Institute for Visualization and Interactive Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit Nr. 55

Multi-View Stereo with Inverse Depth Parameterization

Oliver Goroll

Course of Study: Informatik

Examiner: Prof. Dr. Andrés Bruhn

Supervisor: Daniel Maurer, M.Sc.

Commenced: 2015-10-19

Completed: 2016-04-19

CR-Classification: I.4.8

ABSTRACT

Estimating accurate depth maps by solving the stereo problem is an important step in reconstructing real world surfaces. Variational methods that minimize a global energy functional are considered as especially precise throughout the literature. By using the depth as parameterization directly, the approach is easily extended to multiple views, allowing to considerably enhance the quality of the resulting depth maps. Further improvement can be achieved by adapting the depth parameterization to the regularizer used. In this thesis second-order regularization is used with an INVERSE depth parameterization and compared against the direct depth parameterization to examine its benefits. Several extensions that improve upon the naïve multi-view approach are suggested. The proposed method and its extensions are evaluated by experiments, using artificial as well as real world test scenes.

ZUSAMMENFASSUNG

Das Ermitteln präziser Tiefenkarten durch Lösen des Stereoproblems stellt einen wichtigen Schritt bei der Rekonstruktion real existierender Oberflächen dar. Variationsmethoden, die ein globales Energiefunktional minimieren, haben sich in der Literatur als besonders exakt erwiesen. Durch Verwendung der Tiefe als Parametrisierung kann der Ansatz leicht auf mehrere Bilder erweitert werden, wodurch die Qualität der sich ergebenden Tiefenkarten erheblich verbessert wird. Eine weitere Qualitätssteigerung kann durch die Anpassung der Tiefenparametrisierung an den verwendeten Regularisierer erreicht werden. In dieser Arbeit wird eine Regularisierung zweiter Ordnung zusammen mit einer INVERSEN Tiefenparametrisierung verwendet, um deren Nutzen gegenüber der direkten Tiefenparametrisierung zu vergleichen. Ferner werden verschiedene Erweiterungen vorgestellt, die das einfache Mehr-Bild-Verfahren verbessern. Anhand künstlich erzeugter sowie echter Testbilder werden die vorgeschlagene Methode und die Erweiterungen evaluiert.

Imagine that! It never occurred to me to think
of SPACE as the thing that was moving!

— Montgomery 'Scotty' Scott

ACKNOWLEDGMENTS

I would like to thank Professor Andrés Bruhn for providing me with this topic; his lectures, that were an excellent preparation for this thesis; and his contagious enthusiasm.
Many thanks to my supervisor Daniel Maurer for his support, valuable suggestions and for his patience and thoroughness when answering my questions regarding the research.
For proofreading and general advice, I thank Bernhard Bermeitinger and Jessica Goroll.
For proofreading and suggestions, but also for three years spent discussing countless topics, sharing laughs and diverting my thoughts to more utile things, my thanks go to Matti Grüner.
Finally, I would like to thank my family who never stopped believing in me and always cared.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\mathbf{P} = [X\ Y\ Z]^T$	point in 3D space
$\tilde{\mathbf{P}} = [X\ Y\ Z\ 1]^T$	homogeneous coordinate of $\mathbf{P}$
$\mathbf{p} = [x\ y]^T$	2D projection of $\mathbf{P}$
$\tilde{\mathbf{p}} = [x\ y\ 1]^T$	homogeneous coordinate of $\mathbf{p}$
$g(x_1, x_2, \ldots, x_n)$	function of n variables
$\partial_{x_i} g$, g_{x_i}	partial derivative of g for x_i
$\mathcal{R}$	rotation matrix
t	translation vector
T	extrinsic matrix
K	intrinsic matrix
M	full projection matrix
f	focal length
s_x, s_y	focal length in pixels
o_x, o_y	principal point offset
$I_\nu(\mathbf{p})$	image ν as continuous function
$I_{\nu\, i,j}$	discrete representation of image ν
$\rho(\mathbf{p})$	depth image as continuous function
$\rho_{i,j}$	discrete representation of the depth image
$\pi(\tilde{\mathbf{p}})$	mapping of $\tilde{\mathbf{p}}$ to $\mathbf{p}$
$\Phi(\rho)$	parameterization function of the depth
$E(\rho)$	energy functional
$D(\Phi \circ \rho)$	data term
$S(\rho)$	smoothness term

1 INTRODUCTION

Reconstructing the world or parts of it has been a classical problem of Computer Vision. In the automotive industry, it enables an autonomous car to perceive its surroundings and make sense of its environment [Rab+10]. In cartography, aerial imagery is used to infer elevation data, at a scale ranging from mountains to individual cottages, without human interaction [HSH05]. In museum curation, artifacts are imaged to create digital archives, restore and preserve them without the risk of damage or to create renderings for virtual exhibitions [Hos+12]. These are only three examples, but the variety of possible applications is huge. They have in common that they take a set of input images and create a set of 3-D geometry as output, which represents the visible scene. This process is known as STEREO RECONSTRUCTION; at its core lies the solution to the STEREO PROBLEM.

To reconstruct 3-D geometry from images two steps are necessary. In the first step, correspondences between the image points of an image pair are identified; in the second step, the correspondences are triangulated using the relative positions of the cameras involved to obtain the 3-D coordinates of the scene geometry. With known camera parameters the triangulation is the trivial part of the reconstruction, leaving the detection of correspondences as the problem to solve. Of the methods that solve the stereo problem, global methods emerged as the most accurate. Instead of finding correspondences for each image point individually or in a small neighborhood, they find correspondences for all points at once by defining a global energy functional that measures similarity of related image points and tries to enforce a smooth solution. The energy formulation transforms the stereo problem to a minimization problem which can be solved with common numerical methods.

In most cases the so-called DISPARITY, a displacement vector in the image domain that describes the relative motion between corresponding points, is used as a parameterization. Although disparity computation is similar to optic flow computation, which allows to employ similar concepts and methods, using the disparity as parameterization has drawbacks [SBW05]. Most importantly, the disparity is defined between image pairs, hence extending the stereo method to arbitrary settings is not easy and has limitations [Hir05]. As a way to remedy these limitations, the correspondences are parameterized using the depth of the 3-D points, allowing computations to be performed in the 3-D space of the scene. Using such a parameterization, a correspondence between multiple images can be expressed by a

single parameter, that is the depth of the 3-D point. However, using the depth directly as a parameterization is not free of problems. Depending on the chosen regularizer the back-projection favors curved surfaces. This results in planar surfaces not being reconstructed accurately. Instead, the correspondences are parameterized by the INVERSE DEPTH, which promises better performance in the presence of planar or piecewise planar scene elements.

1.1 RELATED WORK

To solve the stereo problem, Robert and Deriche [RD96] presented a variational approach for two images using a depth parameterization. A constant intensity between corresponding points was assumed for the similarity measure. To preserve depth continuities a regularizer was introduced that smooths only along isophotes but not across depth boundaries. To increase the robustness of the smoothness term, different penalizer functions were considered and the two most promising ones examined further. The authors also highlight the ease of extending their approach to multiple views.

Stühmer et al. [SGC10] proposed a real-time capable approach for multiple views using images from a single hand-held camera. As before, Lambertian surfaces were assumed. Regularization of the solution was achieved by using the TV-norm to preserve discontinuities [ROF92]. Since a continuous video stream is used to provide different views the approach works best for rigid, stationary objects. To handle large displacements a coarse-to-fine warping strategy was employed.

Basha et al. [BMK12] introduced a method to obtain geometry and 3-D flow jointly. Correspondences were parameterized using the direct depth at time steps t and $t+1$. Additionally, robust penalizer functions were used to reduce the influence of outliers in both the data and the smoothness term, which is a first-order regularization of the depth and the 3-D flow. Resulting in a non-convex energy term, a coarse-to-fine approach was used to find the global minimizer.

In his master's thesis, Maurer [Mau14] developed a variational approach using a direct depth parameterization to solve the stereo problem for two views. Similar to Basha et al., robust penalizer functions were used and a coarse-to-fine warping strategy employed to solve the energy functional. Extending his method he introduced several smoothness terms, identifying three particularly accurate ones: isotropic depth-driven, isotropic image- and depth-driven and anisotropic depth-driven regularization. His implementation was used as basis for this thesis.

Schroers et al. [SHW15] introduced the inverse depth parameterization, using a robust, variational, multi-view approach. The authors provide justification for their choice of depth parameterization and

back-projection by examining the curvature of the reconstructed surfaces. They found that the direct depth parameterization favors curved surfaces when used in conjunction with a second-order regularizer. In contrast to the other methods mentioned, their choice of regularizer is more suitable in the presence of slanted surfaces.

Graber et al. [Gra+15] proposed a minimal area regularizer that aims to better respect the surface geometry than TV regularization. Because their regularizer is non-convex, they perform a reparameterization of the depth, resulting in a linear representation, which is easier to solve. They show that their method is able to better reconstruct geometry in artificial and real world test cases.

1.2 AIM AND ORGANIZATION OF THIS THESIS

The goal of this thesis is to evaluate the utility of the inverse depth parameterization in various cases to examine its benefits compared to the traditional direct depth parameterization. To achieve this, the relevant theoretical foundations are established (Chapter 2) before the approach by Maurer is extended to arbitrary parameterizations and the inverse depth is introduced (Sections 3.1.2 and 3.2). For increased accuracy, the approach is further extended to use multiple images (Section 3.3) and to handle the increased amount of occlusions (Section 4.1). The minimization, discretization and a method to solve the energy functional are presented (Sections 3.3.1 to 3.3.3). Possible further extension are explored (Chapter 4) before experimental results are presented and discussed (Chapter 5). A summary of the work done and suggestions for future work conclude this thesis (Chapter 6).

2 FOUNDATIONS

This chapter aims to setup a consistent notation that is used throughout the thesis and to introduce the concepts on which the subsequent chapters rely upon.

2.1 IMAGES AND PICTURES

The terms *image* and *picture* are commonly used interchangeably when talking about the output produced by a camera. In mathematics, *image* is a well defined term, referring to the set of values a function g maps to. To avoid confusion and to relate images to pictures and vice versa, both terms are formally defined.

Definition 2.1. *A gray value image* $I : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ *is a continuous, differentiable function over a rectangular domain* Ω *that maps a tuple to a scalar.*
A color image $I : \Omega \subset \mathbb{R}^2 \to \mathbb{R}^3$ *is a continuous, differentiable function over a rectangular domain* Ω *that maps a tuple to a triplet. The components of the triplet are called* channels.
The value $I(\mathbf{p})$ *with* $\mathbf{p} = [x\ y]^\mathsf{T}$ *is called* signal *or* response.

In addition, the function $I(\mathbf{p}, c)$ with $c \in \{1, 2, 3\}$ maps to one of the three possible channels. Differentiability by c, that is $\partial_c I$, is not required.

A suitable definition for pictures is the following:

Definition 2.2. *A gray value picture* $P : \Omega \subset \mathbb{N}^2 \to \mathbb{N}$ *maps tuples of discrete values of* Ω *to a scalar* g.
A color picture $P : \Omega \subset \mathbb{N}^2 \to \mathbb{N}^3$ *maps tuples of discrete values of* Ω *to a triplet* (r, g, b).
The elements $P(\mathbf{p})$ *with* $\mathbf{p} = [x\ y]^\mathsf{T}$ *are called picture elements or* pixels.

The triplet (r, g, b) represents the amounts of red, green and blue that are necessary to form a color perceivable by the human visual system. The magnitudes of the picture values, either gray or color, depend on the range of possible values. Common ranges are $[0, 255]$ for eight bit pictures and $[0, 65535]$ for 16 bit pictures. Black and white are defined as the lowest and highest values respectively. As with images, the function $P(\mathbf{p}, c)$ with $c \in \{1, 2, 3\}$ selects the channel of a pixel.

Images can be transformed to pictures by sampling. It is the duty of a digital camera to turn the continuous signal into a quantized, discrete signal. However, the exact process is of no concern for this

thesis. The other way, though, turning a picture into an image, is of importance.

There are three things to consider:

1. Images are differentiable, pictures are not
2. Images are continuous, pictures are not
3. The co-domain of images is continuous, the co-domain of pictures is discrete

The last point is easily remedied by the fact that $\mathbb{N}$ is a subset of $\mathbb{R}$ and the picture values are directly used as image values. This applies to the individual channels of color images, too. The second point is solved by interpolation, e. g. bilinear interpolation, where the image value is computed from the four nearest picture values at discrete locations. The problem in the first point is solved by pre-smoothing the image with a Gaussian kernel with small σ to get rid of the hard discontinuities between neighboring pixel values [BWS05].

A NOTE ON COLOR SPACES The values of images and pictures are used in computations to determine equality and similarity. It is therefore advisable that the gray and color values are linear: A pixel of twice the magnitude of another pixel must represent a color twice as bright. Pictures with non-linear colors should be converted to a linear color space before using them with the method proposed in Chapter 3.

2.2 HOMOGENEOUS COORDINATES AND PROJECTIVE GEOMETRY

Throughout this thesis the coordinates of points are often expressed in a homogeneous representation. At other occasions, coordinates are transformed from their non-homogeneous representation to their homogeneous one or vice versa. That is because homogeneous coordinates provide two properties that are useful in context of this thesis: First, it is possible to express affine transformations of points and vectors using matrix multiplications only. Second, the transformation from a homogeneous coordinate to its euclidean counterpart describes a projective mapping that will become useful in the next section.

To transform between euclidean and homogeneous coordinates a mapping is introduced:

$$\begin{aligned} \Pi : \quad & \mathbb{R}^n \to \mathbb{P}^n \\ \mathbf{p} = \begin{bmatrix} p_1 & \dots & p_n \end{bmatrix}^\mathsf{T} & \mapsto \begin{bmatrix} p_1 & \dots & p_n & 1 \end{bmatrix}^\mathsf{T} = \tilde{\mathbf{p}} \end{aligned} \tag{2.1}$$

The mapping Π transforms a vector with n components in euclidean space to its homogeneous representation with $n+1$ components in projective space $\mathbb{P}^n$. The reverse transformation is defined as:

$$\begin{aligned} \pi : \quad & \mathbb{P}^n \to \mathbb{R}^n \\ \tilde{\mathbf{p}} = \begin{bmatrix} p_1 & \dots & p_n & p_{n+1} \end{bmatrix}^\top & \mapsto \frac{1}{p_{n+1}} \begin{bmatrix} p_1 & \dots & p_n \end{bmatrix}^\top = \mathbf{p} \end{aligned} \tag{2.2}$$

The notation $\tilde{\mathbf{p}}$ is used as a shorthand for $\Pi(\mathbf{p})$. A comprehensive introduction to homogeneous coordinates and projective geometry can be found in *Multiple View Geometry in Computer Vision* by Hartley and Zisserman [HZ03].

2.2.1 *Application to Affine Transformations*

For this thesis, transformations of three-dimensional points $\mathbf{P} \in \mathbb{R}^3$ are relevant. Especially rotation and translation are of particular importance because in general the points considered for a transformation are part of a rigid body. This excludes any transformation that alters the absolute distance between two points, like shearing or scaling. Thus, the following explanations are restricted to isometries but are nevertheless valid for any affine transformation [HZ03].

An isometric transformation can be expressed as

$$\mathbf{P}' = \mathcal{R} \cdot \mathbf{P} + t,$$

where $\mathcal{R}$ is a 3×3 rotation matrix and t is a 3×1 translation vector. Consecutive applications of such transformations become complicated quickly. Consider a second pair $\mathcal{R}'$, t' that transforms $\mathbf{P}'$:

$$\mathbf{P}'' = \mathcal{R}' \cdot (\mathcal{R} \cdot \mathbf{P} + t) + t'$$

The problem here is, it is not possible to express the translation with a transformation matrix in euclidean coordinates. It can be shown, however, using homogeneous coordinates, it is possible to express any affine transformation as a matrix multiplication:

$$\tilde{\mathbf{P}}' = \begin{bmatrix} I & t \\ 0^\top & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathcal{R} & 0 \\ 0^\top & 1 \end{bmatrix} \cdot \tilde{\mathbf{P}} = \begin{bmatrix} \mathcal{R} & t \\ 0^\top & 1 \end{bmatrix} \cdot \tilde{\mathbf{P}},$$

where I is the identity matrix.

Another valuable observation is that affine transformations keep the fourth component of $\tilde{\mathbf{P}}$ intact. This allows to simply drop that component in order to transform the point to euclidean coordinates.

2.2.2 *Projective Properties*

As in the previous section, only the relevant space is considered. In this case, points in euclidean space with $\mathbf{p} \in \mathbb{R}^2$, and homogeneous points with $\tilde{\mathbf{p}} \in \mathbb{P}^2$.

Given any point $\tilde{\mathbf{p}} = [x\ y\ 1]^T$, its euclidean representation is $\mathbf{p} = [x\ y]^T$. If $\tilde{\mathbf{p}}$ is multiplied with a scalar $k \neq 0$ the euclidean point $\mathbf{p}$ is the same:

$$k \cdot \tilde{\mathbf{p}} = \begin{bmatrix} k \cdot x & k \cdot y & k \end{bmatrix}^T$$
$$\mathbf{p} = \pi(k \cdot \tilde{\mathbf{p}}) = \frac{1}{k} \begin{bmatrix} k \cdot x & k \cdot y \end{bmatrix}^T = \begin{bmatrix} x & y \end{bmatrix}^T$$

This shows that every multiple of $\tilde{\mathbf{p}}$ is projected onto the same $\mathbf{p}$. In reverse, for a single point $\mathbf{p}$ in euclidean coordinates there is an infinite number of valid homogeneous representations of said point. Or, to frame it more tangible: The color of a single pixel in an image could result from any object between the camera's lens and the horizon. The only restriction is: The possible homogeneous points all lie one a single line through the origin and the canonical point $\Pi(\mathbf{p})$. Given the coefficients of a line through the origin, $\mathbf{l} = [a\ b\ c]^T$, a point $\tilde{\mathbf{p}}$ is part of the line if the inner product $\mathbf{l}^T \cdot \tilde{\mathbf{p}}$ is 0. It follows, every multiple of $\tilde{\mathbf{p}}$, $k \cdot \tilde{\mathbf{p}}$, is also part of the line:

$$\begin{aligned} \mathbf{l}^T \cdot (k \cdot \tilde{\mathbf{p}}) &= 0 \\ a \cdot k \cdot x + b \cdot k \cdot y + c \cdot k &= 0 \\ k \cdot (a \cdot x + b \cdot y + c) &= 0 \\ \mathbf{l}^T \cdot \tilde{\mathbf{p}} &= 0 \end{aligned}$$

To summarize, homogeneous coordinates are useful to simplify the application of affine transformations and provide a simple way to formalize projective mappings from an $n+1$-dimensional space to the n-dimensional one.

A REMARK ON THE NOTATION As stated in the list of symbols, a boldface lowercase letter, $\mathbf{p}$, denotes a 2-D point in euclidean coordinates. With a tilde, $\tilde{\mathbf{p}}$, the point becomes a 3-D point in homogeneous coordinates. However, there is also the boldface uppercase letter, $\mathbf{P}$, that denotes a 3-D point in euclidean coordinates. Strictly speaking, they are not the same because they are elements of different vector spaces. For all practical purposes though, $\tilde{\mathbf{p}}$ and $\mathbf{P}$ are treated the same and used interchangeably throughout the thesis.

2.3 THE PINHOLE MODEL AND THE CALIBRATED CAMERA

To model how objects in a three-dimensional world are mapped onto a two-dimensional representation of the world — also called an image or picture — the basic pinhole camera model is employed. It consists of an optical center, $\mathbf{C}$, that is located at the origin of the camera's coordinate frame, an image plane, onto which three-dimensional points are mapped, and the focal length, f, that determines the distance between optical center and image plane. Figure 2.1 shows a schematic.

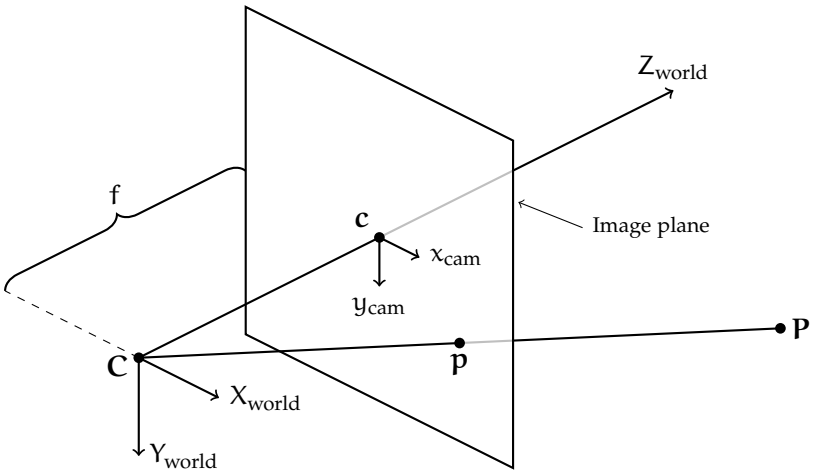


Figure 2.1: The pinhole camera model. The optical center is marked by $\mathbf{C}$; the optical axis is the line through $\mathbf{C}$ and the principal point, $\mathbf{c}$; the distance between optical center and the image plane is the focal length, f.

Although the pinhole camera is a simple model, it is a useful approximation of real cameras and serves as a basis for more complicated models [HZ03].

To compute the coordinates of the projected point, $\mathbf{p}$, consider the diagram in Figure 2.2. The line through the optical center and $\mathbf{P}$ determines where $\mathbf{p}$ is located on the image plane. Using the theorem of intersecting lines, the relations

$$\begin{aligned} \frac{x}{X} &= \frac{f}{Z}, & x &= f \cdot \frac{X}{Z}, \\ \frac{y}{Y} &= \frac{f}{Z}, & y &= f \cdot \frac{Y}{Z} \end{aligned}$$

hold. The focal length acts as scaling factor: It sets the z-coordinate of the homogeneous representation of $\mathbf{p}$ equal to the focal length f, scales the remaining two components and thus places the point on the image plane.

Because the focal length scales X and Y, the above behavior is described by a scale matrix:

$$\mathcal{P} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$\mathcal{P}$ is called the projection matrix and projects from projective space $\mathbb{P}^2$ to euclidean space $\mathbb{R}^2$. Using the projection matrix, it is easy to see, that the coordinates of $\mathbf{p}$ are computed by

$$\mathbf{p} = \pi(\mathcal{P} \cdot \mathbf{P}) = \frac{1}{\mathbf{P}_z} \cdot \begin{bmatrix} f \cdot \mathbf{P}_x \\ f \cdot \mathbf{P}_y \end{bmatrix}.$$

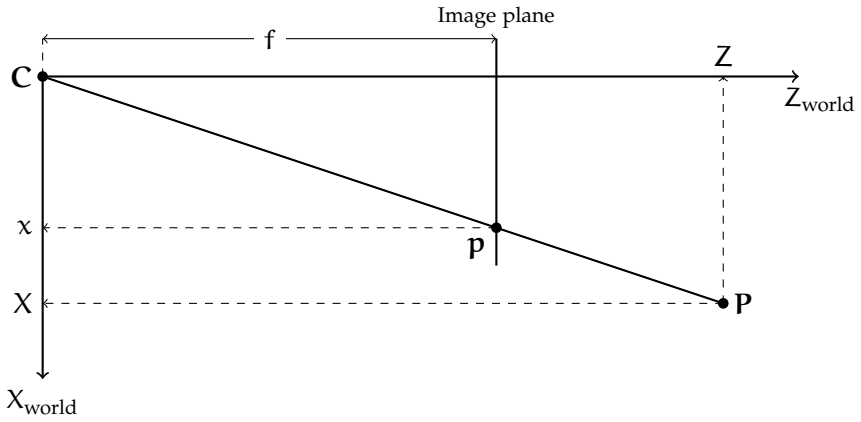


Figure 2.2: The pinhole camera as viewed from above. A point $\mathbf{P}$ is mapped to its counterpart $\mathbf{p}$.

Unfortunately, the camera model imposes several restrictions that limit its usefulness.

1. The camera's coordinate frame coincides with the world coordinate frame. For a single camera this restriction would not matter, for multiple cameras it is necessary to express that the camera can be positioned freely in the world coordinate system.

2. The principal point is usually located in the center of the image, whereas the origin of a picture is commonly one of the corners, in this case it is the upper left corner. A translation of the principal point is needed.

