial^2}{\partial_x^2} F_{u_{xx}} + \frac{\partial^2}{\partial_x \partial_y} F_{u_{xy}} + \frac{\partial^2}{\partial_y \partial_x} F_{u_{yx}} + \frac{\partial^2}{\partial_y^2} F_{u_{yy}} \Big) \cdot \eta \; dxdy \\ & + \int_{\partial\Omega} \left(F_{u_x} - \frac{\partial}{\partial_x} F_{u_{xx}} - \frac{\partial}{\partial_y} F_{u_{xy}} \right) \cdot \eta n_x \; dxdy \\ & + \int_{\partial\Omega} \left(F_{u_y} - \frac{\partial}{\partial_x} F_{u_{yx}} - \frac{\partial}{\partial_y} F_{u_{yy}} \right) \cdot \eta n_y \; dxdy \\ & + \int_{\partial\Omega} \left(F_{u_{xx}} n_x + F_{u_{xy}} n_y \right) \cdot \eta_x \; dxdy \\ & + \int_{\partial\Omega} \left(F_{u_{yx}} n_x + F_{u_{yy}} n_y \right) \cdot \eta_y \; dxdy\end{aligned}$$

Since u is a minimizer it must fulfill Equation 18. Further one can choose a function η with the properties $\eta|_{\partial\Omega} \neq 0$ and $\nabla\eta|_{\partial\Omega} = 0$, what results in the following case

$$\delta E = \int_{\partial\Omega} \left(F_{u_x} - \frac{\partial}{\partial_x} F_{u_{xx}} - \frac{\partial}{\partial_y} F_{u_{xy}} \right) \cdot \eta n_x \; dxdy \tag{109}$$

$$+ \int_{\partial\Omega} \left(F_{u_y} - \frac{\partial}{\partial_x} F_{u_{yx}} - \frac{\partial}{\partial_y} F_{u_{yy}} \right) \cdot \eta n_y \; dxdy \tag{110}$$

Making use the fundamental lemma of the calculus of variations, this results in

$$\begin{aligned}0 = & \left(F_{u_x} - \frac{\partial}{\partial_x} F_{u_{xx}} - \frac{\partial}{\partial_y} F_{u_{xy}} \right) n_x \\ & + \left(F_{u_y} - \frac{\partial}{\partial_x} F_{u_{yx}} - \frac{\partial}{\partial_y} F_{u_{yy}} \right) n_y \quad \forall (x,y) \in \partial\Omega\end{aligned} \tag{111}$$

Similar one can choose $\eta|_{\partial\Omega} = \eta_y|_{\partial\Omega} = 0$ and $\eta_y|_{\partial\Omega} \neq 0$, what results in

$$0 = F_{u_{xx}} n_x + F_{u_{xy}} n_y \quad \forall (x,y) \in \partial\Omega \tag{112}$$

and analog

$$0 = F_{u_{yx}} n_x + F_{u_{yy}} n_y \quad \forall (x,y) \in \partial\Omega \tag{113}$$

So it follows that the minimizer u fulfills the boundary conditions

$$0 = n^\top \begin{pmatrix} F_{u_x} - \frac{\partial}{\partial x} F_{u_{xx}} - \frac{\partial}{\partial y} F_{u_{xy}} \\ F_{u_y} - \frac{\partial}{\partial x} F_{u_{yx}} - \frac{\partial}{\partial y} F_{u_{yy}} \end{pmatrix} \quad \forall (x,y) \in \partial\Omega \tag{114}$$

$$0 = n^\top \begin{pmatrix} F_{u_{xx}} \\ F_{u_{xy}} \end{pmatrix}, \quad 0 = n^\top \begin{pmatrix} F_{u_{yx}} \\ F_{u_{yy}} \end{pmatrix} \quad \forall (x,y) \in \partial\Omega \tag{115}$$