3. Real world pictures are seldom square and a picture's contents are accessed in terms of pixels. A conversion from coordinates in distance units to pixel coordinates with different scaling factors per coordinate is required.

Lifting the above restrictions gives rise to the external, or extrinsic, and internal, or intrinsic, camera parameters that allow to model arbitrary cameras at arbitrary positions.

2.3.1 *Extrinsic Camera Parameters*

The extrinsic parameters encapsulate the external configuration of the camera, that is, its orientation $\mathcal{R}_c$ and location $\mathcal{C}$. Both, $\mathcal{R}_c$ and $\mathcal{C}$ are relative to a world coordinate frame and specify the camera's transformation in it.

A camera is a rigid body, therefore the camera's transformation is expressed by isometric transformations, namely rotations and trans-

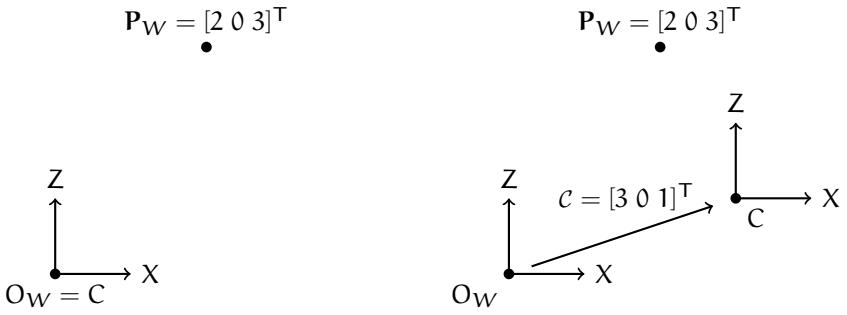


Figure 2.3: Left: The world and camera coordinate frame align, and the coordinate of the point $\mathbf{P}$ relative to each frame is the same.
Right: The camera was moved by $\mathcal{C} = (3\ 0\ 1)^\mathsf{T}$, the coordinate of $\mathbf{P}$ relative to the camera is $\mathbf{P}_C = (-1\ 0\ 2)^\mathsf{T} = \mathbf{P}_w - \mathcal{C}$; the translation of the point relative to the camera is exactly the inverse of the translation of the camera relative to the world.

lations [HZ03]. In homogeneous coordinates this transformation is represented by a matrix

$$\mathsf{T}_{\text{world}\to\text{cam}} = \begin{bmatrix} \mathcal{R}_c & \mathcal{C} \\ 0^\mathsf{T} & 1 \end{bmatrix}.$$

$\mathsf{T}_{\text{world}\to\text{cam}}$ moves the camera in the world coordinate frame, but a transformation is required that expresses the contents of the world relative to the camera coordinate frame. The transformation matrix to achieve this, $\mathsf{T}_{\text{cam}\to\text{world}}$, is the inverse of the already known camera transformation matrix. Consider Figure 2.3, the diagram on the left shows a state, in which camera and world coordinate frame are aligned, and a point in the world called $\mathbf{P}$. The coordinates of $\mathbf{P}$ in relation to both frames are exactly the same. In the diagram on the right, the camera coordinate frame was translated by $\mathcal{C}$. The coordinates of $\mathbf{P}$ relative to the world frame are the same as in the left diagram. However, the coordinates of the point relative to the camera frame have changed by the inverse of the camera translation, $\mathcal{C}^{-1}$. The matrix to accomplish the transformation is

$$\mathsf{T} = \mathsf{T}_{\text{cam}\to\text{world}} = \begin{bmatrix} \mathcal{R}_c^\mathsf{T} & -\mathcal{R}_c\mathcal{C} \\ 0^\mathsf{T} & 1 \end{bmatrix} = \begin{bmatrix} \mathcal{R} & t \\ 0^\mathsf{T} & 1 \end{bmatrix}. \tag{2.3}$$

Its derivation is found in Appendix A.1. The matrix T is called the extrinsic matrix and with it the first restriction is lifted.

2.3.2 *Intrinsic Camera Parameters*

For images it is customary that the origin is located in one of the corners — here in the upper left corner, see Figure 2.4. A point $\mathbf{P}$ shall be mapped to $\mathbf{p}$, such that

$$\mathbf{P} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \mapsto \begin{bmatrix} f \cdot \frac{X}{Z} + c_x \\ f \cdot \frac{Y}{Z} + c_y \end{bmatrix} = \frac{1}{z} \cdot \begin{bmatrix} f \cdot X + Z \cdot c_x \\ f \cdot Y + Z \cdot c_y \end{bmatrix} = \mathbf{p}.$$

$[c_x \; c_y]^T$ describes a translation after the projection onto the image plane and the projection matrix, including an adjusted origin, is expressed by

$$\mathcal{P} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.4}$$

where $[c_x \; c_y]^T$ is the coordinate of the principal point, $\mathbf{c}$, in relation to the image frame [HZ03].

The coordinates of the projected points are not directly mapped to the pixels of a picture. Moreover, the ratio between width and height of an image is generally not equal to one. Therefore, additional parameters are required that relate the projected coordinates to pixel coordinates and allow different scaling factors per axis. The matrix then reads

$$K = \begin{bmatrix} \gamma_x & 0 & 0 \\ 0 & \gamma_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.5}$$

where γ_x and γ_y are the scaling factors that relate quantities in distance units to pixels and are measured in pixels per unit distance; s_x and s_y are the focal lengths per axis, now measured in pixels; and o_x and o_y are the offsets of the principal point in pixels as well. The matrix K is called the intrinsic matrix [FLM92].

K is an upper triangular matrix, thus its determinant is $\det(K) = s_x \cdot s_y \neq 0$ and K is invertible. The inverse reads

$$K^{-1} = \begin{bmatrix} \frac{1}{s_x} & 0 & \frac{-o_x}{s_x} \\ 0 & \frac{1}{s_y} & \frac{-o_y}{s_y} \\ 0 & 0 & 1 \end{bmatrix}.$$

2.3.3 *The Projection Matrix*

With the extrinsic and intrinsic parameters in place, the full projection from $\mathbb{P}^3$ to $\mathbb{R}^2$ is defined. The extrinsic matrix, however, is a 4×4

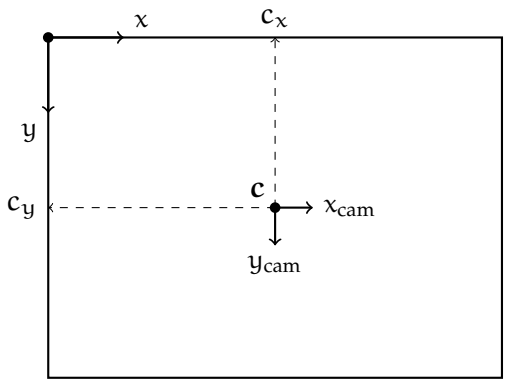


Figure 2.4: The coordinates of the principal point, $\mathbf{c}$, relative to the image frame.

matrix, whereas the intrinsic matrix is a 3×3 matrix. The extrinsic matrix, though, consists only of isometric transformations and for every homogeneous coordinate that is transformed with the extrinsic matrix the fourth component remains as it is. For all considerations in this thesis the fourth component is always one; dividing by the fourth component is the same as simply dropping it. This is achieved by augmenting the intrinsic matrix by a fourth column that is entirely zero. The full projection thus reads

$$\mathbf{p} = \pi\left([K \mid 0] \cdot T \cdot \tilde{\mathbf{P}}\right) = \pi\left(M \cdot \tilde{\mathbf{P}}\right), \tag{2.6}$$

where $M = [K \mid 0] \cdot T$ is called the full projection matrix and encapsulates the entire camera configuration.

2.4 MULTIPLE CAMERAS AND EPIPOLAR GEOMETRY

Equipped with the knowledge of Section 2.3, it is now possible to describe an arbitrary number of cameras in the world. The depth, however, is computed for a single camera, called the reference camera. It shall be denoted by C_0. With a total of n cameras, the remaining $n-1$ cameras are called the match cameras and are denoted by C_ν, with $\nu \in [1, n]$. The matrices of these cameras — extrinsic, intrinsic and full projection matrix — are labeled accordingly.

For any point $\tilde{\mathbf{P}}_\nu$ in the coordinate frame of camera ν, the coordinate of $\tilde{\mathbf{P}}_j$ in the frame of camera j is expressed by

$$\tilde{\mathbf{P}}_j = T_j \cdot T_\nu^{-1} \cdot \tilde{\mathbf{P}}_\nu.$$

Having a dedicated reference camera, it makes sense to align its coordinate frame with the world frame. This simplifies some equations later and does not change the relative arrangement of all involved cameras. The alignment is performed by multiplying all extrinsic matrices by the inverse extrinsic matrix of the reference camera,

$$\check{T}_\nu = T_\nu \cdot T_0^{-1}, \quad \forall \nu \in [0, n].$$

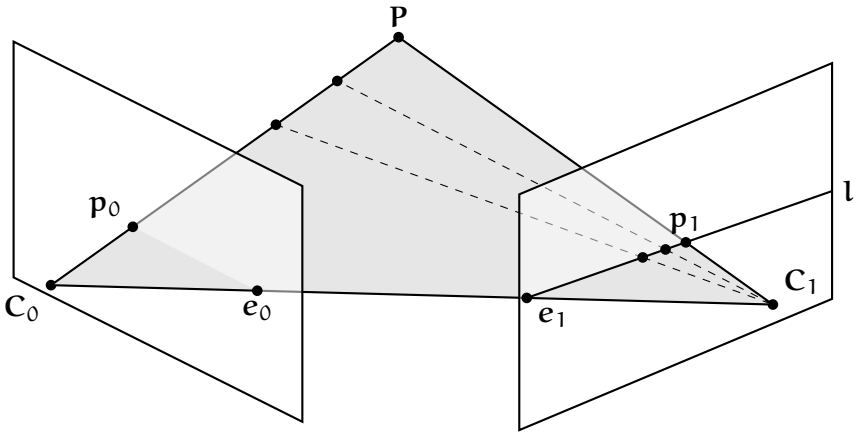


Figure 2.5: Epipolar geometry between two cameras in a converging setup.

The extrinsic matrix of the reference camera becomes the identity and the projection matrix simplifies to $M_0 = [K_0 \mid 0]$.

The relation of image points for any pair of cameras is described by the epipolar geometry. In Figure 2.5 the setup for two cameras with centers $\mathbf{C}_0$ and $\mathbf{C}_1$ is depicted. The projections of the center of one camera onto the image plane of the other camera are called the epipoles, $\boldsymbol{e}_0$ and $\boldsymbol{e}_1$. As mentioned earlier, all possible three-dimensional representations of a point $\mathbf{p}$ lie on a straight line through the camera center and $\mathbf{p}$. The camera centers and $\mathbf{p}$ define a plane, called the epipolar plane; the epipoles and all possible $\mathbf{P}$ are part of this plane. Figure 2.5 also depicts how the points $\mathbf{P}$ are projected onto the image plane of camera $\mathbf{C}_1$. The line $\overline{\mathbf{C}_0\mathbf{P}}$ is projected onto I_1 as a line through the epipole and $\mathbf{p}_1$, $\mathbf{l} = \overline{\boldsymbol{e}_1\mathbf{p}_1}$. Alternatively, $\mathbf{l}$ is the result of the intersection between the image plane and the epipolar plane. The line $\mathbf{l}$ is called the epipolar line. The relation between points in the first and the second camera can be expressed by a matrix F, called the fundamental matrix, and leads to the epipolar constraint

$$\tilde{\mathbf{p}}_1^T \cdot F \cdot \tilde{\mathbf{p}}_0 = 0. \tag{2.7}$$

F is a 3×3 matrix but has only rank 2 and is thus not invertible. However, by substituting $\mathbf{l}$ for $F \cdot \tilde{\mathbf{p}}_0$, it can be seen that F describes the precondition that all $\mathbf{p}_1$ lie on the epipolar line by

$$\tilde{\mathbf{p}}_1^T \cdot \mathbf{l} = 0. \tag{2.8}$$

For calibrated cameras the fundamental matrix is computed from the projection matrices and the fact that all correspondences of $\mathbf{p}_0$ lie on the epipolar line; for uncalibrated cameras F can be computed from known point correspondences. The epipolar constraint has an important role in solving the stereo problem: It restricts the two-dimensional search space for correspondences to a one-dimensional one, greatly reducing complexity and potential for erroneous matches.

An important special case is the ortho-parallel camera setup. In this scenario, the view direction of all cameras is parallel to the view direction of the reference camera; the image planes of all cameras are parallel, too. If the same settings for all cameras were used, the intrinsic matrices would be equal to each other. Additionally, the camera centers can be made to all lie on a line. If both of those conditions are true, all epipolar lines are parallel to each other and two corresponding points have the same y-coordinate.

2.5 CALCULUS OF VARIATIONS

As mentioned in Chapter 1, the solution to the stereo problem will be the minimization of an energy functional. A functional can be seen as a function of functions and Calculus of Variations deals with finding functions that maximize or minimize a given functional. A functional can be defined as

$$E(g) = \int_a^b F(x, g, g_x)\,dx, \tag{2.9}$$

such that F is a function $[a, b] \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, which is differentiable on $[a, b]$ with respect to g and g_x; g is a differentiable function $\mathbb{R} \to \mathbb{R}$ and g_x is the derivative of g. If g is a minimizer of the functional the Euler-Lagrange equation has to be satisfied [Fox50]. It reads

$$0 = F_g - \frac{d}{dx} F_{g_x}, \tag{2.10}$$

where F_g is the partial derivative of F with respect to g and F_{g_x} is the partial derivative of F with respect to g_x. The boundary conditions are

$$F_{g_x}(a, g(a), g_x(a)) = 0 \text{ and}$$
$$F_{g_x}(b, g(b), g_x(b)) = 0.$$

2.5.1 *Functionals of Multiple Variables*

In the case of a rectangular domain $\Omega \subset \mathbb{R}^2$ the energy functional takes the form

$$E(g) = \int_\Omega F(x, y, g, g_x, g_y)\,dx\,dy.$$

As before, F is differentiable on Ω with respect to g, g_x and g_y, where g_x and g_y are the partial derivatives of g with respect to x and y. The Euler-Lagrange equation can be generalized to the case of several variables and provides the necessary condition for the minimizer g. It reads

$$0 = \partial_g F - \partial_x F_{g_x} - \partial_y F_{g_y}, \tag{2.11}$$

with the boundary condition

$$0 = \mathbf{n}^T \cdot \begin{bmatrix} F_{g_x} \\ F_{g_y} \end{bmatrix},$$

where $\mathbf{n}$ is the normal vector at the boundary. Using the divergence operator, Equation (2.11) can be formulated in a more concise way by

$$0 = \partial_g F - \operatorname{div} \begin{bmatrix} F_{g_x} \\ F_{g_y} \end{bmatrix}. \tag{2.12}$$

2.5.2 *Functionals of Higher-Order Derivatives*

In the presence of second-order derivatives the energy functional for two variables reads

$$E(g) = \int_\Omega F(x, y, g, g_x, g_y, g_{xx}, g_{xy}, g_{yx}, g_{yy})\, dx\, dy.$$

In this case F and g must be two times differentiable over the domain Ω. The functions g_{xx}, g_{xy}, g_{yx} and g_{yy} are the partial second derivatives of g. The corresponding Euler-Lagrange equation is an extension of Equation (2.11) and reads

$$\begin{aligned} 0 = {} & \partial_g F - \partial_x F_{g_x} - \partial_y F_{g_y} \\ & + \partial_{xx} F_{g_{xx}} + \partial_{xy} F_{g_{xy}} + \partial_{yx} F_{g_{yx}} + \partial_{yy} F_{g_{yy}}, \end{aligned} \tag{2.13}$$

with the boundary conditions [Mau14]

$$\begin{aligned} 0 &= \mathbf{n}^T \cdot \begin{bmatrix} F_{g_x} - \partial_x F_{g_{xx}} - \partial_y F_{g_{xy}} \\ F_{g_y} - \partial_x F_{g_{yx}} - \partial_y F_{g_{yy}} \end{bmatrix}, \\ 0 &= \mathbf{n}^T \cdot \begin{bmatrix} F_{g_{xx}} \\ F_{g_{xy}} \end{bmatrix}, \quad 0 = \mathbf{n}^T \cdot \begin{bmatrix} F_{g_{yx}} \\ F_{g_{yy}} \end{bmatrix}. \end{aligned}$$

3 METHODOLOGY

This chapter introduces a depth parameterization used in the multi-view stereo approach and explains the reasoning for choosing an alternative depth parameterization. A model using a global energy functional is specified and the steps necessary to find the minimizer are described. To better utilize the properties of the new depth parameterization, the model is adapted to use a second-order regularizer.

3.1 RELATING IMAGE POINTS

The previous chapter explained how to transform points in three- and four-dimensional space and how to project points onto image planes. This assumes known 3-D objects. The goal of stereo reconstruction, however, is the computation of 3-D points, where the points are the unknowns. A set of calibrated cameras with images of the scene corresponding to the camera's location and orientation is given. Additionally, there is a dedicated reference camera, its coordinate frame is aligned to the world's coordinate frame and the 3-D coordinates of the points in the scene are computed relative to the reference camera. The problem of finding the corresponding 3-D point to a point $\mathbf{p}$ on the image plane is dual to finding the depth value at $\mathbf{p}$.

3.1.1 *Back-Projection to a Surface*

As explained in Section 2.2.2, all possible 3-D representations of $\mathbf{p}$ lie on a line and the depth at $\mathbf{p}$ correlates with the scaling factor k of $\mathbf{l}$. Formally, the depth is defined as a function

$$\begin{aligned} \rho : \mathbb{R}^2 &\to \mathbb{R} \\ (\mathbf{p}) &\mapsto \rho(\mathbf{p}), \end{aligned} \tag{3.1}$$

such that it relates the image points of the reference camera C_0 to 3-D points. To compute those 3-D points, recall how the point projected onto the image plane is computed from $\mathbf{P}$:

$$\begin{aligned} \mathbf{p} &= \pi(M_0 \cdot \tilde{\mathbf{P}}) = \pi(K_0 \cdot \mathbf{P}) \\ &= \pi\left(\begin{bmatrix} X \cdot s_x + Z \cdot o_x \\ Y \cdot s_y + Z \cdot o_y \\ Z \end{bmatrix}\right) = \begin{bmatrix} \frac{X}{Z} \cdot s_x + o_x \\ \frac{Y}{Z} \cdot s_y + o_y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \end{aligned}$$

This provides two equations with three unknowns, X, Y and Z. From Figure 2.2 it is evident that there are two possibilities to relate

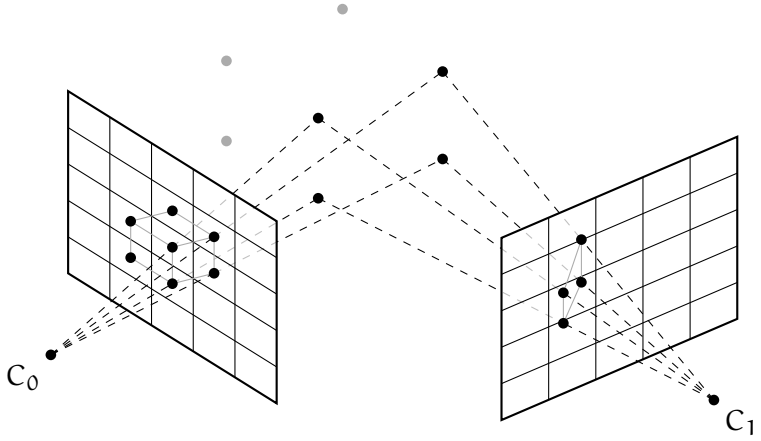


Figure 3.1: The pixels of I_0 are back-projected to a point cloud. The result is then projected onto I_1.

the Z-coordinate of $\mathbf{P}$ with the depth. The first is to project $\mathbf{p}$ back along the optical axis, that is the $Z = \rho(\mathbf{p})$. The second possibility is to back-project along the viewing ray. Using the euclidean distance, it follows that $Z = \sqrt{\rho(\mathbf{p})^2 - X^2 - Y^2}$. Schroers et al. found, however, that the latter back-projection in conjunction with a standard regularizer results in a curved surface even for constant depth [SHW15]. Consequently, the first back-projection is used.

Substituting Z with $\rho(\mathbf{p})$ using the back-projection along the optical axis leaves only two unknowns; solving for X and Y respectively gives the following equations:

$$\begin{aligned} X &= \rho(\mathbf{p}) \cdot \frac{x - o_x}{s_x} \\ Y &= \rho(\mathbf{p}) \cdot \frac{y - o_y}{s_y} \\ Z &= \rho(\mathbf{p}) \end{aligned} \tag{3.2}$$

From Equations 3.2 follows, the points $\mathbf{P}$ can be specified in terms of $\mathbf{p}$, leading to a parameterization of the point cloud resulting from the back-projection:

$$\mathcal{P}(\mathbf{p}) = \rho(\mathbf{p}) \cdot K^{-1} \cdot \Pi(\mathbf{p}) \tag{3.3}$$

The process is depicted in Figure 3.1. Each image point is projected back into 3-D space using its depth value. For a continuous image the back-projection results in a closed surface; for a discrete picture, however, only a sparse representation of the original surface is reconstructed, forming a point cloud. The elements of the point cloud are then projected onto the image plane of the second camera, which is covered in Section 3.1.3.

3.1.2 *Introducing a Depth Parameterization*

In Equations 3.2 the depth $\rho(\mathbf{p})$ is directly related to the Z coordinate of $\mathbf{P}$. There is no requirement for that to be the case, though. Suppose a function Φ exists, such that $Z = \Phi(\rho(\mathbf{p}))$, and that the objective remains to find $\rho(\mathbf{p})$. Obviously, in this case the found depth would not be the same, but the back-projected points and so the reconstructed 3-D scene would. If the result does not change, why then should an additional function be introduced? Before this question is answered in Section 3.2, two extra steps are necessary. The first is to introduce a formal depth parameterization Φ of the following form:

$$\begin{aligned} \Phi : \mathbb{R} &\to \mathbb{R} \\ \rho &\mapsto \Phi(\rho) \end{aligned} \tag{3.4}$$

Although not a necessity, Φ should be invertible to compute ρ from Z. As seen later, the second derivative of Φ is required; Φ has to be two times differentiable. The relation between ρ and Z is now established as

$$Z = \Phi(\rho(\mathbf{p}))$$

Thus, the point cloud parameterization of Equation (3.3) can be reformulated:

$$\mathcal{P}(\mathbf{p}, \Phi \circ \rho) = \Phi(\rho(\mathbf{p})) \cdot K^{-1} \cdot \Pi(\mathbf{p}) \tag{3.5}$$

3.1.3 *Projection onto Match Images*

The last step is to project the point cloud onto the match images to relate the pixels from image I_0 to pixels in I_ν at their corresponding coordinates. The point cloud is given in its parameterized form, $\mathcal{P}(\mathbf{p}, \Phi \circ \rho)$. Because the coordinate frame of camera C_0 is aligned with the world coordinate frame, the point cloud is already in world coordinates and the projection onto I_ν is achieved by

$$\mathbf{p}_\nu(\mathbf{p}, \Phi \circ \rho) = \pi(M_\nu \cdot \Pi(\mathcal{P}(\mathbf{p}, \Phi \circ \rho))). \tag{3.6}$$

This leads to two equations

$$\begin{aligned} x_\nu(\mathbf{p}, \Phi \circ \rho) &= \frac{M_\nu^{[1]} \cdot \tilde{\mathcal{P}}(\mathbf{p}, \Phi \circ \rho)}{M_\nu^{[3]} \cdot \tilde{\mathcal{P}}(\mathbf{p}, \Phi \circ \rho)}, \\ y_\nu(\mathbf{p}, \Phi \circ \rho) &= \frac{M_\nu^{[2]} \cdot \tilde{\mathcal{P}}(\mathbf{p}, \Phi \circ \rho)}{M_\nu^{[3]} \cdot \tilde{\mathcal{P}}(\mathbf{p}, \Phi \circ \rho)}, \end{aligned} \tag{3.7}$$

where $M_v^{[r]}$ denotes the r-th row of M_v. The individual elements of M_v are denoted by m_v^{rc}, where r is the row and c the column index. The explicit formulation of Equations 3.7 is

$$x_v(\mathbf{p}, \Phi \circ \rho) = \frac{\Phi(\rho(\mathbf{p})) \cdot \overbrace{\left(m_v^{11} \cdot \frac{x - o_x}{s_x} + m_v^{12} \cdot \frac{y - o_y}{s_y} + m_v^{13}\right)}^{a(\mathbf{p})} + \overbrace{m_v^{14}}^{b(\mathbf{p})}}{\Phi(\rho(\mathbf{p})) \cdot \underbrace{\left(m_v^{31} \cdot \frac{x - o_x}{s_x} + m_v^{32} \cdot \frac{y - o_y}{s_y} + m_v^{33}\right)}_{c(\mathbf{p})} + \underbrace{m_v^{34}}_{d(\mathbf{p})}}$$
$$= \frac{\Phi(\rho(\mathbf{p})) \cdot a(\mathbf{p}) + b(\mathbf{p})}{\Phi(\rho(\mathbf{p})) \cdot c(\mathbf{p}) + d(\mathbf{p})}, \tag{3.8}$$

$$y_v(\mathbf{p}, \Phi \circ \rho) = \frac{\Phi(\rho(\mathbf{p})) \cdot \overbrace{\left(m_v^{21} \cdot \frac{x - o_x}{s_x} + m_v^{22} \cdot \frac{y - o_y}{s_y} + m_v^{23}\right)}^{\check{a}(\mathbf{p})} + \overbrace{m_v^{24}}^{\check{b}(\mathbf{p})}}{\Phi(\rho(\mathbf{p})) \cdot \underbrace{\left(m_v^{31} \cdot \frac{x - o_x}{s_x} + m_v^{32} \cdot \frac{y - o_y}{s_y} + m_v^{33}\right)}_{c(\mathbf{p})} + \underbrace{m_v^{34}}_{d(\mathbf{p})}}$$
$$= \frac{\Phi(\rho(\mathbf{p})) \cdot \check{a}(\mathbf{p}) + \check{b}(\mathbf{p})}{\Phi(\rho(\mathbf{p})) \cdot c(\mathbf{p}) + d(\mathbf{p})}. \tag{3.9}$$

The functions x_v, y_v, a, b, c, d, $\check{a}$, $\check{b}$ as well as ρ are all functions of a variable $\mathbf{p}$. In addition x_v and y_v are functions of the depth parameterization. From now on — for the sake of clarity — the arguments are omitted.