B

STENCILS

B.1 ISOTROPIC SMOOTHNESS TERM

	$i-1$	i	$i+1$
$j-1$		$\frac{\alpha}{2 \cdot h_y^2} \left(\Psi'^{k,l}_{i,j-1} + \Psi'^{k,l}_{i,j} \right)$	
j	$\frac{\alpha}{2 \cdot h_x^2} \left(\Psi'^{k,l}_{i-1,j} + \Psi'^{k,l}_{i,j} \right)$	$-\frac{\alpha}{2 \cdot h_x^2} \left(\Psi'^{k,l}_{i-1,j} + 2 \cdot \Psi'^{k,l}_{i,j} + \Psi'^{k,l}_{i+1,j} \right)$ $-\frac{\alpha}{2 \cdot h_y^2} \left(\Psi'^{k,l}_{i,j-1} + 2 \cdot \Psi'^{k,l}_{i,j} + \Psi'^{k,l}_{i,j+1} \right)$	$\frac{\alpha}{2 \cdot h_x^2} \left(\Psi'^{k,l}_{i+1,j} + \Psi'^{k,l}_{i,j} \right)$
$j+1$		$\frac{\alpha}{2 \cdot h_y^2} \left(\Psi'^{k,l}_{i,j+1} + \Psi'^{k,l}_{i,j} \right)$	

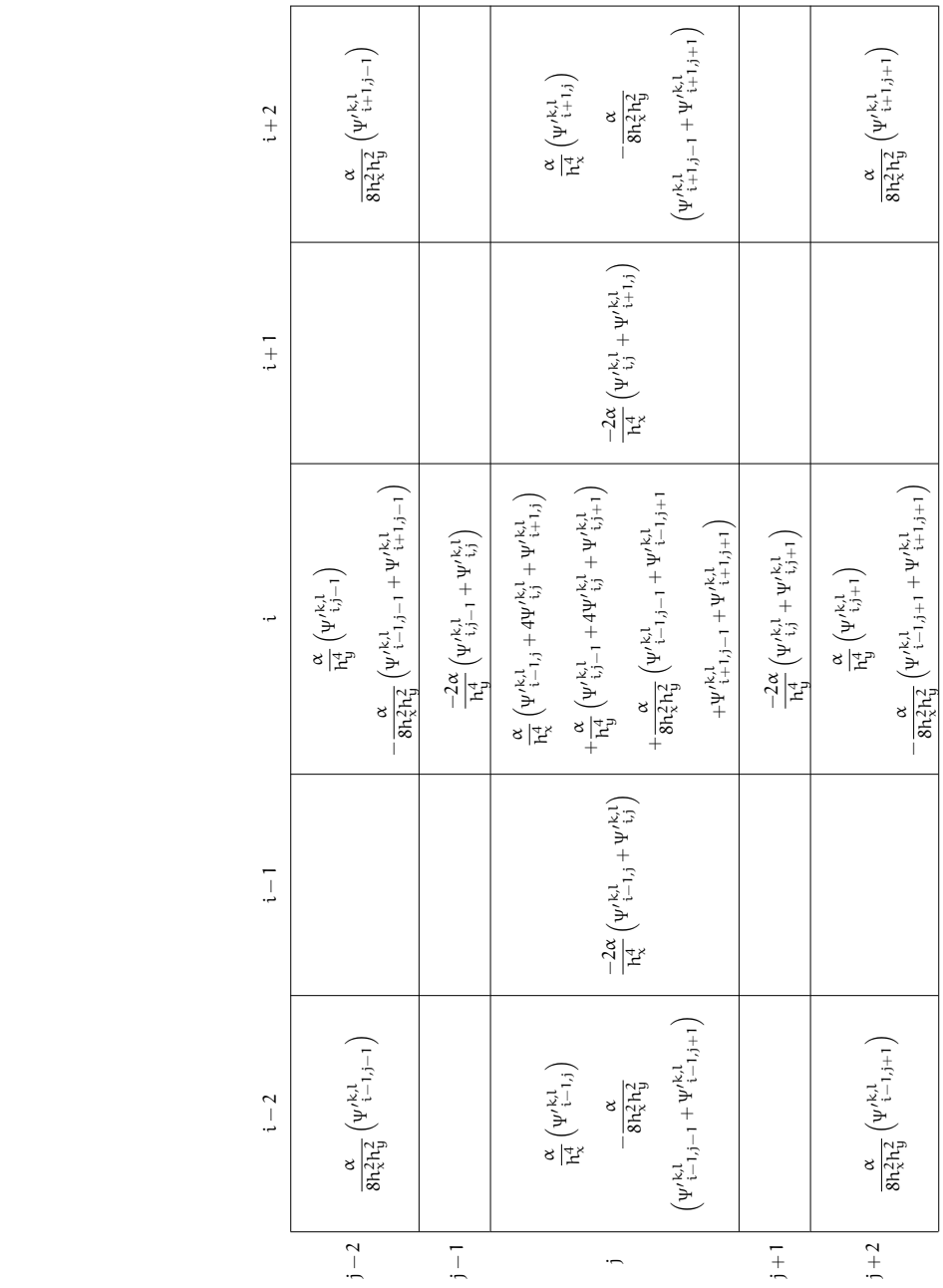
Table 5: Stencil for the isotropic smoothness term.