3.2 THE CASE FOR AN INVERSE DEPTH PARAMETERIZATION

The common practice regarding the depth parameterization is to use the depth directly. A second parameterization that will be examined in this thesis is the inverse depth. This results in the two parameterizations

$$\Phi_{\text{direct}}(\rho) = \rho \text{ and}$$
$$\Phi_{\text{inverse}}(\rho) = \frac{1}{\rho}. \tag{3.10}$$

To explain the benefit of the inverse depth parameterization, the effect of Φ_{direct} and Φ_{inverse} on back-projection and linearization are now examined.

3.2.1 *Linearization Error for Converging Setups*

In a typical setup the cameras keep the object of interest in the center of the view, maintain the distance to the object and circle it with only small lateral motion [SHW15]. Consider Figure 3.2, in which this setup is illustrated in a top view; both cameras have the same

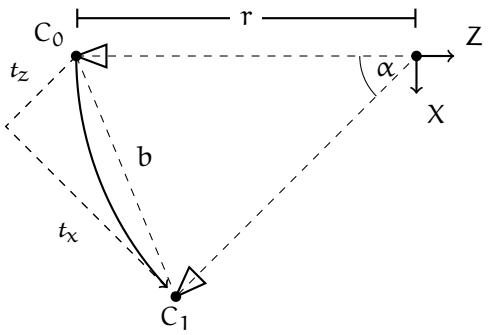


Figure 3.2: Two cameras with the same distance to the origin, oriented to look directly at the origin. The Y-axis points 'inside' the page.

distance from an object located at the origin and both cameras point directly at the object. Furthermore, in comparison to the distance to the object, the z-translation t_z is significantly smaller and vanishes with increasing radius r for a fixed t_x. Consider again Figure 3.2; for a fixed t_x the angle α is calculated as

$$\alpha = \arcsin \frac{t_x}{r}.$$

Using the cosine law, it follows for the baseline b that

$$b^2 = 2 \cdot r^2 - 2 \cdot r^2 \cdot \cos \alpha = 2 \cdot r^2 \cdot (1 - \cos \alpha).$$

The Pythagorean theorem allows to solve for t_z,

$$t_z = \sqrt{2 \cdot r^2 \cdot \left(1 - \cos\left(\arcsin \frac{t_x}{r}\right)\right) - t_x^2}.$$

With increasing distance to the object t_z approaches zero. See Figure 3.3 for a plot with varying r and fixed $t_x = 0.33$. For common camera setups it is safe to assume that the lateral camera movement is small to limit the occurrence of occlusions. In consequence, the assumption can be made that t_z is approximately zero. This concerns the converging setup; in the ortho-parallel setup this is always the case. For the extrinsic camera matrices now holds that

$$T_v = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{32} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Multiplying by the intrinsic matrix results in a new projection matrix, in which the component m_v^{34} is zero. This alters Equations 3.7. Exemplary for x_v, the result is

$$x_v = \frac{\Phi(\rho) \cdot a + b}{\Phi(\rho) \cdot c}, \tag{3.11}$$

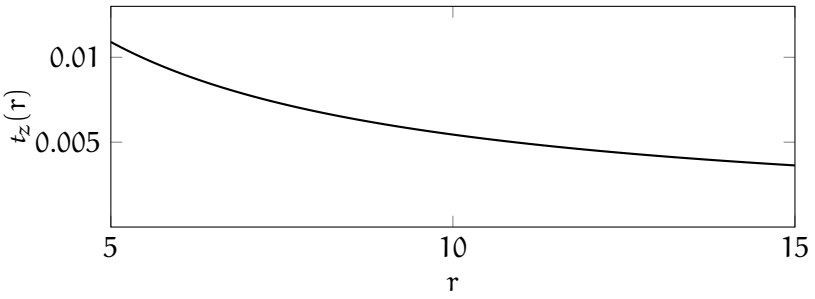


Figure 3.3: t_z approaches 0 with increasing distance r to an object using a fixed $t_x = 0.33$.

because $d = m_\nu^{34}$. The characteristics of x_ν in Equation (3.11) depend on the particular form of Φ. By applying each parameterization to Equation (3.11) different observations can be made. In the case of the direct depth parameterization Equation (3.11) becomes

$$x_\nu = \frac{a}{c} + \frac{1}{\rho} \cdot \frac{b}{c}. \tag{3.12}$$

In the case of the inverse depth parameterization the result is

$$x_\nu = \frac{\frac{1}{\rho} \cdot a + b}{\frac{1}{\rho} \cdot c} = \frac{a}{c} + \rho \cdot \frac{b}{c}. \tag{3.13}$$

In Equation (3.12) x_ν is a hyperbola with respect to the depth ρ. When linearizing the expression, an additional error is to be expected [SHW15]. In contrast, x_ν in Equation (3.13) is linear with respect to ρ and no linearization error is possible.

3.2.2 *Back-projection of Affine Depth*

Assuming a continuous depth, the back-projection results in a surface ξ, such that

$$\xi(\mathbf{p}) = \Phi(\rho(\mathbf{p})) \cdot K^{-1} \cdot \tilde{\mathbf{p}}.$$

The tangential plane of the surface ξ can be specified by its normal vector and a point on the plane; the normal vector can be computed from two vectors

$$\xi_x := \partial_x \xi(\mathbf{p}) \text{ and } \xi_y := \partial_y \xi(\mathbf{p}),$$

which both are tangent to the surface at $\mathbf{p}$. The normal is then computed from the cross product by

$$\mathbf{n} = \frac{\xi_x \times \xi_y}{\|\xi_x \times \xi_y\|_2}.$$

The depth is assumed to be affine, hence it can be described in normal form by $\rho(\mathbf{p}) = \mathbf{d}^T \tilde{\mathbf{p}}$, with $\mathbf{d} = [a\ b\ c]$. This assumption is useful

insofar as the computed depth of piecewise affine objects is again piecewise affine. The back-projection of the affine depth is expected to result in a plane; to examine the deviation from the plane, it is thus sufficient to analyze the direction of the normal, that is the vector

$$\hat{\mathbf{n}} = \xi_x \times \xi_y.$$

DIRECT DEPTH The tangents for the direct depth parameterization result in

$$\xi_x = K^{-1} \cdot \left(\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \mathbf{d}^T \tilde{\mathbf{p}} + \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \cdot a \right) = K^{-1} \cdot \begin{bmatrix} \mathbf{d}^T \tilde{\mathbf{p}} + a \\ a \cdot y \\ a \end{bmatrix} \text{ and}$$

$$\xi_y = K^{-1} \cdot \left(\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \cdot \mathbf{d}^T \tilde{\mathbf{p}} + \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \cdot b \right) = K^{-1} \cdot \begin{bmatrix} b \cdot x \\ \mathbf{d}^T \tilde{\mathbf{p}} + b \\ b \end{bmatrix}.$$

For the direction of the normal follows

$$\hat{\mathbf{n}} = K^T \cdot \begin{bmatrix} -a \\ -b \\ 2 \cdot a \cdot x + 2 \cdot b \cdot y + c \end{bmatrix}.^1$$

The direction of the surface normal depends on x and y and is therefore not constant; the surface diverges from the plane which the affine depth implies.

INVERSE DEPTH For the second parameterization, the inverse depth, the tangents are

$$\xi_x = K^{-1} \cdot \left(\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \frac{1}{\mathbf{d}^T \tilde{\mathbf{p}}} + \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \cdot \frac{-a}{(\mathbf{d}^T \tilde{\mathbf{p}})^2} \right) = \frac{K^{-1}}{(\mathbf{d}^T \tilde{\mathbf{p}})^2} \cdot \begin{bmatrix} \mathbf{d}^T \tilde{\mathbf{p}} - a \cdot x \\ -a \cdot y \\ -a \end{bmatrix},$$

$$\xi_y = K^{-1} \cdot \left(\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \cdot \frac{1}{\mathbf{d}^T \tilde{\mathbf{p}}} + \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \cdot \frac{-b}{(\mathbf{d}^T \tilde{\mathbf{p}})^2} \right) = \frac{K^{-1}}{(\mathbf{d}^T \tilde{\mathbf{p}})^2} \cdot \begin{bmatrix} -b \cdot x \\ \mathbf{d}^T \tilde{\mathbf{p}} - b \cdot y \\ -b \end{bmatrix}.$$

The direction of the surface normal is then derived as

$$\hat{\mathbf{n}} = \frac{K^T}{(\mathbf{d}^T \tilde{\mathbf{p}})^4} \cdot \mathbf{d}.$$

It follows that the normal direction is constant as expected under the assumption of an affine depth; the factor $(\mathbf{d}^T \tilde{\mathbf{p}})^{-4}$ is applied uniformly to all three components of the normal and has no effect on the direction.

1 The cross product $M\mathbf{u} \times M\mathbf{w}$, where $M \in \mathbb{R}^{3\times3}$ and $u, w \in \mathbb{R}^3$, equals $\operatorname{cof}(M)(\mathbf{u} \times \mathbf{w})$. The cofactor matrix of K^{-1} equals the transpose of K, K^T.

In summary, the inverse depth parameterization is in theory superior to the direct depth parameterization. An experimental evaluation will later show if this truly is the case or if the differences in reconstruction accuracy are negligible.

3.3 THE VARIATIONAL MODEL

To estimate the depth from a set of input images a variational approach is used. To that end, an energy functional is defined that takes the depth as a function that minimizes the global energy. It reads:

$$E(\rho) = D(\Phi \circ \rho) + \alpha \cdot S(\rho) \tag{3.14}$$

The energy functional consists of a data term $D(\Phi \circ \rho)$ and a smoothness term $S(\rho)$. The data term defines a measure of similarity between individual elements of the reference and the match image. It is assumed that for every pixel there is a property that remains constant between images. Several properties are used in the literature, be it a constant gradient [Ura+88] or even Hessian [Pap+06], but the most common choice is the use of a constant brightness [Mau14; Bro+04]. For objects with a Lambertian reflectance the brightness and color of a single point on the objects remain the same from all directions. This assumption is expressed by

$$I_0(\mathbf{p}) = I_v(\mathbf{p}_v) \Leftrightarrow I_0(\mathbf{p}) - I_v(\mathbf{p}_v) = 0,$$

where $\mathbf{p}_v = [x_v \; y_v]^T$ is defined by the functions from Equation (3.7). To penalize deviations from the brightness constancy, the squared difference $|I_0(\mathbf{p}) - I_v(\mathbf{p}_v)|^2$ could be used. This kind of penalizer, however, is quadratic and outliers produce an disproportionate influence on the energy. To overcome this behavior, a robust, sub-quadratic penalizer Ψ is introduced, which reads

$$\Psi(s^2) = \sqrt{s^2 + \epsilon^2}, \tag{3.15}$$

where ϵ is close to zero and ensures that Ψ is differentiable for $s^2 = 0$ [Mau14; Bro+04; SHW15; BMK12]. A first version of the data term considering only two views would read

$$\hat{D}(\Phi \circ \rho) = \int_\Omega \Psi_D(|I_0(\mathbf{p}) - I_1(\mathbf{p}_1)|^2)\, dx\, dy. \tag{3.16}$$

Since the point correspondences are parameterized using the depth and the resulting depth map is the same even using different match views, the extension to multiple views is simple. All views should have the same influence on the energy, thus no weighting per image is required; additional views add more constraints to the energy functional and this is expressed by adding further brightness constancy

terms for each pair of reference and match view. To keep the energy from the data term at approximately the same level when using a different number of views the data term is normalized by the amount of views. The resulting data term then reads

$$D(\Phi \circ \rho) = \frac{1}{n} \int_{\Omega} \sum_{\nu=1}^{n} \Psi_D(|I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)|^2) \, dx \, dy, \tag{3.17}$$

where n is the number of match cameras. A separate robustification scheme is used because occlusions appear independently. Using only the data term, though, results in an ill-posed problem. When comparing single pixels, it is likely that multiple pixels in the match image have the same brightness or color as the pixel in the reference image. Further, the correctly matching pixel could be occluded and the minimizer would choose the closest match in terms of similarity rather than in terms of locality. It is therefore necessary to impose an additional constraint on the form of the minimizer ρ by introducing a prior. It is reasonable to assume that for two neighboring pixels their respective depth values are the same, up to a small difference. This is obviously not true at depth discontinuities, that is on edges where an object ends and another one, closer or farther away, begins. Such an isotropic first-order penalizer, however, proves good enough when used in conjunction with a suitable penalizer function [Mau14]. The required behavior can be modeled by penalizing deviations of the gradient of ρ from zero. Again, the most simple penalizer would be quadratic. The violation of the smoothness term is desirable, though; hence the influence is weighted down using a robust penalizer. The smoothness term then reads

$$S(\rho) = \int_{\Omega} \Psi_S(|\nabla \rho|^2) \, dx \, dy. \tag{3.18}$$

A side effect of the smoothness term is its capability to fill in depth values from neighboring pixels in regions where the data term fails [HS81]. This is particularly useful in occluded or insufficiently textured regions.

This results in the final energy functional with robust penalizers for both the data and the smoothness term; a variable amount of views is used and a dense reconstruction of the depth map is achieved due to the fill-in-effect of the smoothness term. For the data and smoothness term the same sub-quadratic penalizer Ψ is used, which simplifies the energy functional to

$$E(\rho) = \int_{\Omega} \frac{1}{n} \sum_{\nu=1}^{n} \Psi(|I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)|^2) + \alpha \cdot \Psi(|\nabla \rho|^2) \, dx \, dy. \tag{3.19}$$

Using the regularization factor α, it is possible to control the influence of the smoothness term. It should be noted, however, that the

current energy functional handles only gray-value images. The inclusion of color images is treated with the introduction of the depth tensor in Section 3.3.1.3.

3.3.1 *Minimization*

To find the minimizer ρ for Equation (3.19), Calculus of Variations provide the necessary framework. The energy functional has the form

$$E(\rho) = \int_{\Omega} F(x, y, \rho, \rho_x, \rho_y)\, dx\, dy$$

and following the Euler-Lagrange Equations from Section 2.5 the necessary condition the minimizer has to fulfill is

$$0 = \partial_\rho F - \partial_x F_{\rho_x} - \partial_y F_{\rho_y}.$$

The partial derivatives of F are

$$\begin{aligned} F_\rho &= 2 \cdot \frac{1}{n} \sum_{\nu=1}^{n} \Psi'(|I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)|^2) \cdot (I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)) \\ &\qquad \cdot \partial_\rho (I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)), \\ F_{\rho_x} &= 2 \cdot \alpha \cdot \Psi'(|\nabla \rho|^2) \cdot \rho_x, \\ F_{\rho_y} &= 2 \cdot \alpha \cdot \Psi'(|\nabla \rho|^2) \cdot \rho_y. \end{aligned} \tag{3.20}$$

To achieve a shorter and more concise notation for the brightness constancy, the definition $\Delta_\nu := I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)$ is introduced; including the divergence notation for the Euler-Lagrange equation, the condition for the minimizer becomes

$$0 = \frac{1}{n} \sum_{\nu=1}^{n} \left(\Psi'(|\Delta_\nu|^2) \cdot \Delta_\nu \cdot \partial_\rho \Delta_\nu \right) - \alpha \operatorname{div} \left(\Psi'(|\nabla \rho|^2) \cdot \nabla \rho \right), \tag{3.21}$$

with boundary condition

$$0 = \mathbf{n}^T \cdot \begin{bmatrix} \Psi'(|\nabla \rho|^2) \cdot \rho_x \\ \Psi'(|\nabla \rho|^2) \cdot \rho_y \end{bmatrix},$$

where $\mathbf{n}$ is the normal vector at the boundary. Unfortunately, due to the non-linearized data term and the perspective projection the energy functional is not convex and thus possibly has multiple local minimizer. To find a good local or a global minimizer, a multi-level warping approach is introduced in Section 3.3.1.2. Another shortcoming is the non-linearity due to the sub-quadratic penalizers which prevents the application of a linear solver.

3.3.1.1 *Linearization*

The first step in solving the energy functional is to linearize Equation (3.21). To achieve this, similar to the method of [Bro+04], a fixed point iteration is introduced. The depth ρ^{k+1} is then a solution to Equation (3.21) and it can be rewritten as

$$\begin{aligned} 0 = \frac{1}{n} \sum_{\nu=1}^{n} \left(\Psi'(|\Delta_\nu^{k+1}|^2) \cdot \Delta_\nu^{k+1} \cdot \partial_\rho \Delta_\nu^k \right) \\ - \alpha \operatorname{div} \left(\Psi'(|\nabla \rho^{k+1}|^2) \cdot \nabla \rho^{k+1} \right). \end{aligned} \tag{3.22}$$

The unknown depth ρ^{k+1} is split into

$$\rho^{k+1} = \rho^k + d\rho^k,$$

such that the depth in the next time step depends on the known depth of the current iteration step and an unknown depth increment $d\rho^k$. The difference term at the next iteration step, Δ_ν^{k+1}, is dependent on the depth at $k+1$ and a linearization is required, too. It can be split into a linearization of each image. To linearize $I_\nu(\mathbf{p}_\nu^{k+1})$ a first order Taylor expansion around ρ^k is performed which yields the approximation

$$\begin{aligned} I_\nu(\mathbf{p}_\nu^{k+1}) &\approx I_\nu(\mathbf{p}_\nu^k) + \left(\rho^{k+1} - \rho^k\right) \cdot \partial_\rho I_\nu(\mathbf{p}_\nu^k) \\ &= I_\nu(\mathbf{p}_\nu^k) + d\rho^k \cdot \partial_\rho I_\nu(\mathbf{p}_\nu^k). \end{aligned}$$

For Δ_ν^{k+1} the linearization is derived by

$$\begin{aligned} \Delta_\nu^{k+1} &= I_0(\mathbf{p}^{k+1}) - I_\nu(\mathbf{p}_\nu^{k+1}) \\ &\approx I_0(\mathbf{p}^k) + d\rho^k \cdot \partial_\rho I_0(\mathbf{p}^k) - I_\nu(\mathbf{p}_\nu^k) - d\rho^k \cdot \partial_\rho I_\nu(\mathbf{p}_\nu^k) \\ &= \left(I_0(\mathbf{p}^k) - I_\nu(\mathbf{p}_\nu^k)\right) + d\rho^k \cdot \partial_\rho (I_0(\mathbf{p}^k) - I_\nu(\mathbf{p}_\nu^k)) \\ &= \Delta_\nu^k + d\rho^k \cdot \partial_\rho \Delta_\nu^k. \end{aligned}$$

For the data and smoothness term related parts of the Euler-Lagrange equation, the following abbreviations are introduced:

$$\begin{aligned} (\Psi'_\nu)_D^k &:= \Psi'(|\Delta_\nu^k + d\rho^k \cdot \partial_\rho \Delta_\nu^k|^2) \\ (\Psi')_S^k &:= \Psi'(|\nabla(\rho^k + d\rho^k)|^2). \end{aligned} \tag{3.23}$$

Putting it all together results in a new condition

$$\begin{aligned} 0 = \frac{1}{n} \sum_{\nu=1}^{n} (\Psi'_\nu)_D^k \cdot \left(\Delta_\nu^k + d\rho^k \cdot \partial_\rho \Delta_\nu^k\right) \cdot \partial_\rho \Delta_\nu^k \\ - \alpha \cdot \operatorname{div} \left((\Psi')_S^k \cdot \nabla(\rho^k + d\rho^k) \right). \end{aligned} \tag{3.24}$$

This new equation is only partially linearized, though, because the unknown $d\rho^k$ appears quadratically in the penalizer terms. Brox et al. faced a similar problem and introduced an additional, inner, fixed

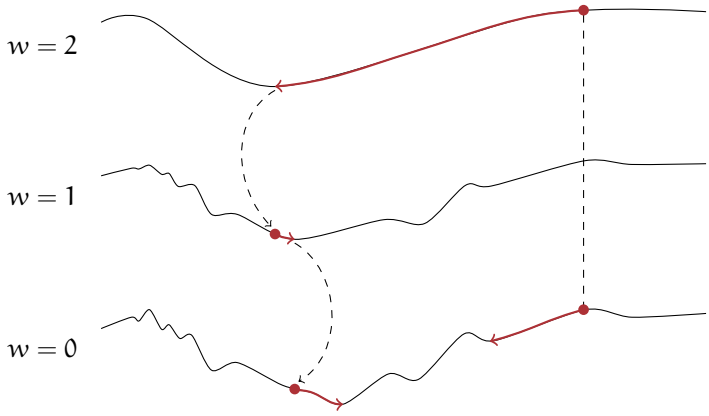


Figure 3.4: Non-convex energy with multiple local minima. A global minimizer is found on a coarse level and used as initialization on the next finer level.

point iteration scheme for each iteration k of the outer fixed point scheme to overcome this problem [Bro+04]. The same scheme is applied here. Since the chosen penalizer function Ψ is strictly convex, a minimizer for the inner fixed point iterations of the penalizer terms can be found. The fixed point of the inner scheme is $d\rho^{k,l+1}$, which results in a final linearized equation

$$\begin{aligned} 0 = \frac{1}{n} \sum_{\nu=1}^{n} (\Psi'_\nu)_D^{k,l} \cdot \left(\Delta_\nu^k + d\rho^{k,l+1} \cdot \partial_\rho \Delta_\nu^k\right) \cdot \partial_\rho \Delta_\nu^k \\ - \alpha \cdot \operatorname{div}\left((\Psi')_S^{k,l} \cdot \nabla\left(\rho^k + d\rho^{k,l+1}\right)\right), \end{aligned} \tag{3.25}$$

where l denotes the iteration step of the inner fixed point scheme.