B.2 ANISOTROPIC SMOOTHNESS TERM

	$i-1$	i	$i+1$
$j-1$	$\frac{\alpha}{4 \cdot h_x h_y} \left(\lvert b_{i-1,j} \rvert + b_{i-1,j} \right)$ $+\frac{\alpha}{4 \cdot h_x h_y} \left(\lvert b_{i,j-1} \rvert + b_{i,j-1} \right)$	$\frac{\alpha}{2 \cdot h_y^2} \left(c_{i,j-1} + c_{i,j} \right)$ $-\frac{\alpha}{2 \cdot h_x h_y} \left(\lvert b_{i,j-1} \rvert + \lvert b_{i,j} \rvert \right)$	$\frac{\alpha}{4 \cdot h_x h_y} \left(\lvert b_{i+1,j} \rvert - b_{i+1,j} \right)$ $+\frac{\alpha}{4 \cdot h_x h_y} \left(\lvert b_{i,j-1} \rvert - b_{i,j-1} \right)$
j	$\frac{\alpha}{2 \cdot h_x^2} \left(a_{i-1,j} + a_{i,j} \right)$ $-\frac{\alpha}{2 \cdot h_x h_y} \left(\lvert b_{i-1,j} \rvert + \lvert b_{i,j} \rvert \right)$	$-\frac{\alpha}{2 \cdot h_x^2} \left(a_{i-1,j} + 2a_{i,j} + a_{i+1,j} \right)$ $+\frac{2 \cdot \alpha}{h_x h_y} \left(\lvert b_{i,j} \rvert \right)$ $-\frac{\alpha}{2 \cdot h_y^2} \left(c_{i,j-1} + 2c_{i,j} + c_{i,j+1} \right)$	$\frac{\alpha}{2 \cdot h_x^2} \left(a_{i+1,j} + a_{i,j} \right)$ $-\frac{\alpha}{2 \cdot h_x h_y} \left(\lvert b_{i+1,j} \rvert + \lvert b_{i,j} \rvert \right)$
$j+1$	$\frac{\alpha}{4 \cdot h_x h_y} \left(\lvert b_{i-1,j} \rvert - b_{i-1,j} \right)$ $+\frac{\alpha}{4 \cdot h_x h_y} \left(\lvert b_{i,j+1} \rvert - b_{i,j+1} \right)$	$\frac{\alpha}{2 \cdot h_y^2} \left(c_{i,j+1} + c_{i,j} \right)$ $-\frac{\alpha}{2 \cdot h_x h_y} \left(\lvert b_{i,j+1} \rvert + \lvert b_{i,j} \rvert \right)$	$\frac{\alpha}{4 \cdot h_x h_y} \left(\lvert b_{i+1,j} \rvert + b_{i+1,j} \right)$ $+\frac{\alpha}{4 \cdot h_x h_y} \left(\lvert b_{i,j+1} \rvert + b_{i,j+1} \right)$

Table 6: Stencil for the anisotropic smoothness terms.