3.3.1.2 *Coarse-to-Fine Warping*

Due to the non-convex data term the whole energy functional in Equation (3.19) is not convex. This poses a problem during minimization. Ideally, the energy term is convex and has a single minimum that is the global minimum. A non-convex function may have several local minima. The minimization shifts from finding *the* global minimum to finding a local minimum that is good enough. If it so happens that this solution is the best possible solution, the better. However, this still poses a challenge; finding a minimum depends heavily on the initialization: A bad initialization may lead to an insufficient local minimum and the solver will get stuck there. A solution could be to change the initialization manually, but for large images, which are two-dimensional functions, this is neither feasible nor desirable. A solution for this dilemma is to introduce a coarse-to-fine warping strategy, finding a minimizer on a coarse level and using this solution as initialization of the next finer level [Bro+04; MP98]. A downscaling factor, $\eta \in (0, 1)$, is introduced that relates the number of samples

per coordinate axis f^w of an image at the finest level, $w = 0$, to the number of samples f^{w+1} at the next coarser level by $f^{w+1} = \eta \cdot f^w$. Using an appropriate sampling method like area based sampling, the downscaling smooths the images and only large scale structures remain [Mau14]. The energy on a coarse level is then a smooth approximation of the energy at the next finer level. This process is illustrated in Figure 3.4. The result is that on a coarser level less local minima are present and a good local or the global minima is found. Initializing the next finer level with the solution from the coarse level reduces the risk of getting stuck near a bad initialization. Since the solution on the coarse level has fewer samples as needed on the finer level, a resampling of the computed depth is necessary.

The intrinsic matrix encodes the number of pixels per unit distance in the factors γ_x and γ_y; since the metric size of the image does not change by warping, but the number of pixels, the intrinsic matrix has to be adapted for each warping level. This is simply achieved by a multiplication with a scaling matrix, such that

$$K_\nu^{k,w+1} = \begin{bmatrix} \eta & 0 & 0 \\ 0 & \eta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot K_\nu^{k,w}, \tag{3.26}$$

where $K_\nu^{k,0} = K_\nu^k$.

At low resolution levels the regularization parameter has a higher smoothing effect than at high resolution levels. To reduce this effect the influence of the smoothness term is decreased by downscaling α, such that $\alpha^{w+1} = \eta \cdot \alpha^w$ [BMK12].

3.3.1.3 *The Depth Tensor Notation*

In dense motion estimation it is common to express the penalizer in the data term using a tensor notation; see Bruhn and Weickert [BW05] or Valgaerts et al. [Val+10], for example. Following the idea of the motion tensor, Maurer devised a depth tensor notation that proves useful to shorten the notation and for adding additional data terms [Mau14]. Let $S_{\nu\nabla}^k := [\partial_\rho \Delta_\nu^k \; \Delta_\nu^k]^T$, then the depth tensor is defined as

$$T_\nu^k := {S_{\nu\nabla}^k}^T S_{\nu\nabla}^k = \begin{bmatrix} \left(\partial_\rho \Delta_\nu^k\right)^2 & \partial_\rho \Delta_\nu^k \cdot \Delta_\nu^k \\ \partial_\rho \Delta_\nu^k \cdot \Delta_\nu^k & \left(\Delta_\nu^k\right)^2 \end{bmatrix}.$$

For the depth increment $d\rho^{k,l}$ a vector form is introduced which reads

$$\mathbf{d}\rho^{k,l} := \begin{bmatrix} d\rho^{k,l} \\ 1 \end{bmatrix}.$$

Using the depth tensor, the abbreviation from Equation (3.23) can be redefined as

$$(\Psi'_\nu)_D^{k,l} := \Psi'(\mathbf{d}\rho^{k,l\,T} T_\nu^k \, \mathbf{d}\rho^{k,l}).$$

Equation (3.25) can now be rewritten as

$$\begin{aligned} 0 =& \frac{1}{n} \sum_{\nu=1}^{n} (\Psi'_\nu)^{k,l}_D \cdot \left(T_{\nu 12} + d\rho^{k,l+1} \cdot T_{\nu 11} \right) \\ & - \alpha \cdot \operatorname{div} \left((\Psi')^{k,l}_S \cdot \nabla \left(\rho^k + d\rho^{k,l+1} \right) \right), \end{aligned} \tag{3.27}$$

with $(\Psi')^{k,l}_S$ as in Equation (3.23).

ADDING COLOR IMAGES The benefit of the depth tensor becomes immediately clear when color images are considered. Assume there are n color images as defined in Section 2.1, with three channels and $I(\mathbf{p}, c)$ is the value of the image at $\mathbf{p}$ and channel c. A possible constancy assumption that is obtained, reads

$${}^c\Delta_\nu \coloneqq I_0(\mathbf{p}, c) - I_\nu(\mathbf{p}, c).$$

It seems as if using color images increases the number of available assumptions manifold; this is only partially true, however. In the case of the common RGB color space the individual channels are not independent. Increasing the brightness the scene is exposed to — or increasing the exposure time — increases the response across all channels. RGB color images thus provide a joint constancy assumption and the violation should be penalized jointly. Hence, an appropriate data term reads

$$D(\Phi \circ \rho) = \frac{1}{n} \int_\Omega \sum_{\nu=1}^{n} \Psi \left(\sum_{c=1}^{3} |{}^c\Delta_\nu|^2 \right) dx\, dy. \tag{3.28}$$

This leads to a reformulation of the relevant part of the Euler-Lagrange equations:

$$F_\rho = 2 \cdot \frac{1}{n} \sum_{\nu=1}^{n} \Psi' \left(\sum_{c=1}^{3} |{}^c\Delta_\nu|^2 \right) \cdot \sum_{c=1}^{3} {}^c\Delta_\nu \cdot \partial_\rho\, {}^c\Delta_\nu \tag{3.29}$$

After the linearization of F_ρ as in Section 3.3.1.1 the application of the tensor notation results in

$$F_\rho = 2 \cdot \frac{1}{n} \sum_{\nu=1}^{n} \Psi' \left(\overbrace{\sum_{c=1}^{3} \mathbf{d}\rho^{k,l\mathsf{T}}\, {}^cT^k_\nu\, \mathbf{d}\rho^{k,l+1}}^{A \coloneqq} \right) \cdot \overbrace{\sum_{c=1}^{3} {}^cT^k_{\nu 12} + d\rho \cdot {}^cT^k_{\nu 11}}^{B \coloneqq} \tag{3.30}$$

It is easy to see, that A and B can be transformed, such that

$$\begin{aligned} A &= \mathbf{d}\rho^{k,l\mathsf{T}} \cdot \sum_{c=1}^{3} {}^cT^k_\nu \cdot \mathbf{d}\rho^{k,l}, \text{ and} \\ B &= \sum_{c=1}^{3} {}^cT^k_{\nu 12} + d\rho^{k,l+1} \cdot \sum_{c=1}^{3} {}^cT^k_{\nu 11}. \end{aligned}$$

Equation (3.27) can now be rewritten as

$$0 = \frac{1}{n}\sum_{\nu=1}^{n} \Psi'(\mathbf{d}\rho^{k,l\mathsf{T}}\mathbf{T}_\nu^k\,\mathbf{d}\rho^{k,l})\cdot\left(\mathbf{T}_{\nu\,12}^k + \mathrm{d}\rho^{k,l+1}\cdot\mathbf{T}_{\nu\,11}^k\right) - \alpha\cdot\operatorname{div}\left(\Psi_S'^{k,l}\cdot\nabla\left(\rho^k+\mathrm{d}\rho^{k,l+1}\right)\right), \tag{3.31}$$

where $\mathbf{T}_\nu^k$ is defined as

$$\mathbf{T}_\nu^k \coloneqq \sum_{c=1}^{3} {}^c\mathbf{T}_\nu^k = \begin{bmatrix} \sum_{c=1}^{3}(\partial_\rho\,{}^c\Delta_\nu^k)^2 & \sum_{c=1}^{3}\partial_\rho\,{}^c\Delta_\nu^k\cdot{}^c\Delta_\nu^k \\ \sum_{c=1}^{3}\partial_\rho\,{}^c\Delta_\nu^k\cdot{}^c\Delta_\nu^k & \sum_{c=1}^{3}({}^c\Delta_\nu^k)^2 \end{bmatrix}$$

This shows that adding new constancy assumptions can be expressed by transforming the assumption into a tensor and adding it to the already existing tensor. As a consequence, the discretization and the solver does not have to change when adding new assumptions.

3.3.2 *Discretization*

To solve Equation (3.27) numerically, it is necessary to discretize the continuous functions used so far. In the context of image processing, it is common to use a rectangular grid of extent $W \times H$, where W and H are the width and height in pixels of the picture taken by the reference camera; the spacing between pixels is denoted by h_x and h_y. The domain Ω is then defined as $\Omega \coloneqq [0, W) \times [0, H) \subset \mathbb{N}^2$. As depicted in Figure 3.1, the back-projection is performed at discrete locations only. This allows to express the discretized reference image by

$$I_{0\,i,j} \coloneqq I_0(i,j), \quad \forall (i,j) \in \Omega.$$

The depth and depth increment are defined over the domain Ω, too; the same discretization scheme is applied, such that $\mathrm{d}\rho_{i,j}^{k,l+1}$ and $\rho_{i,j}^{k,l}$ are the discretized depth increment and depth.

To compute the derivatives of the discretized functions, finite difference schemes are used. At the boundary, this requires to access values at locations outside of Ω, which are not available. The boundary conditions of the Euler-Lagrange equation determine the values outside the domain. For the first-order isotropic regularizer, the values inside the domain are mirrored along the boundary. At the left and top boundary this is expressed by

$$g_{i,j} = g_{|i-1|,|j-1|}, \quad \forall i,j < 0.$$

At the right and bottom boundary the width and height have to be considered.

The indexing tuple (i,j) refers to a location in the domain of the reference image; therefore, the tuple cannot be used directly to refer

to corresponding points in I_ν. However, using the current depth, the coordinate of the corresponding point can be computed, as seen in Equation (3.9). By warping the match images towards the reference view, a new image can be defined, such that the values are accessible using $\mathbf{p}$. The warped image $\mathcal{I}_\nu^k(\mathbf{p})$ is defined as

$$\mathcal{I}_\nu^k(\mathbf{p}) \coloneqq I_\nu(\mathbf{p}_\nu^k),$$

where $\mathbf{p}_\nu^k$ is the coordinate of the corresponding point using the current depth. The discretized match image is then denoted by $\mathcal{I}_{\nu\,i,j}^k$. Because the x_ν^k and y_ν^k are in general not natural numbers, the value at $\mathbf{p}_\nu^k$ is obtained by performing a bi-linear interpolation [BMK12; Mau14].

The penalizer function has the analytic derivative

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

the discrete penalizers $(\Psi'_D)_{i,j}$ and $(\Psi'_S)_{i,j}$ are computed by their discretized arguments.

THE DISCRETIZED DEPTH TENSOR The discretized representation of the part related to the data term reads

$$\frac{1}{n} \sum_{\nu=1}^{n} (\Psi'_\nu)_{D\,i,j}^{k,l} \cdot \left((\mathbf{T}_{\nu\,12})_{i,j} + d\rho_{i,j}^{k,l+1} \cdot (\mathbf{T}_{\nu\,11})_{i,j} \right). \tag{3.32}$$

Therefore, a discretization of the depth tensor is required, which in turn requires discretized $\Delta_{\nu\,i,j}^k$ and $\partial_\rho \Delta_{\nu\,i,j}^k$. The former is defined by its individual discretized parts, $\Delta_{\nu\,i,j}^k = \mathcal{I}_{0\,i,j}^k - \mathcal{I}_{\nu\,i,j}^k$; the latter requires handling the derivative by ρ. The difference term is defined as $\Delta_\nu \coloneqq I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)$, using the sum rule of differentiation it follows that

$$\partial_\rho \Delta_\nu^k = \partial_\rho I_0(\mathbf{p}) - \partial_\rho I_\nu(\mathbf{p}_\nu^k),$$

and that it is sufficient to treat the differentiation by ρ on a per image basis. Further, I_0 does not depend on ρ and the image's derivative by ρ is 0. Using the chain rule it follows

$$\begin{aligned} \partial_\rho I_\nu(\mathbf{p}_\nu^k) &= (\nabla_\nu I_\nu(\mathbf{p}_\nu^k))^\mathsf{T} \cdot \partial_\rho \begin{bmatrix} x_\nu^k \\ y_\nu^k \end{bmatrix} \\ &= \partial_{x_\nu} I_\nu \cdot \partial_\rho x_\nu^k + \partial_{y_\nu} I_\nu \cdot \partial_\rho y_\nu^k, \end{aligned} \tag{3.33}$$

where $\nabla_\nu = \begin{bmatrix} \partial_{x_\nu^k} & \partial_{y_\nu^k} \end{bmatrix}^\mathsf{T}$ is the gradient by x_ν^k and y_ν^k, respectively. The terms x_ν^k and y_ν^k can be derived analytically and using the short notation from Equation (3.9) they read

$$\begin{aligned} \partial_\rho x_\nu^k &= \partial_\Phi x_\nu \cdot \partial_\rho \Phi = \Phi'(\rho) \cdot \frac{a \cdot d - b \cdot c}{(\Phi(\rho) \cdot c + d)^2}, \text{ and} \\ \partial_\rho y_\nu^k &= \partial_\Phi y_\nu \cdot \partial_\rho \Phi = \Phi'(\rho) \cdot \frac{\check{a} \cdot d - \check{b} \cdot c}{(\Phi(\rho) \cdot c + d)^2}. \end{aligned} \tag{3.34}$$

The derivations of both can be found in Appendix B.5. The derivatives of Φ are

$$\Phi'_{\text{direct}}(\rho) = 1, \text{ and}$$
$$\Phi'_{\text{inverse}}(\rho) = \frac{-1}{\rho^2}.$$

Computing the derivatives $(\nabla_v I_v)^T$ is performed using the warped image [BMK12]. Between the gradient of the image and the gradient of the warped image exists the relation

$$(\nabla_v I_v(\mathbf{p}_v^k))^T = (\nabla \mathcal{I}_v^k(\mathbf{p}))^T \cdot J^{-1},$$

where J is the Jacobian as derived in Appendix B.1. To approximate the partial derivatives of $\mathcal{I}_v$, an eighth order finite difference scheme is used.

THE DISCRETIZED SMOOTHNESS TERM To approximate the divergence expression of the smoothness term, a nested, central second order difference scheme was chosen. Its derivation is found in Appendix B.2. The individual parts are discretized as usual and the smoothness term reads

$$\begin{aligned}
&\alpha \cdot \operatorname{div}\left((\Psi')^{k,l}_{S\,i,j} \cdot \nabla\left(\rho^{k,l}_{i,j} + d\rho^{k,l+1}_{i,j}\right)\right) \\
=& w_1 \cdot \left(\rho^{k,l}_{i+1,j} - \rho^{k,l}_{i,j} + d\rho^{k,l+1}_{i+1,j} - d\rho^{k,l+1}_{i,j}\right) \\
-& w_2 \cdot \left(\rho^{k,l}_{i,j} - \rho^{k,l}_{i-1,j} + d\rho^{k,l+1}_{i,j} - d\rho^{k,l+1}_{i-1,j}\right) \\
+& w_3 \cdot \left(\rho^{k,l}_{i,j+1} - \rho^{k,l}_{i,j} + d\rho^{k,l+1}_{i,j+1} - d\rho^{k,l+1}_{i,j}\right) \\
-& w_4 \cdot \left(\rho^{k,l}_{i,j} - \rho^{k,l}_{i,j-1} + d\rho^{k,l+1}_{i,j} - d\rho^{k,l+1}_{i,j-1}\right),
\end{aligned} \tag{3.35}$$

with the weights w_i defined as

$$\begin{aligned}
w_1 &= \alpha \cdot \frac{(\Psi')^{k,l}_{S\,i+1,j} + (\Psi')^{k,l}_{S\,i,j}}{2 \cdot h_x^2}, \\
w_2 &= \alpha \cdot \frac{(\Psi')^{k,l}_{S\,i,j} + (\Psi')^{k,l}_{S\,i-1,j}}{2 \cdot h_x^2}, \\
w_3 &= \alpha \cdot \frac{(\Psi')^{k,l}_{S\,i,j+1} + (\Psi')^{k,l}_{S\,i,j}}{2 \cdot h_y^2}, \\
w_4 &= \alpha \cdot \frac{(\Psi')^{k,l}_{S\,i,j} + (\Psi')^{k,l}_{S\,i,j-1}}{2 \cdot h_y^2}.
\end{aligned} \tag{3.36}$$

The expressions $(\Psi')^{k,l}_{S\,i,j}$ are evaluated using the analytic derivative of Ψ_S and an approximation of its argument as per Equation (3.23). The partial derivatives are approximated using an eighth order finite difference scheme.

3.3.3 *Solving the Energy Functional*

Now that Equation (3.25) is discretized, it must be solved. The variable to solve for is $d\rho_{i,j}^{k,l+1}$, where each individual pixel is related only to its four direct neighbors, two in each direction. This relationship forms a sparse matrix and a linear equation system of the form $A \cdot x = b$. Such a system can be solved by an iterative method, a suitable approach is the Gauss-Seidel method. Using successive over-relaxation the convergence rate can be increased [You54]. From the nested fixed point iteration scheme follows a nested scheme for solving the linear equation system; that is at each iteration step k several inner steps l are performed. Using the over-relaxation parameter ω and the iteration variable s for the SOR solver, the resulting scheme reads

$$\begin{aligned}
&d\rho_{i,j}^{k,l+1,s+1} \\
&= (1-\omega) \cdot d\rho_{i,j}^{k,l+1,s} + \omega \cdot \left(\frac{-1}{n} \sum_{\nu=1}^{n} (\Psi'_\nu)_{D\,i,j}^{k,l} \cdot (\mathbf{T}_{\nu\,12})_{i,j} \right. \\
&+ w_1 \cdot \left(\rho_{i+1,j}^{k,l} - \rho_{i,j}^{k,l} + d\rho_{i+1,j}^{k,l+1,s} \right) \\
&+ w_3 \cdot \left(\rho_{i,j+1}^{k,l} - \rho_{i,j}^{k,l} + d\rho_{i,j+1}^{k,l+1,s} \right) \\
&- w_2 \cdot \left(\rho_{i,j}^{k,l} - \rho_{i-1,j}^{k,l} - d\rho_{i-1,j}^{k,l+1,s+1} \right) \\
&\left. - w_4 \cdot \left(\rho_{i,j}^{k,l} - \rho_{i,j-1}^{k,l} - d\rho_{i,j-1}^{k,l+1,s+1} \right) \right) \\
&\cdot \left(w_1 + w_2 + w_3 + w_4 + \frac{1}{n} \sum_{\nu=1}^{n} (\Psi'_\nu)_{D\,i,j}^{k,l} \cdot (\mathbf{T}_{\nu\,11})_{i,j} \right)^{-1}. \qquad (3.37)
\end{aligned}$$

Values of $d\rho_{i,j}^{k,l+1}$ are used as soon as they are available, indicated by a mixture of the iteration steps s and $s+1$.

INITIALIZATION A last prerequisite to solve the energy functional concerns the initialization of $\rho_{i,j}$. The trivial solution 0 comes to mind and with the coarse-to-fine warping the iterative solution converges towards the correct solution. Setting the depth to zero, however, results in an initial point cloud such that every point lies at the origin of the reference camera and might cause difficulties relating the 3-D points to their corresponding locations in the match views. Further, using a constant initial value introduces a systematic error.

Instead, the plane sweeping approach by Maurer is extended to multiple views. Given the range of possible depth values $[\rho_{min}, \rho_{max}]$ the data term is used to compute a normalized sum of squared differences for a number of depth values, evenly spaced across the depth range.

$$\arg\min_{\rho_z} \frac{1}{N} \sum_{i=0}^{W} \sum_{j=0}^{H} \sum_{\nu=1}^{n} (I_{0\,i,j} - \mathcal{I}_{\nu\,i,j}(\rho_z))^2, \ \rho_z \in [\rho_{min}, \rho_{max}]. \qquad (3.38)$$

$\mathcal{I}_\nu$ represents the ν-th image warped towards the reference view and N is the number of points that are visible in all images.

3.4 SECOND-ORDER ISOTROPIC REGULARIZATION

The previous section described the process of developing a model to solve the stereo problem using a first-order isotropic regularization; however, such a first-order regularizer favors a constant depth. In Section 3.2.2 an analysis was performed which shows that the inverse depth parameterization is better suited for reconstruction because it back-projects affine depths to planes without introducing an error. To make use of this property of the inverse depth parameterization and to allow slanted surfaces, a second-order regularization term is introduced. Instead of the assumption that neighboring image points have the same depth value it is now assumed that the rate of change of the depth is constant between neighboring image points. This constraint is expressed by the second derivatives of ρ. The new smoothness term uses the Frobenius norm of the Hessian and reads

$$S(\rho) = \int_\Omega \Psi_S(\|\mathcal{H}\rho\|_F^2)\,dx\,dy, \tag{3.39}$$

where $\mathcal{H}\rho$ is the Hessian of ρ and $\|\mathcal{H}\rho\|_F = \sqrt{\rho_{xx}^2 + \rho_{xy}^2 + \rho_{yx}^2 + \rho_{yy}^2}$ the Frobenius norm of the Hessian. This results in the energy functional

$$E(\rho) = \int_\Omega \frac{1}{n}\sum_{\nu=1}^{n} \Psi(|I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)|^2) + \alpha\cdot\Psi_S(\|\mathcal{H}\rho\|_F^2)\,dx\,dy. \tag{3.40}$$

3.4.1 *Minimization*

The energy functional has the form

$$E(\rho) = \int_\Omega F(x, y, \rho, \rho_{xx}, \rho_{xy}, \rho_{yx}, \rho_{yy})\,dx\,dy$$

and the Euler-Lagrange equation provides the necessary condition for the minimizer ρ. The data term has not changed. Using the abbreviation $\Delta_\nu = I_0(\mathbf{p}) - I_\nu(\mathbf{p}_\nu)$ F_ρ is the same as in Equation (3.20):

$$F_\rho = \frac{1}{n}\sum_{\nu=1}^{n}(\Psi'(|\Delta_\nu|^2)\cdot\Delta_\nu\cdot\partial_\rho\Delta_\nu)$$

For the remaining parts of the Euler-Lagrange equation follows

$$\begin{aligned}
&F_{g_x} = 0, \quad F_{g_y} = 0,\\
&F_{g_{xx}} = \alpha\cdot\Psi_S'(\|\mathcal{H}\rho\|_F^2)\cdot 2\cdot\rho_{xx}, \quad F_{g_{xy}} = \alpha\cdot\Psi_S'(\|\mathcal{H}\rho\|_F^2)\cdot 2\cdot\rho_{xy},\\
&F_{g_{yx}} = \alpha\cdot\Psi_S'(\|\mathcal{H}\rho\|_F^2)\cdot 2\cdot\rho_{yx}, \quad F_{g_{yy}} = \alpha\cdot\Psi_S'(\|\mathcal{H}\rho\|_F^2)\cdot 2\cdot\rho_{yy}.
\end{aligned}$$

Using the symmetry of second derivatives, it holds that $F_{g_{xy}} = F_{g_{yx}}$ because $\rho_{xy} = \rho_{yx}$. Using the operator $\nabla_{\mathcal{H}} \coloneqq [\partial_{xx}\ \partial_{xy}\ \partial_{yx}\ \partial_{yy}]^T$ [Mau14] the Euler-Lagrange equation reads

$$0 = \frac{1}{n} \sum_{v=1}^{n} (\Psi'(|\Delta_v|^2) \cdot \Delta_v \cdot \partial_\rho \Delta_v) + \alpha \cdot \nabla_{\mathcal{H}}^T \left(\Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \nabla_{\mathcal{H}} \rho\right), \quad (3.41)$$

with the boundary conditions

$$0 = \mathbf{n}^T \cdot \begin{bmatrix} -\partial_x \left(\Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \rho_{xx}\right) - \partial_y \left(\Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \rho_{xy}\right) \\ -\partial_x \left(\Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \rho_{yx}\right) - \partial_y \left(\Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \rho_{yy}\right) \end{bmatrix},$$

$$0 = \mathbf{n}^T \cdot \begin{bmatrix} \Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \rho_{xx} \\ \Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \rho_{xy} \end{bmatrix}, \quad 0 = \mathbf{n}^T \cdot \begin{bmatrix} \Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \rho_{yx} \\ \Psi_S'(\|\mathcal{H}\rho\|_F^2) \cdot \rho_{yy} \end{bmatrix}.$$

With the same reasoning as for the first-order regularizer a nested fixed-point iteration scheme is introduced. By applying the same linearization this results in

$$\begin{aligned} 0 = \frac{1}{n} \sum_{v=1}^{n} & (\Psi_v')_D^{k,l} \cdot \left(\Delta_v^k + d\rho^{k,l+1} \cdot \partial_\rho \Delta_v^k\right) \cdot \partial_\rho \Delta_v^k \\ & + \alpha \cdot \nabla_{\mathcal{H}}^T \left((\Psi')_S^{k,l} \cdot \nabla_{\mathcal{H}} \left(\rho^k + d\rho^{k,l+1}\right)\right), \end{aligned} \quad (3.42)$$

with the abbreviations

$$\begin{aligned} (\Psi_v')_D^{k,l} &\coloneqq \Psi'(|\Delta_v^k + d\rho^k \cdot \partial_\rho \Delta_v^k|^2), \text{ and} \\ (\Psi')_S^{k,l} &\coloneqq \Psi'(\|\nabla_{\mathcal{H}}(\rho^k + d\rho^k)\|_F^2). \end{aligned} \quad (3.43)$$

3.4.2 *Discretization*

As already noted, the data term is identical to the data term that was used together with the first-order regularizer; hence, the discretization of the part related to the data term is the same as in Section 3.3.2. Of interest, however, is the discretization of the smoothness term. The second derivatives in the argument to the term $(\Psi')_{S\,i,j}^{k,l}$ are approximated by a fourth-order central finite difference scheme; the term

$$\nabla_{\mathcal{H}}^T \left((\Psi')_S^{k,l} \cdot \nabla_{\mathcal{H}} \left(\rho^k + d\rho^{k,l+1}\right)\right)$$

is approximated using a nested central finite difference scheme. Its derivation is found in Appendix B.3.

3.4.3 *Adapted Successive Over-Relaxation*

The discretized Euler-Lagrange equation forms a system of linear equations in which the $d\rho_{i,j}^{k,l+1}$ are the unknowns. Due to the increased neighborhood to approximate the part related to the smooth-

ness term the resulting matrix is not as sparsely populated as before, but still diagonally dominant; the successive over-relaxation remains a suitable method to find the minimizer. Using the abbreviation $P^{k,l}_{i,j} \coloneqq \rho^{k,l}_{i,j} + d\rho^{k,l+1}_{i,j}$ the SOR scheme reads

$$
\begin{aligned}
& d\rho^{k,l+1,s+1}_{i,j} \\
&= (1-\omega)\cdot d\rho^{k,l+1,s}_{i,j} + \omega \cdot \Bigg(\frac{-1}{n} \sum_{\nu=1}^{n} (\Psi'_\nu)^{k,l}_{D\,i,j} \cdot (\mathbf{T}_{\nu\,12})_{i,j} \\
&- w_0 \cdot \left(P^{k,l,s}_{i+2,j} - 2\cdot P^{k,l,s}_{i+1,j} + \rho^{k,l}_{i,j} \right) \\
&+ w_1 \cdot \left(P^{k,l,s}_{i+1,j} - 2\cdot \rho^{k,l}_{i,j} + P^{k,l,s+1}_{i-1,j} \right) \\
&- w_2 \cdot \left(\rho^{k,l}_{i,j} - 2\cdot P^{k,l,s+1}_{i-1,j} - P^{k,l,s+1}_{i-2,j} \right) \\
&- w_3 \cdot \left(P^{k,l,s}_{i,j+2} - 2\cdot P^{k,l,s}_{i,j+1} + \rho^{k,l}_{i,j} \right) \\
&+ w_4 \cdot \left(P^{k,l,s}_{i,j+1} - 2\cdot \rho^{k,l}_{i,j} + P^{k,l,s+1}_{i,j-1} \right) \\
&- w_5 \cdot \left(\rho^{k,l}_{i,j} - 2\cdot P^{k,l,s+1}_{i,j-1} - P^{k,l,s+1}_{i,j-2} \right) \\
&- w_6 \cdot \left(P^{k,l,s}_{i+2,j+2} - P^{k,l,s}_{i,j+2} - P^{k,l,s}_{i+2,j} + \rho^{k,l}_{i,j} \right) \\
&+ w_7 \cdot \left(P^{k,l,s}_{i,j+2} - P^{k,l,s}_{i-2,j+2} - \rho^{k,l}_{i,j} + P^{k,l,s+1}_{i-2,j} \right) \\
&+ w_8 \cdot \left(P^{k,l,s}_{i+2,j} - \rho^{k,l}_{i,j} - P^{k,l,s}_{i+2,j-2} + P^{k,l,s+1}_{i,j-2} \right) \\
&- w_9 \cdot \left(\rho^{k,l}_{i,j} - P^{k,l,s+1}_{i-2,j} - P^{k,l,s+1}_{i,j-2} + P^{k,l,s+1}_{i-2,j-2} \right) \Bigg) \\
&\cdot \left(\sum_{r=0}^{9} w_r + \frac{1}{n} \sum_{\nu=1}^{n} (\Psi'_\nu)^{k,l}_{D\,i,j} \cdot (\mathbf{T}_{\nu\,11})_{i,j} \right)^{-1},
\end{aligned}
\tag{3.44}
$$

where the weights w_r are defined as follows:

$$
\begin{aligned}
w_0 &\coloneqq \frac{\alpha \cdot (\Psi')^{k,l}_{S\,i+1,j}}{h_x^4}, & w_1 &\coloneqq \frac{-\alpha \cdot (\Psi')^{k,l}_{S\,i,j}}{h_x^4}, \\
w_2 &\coloneqq \frac{\alpha \cdot (\Psi')^{k,l}_{S\,i-1,j}}{h_x^4}, & w_3 &\coloneqq \frac{\alpha \cdot (\Psi')^{k,l}_{S\,i,j+1}}{h_y^4}, \\
w_4 &\coloneqq \frac{-\alpha \cdot (\Psi')^{k,l}_{S\,i,j}}{h_y^4}, & w_5 &\coloneqq \frac{\alpha \cdot (\Psi')^{k,l}_{S\,i,j-1}}{h_y^4}, \\
w_6 &\coloneqq \frac{\alpha \cdot (\Psi')^{k,l}_{S\,i+1,j+1}}{8\cdot h_x^2 \cdot h_y^2}, & w_7 &\coloneqq \frac{-\alpha \cdot (\Psi')^{k,l}_{S\,i-1,j+1}}{8\cdot h_x^2 \cdot h_y^2}, \\
w_8 &\coloneqq \frac{-\alpha \cdot (\Psi')^{k,l}_{S\,i+1,j-1}}{8\cdot h_x^2 \cdot h_y^2}, & w_9 &\coloneqq \frac{\alpha \cdot (\Psi')^{k,l}_{S\,i-1,j-1}}{8\cdot h_x^2 \cdot h_y^2}.
\end{aligned}
$$

The iteration variable for the SOR scheme is s and $s+1$ indicates that values which are already computed are used.

4 EXTENSIONS

This chapter introduces extensions that are useful in a multi-view configuration, improving the fidelity of or the confidence in the solution. Furthermore, with automatic view selection the runtime can be reduced by discarding views that do not contribute to the solution.

4.1 OCCLUSION HANDLING

When a scene with foreground and background objects is observed from different view points it so happens that regions visible in one view are occluded in the other view. The foreground and the background differ in their perceived relative motion and at the depth discontinuity occlusions become apparent. For pixels in the occluded region it is not only difficult but impossible to find the corresponding pixel in the match view. At least this is the case in a two-view stereo case, in a multi-view case another view might provide a correspondence. Regardless of the actual case, trying to use occluded pixels might have detrimental effects because the data term either reports a high matching cost for a potentially correct depth or the opposite, a low matching cost for an incorrect depth. Either way, inferring truth from incomplete data should be avoided. In addition, although the multi-view case could provide information from one match view if another fails to do so, the overall occluded regions increase proportional with the number of views and it becomes more important to handle occlusions [BMK12]. To disable the data term in occluded regions, for each image an occlusion map $g_{\nu\,i,j}$ is computed that assigns each image point a binary visibility score, such that

$$g_{\nu\,i,j} = \begin{cases} 1 & \text{if } \mathbf{P}_{i,j} \text{ is visible in } I_\nu \\ 0 & \text{else,} \end{cases} \tag{4.1}$$

where $\mathbf{P}_{i,j}$ is the 3-D point that is projected onto (i, j). Equation (3.27) is modified to use the visible score by

$$\frac{1}{n} \sum_{\nu=1}^{n} g_{\nu\,i,j} \cdot (\Psi'_\nu)^{k,l}_{D\,i,j} \cdot \left((\mathbf{T}_{\nu\,12})_{i,j} + d\rho^{k,l+1}_{i,j} \cdot (\mathbf{T}_{\nu\,11})_{i,j} \right). \tag{4.2}$$

In essence, an occluded pixel does not contribute to the cost and if the pixel is occluded across all views the method solely relies on the smoothness term's fill-in effect. A Z-Buffer approach is used to compute the visibility score. Figure 4.1 illustrates the case of an occluded pixel [BMK12; Val+10]. Using the depth at the current warping level, every point $\mathbf{P}_{i,j}$ is projected onto the image plain of match

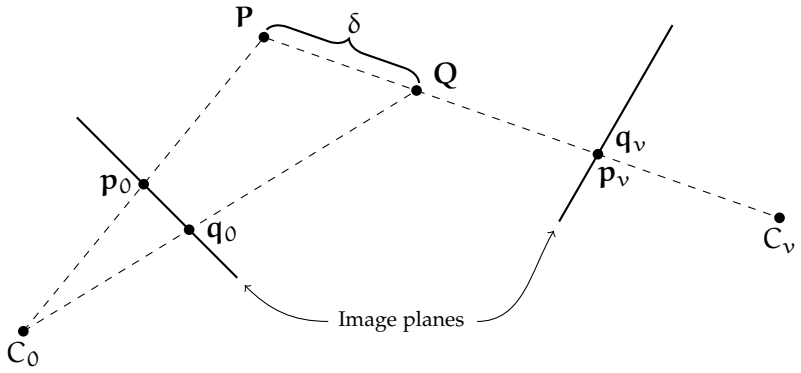


Figure 4.1: The image points $\mathbf{p}_0$ and $\mathbf{q}_0$ are mapped to the same image point $\mathbf{p}_\nu = \mathbf{q}_\nu$. The scene point $\mathbf{Q}$ occludes $\mathbf{P}$ if the distance δ is larger than some threshold. If that is the case, $\mathbf{p}_0$ is occluded and its occlusion score is set to 0.

view ν; if multiple $\mathbf{P}_{i,j}$ are projected onto the same corresponding pixel $\mathbf{p}_\nu$, an occlusion is detected, provided the distances between the back-projected points exceeds a predefined threshold δ. In Figure 4.1 there are two points $\mathbf{P}$ and $\mathbf{Q}$ that are projected onto the same point $\mathbf{p}_\nu = \mathbf{q}_\nu$. As seen from C_ν, $\mathbf{P}$ is occluded by $\mathbf{Q}$ and the visibility score of $\mathbf{p}_0$ is 0 if $\|\mathbf{P} - \mathbf{Q}\|_2 > \delta$. As a result, view ν has no influence on the data term at $\mathbf{p}_0$.

4.2 FORWARD-BACKWARD CONSISTENCY CHECKING

Although similar to occlusion detection, the forward-backward consistency check is a post-processing step to find and disable inconsistent depth values across multiple views. The idea is based on the left/right check and the symmetric distance error measure used for estimating the fundamental matrix [Hir05; FL04]. The depth values from two different views are said to be consistent if they back-project onto the same 3-D coordinate. This requires the computed depth map for all views involved and the forward-backward check is therefore only possible after a sufficient number — at least two — of depth maps have been computed for a scene.

As input, the depth map of the reference view and the depth maps of the match views are given. The consistency map is an integer score, instead of a binary one as used in occlusion detection. For every pixel in the reference view each match view ν casts a vote if the depth in both views is consistent. For the consistent depth map $\check{\rho}$ follows

$$\check{\rho}(\mathbf{p}) = \begin{cases} \rho(\mathbf{p}) & \text{if } \sum_{\nu=1}^{n} c_\nu(\mathbf{p}) > \omega \\ 0 & \text{else,} \end{cases} \tag{4.3}$$

where c_ν is the consistency map of view ν and ω is the number of necessary votes. The consistency map is defined by

$$c_\nu(\mathbf{p}) = \begin{cases} 1 & \text{if } |\mathbf{p} - \mathbf{p}'| < \delta \\ 0 & \text{else,} \end{cases} \tag{4.4}$$

where

$$\mathbf{p}' = \pi(M_0 \cdot \Pi(\Phi(\rho_\nu(\mathbf{p}_\nu)) \cdot K_\nu^{-1} \cdot \Pi(\mathbf{p}_\nu)) \text{ and}$$
$$\mathbf{p}_\nu = \pi(M_\nu \cdot \Pi(\Phi(\rho_0(\mathbf{p})) \cdot K_0^{-1} \cdot \Pi(\mathbf{p})).$$

Every point $\mathbf{p}$ is projected onto the match view ν using the depth map of the reference view to obtain the corresponding point $\mathbf{p}_\nu$. This point in turn is then projected onto the reference view and $\mathbf{p}'$ is obtained. The depth at $\mathbf{p}$ is deemed consistent if the distance between $\mathbf{p}$ and its forward-backward projection does not exceed a certain threshold δ.

4.3 AUTOMATIC VIEW SELECTION

In a multi-view stereo setting there exists a paradox: Having a larger amount of images from different angles and distances available allows to cover more elements of the scene by providing vantage points that complement each other. But not all images are equally useful considering a specific reference view. Two camera pairs could be oriented so that their configuration is diverging; in the extreme case both cameras could point in opposite directions, for example when cameras were circling an object. Thus, selecting appropriate views is mandatory for good results. Using certain criteria, this process can be automated and two different approaches are described in this section.

4.3.1 *Geometry-Based Approach*

The first approach uses the poses of the match views relative to the reference view. It is assumed that images were taken such that the objects of interest are visible in the image. The extrinsic matrix contains two distinct properties, the orientation of a match camera and the translation, that allows to compute the origin of the match camera in the coordinate frame of the reference view. The relative motion of foreground objects is larger than for background objects; with increasing distance between the camera centers, that is increasing baseline b, the motion of the foreground object might become so large that reliable matching is not possible anymore. It is therefore necessary to restrict the baseline to not exceed a certain threshold. This forms the first condition for selecting valid match views. One could impose a constraint on the minimal baseline, thus excluding views that are to close to the reference view. However, this introduces another parameter and it is assumed that the typical use case as described in

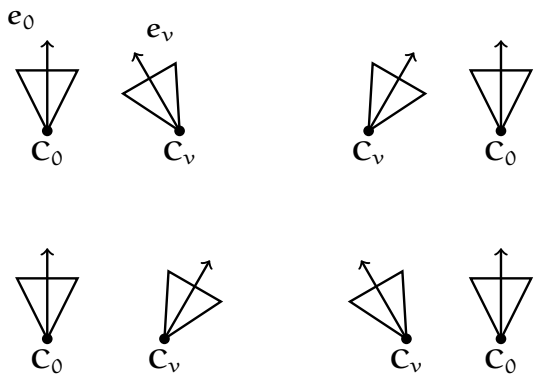


Figure 4.2: A top-down view of the four different cases regarding the orientation between match camera C_ν and reference camera C_0. The top row shows valid converging setups, the bottom row shows the two invalid setups.

Section 3.2.1 results in baselines that are too large rather than too small.

The second condition is based on the camera orientations. In the ideal case, all cameras involved point to the same object and form a converging setup. The viewing direction is along the Z-axis and the vector $\boldsymbol{e}_\nu = [0\ 0\ 1\ 0]^T$, called the eye vector, of camera ν is expressed in its coordinate frame. Note that the 4th component of $\boldsymbol{e}_\nu$ is zero, which prevents the addition of the translation vector. The eye vector $\boldsymbol{e}_\nu$ is transformed to the coordinate frame of the reference camera by

$$\check{\boldsymbol{e}}_\nu = T_0 \cdot T_\nu^{-1} \cdot \boldsymbol{e}_\nu$$

and the angle ϕ between $\check{\boldsymbol{e}}_\nu$ and $\boldsymbol{e}_0$ is computed using the relation

$$\cos\phi = \frac{\boldsymbol{e}_0^T \cdot \check{\boldsymbol{e}}_\nu}{\|\boldsymbol{e}_0\|_2 \cdot \|\check{\boldsymbol{e}}_\nu\|_2}.$$

Positive angles would indicate a converging, negative angles a diverging setup. However, the dot product computes the smallest positive angle between two vectors. Furthermore, since both eye vectors span a plane all information regarding the relative position of the cameras, that is left or right, above or below, is lost. Consider Figure 4.2 which depicts the four possible configurations considering the 2-D case. A converging setup exists, if the camera lies on one side of the reference camera but points towards the other side. It is therefore necessary to first determine the relative position of the match camera. To achieve this, the origin of the match camera is transformed to the coordinate frame of the reference camera by

$$\check{C}_\nu = T_0 \cdot T_\nu^{-1} \cdot [0\ 0\ 0\ 1]^T.$$

The match camera lies to the left of the reference camera, if the X-component is negative, otherwise it lies to the right; if the Y-component is positive, the match camera lies above, otherwise below

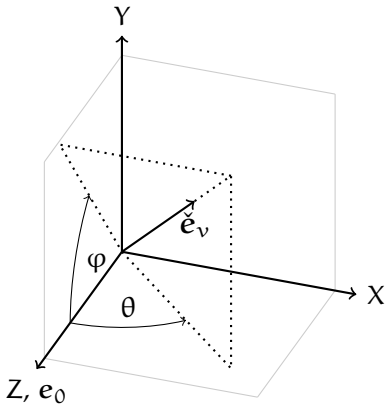


Figure 4.3: The eye vector $\check{e}_\nu$ of camera ν is transformed to the coordinate frame of camera C_0. The angles φ and θ are used to determine the view direction of camera C_ν relative to the reference camera.

the reference camera. After the relative position is established, two angles are computed by

$$\varphi = \arctan\left(\frac{y}{z}\right), \quad \theta = \arctan\left(\frac{x}{z}\right).$$

φ and θ are the angles between the Z-axis and the projections of the eye vector $\check{e}_\nu$ onto the YZ- and XZ-plane respectively, as is shown in Figure 4.3. Both angles represent the deviation of $\check{e}_\nu$ from their respective plane, determining how much the match camera points upwards and sideward. The condition for a valid camera becomes

$$\begin{aligned}\texttt{valid} = b < \beta \wedge ((\texttt{left} \wedge \theta \in [0, \Theta]) \quad &\vee \quad (\neg\texttt{left} \wedge \theta \in [-\Theta, 0])) \\ \wedge ((\texttt{above} \wedge \varphi \in [0, \Theta]) \quad &\vee \quad (\neg\texttt{above} \wedge \varphi \in [-\Theta, 0])),\end{aligned}$$

where β is the threshold for the baseline $b = \|\mathbf{C}_\nu - \mathbf{C}_0\|_2$, Θ is the threshold for both angles, `left` is true if $\mathbf{C}_\nu - \mathbf{C}_0$ has a negative X-component and `above` is true if $\mathbf{C}_\nu - \mathbf{C}_0$ has a negative Y-component.

4.3.2 *Feature-Based Approach*

The previous approach to view selection made the assumption that cameras oriented in a converging manner observe similar parts of the scene. Another possibility is to examine the contents of the pictures and determine whether the match cameras see similar parts of the scene. A way to achieve this is to extract distinctive features like edges and corners from the picture of the reference view and find those again in the picture from the match view. If enough features can be matched, the images are deemed similar.

Given a feature of the scene, there are three major transformations in image space the feature can undergo when switching between cameras: Due to lateral movement of the camera, the feature's position is

translated; moving the camera closer or farther away from the scene or changing the focal length results in scaling the feature; the feature is rotated, if the camera was rotated around its optical axis. To be of use, the feature extractor should be invariant under those transformations. Hence, the Scale-Invariant Feature Transform (SIFT) was chosen [Low04]. Lowe also provides a descriptor for each SIFT feature, that is used for matching.

For the reference image I_0 and a match image I_ν, the first step is to extract distinctive features at locations $\mathbf{p}_0$ and $\mathbf{p}_\nu$, with descriptors $\mathbf{d}_0$ and $\mathbf{d}_\nu$. This results in two sets $\mathcal{F}_0$ and $\mathcal{F}_\nu$. The second step is to find for each point $\mathbf{p}_0$ with descriptor $\mathbf{d}_0$ a matching point $\mathbf{p}_\nu$ with $\mathbf{d}_\nu$ such that the distance between those descriptors is minimal. However, this does not guarantee, that the match is useful; a minimal match will be found for every feature. Hence, some criteria are needed to determine whether the found matches are good. Lowe made the observation that for correct matches the distance of the second best match is considerably larger than for the best match [Low04]. This results in a first test, where a match is valid if

$$\frac{\|\mathbf{d}_0 - \mathbf{d}_\nu\|_2}{\|\mathbf{d}_0 - \hat{\mathbf{d}}_\nu\|_2} \leqslant \tau, \tag{4.5}$$

where $\hat{\mathbf{d}}_\nu$ is the second-closest descriptor and $\tau \in [0, 1]$ the threshold for inclusion. It is possible to impose another constraint on the matches. Having the extrinsic and intrinsic matrices available, the fundamental matrix for each image pair can be computed, see the derivation in Appendix A.2. From the epipolar constraint follows that

$$\mathbf{p}_\nu^T \cdot F_\nu \cdot \mathbf{p}_0 = 0,$$

where F_ν is the fundamental matrix for the cameras C_0 and C_ν. This leads to a criterion every feature has to satisfy,

$$|\mathbf{p}_\nu^T \cdot F_\nu \cdot \mathbf{p}_0| \leqslant \delta, \tag{4.6}$$

where δ is a positive, small threshold. The two criteria from Equation (4.5) and Equation (4.6) are combined and the process of selecting a match view ν is as follows:

1. Extract features $f_{i,0} \coloneqq (\mathbf{p}_0, \mathbf{d}_0)$ and $f_{i,\nu} \coloneqq (\mathbf{p}_\nu, \mathbf{d}_\nu)$ resulting in sets $\mathcal{F}_0$ and $\mathcal{F}_\nu$
2. Find the best and second best matches and apply criterion 4.5, resulting in a set $\mathcal{M}_r$ that contains all correct matches
3. For each match $m_i \in \mathcal{M}_r$ apply criterion 4.6, resulting in a set $\mathcal{M}_{EC}$ that contains all valid matches
4. Discard view ν if $\frac{|\mathcal{M}_{EC}|}{|\mathcal{M}_r|} < \kappa$

In the above algorithm $\kappa \in [0, 1]$ allows to control what percentage of matches from $\mathcal{M}_r$ has to satisfy the epipolar constraint. By setting $\tau = 1$ it is also possible to only use the epipolar constraint.

5 EVALUATION

In this chapter the proposed method to solve the stereo problem as well as the introduced extensions are evaluated experimentally. As test data, artificial scenes were used to control the properties of lighting, material and cameras. Additional real world test scenes were used to evaluate the methods under natural conditions.

5.1 EVALUATION METHOD

The stereo reconstruction is performed on numerous test scenes exhibiting different characteristics regarding the camera configuration. The ground-truth depth map for the reference view is available for each scene. Using it, it is possible to compare the computed depth to the expected depth and determine the absolute error. However, projection does not preserve relative distances and it is therefore better to compute the error of the back-projected instead of the projected points [BMK12; Mau14]. For a single point this is formulated by

$$e = \|\mathbf{P}_{gt} - \mathbf{P}\|_2,$$

where $\mathbf{P} \in \mathcal{P}$ is the back-projection of $\mathbf{p}$ using the computed depth and $\mathbf{P}_{gt}$ is the back-projection using the ground-truth depth value. Using the above formula the absolute error for every image point can be computed. The resulting error values represent the distance between the correct and the computed point. To determine whether the computed point is closer or farther away than the correct point, the error is multiplied with a sign s, such that

$$s = \begin{cases} 1 & \text{if } Z_{gt} - Z > 0 \\ -1 & \text{else.} \end{cases}$$

Sign and error combined allow a color coding as in Figure 5.1 in which green values represent a close match, red values represent

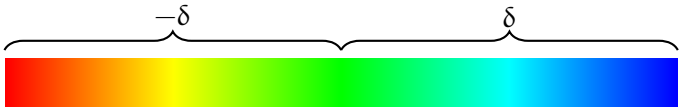


Figure 5.1: Color coding of the 3-D error. A back-projected point that is closer to the camera compared to the correct location is colored towards blue; a point farther away is colored towards red. δ is the threshold above which error values are truncated.