B.3 SECOND ORDER ISOTROPIC SMOOTHNESS TERM



	i − 2	i − 1	i	i + 1	i + 2
j − 2	$\frac{\alpha}{8h_x^2h_y^2}\left(\psi'^{k,l}_{i-1,j-1}\right)$		$\frac{\alpha}{h_y^4}\left(\psi'^{k,l}_{i,j-1}\right)$ $-\frac{\alpha}{8h_x^2h_y^2}\left(\psi'^{k,l}_{i-1,j-1}+\psi'^{k,l}_{i+1,j-1}\right)$		$\frac{\alpha}{8h_x^2h_y^2}\left(\psi'^{k,l}_{i+1,j-1}\right)$
j − 1			$\frac{-2\alpha}{h_y^4}\left(\psi'^{k,l}_{i,j-1}+\psi'^{k,l}_{i,j}\right)$		
j	$\frac{\alpha}{h_x^4}\left(\psi'^{k,l}_{i-1,j}\right)$ $-\frac{\alpha}{8h_x^2h_y^2}$ $\left(\psi'^{k,l}_{i-1,j-1}+\psi'^{k,l}_{i-1,j+1}\right)$	$\frac{-2\alpha}{h_x^4}\left(\psi'^{k,l}_{i-1,j}+\psi'^{k,l}_{i,j}\right)$	$\frac{\alpha}{h_x^4}\left(\psi'^{k,l}_{i-1,j}+4\psi'^{k,l}_{i,j}+\psi'^{k,l}_{i+1,j}\right)$ $+\frac{\alpha}{h_y^4}\left(\psi'^{k,l}_{i,j-1}+4\psi'^{k,l}_{i,j}+\psi'^{k,l}_{i,j+1}\right)$ $+\frac{\alpha}{8h_x^2h_y^2}\left(\psi'^{k,l}_{i-1,j-1}+\psi'^{k,l}_{i-1,j+1}\right.$ $\left.+\psi'^{k,l}_{i+1,j-1}+\psi'^{k,l}_{i+1,j+1}\right)$	$\frac{-2\alpha}{h_x^4}\left(\psi'^{k,l}_{i,j}+\psi'^{k,l}_{i+1,j}\right)$	$\frac{\alpha}{h_x^4}\left(\psi'^{k,l}_{i+1,j}\right)$ $-\frac{\alpha}{8h_x^2h_y^2}$ $\left(\psi'^{k,l}_{i+1,j-1}+\psi'^{k,l}_{i+1,j+1}\right)$
j + 1			$\frac{-2\alpha}{h_y^4}\left(\psi'^{k,l}_{i,j}+\psi'^{k,l}_{i,j+1}\right)$		
j + 2	$\frac{\alpha}{8h_x^2h_y^2}\left(\psi'^{k,l}_{i-1,j+1}\right)$		$\frac{\alpha}{h_y^4}\left(\psi'^{k,l}_{i,j+1}\right)$ $-\frac{\alpha}{8h_x^2h_y^2}\left(\psi'^{k,l}_{i-1,j+1}+\psi'^{k,l}_{i+1,j+1}\right)$		$\frac{\alpha}{8h_x^2h_y^2}\left(\psi'^{k,l}_{i+1,j+1}\right)$

Table 7: Stencil for the second order isotropic smoothness term.

BIBLIOGRAPHY

[1] L. Alvarez, J. Esclarín, M. Lefébure, and J. Sánchez. A PDE model for computing the optical flow. In *Proc. XVI Congreso de Ecuaciones Diferenciales y Aplicaciones*, pages 1349–1356, Las Palmas de Gran Canaria, Spain, September 1999.

[2] T. Basha, Y. Moses, and N. Kiryati. Multi-view scene flow estimation: A view centered variational approach. *International Journal of Computer Vision*, 101(1):6–21, January 2013.

[3] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, March 2011.

[4] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *Proc. European Conference on Computer Vision*, pages 25–36, Prague, Czech Republic, May 2004.

[5] A. Bruhn and J. Weickert. Towards ultimate motion estimation: Combining highest accuracy with real-time performance. In *Proc. IEEE International Conference on Computer Vision*, pages 749–755, Beijing, China, October 2005.