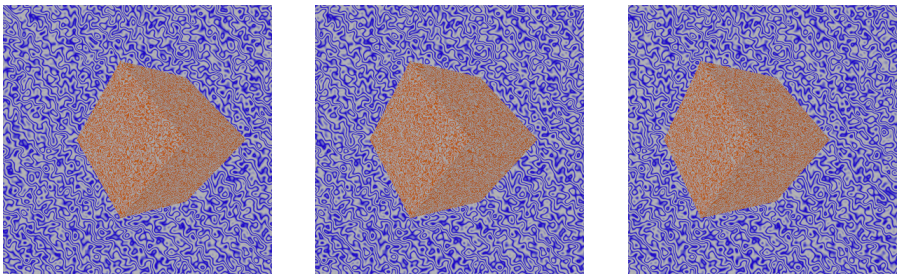


Figure 5.2: Three views from the rotated cube scene with an ortho-parallel camera configuration. The picture in the middle shows the reference view, the pictures left and right to the respective match views.

points that are farther away from the camera than the correct points and blue values represent points that are closer to the camera than the correct points.

To compare the overall result of different methods, the *root mean square* (RMS) error of the back-projected points is used. For $\mathbf{P}_{gt}$ and $\mathbf{P}$ as above, the error is computed as

$$e_{RMS} = \sqrt{\frac{1}{N} \cdot \sum_{i=1}^{N} \left(\mathbf{P}_{i,gt} - \mathbf{P}_i\right)^T \left(\mathbf{P}_{i,gt} - \mathbf{P}_i\right)},$$

where N denotes the cardinality of the point cloud, $|\mathcal{P}|$, and i the i-th element of the point cloud.

To control lighting and camera positions more precisely, artificial test scenes were created using the tool *Blender*[1]. All objects consist of a material with diffuse reflectance and were textured using colored, procedural noise. The latter gives each image point a distinguishable neighborhood to reflect the imperfections found in actual materials, which are not homogeneous.

Some of the adjustable parameters were found to work well across all tested scenes or to have only little influence, such that it made sense to keep these fixed. The parameters of the penalizer functions, ϵ_{data} and ϵ_{smooth}, were set to 0.001. The number of outer iterations was set to 1, the number of inner iterations was set to 3 and the number of SOR iterations was set to 10. The over-relaxation parameter ω was set to 1.8 and the parameter used for pre-smoothing the images, σ, was set to 0.5. Exceptions for any parameter are pointed out in the text.

5.2 MULTIPLE VIEWS

In a first experiment, the extension to multiple images is evaluated. Because the occlusion handling was described as a sensible enhancement of the basic multi-view case, the effects are studied here, too. Finally, the forward-backward consistency check bears resemblance

1 https://www.blender.org

Test Case	n	ω	RMS	Density [%]
single	1		0.906	100
multi	2		0.831	100
	4		0.843	100
occlusion	2		0.595	100
	4		0.652	100
consistency	2	1	0.543	99.3
	2	2	0.477	97.7
	2	3	0.424	84.0
	2	4	0.415	65.0

Table 5.1: Results for the rotated cube scene in ortho-parallel camera configuration with $\alpha = 8.8 \times 10^{-2}$ and first-order isotropic regularization, rounded to three significant digits.

to the occlusion handling and its results are discussed alongside the other two cases.

As a test scene, a rotated cube in front of a plane parallel to the reference camera's image plane was created. The reference view and the left and right match views are shown in Figure 5.2. Two additional views farther to the left and right respectively exist, resulting in a total of six match views in an ortho-parallel camera setup.

The direct depth parameterization is used with the first-order isotropic regularization. For the test case *single*, the view to the right of the reference view was used, which corresponds to the rightmost picture in Figure 5.2. In the *multi* and *occlusion* case, the two views closest to the reference view were used for the first experiment and the closest four views for the second one. The resulting errors are shown in Table 5.1, the computed depth values and error visualization are presented in Figure 5.3. The downscaling factor η was set to 0.96 for all test cases. The regularization parameter α was set to 8.8×10^{-2} because it was found to be the value that resulted in a minimal RMS error for *single*. It is possible, that setting α to different values, better results could be achieved for the different test cases. However, this step was omitted because changes in the RMS error would then depend on an additional parameter. Furthermore, the normalization of the data term by the number of match views is intended to keep the influence of the data term approximately the same for an increasing amount of match views.

The case with a single view establishes a baseline with an RMS error of 0.906. Figure 5.3b shows that the main contribution to the error is at the left side of the cube; the reason is that this part is

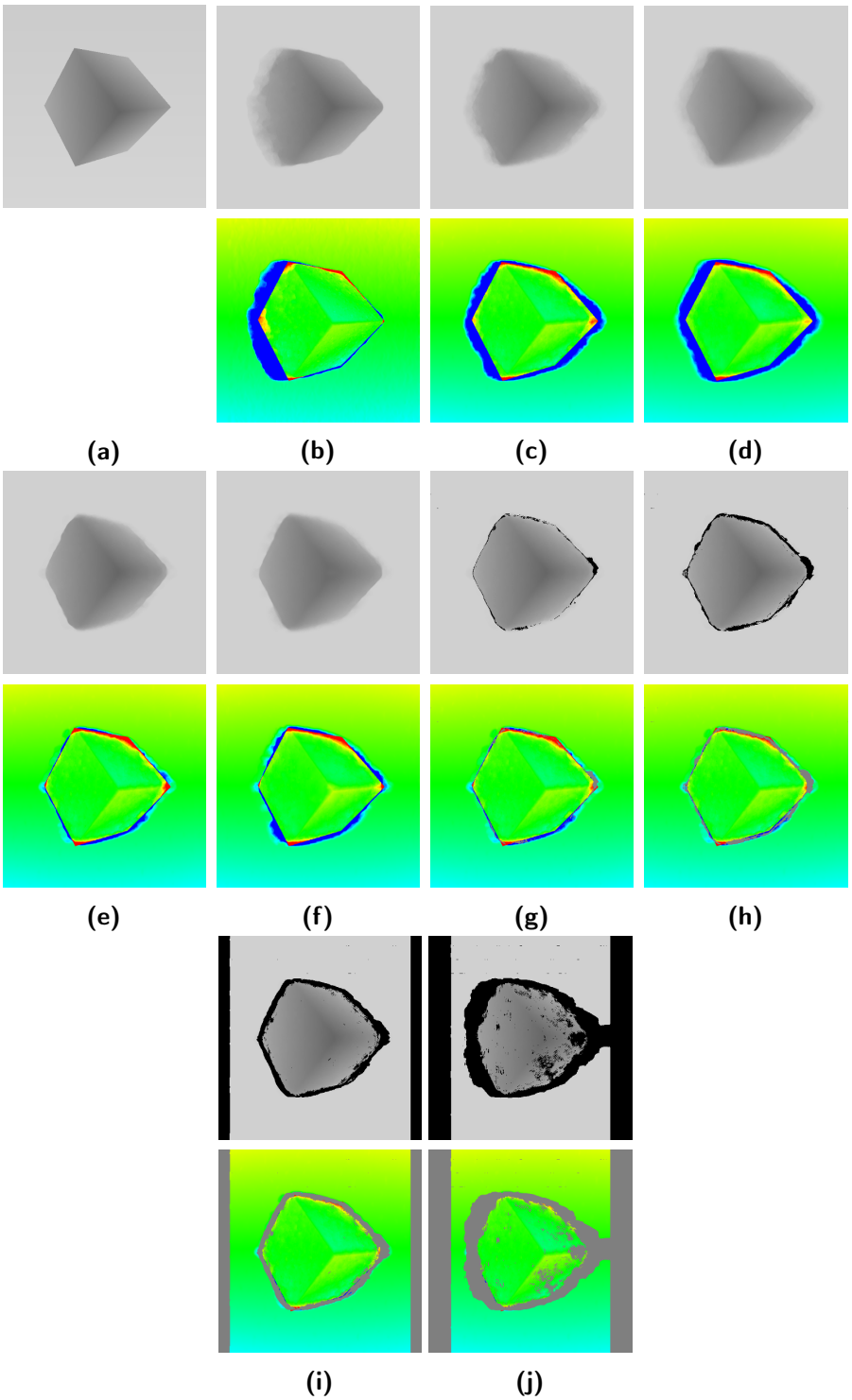


Figure 5.3: Depth maps and error visualization for the test cases from Table 5.1. (a) Ground-truth depth values; (b) *single*; (c) *multi*, $n = 2$; (d) *multi*, $n = 4$; (e) *occlusion*, $n = 2$; (f) *occlusion*, $n = 4$; (g) *consistency*, $\omega = 1$; (h) *consistency*, $\omega = 2$; (i) *consistency*, $\omega = 3$; (j) *consistency*, $\omega = 4$.

occluded in the match view. Accordingly, on the right side of the cube, where no occlusions occur, the depth discontinuities are preserved.

Basha et al. reasoned that occlusions are expected to occur more frequently with increasing number of match views [BMK12]. In the case at hand, this argument is easy to follow: Parts to the right of the cube are occluded in views to the left of the reference view and vice versa. It is expected that these occluded regions have a detrimental effect on the reconstruction quality. However, Table 5.1 shows that with one additional view the error decreases, by 8.3%. Using three additional views, however, the error is reduced only by 6.9%. In this case, the two new views have a large baseline, resulting in more occlusions which are more difficult to reconstruct correctly. Looking at the error visualizations in Figure 5.3c and 5.3d, the depth discontinuities at the right side of the cube are not preserved as well as in the single-view case, but the error on the left is reduced. If the increase in occluded regions is responsible for the less accurate depth discontinuities, occlusion handling should help mitigate the influence of occlusions. Using an occlusion threshold of 0.25, it follows from Table 5.1 that this is indeed true: In both cases with two and four match views respectively, the RMS error is reduced by 22.7 – 28.4%. Consistent with prior observations, the reconstruction quality is better using two views instead of four. The same reasoning as for the multi-view case applies. In Figures 5.3e and 5.3f can be seen that depth discontinuities are preserved better than in the previous cases. As a result, the cube is easier to distinguish from the background and the edges are sharper.

For the *consistency* case, five of the seven views from the scene were used as reference view, each with one match view to either side. As a result, five depth maps were computed, with activated occlusion handling and a threshold of 0.25. The forward-backward consistency check was executed with $\delta = \sqrt{2}$ and the number of votes ω as in Table 5.1. When computing the RMS error and the error visualization, inconsistent values were not considered. The reason behind this is that the consistency check does not try to correct false values but only to detect them. The remaining values are considered correct by at least ω views, thus increasing the confidence in the solution. Table 5.1 shows that the error continually decreases, but the amount of the reduction becomes smaller with each increase of the required number of votes. This indicates that the computed depth values which result in a high error are detected even with a small number of necessary votes. Judging from the error visualizations in Figures 5.3g to 5.3j this interpretation is confirmed: The bright red and blue regions representing depth values with large errors are among the first to be detected as inconsistent, marked by gray color in the visualization. However, a higher ω results in a reduction of the total number of reconstructed points, that is $|\mathcal{P}|$. A compromise between the density of

Figure 5.4: The images from the reference views of scenes used to evaluate the inverse depth parameterization. From left to right: *slanted*[2], *cloth1*, *venus*.

the reconstruction and the confidence in the solution has to be found. In the case at hand, a value of ω of 2 or 3 seems reasonable, as the most erroneous points are detected while 84.0 – 97.7% of the points still remain.

5.3 INVERSE DEPTH PARAMETERIZATION

To evaluate the benefit of the inverse depth parameterization, two things have to be considered: First, for camera configurations in which t_z is approximately zero, the error using the inverse depth parameterization is expected to be less than the error using the direct depth parameterization. Second, affine depth values are back-projected to an affine surface when using the inverse depth parameterization. A regularizer that favors affine depths is the second-order isotropic regularizer, which is used for all experiments in this section. To examine whether the inverse depth benefits from $t_z \approx 0$, the test scene from before is reused, but the camera positions of the match views were altered. The test case *random* consists of four match cameras placed in proximity to the reference camera and pointing in the general direction of the reference camera. For *spherical*, the cameras where distributed such that each camera satisfies the setup depicted in Figure 3.2. The camera centers thus lie on a sphere around the object origin and the view vectors point to the object origin. The case *ortho-parallel* is the same as in the previous section and is used here because for an ortho-parallel setup t_z is exactly zero.

Three additional scenes were used to compare the depth parameterizations. The scene *slanted* consists of a plane angled towards the reference camera and textured with the image of a stone wall. The match cameras were placed randomly. To examine the method under more natural lighting conditions, two scenes from the Middlebury stereo

2 *Wallstone* texture (http://gryllus.net/Blender/PDFTutorials/02BCastleTexturing_ITunesU/wallstone.jpg) by Neal Hirsig (http://gryllus.net) is licensed under CC-BY-NC-SA 3.0 (http://creativecommons.org/licenses/by-nc-sa/3.0/).

Scene	Φ	α	n	RMS
random	direct	5.66×10^{-2}	4	1.41
	inverse	4.15×10^{1}	4	1.30
spherical	direct	2.08×10^{-2}	4	1.09
	inverse	3.05×10^{1}	4	1.07
ortho-parallel	direct	2.75×10^{-2}	2	0.889
	inverse	9.30	2	0.994
slanted	direct	5.32×10^{-1}	4	0.481
	inverse	4.30×10^{1}	4	0.475
venus	direct	4.05×10^{-5}	8	0.631
	inverse	3.00×10^{-5}	8	0.586
cloth1	direct	8.50×10^{-2}	6	0.618
	inverse	4.37×10^{1}	6	0.201

Table 5.2: Results for the multi-view case with second-order isotropic regularization, rounded to three significant digits.

benchmark[3] [SS02; SP07; HS07] were used. The available scenes usually consist of multiple views, the cameras are always placed equidistantly and in an ortho-parallel fashion. Furthermore, dense ground truths images are provided in the form of disparity maps. These are easily converted to depth maps by

$$Z(\mathbf{p}) = \frac{s_x \cdot B}{D(\mathbf{p})},$$

where $Z(\mathbf{p})$ is the ground truth depth at $\mathbf{p}$, B the baseline of the cameras for which the disparity is provided, and $D(\mathbf{p})$ the ground truth disparity values. Two scenes from the data sets were used. The scene *cloth1* [SP07; HS07] consists of a patterned sheet of cloth draped in a way that creates slanted surfaces. Creases in the sheet pose additional difficulties compared to pure planar surfaces. The scene contains seven cameras in total and the camera matrices were generated as described in Appendix C.1. The *venus* scene [SS02] consists of different planes angled towards and away from the reference camera. A total of eight cameras are available for matching. No calibration data is available for the *venus* scene, therefore the cameras were calibrated as follows. The principal point is assumed to lie in the center of the image, o_x and o_y are then half the width and height of the image, respectively. A value of 0.01 between view 2 and view 6 is used as baseline. Then s_x was chosen such that the depth values lie between

3 http://vision.middlebury.edu/stereo/data

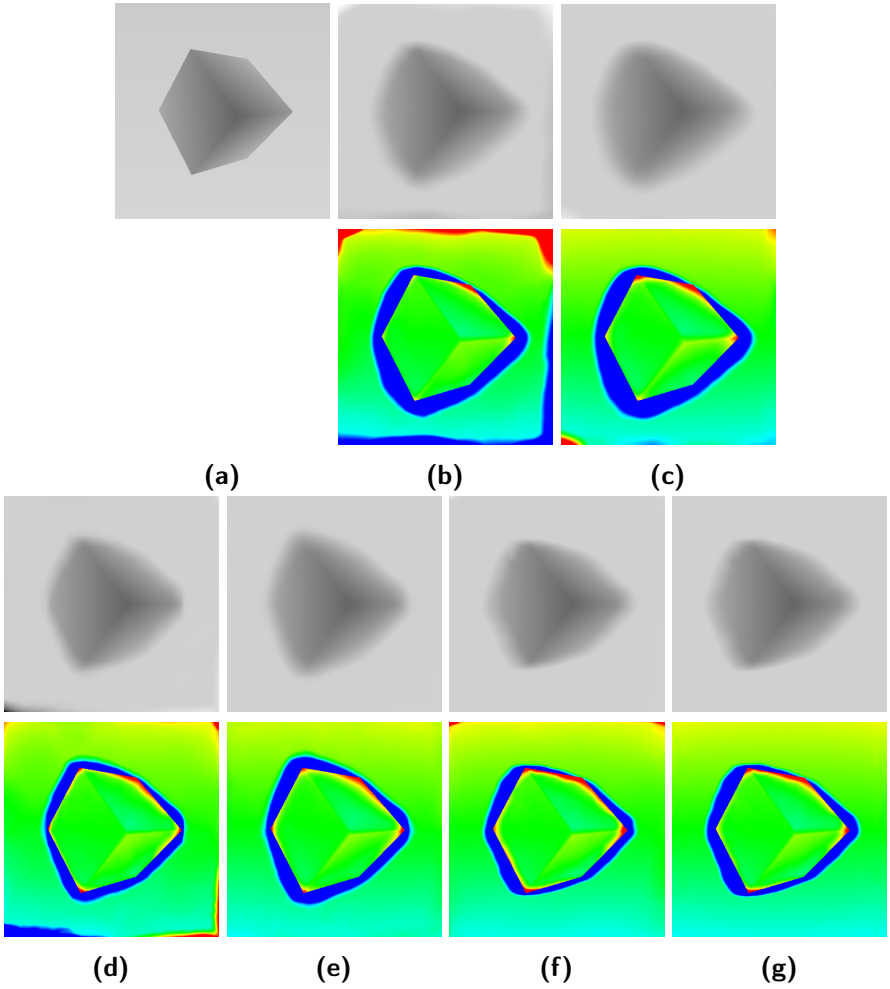


Figure 5.5: Depth values and error visualization for the cube scenes from Table 5.2. (a) Ground-truth depth for the *cube* scenes; (b) *random*, direct; (c) *random*, inverse; (d) *spherical*, direct; (e) *spherical*, inverse; (f) *ortho-parallel*, direct; (g) *ortho-parallel*, inverse.

0.4 and 3; s_y was set to a value such that the ratio between s_x and s_y matches the aspect ratio of the images. In Figure 5.4 the images for the reference views are depicted.

In all tests, occlusion handling was active, the threshold was set to 0.25 for the cube scenes and to 0.0001 for all other scenes. The downscaling factor was set to 0.96 for the cube scenes, to 0.93 for *slanted*, to 0.95 for *cloth1*, and to 0.92 for *venus*. For the scene *random* with direct depth parameterization an η of 0.94 was used. The pre-smoothing parameter for *cloth1* was changed to 0.7 and to 1.5 for *slanted*. The over-relaxation parameter ω was set to 1.9 for *slanted* and to 1.7 for *venus*. Although more views do not necessarily result in a better reconstruction, the number of match views used for each test case was chosen, such that the RMS error was minimal.

The results of the experiments are shown in Table 5.2. In both the *random* and *spherical* scene, the inverse depth parameterization per-

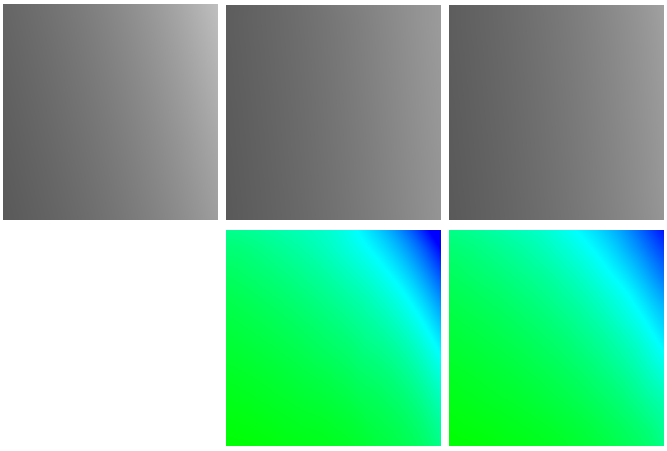
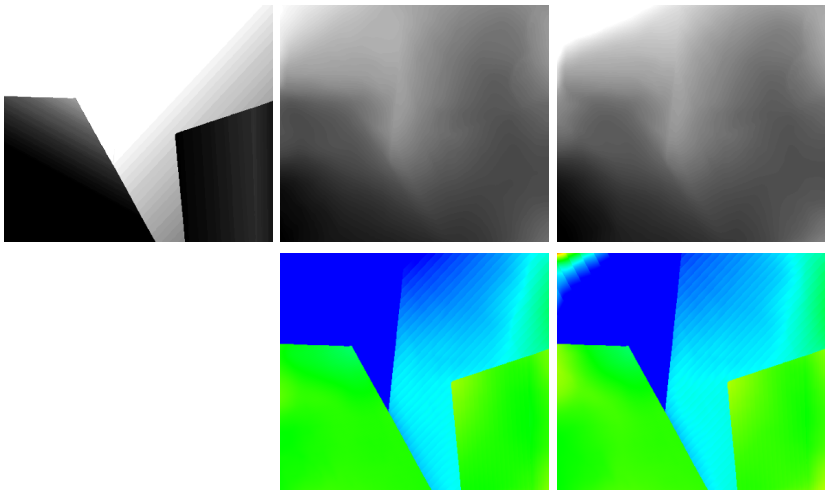


Figure 5.6: Depth values and error visualization for *slanted* scene from Table 5.2. (a) Ground-truth depth for the *slanted* scene; (b) direct depth; (c) inverse depth.

forms better in terms of the RMS error. However, the results seem to indicate that with the direct depth the decrease of the error is larger for scenes where t_z approaches zero. For *spherical* the error for the inverse depth is only slightly smaller than for the direct depth. Looking at Figures 5.5b to 5.5e, it becomes apparent that the direct depth has difficulties reconstructing the surface at the image boundaries, while for the inverse depth the edges of the cube are less sharp. However, in the *ortho-parallel* case using the direct depth parameterization results in a lower RMS error compared to the inverse depth. For all cameras involved in this case, t_z is exactly zero and one would also suspect the inverse depth to have a smaller error. Because of the square in the computation of the RMS error, larger individual error values have a higher influence on the resulting error, but the visualizations in Figures 5.5f and 5.5g show that the red areas with a high negative error are smaller for the inverse depth. On the other hand, the blue area with high positive error values is larger. In all those scenes the depth difference between foreground and background object is high and the larger erroneous region around the cube seems to indicate that the inverse depth performs worse in this case.

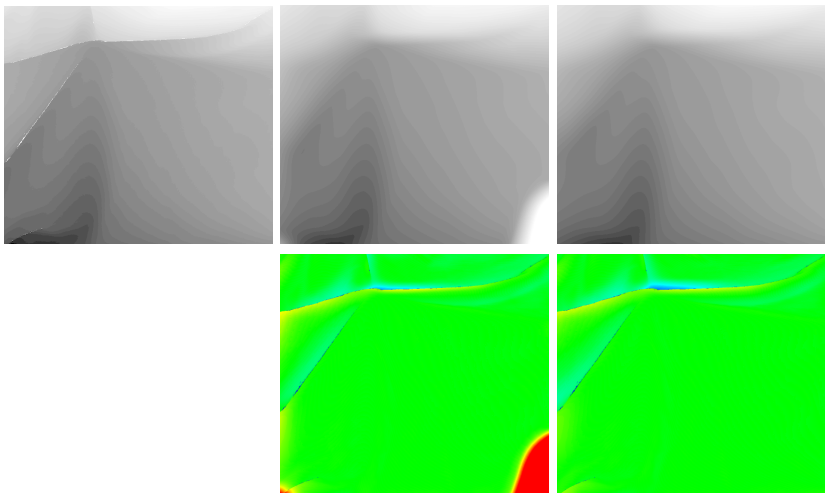
For the *slanted* scene, Table 5.2 shows that the error for both parameterizations is approximately the same; the images in Figure 5.6 confirm that the computed depth values for both parameterizations are similar. The blue area in the upper right corner of both error visualizations is due to the higher depth in this region, which results in a higher error reported. For a scene as simple as *slanted*, both parameterizations produce similar results, with the inverse depth performing slightly better.

The *venus* scene proved difficult for both parameterizations. While the RMS error in Table 5.2 is low compared to the previously dis-



(a) (b) (c)

Figure 5.7: Depth maps and error visualization for the *venus* scene from Table 5.2. (a) Ground-truth depth values; (b) direct depth; (c) inverse depth.



(a) (b) (c)

Figure 5.8: Depth values and error visualization for the *cloth1* scene from Table 5.2. (a) Ground-truth depth values; (b) direct depth; (c) inverse depth.

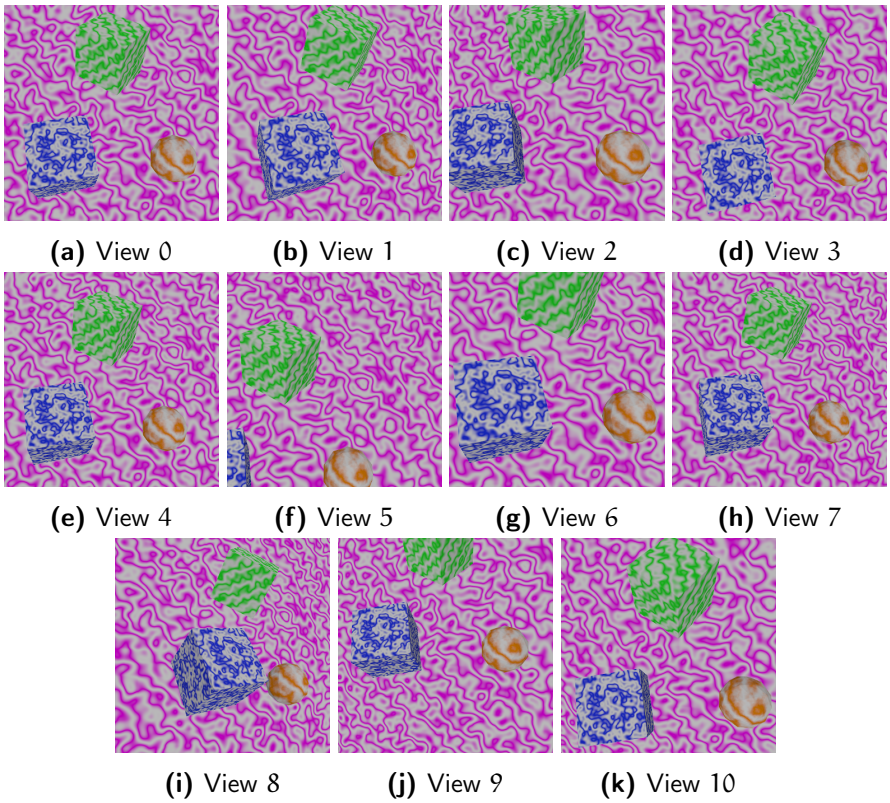
cussed scenes, the computed depth maps and the visualization of the errors in Figure 5.7 show that only the foreground objects were reconstructed accurately. To distinguish the foreground from the background, the contrast of the depth maps had to be increased. At the boundaries of the foreground objects the depth difference is large and neither parameterization was able to to accurately develop the edges. However, the *venus* data set shows that the inverse depth parameterization can produce better results for ortho-parallel camera configurations, too.

The last test scene is *cloth1* and the RMS error in Table 5.2 suggests that the inverse depth parameterization performs far better than the direct depth. However, the depth maps and error visualizations in Figure 5.8 show that the reconstruction quality in both cases is similar. Apart from a high error in the lower right corner and a smaller error at the horizontal crease, the result from the direct depth is indistinguishable from the result from the inverse depth. For the most part, the surface of the cloth is closed, which favors both depth parameterizations. At the depth discontinuities the errors with a blue tint are more frequent for the inverse depth than for the direct depth. However, for errors with a yellow tint the opposite is true. As with the previous scene, the results for *cloth1* show that the inverse depth parameterization does not have a disadvantage in ortho-parallel camera setups.

5.4 AUTOMATIC VIEW SELECTION

To evaluate the automatic view selection, two different scenes were used. The first is a newly created scene that consists of three different shapes in front of a plane parallel to the reference camera's image plane. Ten additional cameras were randomly placed, so that at least some parts of the scene that are visible in the reference camera are visible in any match camera; some cameras are angled towards and some away from the reference camera. The reference view and all match views are depicted in Figure 5.9, showing a variety of different camera angles and positions. The second scene is the familiar *cloth1* data set. The previous two sections showed that the first-order isotropic regularization is preferable in scenes with large depth differences between objects and that the inverse depth performs better in the case of random camera placement. Consequently, the inverse depth with first-order regularization was used for the *shapes* scene. For the *cloth1* scene, second-order isotropic regularization was used with the inverse depth. Occlusion handling was used for both scenes, the threshold was set to 0.001 for *shapes*; the downscaling factor η was set to 0.93. The parameters for *cloth1* were not changed.

The goal of automatic view selection is twofold: First, the process should discard the views that have a detrimental effect on the so-



(a) View 0 **(b)** View 1 **(c)** View 2 **(d)** View 3

(e) View 4 **(f)** View 5 **(g)** View 6 **(h)** View 7

(i) View 8 **(j)** View 9 **(k)** View 10

Figure 5.9: The 11 views from the *shapes* data set. View 0 is used as the reference view.

lution. Second, those views that contribute the most to an accurate reconstruction should be selected. If this is not possible the view selection should not worsen the result by much, but still reduce the overall execution time. To measure the quality, the RMS error is used again and the execution times using an Intel Xeon E3-1231 v3 @ 4×3.4 GHz were recorded for the selected parameters, averaged over four runs. As a baseline, the depth maps for both test cases were computed using all available views. For the *shapes* data set this resulted in an RMS error of 0.707 with an average runtime of 50.8 seconds. The RMS error for *cloth1* is 0.201 with a mean execution time of 196 seconds. All numbers were rounded to three significant digits. In Tables 5.3 and 5.4, the first line for each scene shows the results using all views.

The results for the geometry-based view selector are presented in Table 5.3, the computed depth maps and error visualizations for *shapes* are shown in Figure 5.10. For the *shapes* scene, the table shows that the view selector is able to find a subset of the available views that reduces the RMS error as well as the runtime. By choosing the views 3, 4 and 6 the error is reduced to 63.6% of the original error while the computation of the depth values took less than half the time. The results also show that to increase the quality of the reconstruction it is not sufficient to simply add or remove views. In both cases with one view fewer and one additional view the RMS error

Scene	Θ [°]	β	ν	t [s]	RMS
shapes	-	-	$\{1, \ldots, 10\}$	50.8	0.707
	8	1.4	$\{4, 6\}$	16.3	0.483
	9	2.0	$\{3, 4, 6\}$	19.8	0.450
	10	2.5	$\{3, 4, 6, 7\}$	26.2	0.460
cloth1	-	-	$\{0, 2, \ldots, 6\}$	196	0.201
	-	0.6	$\{0, 2\}$	97.3	0.215
	-	1.0	$\{0, 2, 3\}$	123	0.204
	-	1.4	$\{0, 2, 3, 4\}$	150	0.201

Table 5.3: Results for the *shapes* and *cloth1* scenes using the geometry-based view selector with different baselines and angles. For *cloth1* the angle has no effect. Results rounded to three significant digits and timings averaged over four executions.

is slightly worse. In Figures 5.10b and 5.10d it can be seen that the sphere and the cube at the top are reconstructed well compared to the blue cube. The main contribution to the error stems from the latter. The distance of the objects from the reference camera decreases from the blue to the green cube and then to the sphere. The relative motion decreases in the same order and it can be seen that the reconstruction quality increases accordingly.

In the *cloth1* case, no increase of quality was achieved. Using the views 0 and 2–4 maintains the original error, while removing one or two of the cameras farthest away from the reference camera increases the error to 0.204 and 0.215, respectively. However, this scene illustrates the case in which the view selection is at least able to reduce the required execution time. With four views and no change of the error the runtime is reduced by 23.4%. If a higher error is acceptable a runtime reduction of up to 50.3% is possible. An explanation for this outcome is the ortho-parallel camera setup. As seen during the discussion of the results of the cube scenes, such a setup is more easily reconstructed. This limits the potential for further error reduction.

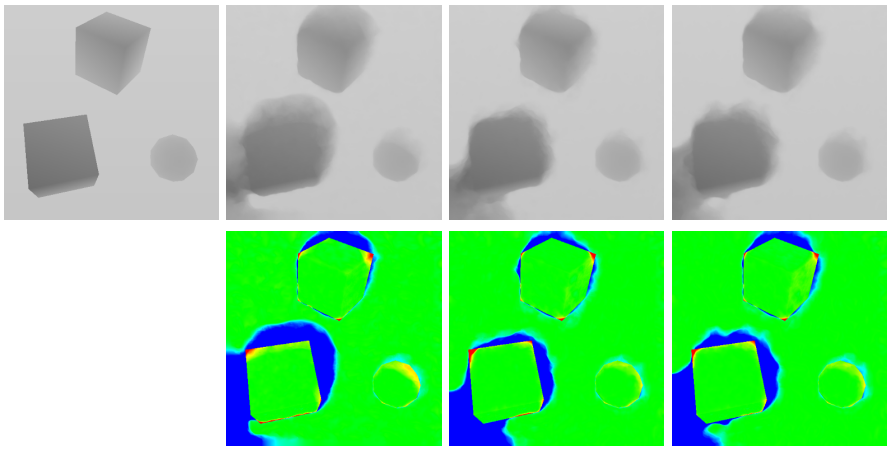
The results for the feature-based view selection with $\tau = 0.8$ and $\delta = 0.001$ are shown in Table 5.4, the computed depth maps and error visualizations can be found in Figure 5.11. For *shapes*, with $\kappa = 0.7$ only the views 2, 3, 8 and 9 are discarded, but the error is reduced by 17.8%. Those views do not have enough overlap with the reference view and too many feature matches violated the epipolar constraint. Of note is view 3, which was chosen by the geometry-based view selector and lead to a minimal RMS error, but was discarded by the feature-based view selection. For $\kappa = 0.75$ only three views remain, but the error is not reduced much further. However, by increasing κ to 0.8 one more view was removed and the error was more than halved.

Scene	κ	ν	t [s]	RMS
shapes	-	$\{1,\ldots,10\}$	50.8	0.707
	0.70	$\{1,4,\ldots,7,10\}$	35.6	0.581
	0.75	$\{4,5,7\}$	20.9	0.541
	0.80	$\{4,5\}$	16.2	0.232
cloth1	-	$\{0,2,\ldots,6\}$	196	0.201
	0.70	$\{0,2,3,4\}$	150	0.201
	0.75	$\{0,2,3\}$	123	0.204
	0.80	$\{2\}$	69.2	0.204

Table 5.4: Results for the *shapes* and *cloth1* scenes using the feature-based view selector with different set ratios, $\tau = 0.8$ and $\delta = 0.001$. Results rounded to three significant digits and timings averaged over four executions.

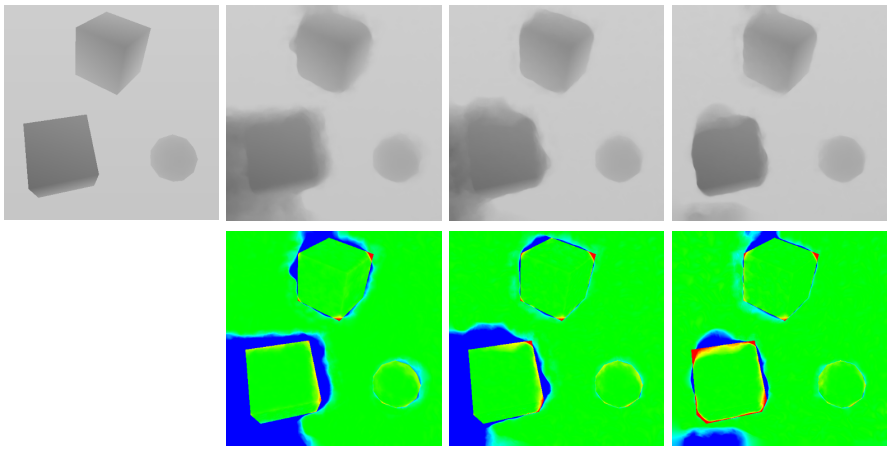
As a result, it was possible to reduce the error by 67.2% using the feature-based view selector, achieving a decrease in runtime of 68.2%. The results in Figures 5.11b to 5.11d are analogous to those from the geometry-based view selector. Even in the worst case, Figure 5.11b, the green cube and sphere are reconstructed as good as in the best case of the geometry-based view selection, as shown in Figure 5.10c. Initially, the blue cube does exhibit more errors at the edges. But by removing more views the edges become more pronounced and the computed depth values match the ground truth more closely than those computed using the other view selector.

The results for the *cloth1* scene are similar to those for the geometry-based view selection. For $\kappa = 0.7$ and $\kappa = 0.75$ the same sets of views are found as for $\beta = 1.4$ and $\beta = 1.0$. This indicates that for views closer to the reference camera less feature matches violate the epipolar constraint. However, for $\kappa = 0.8$ view 0 was discarded, resulting in the same error as for $\kappa = 0.75$, but greatly reducing the execution time. Compared to the geometry-based approach, the feature-based view selection further reduced the runtime to 35.3% of the original runtime at the cost of a 1.49% increase of the RMS error.



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

Figure 5.10: Depth values and error visualization for the *shapes* scene from Table 5.3 with geometry-based view selection. (a) Ground-truth depth values; (b) $\beta = 1.4$, $\Theta = 8°$; (b) $\beta = 2.0$, $\Theta = 9°$; (c) $\beta = 2.5$, $\Theta = 10°$.



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

Figure 5.11: Depth values and error visualization for the *shapes* scene from Table 5.3 with feature-based view selection. (a) Ground-truth depth values; (b) $\kappa = 0.70$; (b) $\kappa = 0.75$; (c) $\kappa = 0.80$.

6 CONCLUSION

In this thesis an approach was presented that extends the variational method for stereo reconstruction by Maurer [Mau14] to a multi-view approach. In addition, a parameterization function was introduced, that allows to utilize arbitrary depth parameterizations, such as the inverse depth. While the depth parameterization allowed a straightforward extension to multiple views, the evaluation showed that not only the number of match images used is important, but also the strategy to select them. Evaluation also showed that using multiple match images consistently improved the quality of the resulting depth images. Furthermore, evaluation revealed that only few additional match images are required, with more images either resulting in little improvement or even decreasing the quality. With proper occlusion handling it was possible to further reduce the reconstruction error.

The inverse depth was presented as an alternative to the common direct depth parameterization. Examination of the computed depths and the resulting surface for both parameterizations showed that the inverse depth is less susceptible to linearization errors, if the translation from the reference camera to a match camera has mainly lateral movement. Further, it was shown that combined with a second-order isotropic regularization the inverse depth does not exhibit a systematic error. Using the direct depth, affine surfaces cannot be reconstructed accurately because the back-projection of affine depth values results in a curved surface. This is not the case when using the inverse depth. Evaluation showed that using the inverse depth with the second-order isotropic regularization, better results are achievable.

With the forward-backward consistency check, a method was presented that allows to remove depth values which are inconsistent across different views. Evaluation of the method showed that the removed values were the ones with the largest errors. As a consequence, the confidence in the correctness of the remaining depth values is increased.

Two different approaches to select a suitable subset of match views from all available views were presented: A geometry-based approach that takes the position and orientation of the match cameras relative to the reference camera into consideration, enforcing converging setups with adjustable angles and baseline; and a feature-based approach that considers the image contents and maximizes overlap. Evaluation showed that both approaches are able to reduce the over-

all error or at least remove views that have only small influence on the solution, thus decreasing runtime.

FUTURE WORK The presented framework allows to use any invertible function as a depth parameterization, the effects and benefits of further ones have to be examined. The inverse depth was chosen because it works well with second-order regularization, Graber et al. [Gra+15] chose their parameterization to address the specific problem of non-convexity. Different parameterizations that cope with other aspects of the variational approach might be possible.

So far, second-order isotropic regularization based on the Frobenius norm of the Hessian was used to evaluate the inverse depth parameterization. Another possibility to design this smoothness term is to use a coupling term, such as done by Schroers et al. [SHW15]. A comparison between both approaches could determine which one is favorable in combination with the inverse depth.

Using multiple views not only improves the resulting depth maps but also allows to perform the reconstruction from multiple vantage points, thus covering parts of the scene that are occluded in the two views used in regular stereo reconstruction. This results in multiple point clouds. Using range image integration as proposed by Zach [Zac08] or Schroers et al. [Sch+12] for example, the depth maps could be combined to retrieve a closed surface.

Geometry- and feature-based view selection are currently two independent approaches. Goesele et al. [Goe+07] for example used aspects of both methods. The techniques presented here could be combined to enforce converging setups with large overlap, for example.

A

GEOMETRIC DERIVATIONS

A.1 CAMERA COORDINATE TRANSFORMATIONS

Given are a matrix $\mathcal{R} \in \mathbb{R}^{3\times3}$ describing a camera's orientation in relation to the world coordinate frame and a vector $\mathcal{C} \in \mathbb{R}^3$ representing the location of the camera in world coordinates. Together they form the transformation of the camera relative to the world coordinate frame:

$$T_{\text{world}\to\text{camera}} = \begin{bmatrix} \mathcal{R} & \mathcal{C} \\ 0 & 1 \end{bmatrix}$$

The inverse transformation matrix that transforms points from the camera coordinate frame to the world coordinate frame is derived as follows:

$$\begin{aligned} T_{\text{camera}\to\text{world}} &= T^{-1}_{\text{world}\to\text{camera}} \\ &= \begin{bmatrix} \mathcal{R} & \mathcal{C} \\ 0 & 1 \end{bmatrix}^{-1} = \left[\begin{bmatrix} \mathcal{I} & \mathcal{C} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathcal{R} & 0 \\ 0 & 1 \end{bmatrix} \right]^{-1} \\ &= \begin{bmatrix} \mathcal{R} & 0 \\ 0 & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathcal{I} & \mathcal{C} \\ 0 & 1 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \mathcal{R}^T & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathcal{I} & -\mathcal{C} \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \mathcal{R}^T & -\mathcal{R}^T\mathcal{C} \\ 0 & 1 \end{bmatrix} \end{aligned}$$

A.2 COMPUTATION OF THE FUNDAMENTAL MATRIX

For two calibrated cameras C_0 and C_1, the fundamental matrix F can be computed from the projection matrices M_0 and M_1. The epipolar constraint states that

$$\tilde{\mathbf{p}}_1^T \cdot F \cdot \tilde{\mathbf{p}}_0 = 0 \Leftrightarrow \mathbf{l}_1^T \cdot \tilde{\mathbf{p}}_1 = 0,$$

where $\mathbf{l}_1$ is the epipolar line defined by the epipole $\boldsymbol{e}_1$ and $\mathbf{p}_1$. For the epipolar line, the condition $\mathbf{l}_1 = F \cdot \tilde{\mathbf{p}}_0$ holds. This allows to compute the fundamental matrix by deriving $\mathbf{l}_1$. The epipole $\boldsymbol{e}_1$ is computed by

$$\tilde{\boldsymbol{e}}_1 = M_1 \cdot \tilde{\mathcal{C}}_0,$$

where $\tilde{\mathcal{C}}_0$ is the homogeneous representation of the location of the reference camera in the world coordinate frame. The point $\tilde{\mathbf{p}}_1$ can be computed by

$$\tilde{\mathbf{p}}_1 = M_1 \cdot M_0^+ \cdot \tilde{\mathbf{p}}_0,$$

where M_0^+ is the pseudoinverse of the projection matrix M_0. The projection matrix is not invertible; however, from the extrinsic and intrinsic matrices a pseudoinverse can be computed, such that M_ν^+ satisfies the Moore-Penrose criteria [Moo20; Pen55]:

$$M_\nu^+ = T_\nu^{-1} \cdot \begin{bmatrix} K_\nu^{-1} \\ 0 \end{bmatrix}$$

In $\mathbb{P}^2$ two points $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{q}}$ lie on the line $\mathbf{l}$, if $\mathbf{l}^T\tilde{\mathbf{p}} = \mathbf{l}^T\tilde{\mathbf{q}} = 0$. From this condition it is derived that $\mathbf{l} = \tilde{\mathbf{p}} \times \tilde{\mathbf{q}}$. The epipolar line $\mathbf{l}_1$ is then computed by

$$\begin{aligned} \mathbf{l}_1 &= \tilde{\mathbf{e}}_1 \times \tilde{\mathbf{p}}_0 \\ &= (M_1 \cdot \tilde{\mathcal{C}}_0) \times \left(M_1 \cdot M_0^+ \cdot \tilde{\mathbf{p}}_0\right) \\ &= \left([\tilde{\mathbf{e}}_1]_\times \cdot M_1 \cdot M_0^+\right) \cdot \tilde{\mathbf{p}}_0, \end{aligned}$$

where $[\tilde{\mathbf{e}}_1]_\times$ is a skew-symmetric matrix that expresses the cross product by a matrix multiplication:

$$[a]_\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

For the fundamental matrix finally follows

$$F = [\tilde{\mathbf{e}}_1]_\times \cdot M_1 \cdot M_0^+.$$

B

DERIVATIVES

B.1 IMAGE DERIVATIVES

The gradient ∇ of the back-projected image $\mathcal{I}_\nu$ is related to the gradient ∇_ν, this is derived as follows:

$$\begin{aligned}
(\nabla \mathcal{I}_\nu(\mathbf{p}))^\mathsf{T} &= \begin{bmatrix} \partial_x \mathcal{I}_\nu(\mathbf{p}) & \partial_y \mathcal{I}_\nu(\mathbf{p}) \end{bmatrix} \\
&= \begin{bmatrix} \partial_x I_\nu(x_\nu, y_\nu) & \partial_y I_\nu(x_\nu, y_\nu) \end{bmatrix} \\
&= \begin{bmatrix} (\nabla_\nu I_\nu(x_\nu, y_\nu))^\mathsf{T} \cdot \partial_x \begin{bmatrix} x_\nu \\ y_\nu \end{bmatrix} & \nabla_\nu I_\nu(x_\nu, y_\nu))^\mathsf{T} \cdot \partial_y \begin{bmatrix} x_\nu \\ y_\nu \end{bmatrix} \end{bmatrix} \\
&= (\nabla_\nu I_\nu(x_\nu, y_\nu))^\mathsf{T} \cdot \underbrace{\begin{bmatrix} \partial_x x_\nu & \partial_y x_\nu \\ \partial_x y_\nu & \partial_y y_\nu \end{bmatrix}}_{J}
\end{aligned}$$

This results in

$$(\nabla_\nu I_\nu(x_\nu, y_\nu))^\mathsf{T} = (\nabla \mathcal{I}_\nu(\mathbf{p}))^\mathsf{T} \cdot J^{-1}.$$

The required inverse Jacobian J^{-1} is computed by

$$J^{-1} = \frac{1}{\det J} \cdot \begin{bmatrix} \partial_y y_\nu & -\partial_x y_\nu \\ -\partial_y x_\nu & \partial_x x_\nu \end{bmatrix},$$

where the partial derivatives of x_ν and y_ν are computed as described in Appendix B.5.