[6] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, February 2005.

[7] R. Courant and D. Hilbert. *Methods of Mathematical Physics*. Interscience Publishers, Inc., 1953.

[8] W. Förstner and E. Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proc. ISPRS Intercommission Conference on Fast Processing of Photogrammetric Data*, pages 281–305, Interlaken, Switzerland, 1987.

[9] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12 (1):16–22, July 2000.

[10] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, Providence, RI, USA, June 2012.

[11] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[12] H. Kielhöfer. *Variationsrechnung: Eine Einführung in die Theorie einer unabhängigen Variablen mit Beispielen und Aufgaben*. Vieweg+Teubner Verlag, 2010.

[13] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, July 1990.

[14] C. Rabe, T. Müller, A. Wedel, and U. Franke. Dense, robust, and accurate motion field estimation from stereo image sequences in real-time. In *Proc. European Conference on Computer Vision*, pages 582–595, Heraklion, Crete, Greece, September 2010.

[15] L. Robert and R. Deriche. Dense depth map reconstruction: Aminimization and regularization approach which preserves discontinuities. In *Proc. European Conference on Computer Vision*, pages 439–451, Cambridge, UK, April 1996.

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

[17] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 195–202, Madison, WI, USA, June 2003.

[18] K. Schauwecker and A. Zell. On-board dual-stereo-vision for the navigation of an autonomous mav. *Journal of Intelligent & Robotic Systems*, 74(1-2):1–16, April 2014.

[19] J. Stillwell. *Mathematics and Its History*. Springer, 2010.

[20] M. Stoll, S. Volz, and A. Bruhn. Adaptive integration of feature matches into variational optical flow methods. In *Asian Conference on Computer Vision*, pages 1–14, Daejeon, Korea, November 2012.

[21] J. Stühmer, S. Gumhold, and D. Cremers. Real-time dense geometry from a handheld camera. In *Proc. DAGM Symposium on Pattern Recognition*, pages 11–20, Darmstadt, Germany, September 2010.

[22] P. Sturm. A historical survey of geometric computer vision. In *Proc. Computer Analysis of Images and Patterns*, volume 1, pages 1–8, Seville, Spain, August 2011.

[23] D. Sun, S. Roth, J. Lewis, and M. J. Black. Learning optical flow. In *Proc. European Conference on Computer Vision*, pages 83–97, Marseille, France, October 2008.

[24] M. Team. Offical megaPOV website., July 2014. URL `http://megapov.inetart.net/`.

[25] A. Vedaldi. Annotation patch patch for megaPOV 1.2.1, July 2014. URL `http://www.robots.ox.ac.uk/~vedaldi/code/vlpovy.html`.

[26] K. Venkataraman, D. Lelescu, J. Duparré, A. McMahon, G. Molina, P. Chatterjee, R. Mullis, and S. Nayar. PiCam: an ultra-thin high performance monolithic camera array. *ACM Transactions on Graphics*, 32(6):166, November 2013.

[27] J. Weickert. *Anisotropic Diffusion in Image Processing*. Teubner Stuttgart, 1998.

[28] D. Young. Iterative methods for solving partial difference equations of elliptic type. *Transactions of the American Mathematical Society*, pages 92–111, 1954.

[29] H. Zimmer, A. Bruhn, L. Valgaerts, M. Breuß, J. Weickert, B. Rosenhahn, and H.-P. Seidel. Pde-based anisotropic disparity-driven stereo vision. In *Proc. International Workshop on Vision, Modeling and Visualization*, pages 263–272, Konstanz, Germany, October 2008.

[30] H. Zimmer, A. Bruhn, J. Weickert, L. Valgaerts, A. Salgado, B. Rosenhahn, and H.-P. Seidel. Complementary optic flow. In *Proc. International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 207–220, Bonn, Germany, August 2009.

DECLARATION

I hereby declare that the work presented in this thesis is entirely my own. I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

Stuttgart, 21.10.2014

Daniel Maurer

ERKLÄRUNG

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Stuttgart, 21.10.2014

Daniel Maurer