B.2 DISCRETE ISOTROPIC FIRST-ORDER SMOOTHNESS

Here, the nested, central second order derivative of the divergence expression of the smoothness term is derived. While it is possible to use a difference scheme that does not use interpolation and fractional indices, doing so allows to use direct neighbors. The derivation reads

$$\begin{aligned}
&\operatorname{div}\left((\Psi')^{k,l}_S \cdot \nabla (\rho^{k,l} + d\rho^{k,l+1}) \right) \\
&= \left((\Psi')^{k,l}_S \cdot (\rho^{k,l} + d\rho^{k,l+1})_x \right)_x \\
&+ \left((\Psi')^{k,l}_S \cdot (\rho^{k,l} + d\rho^{k,l+1})_y \right)_y \\
&\approx \frac{\left((\Psi')^{k,l}_S \cdot (\rho^{k,l} + d\rho^{k,l+1})_x \right)_{i+1/2,j}}{h_x}
\end{aligned}$$

$$
\begin{aligned}
&- \frac{\left((\Psi')_S^{k,l} \cdot (\rho^{k,l} + d\rho^{k,l+1})_x\right)_{i-1/2,j}}{h_x} \\
&+ \frac{\left((\Psi')_S^{k,l} \cdot (\rho^{k,l} + d\rho^{k,l+1})_y\right)_{i,j+1/2}}{h_y} \\
&- \frac{\left((\Psi')_S^{k,l} \cdot (\rho^{k,l} + d\rho^{k,l+1})_y\right)_{i,j-1/2}}{h_y} \\
&= (\Psi')^{k,l}_{S\,i+1/2,j} \cdot \frac{\partial_x(\rho^{k,l} + d\rho^{k,l+1})_{i+1/2,j}}{h_x} \\
&- (\Psi')^{k,l}_{S\,i-1/2,j} \cdot \frac{\partial_x(\rho^{k,l} + d\rho^{k,l+1})_{i-1/2,j}}{h_x} \\
&+ (\Psi')^{k,l}_{S\,i,j+1/2} \cdot \frac{\partial_y(\rho^{k,l} + d\rho^{k,l+1})_{i,j+1/2}}{h_y} \\
&- (\Psi')^{k,l}_{S\,i,j-1/2} \cdot \frac{\partial_y(\rho^{k,l} + d\rho^{k,l+1})_{i,j-1/2}}{h_y} \\
&= \frac{(\Psi')^{k,l}_{S\,i+1,j} + (\Psi')^{k,l}_{S\,i,j}}{2} \cdot \frac{\left(\rho^{k,l}_{i+1/2,j} + d\rho^{k,l+1}_{i+1/2,j}\right)_x}{h_x} \\
&- \frac{(\Psi')^{k,l}_{S\,i,j} + (\Psi')^{k,l}_{S\,i-1,j}}{2} \cdot \frac{\left(\rho^{k,l}_{i-1/2,j} + d\rho^{k,l+1}_{i-1/2,j}\right)_x}{h_x} \\
&+ \frac{(\Psi')^{k,l}_{S\,i,j+1} + (\Psi')^{k,l}_{S\,i,j}}{2} \cdot \frac{\left(\rho^{k,l}_{i,j+1/2} + d\rho^{k,l+1}_{i,j+1/2}\right)_y}{h_y} \\
&- \frac{(\Psi')^{k,l}_{S\,i,j} + (\Psi')^{k,l}_{S\,i,j-1}}{2} \cdot \frac{\left(\rho^{k,l}_{i,j-1/2} + d\rho^{k,l+1}_{i,j-1/2}\right)_y}{h_y} \\
&\approx \frac{(\Psi')^{k,l}_{S\,i+1,j} + (\Psi')^{k,l}_{S\,i,j}}{2 \cdot h_x^2} \cdot \left(\rho^{k,l}_{i+1,j} - \rho^{k,l}_{i,j} + d\rho^{k,l+1}_{i+1,j} - d\rho^{k,l+1}_{i,j}\right) \\
&- \frac{(\Psi')^{k,l}_{S\,i,j} + (\Psi')^{k,l}_{S\,i-1,j}}{2 \cdot h_x^2} \cdot \left(\rho^{k,l}_{i,j} - \rho^{k,l}_{i-1,j} + d\rho^{k,l+1}_{i,j} - d\rho^{k,l+1}_{i-1,j}\right) \\
&+ \frac{(\Psi')^{k,l}_{S\,i,j+1} + (\Psi')^{k,l}_{S\,i,j}}{2 \cdot h_y^2} \cdot \left(\rho^{k,l}_{i,j+1} - \rho^{k,l}_{i,j} + d\rho^{k,l+1}_{i,j+1} - d\rho^{k,l+1}_{i,j}\right) \\
&- \frac{(\Psi')^{k,l}_{S\,i,j} + (\Psi')^{k,l}_{S\,i,j-1}}{2 \cdot h_y^2} \cdot \left(\rho^{k,l}_{i,j} - \rho^{k,l}_{i,j-1} + d\rho^{k,l+1}_{i,j} - d\rho^{k,l+1}_{i,j-1}\right).
\end{aligned}
$$

B.3 DISCRETE ISOTROPIC SECOND-ORDER SMOOTHNESS

The discretization of the smoothness term related part for the second-order regularizer is derived. The following abbreviations are used:

$$
\begin{aligned}
\Psi' &\coloneqq (\Psi')_S^{k,l} \\
P &\coloneqq \rho^{k,l} + d\rho^{k,l+1}.
\end{aligned}
$$

The part related to the smoothness expands as

$$\nabla_{\mathcal{H}}^{\mathsf{T}}(\Psi' \cdot \nabla_{\mathcal{H}} P) = \underbrace{(\Psi' \cdot (P)_{xx})_{xx}}_{=:A} + \underbrace{2 \cdot (\Psi' \cdot (P)_{xy})_{xy}}_{=:B} + \underbrace{(\Psi' \cdot (P)_{yy})_{yy}}_{=:C}$$

and requires a nested difference scheme to approximate the second and fourth derivatives. The fourth derivatives are approximated by repeating the difference scheme for the second derivatives. The approximations of the individual parts A, B and C then reads

$$\begin{aligned}
A &\approx \frac{1}{h_x^2} \cdot \Big(\Psi'_{i+1,j} \cdot \big(P_{i+1,j}\big)_{xx} - 2 \cdot \Psi'_{i,j} \cdot \big(P_{i,j}\big)_{xx} + \Psi'_{i-1,j} \cdot \big(P_{i-1,j}\big)_{xx} \Big) \\
&= \frac{1}{h_x^4} \cdot \Big(\Psi'_{i+1,j} \cdot \big(P_{i+2,j} - 2 \cdot P_{i+1,j} + P_{i,j}\big) \\
&\qquad - 2 \cdot \Psi'_{i,j} \cdot \big(P_{i+1,j} - 2 \cdot P_{i,j} + P_{i-1,j}\big) \\
&\qquad + \Psi'_{i-1,j} \cdot \big(P_{i,j} - 2 \cdot P_{i-1,j} + P_{i-2,j}\big) \Big),
\end{aligned}$$

$$\begin{aligned}
B &\approx \frac{2}{4 \cdot h_x^2 \cdot h_y^2} \cdot (\Psi'_{i+1,j+1} \cdot (P_{i+1,j+1})_{xy} - \Psi'_{i-1,j+1} \cdot (P_{i-1,j+1})_{xy} \\
&\qquad - \Psi'_{i+1,j-1} \cdot (P_{i+1,j-1})_{xy} + \Psi'_{i-1,j-1} \cdot (P_{i-1,j-1})_{xy}) \\
&= \frac{1}{8 \cdot h_x^4 \cdot h_y^4} \cdot \Big(\Psi'_{i+1,j+1} \cdot \big(P_{i+2,j+2} - P_{i,j+2} - P_{i+2,j} + P_{i,j}\big) \\
&\qquad - \Psi'_{i-1,j+1} \cdot \big(P_{i,j+2} - P_{i-2,j+2} - P_{i,j} + P_{i-2,j}\big) \\
&\qquad - \Psi'_{i+1,j-1} \cdot \big(P_{i+2,j} - P_{i,j} - P_{i+2,j-2} + P_{i,j-2}\big) \\
&\qquad + \Psi'_{i-1,j-1} \cdot \big(P_{i,j} - P_{i-2,j} - P_{i,j-2} + P_{i-2,j-2}\big) \Big),
\end{aligned}$$

$$\begin{aligned}
C &\approx \frac{1}{h_y^2} \cdot \Big(\Psi'_{i,j+1} \cdot \big(P_{i,j+1}\big)_{xx} - 2 \cdot \Psi'_{i,j} \cdot \big(P_{i,j}\big)_{xx} + \Psi'_{i,j-1} \cdot \big(P_{i,j-1}\big)_{xx} \Big) \\
&= \frac{1}{h_y^4} \cdot \Big(\Psi'_{i,j+1} \cdot \big(P_{i,j+2} - 2 \cdot P_{i,j+1} + P_{i,j}\big) \\
&\qquad - 2 \cdot \Psi'_{i,j} \cdot \big(P_{i,j+1} - 2 \cdot P_{i,j} + P_{i,j-1}\big) \\
&\qquad + \Psi'_{i,j-1} \cdot \big(P_{i,j} - 2 \cdot P_{i,j-1} + P_{i,j-2}\big) \Big)
\end{aligned}$$

B.4 DERIVATIVES OF THE DEPTH PARAMETERIZATION

Given are two functions, Φ and ρ, that are defined as

$$\begin{aligned}
\Phi : \quad & \mathbb{R} \to \mathbb{R} \\
& \rho \mapsto \Phi(\rho) \\
\rho : \mathbb{R}^2 & \to \mathbb{R} \\
& (\mathbf{p}) \mapsto \rho(\mathbf{p}),
\end{aligned} \tag{B.1}$$

with $\mathbf{p} = [x\ y]$. Φ and ρ are both at least two times differentiable. To compute the first and second derivatives of $\Phi(\rho(\mathbf{p}))$ application of the chain rule is required. The first derivatives are:

$$\begin{aligned}\partial_x \Phi(\rho(\mathbf{p})) &= \Phi'(\rho(\mathbf{p})) \cdot \rho_x(\mathbf{p}) \\ \partial_y \Phi(\rho(\mathbf{p})) &= \Phi'(\rho(\mathbf{p})) \cdot \rho_y(\mathbf{p})\end{aligned} \tag{B.2}$$

The second derivatives are computed by applying the chain and product rule to the first derivatives. For $\partial_{xx}\Phi(\rho(\mathbf{p}))$ the derivation is:

$$\begin{aligned}\partial_{xx}\Phi(\rho(\mathbf{p})) &= \partial_x \partial_x \Phi(\rho(\mathbf{p})) \\ &= \partial_x \left(\Phi'(\rho(\mathbf{p})) \cdot \rho_x(\mathbf{p})\right) \\ &= \partial_x \Phi'(\rho(\mathbf{p})) \cdot \rho_x(\mathbf{p}) + \Phi'(\rho(\mathbf{p})) \cdot \partial_x \rho_x(\mathbf{p}) \\ &= \Phi''(\rho(\mathbf{p})) \cdot \rho_x(\mathbf{p})^2 + \Phi'(\rho(\mathbf{p})) \cdot \rho_{xx}(\mathbf{p})\end{aligned} \tag{B.3}$$

The other derivatives are derived analogously and since both functions are two times differentiable the symmetry of the second derivatives applies:

$$\begin{aligned}\partial_{yy}\Phi(\rho(\mathbf{p})) &= \Phi''(\rho(\mathbf{p})) \cdot \rho_y(\mathbf{p})^2 + \Phi'(\rho(\mathbf{p})) \cdot \rho_{yy}(\mathbf{p}) \\ \partial_{xy}\Phi(\rho(\mathbf{p})) &= \Phi''(\rho(\mathbf{p})) \cdot \rho_x(\mathbf{p}) \cdot \rho_y(\mathbf{p}) + \Phi'(\rho(\mathbf{p})) \cdot \rho_{xy}(\mathbf{p})\end{aligned} \tag{B.4}$$

B.5 DERIVATIVES OF BACK-PROJECTED POINTS

As noted in Section 3.1.2, the coordinates of the back-projected points are functions of $\mathbf{p}$ and ρ and defined as follows:

$$\begin{aligned}x_v(\mathbf{p}, \Phi \circ \rho) &= \frac{\Phi(\rho(\mathbf{p})) \cdot a(\mathbf{p}) + b(\mathbf{p})}{\Phi(\rho(\mathbf{p})) \cdot c(\mathbf{p}) + d(\mathbf{p})} \\ y_v(\mathbf{p}, \Phi \circ \rho) &= \frac{\Phi(\rho(\mathbf{p})) \cdot \check{a}(\mathbf{p}) + \check{b}(\mathbf{p})}{\Phi(\rho(\mathbf{p})) \cdot c(\mathbf{p}) + d(\mathbf{p})}\end{aligned} \tag{B.5}$$

a, b, $\check{a}$, $\check{b}$, c and d are defined as

$$\begin{aligned}a(\mathbf{p}) &= m_{11} \cdot \frac{x - o_x}{s_x} + m_{12} \cdot \frac{y - o_y}{s_y} + m_{13}, \\ b(\mathbf{p}) &= m_{14}, \\ \check{a}(\mathbf{p}) &= m_{21} \cdot \frac{x - o_x}{s_x} + m_{22} \cdot \frac{y - o_y}{s_y} + m_{23}, \\ \check{b}(\mathbf{p}) &= m_{24}, \\ c(\mathbf{p}) &= m_{31} \cdot \frac{x - o_x}{s_x} + m_{32} \cdot \frac{y - o_y}{s_y} + m_{33}, \\ d(\mathbf{p}) &= m_{34},\end{aligned} \tag{B.6}$$

where m_{ij} are the entries of the full projection matrix M. For the sake of brevity, the arguments to any function are omitted from now on.

Given any two times differentiable function $f : \mathbb{R}^2 \to \mathbb{R}$, where $f(\mathbf{p}) = \frac{N(\mathbf{p})}{D(\mathbf{p})}$. The first and second derivatives are computed by applying the quotient rule once and twice. In general, the derivatives read

$$\begin{aligned}
\partial_x f &= \frac{\partial_x N \cdot D - N \cdot \partial_x D}{D^2}, \\
\partial_y f &= \frac{\partial_y N \cdot D - N \cdot \partial_y D}{D^2}, \\
\partial_{xx} f &= \frac{\partial_{xx} N \cdot D - N \cdot \partial_{xx} D}{D^2} - \frac{2 \cdot \partial_x D \cdot (\partial_x N \cdot D - N \cdot \partial_x D)}{D^3}, \\
\partial_{yy} f &= \frac{\partial_{yy} N \cdot D - N \cdot \partial_{yy} D}{D^2} - \frac{2 \cdot \partial_y D \cdot (\partial_y N \cdot D - N \cdot \partial_y D)}{D^3}, \\
\partial_{xy} f &= \frac{\partial_{xy} N \cdot D - \partial_x N \cdot \partial_y D - \partial_y N \cdot \partial_x D - N \cdot \partial_{xy} D}{D^2} \\
&\quad + \frac{2 \cdot \partial_y D \cdot N \cdot \partial_x D}{D^3}.
\end{aligned} \tag{B.7}$$

From this general form it is easy to derive the sought-after first and second derivatives of x_ν and y_ν simply by inserting the appropriate parts. These read:

$$\begin{aligned}
\partial_x N^x &= \Phi' \cdot \partial_x \rho \cdot a + \Phi \cdot \frac{m_{11}}{s_x}, \\
\partial_y N^x &= \Phi' \cdot \partial_y \rho \cdot a + \Phi \cdot \frac{m_{12}}{s_y}, \\
\partial_{xx} N^x &= \Phi'' \cdot (\partial_x \rho)^2 \cdot a + \Phi' \cdot \partial_{xx} \rho \cdot a + 2\Phi' \cdot \partial_x \rho \cdot \frac{m_{11}}{s_x}, \\
\partial_{yy} N^x &= \Phi'' \cdot (\partial_y \rho)^2 \cdot a + \Phi' \cdot \partial_{yy} \rho \cdot a + 2\Phi' \cdot \partial_y \rho \cdot \frac{m_{12}}{s_y}, \\
\partial_{xy} N^x &= \Phi'' \cdot \partial_x \rho \cdot \partial_y \rho \cdot a + \Phi' \cdot \partial_{yy} \rho \cdot a + \Phi' \cdot \partial_x \rho \cdot \frac{m_{11}}{s_x} \\
&\quad + \Phi' \cdot \partial_y \rho \cdot \frac{m_{12}}{s_y}
\end{aligned}$$

$$\begin{aligned}
\partial_x N^y &= \Phi' \cdot \partial_x \rho \cdot \breve{a} + \Phi \cdot \frac{m_{21}}{s_x}, \\
\partial_y N^y &= \Phi' \cdot \partial_y \rho \cdot \breve{a} + \Phi \cdot \frac{m_{22}}{s_y}, \\
\partial_{xx} N^y &= \Phi'' \cdot (\partial_x \rho)^2 \cdot \breve{a} + \Phi' \cdot \partial_{xx} \rho \cdot \breve{a} + 2\Phi' \cdot \partial_x \rho \cdot \frac{m_{21}}{s_x}, \\
\partial_{yy} N^y &= \Phi'' \cdot (\partial_y \rho)^2 \cdot \breve{a} + \Phi' \cdot \partial_{yy} \rho \cdot \breve{a} + 2\Phi' \cdot \partial_y \rho \cdot \frac{m_{22}}{s_y}, \\
\partial_{xy} N^y &= \Phi'' \cdot \partial_x \rho \cdot \partial_y \rho \cdot \breve{a} + \Phi' \cdot \partial_{yy} \rho \cdot \breve{a} + \Phi' \cdot \partial_x \rho \cdot \frac{m_{21}}{s_x} \\
&\quad + \Phi' \cdot \partial_y \rho \cdot \frac{m_{22}}{s_y}
\end{aligned}$$

$$
\begin{aligned}
\partial_x D &= \Phi' \cdot \partial_x \rho \cdot c + \Phi \cdot \frac{m_{31}}{s_x}, \\
\partial_y D &= \Phi' \cdot \partial_y \rho \cdot c + \Phi \cdot \frac{m_{32}}{s_y}, \\
\partial_{xx} D &= \Phi'' \cdot (\partial_x \rho)^2 \cdot c + \Phi' \cdot \partial_{xx} \rho \cdot c + 2\Phi' \cdot \partial_x \rho \cdot \frac{m_{31}}{s_x}, \\
\partial_{yy} D &= \Phi'' \cdot (\partial_y \rho)^2 \cdot c + \Phi' \cdot \partial_{yy} \rho \cdot c + 2\Phi' \cdot \partial_y \rho \cdot \frac{m_{32}}{s_y}, \\
\partial_{xy} D &= \Phi'' \cdot \partial_x \rho \cdot \partial_y \rho \cdot c + \Phi' \cdot \partial_{yy} \rho \cdot c + \Phi' \cdot \partial_x \rho \cdot \frac{m_{31}}{s_x} \\
&\quad + \Phi' \cdot \partial_y \rho \cdot \frac{m_{32}}{s_y},
\end{aligned}
$$

where N^x denotes the numerator of x_ν, N^y the numerator of y_ν and D denotes the common denominator.

Finally, the derivatives $\partial_\rho x_\nu$ and $\partial_\rho y_\nu$ are derived as follows:

$$
\begin{aligned}
&\partial_\rho x_\nu(\mathbf{p}, \Phi \circ \rho) \\
&\quad = \frac{(\Phi'(\rho) \cdot a) \cdot (\Phi(\rho) \cdot c + d) - (\Phi(\rho) \cdot a + b) \cdot (\Phi'(\rho) \cdot c)}{(\Phi(\rho) \cdot c + d)^2} \\
&\quad = \frac{\Phi'(\rho) \cdot (\Phi(\rho) \cdot c \cdot a + a \cdot d - \Phi(\rho) \cdot a \cdot c - b \cdot c)}{(\Phi(\rho) \cdot c + d)^2} \\
&\quad = \frac{\Phi'(\rho) \cdot (a \cdot d - b \cdot c)}{(\Phi(\rho) \cdot c + d)^2}
\end{aligned}
$$

$$
\begin{aligned}
&\partial_\rho y_\nu(\mathbf{p}, \Phi \circ \rho) \\
&\quad = \frac{(\Phi'(\rho) \cdot \check{a}) \cdot (\Phi(\rho) \cdot c + d) - (\Phi(\rho) \cdot \check{a} + \check{b}) \cdot (\Phi'(\rho) \cdot c)}{(\Phi(\rho) \cdot c + d)^2} \\
&\quad = \frac{\Phi'(\rho) \cdot (\Phi(\rho) \cdot c \cdot \check{a} + \check{a} \cdot d - \Phi(\rho) \cdot \check{a} \cdot c - \check{b} \cdot c)}{(\Phi(\rho) \cdot c + d)^2} \\
&\quad = \frac{\Phi'(\rho) \cdot (\check{a} \cdot d - \check{b} \cdot c)}{(\Phi(\rho) \cdot c + d)^2}
\end{aligned}
\tag{B.8}
$$

C

DATA PREPARATION

C.1 DATA SETS FROM THE MIDDLEBURY STEREO BENCHMARK

To evaluate the proposed stereo algorithm, data sets from the Middlebury Stereo Benchmark were used. The data sets from consist of several scenes with seven rectified views each [SP07; HS07]. For the view 1 and 5 in each set, ground truth disparity maps are provided. Additionally, the baseline between the views 1 and 5 and the focal lengths in pixels are provided. For all sets there is a minimal disparity that must be added to the computed disparities to get the final depth map. The camera setup is ortho-parallel and the resulting depth is computed by

$$\rho(\mathbf{p}) = \frac{s_x \cdot B}{D(\mathbf{p}) + D_{min}}, \tag{C.1}$$

where B is the baseline and $D(\mathbf{p})$ the disparity map. For data sets from 2006 the focal length is 3740 pixels and the baseline is 160 mm.

The stereo approach in this thesis requires fully calibrated cameras, meaning known intrinsic and extrinsic matrices. The Middlebury data sets do not provide matrices, but those can be derived by the known quantities s_x, B and D_{min} and the camera setup. The first observation is that in an ortho-parallel setup the relative transformation between cameras is purely translational, that is all $\mathcal{R}_v$ are the identity. Further, the setup is such that all translations are along the x-axis, the y- and z-components are zero. All cameras are placed equidistant along the x-axis, simplifying the computation of the translational part for all cameras. Without loss of generality view 1 is assumed to be the reference view with $t_1 = [0\ 0\ 0]^T$. For the fifth view $t_5 = [0\ 0\ -1.6]^T$, with the baseline converted to decimeters to avoid large depth values. The equidistant spacing results in a relative translation of 0.4 dm, for the i-th translation vector it holds that

$$t_v = \begin{bmatrix} 0 \\ (1-v)\cdot 0.4 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ B_v \\ 0 \end{bmatrix}, \quad \forall v \in [0, 6].$$

Due to the additional parameter D_{min} the derivation of the intrinsic matrix is more involved. D_{min} leads to an additional shift of the x-component of the corresponding point and is either compensated by a shifted optical center or an adapted translation vector. The latter results in a translation depending on the disparity per pixel and is thus not useful. This leaves an adaption of the optical center as remaining possibility.

The disparity D_v is defined as the distance in pixels between a point $\mathbf{p}$ in the reference view and its corresponding point $\mathbf{p}_v$ in the match view v. For the ortho-parallel setup, the coordinates of the corresponding point are easily computed by

$$\mathbf{p}_v = \begin{bmatrix} x - D_v(\mathbf{p}) \\ y \end{bmatrix}.$$

Under the assumption that the extrinsic matrix of the reference view 1 is the identity, the extrinsic matrix of the match view v is

$$T_v = \begin{bmatrix} I & t_v \\ 0 & 1 \end{bmatrix},$$

with a fixed baseline B_v. The intrinsic matrices are

$$K_1 = \begin{bmatrix} s_x & 0 & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}, \quad K_5 = \begin{bmatrix} s_x & 0 & \check{o}_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}.$$

The focal lengths and o_y are the same, because the images have the same dimensions. Furthermore, in the ortho-parallel case they are required to cancel out. Doing the usual back-projection and projection onto the image plane, the corresponding point is

$$\begin{bmatrix} x_5 \\ y_5 \end{bmatrix} = \begin{bmatrix} x - o_x - \frac{s_x \cdot B_5}{\rho(\mathbf{p})} + \check{o}_x \\ y - o_y + o_y \end{bmatrix} \stackrel{!}{=} \begin{bmatrix} x - D_5(\mathbf{p}) \\ y \end{bmatrix}.$$

Apparently, the interesting equation is x_5, solving for $\check{o}_x$ results in

$$\check{o}_x = o_x + \frac{s_x \cdot B_5}{\rho(\mathbf{p})} - D_5(\mathbf{p}).$$

Substituting in the equation for the resulting depth yields

$$\begin{aligned} \check{o}_x &= o_x + \frac{s_x \cdot B_5 \cdot (D_5(\mathbf{p}) + D_{min})}{s_x \cdot B_5} - D_5(\mathbf{p}) \\ &= o_x + D_5(\mathbf{p}) + D_{min} - D_5(\mathbf{p}) \\ &= o_x + D_{min} \end{aligned}$$

D_{min} is for camera pair 1 and 5, the required offset for each camera C_v depends on the position in the camera setup. The final intrinsic matrices then read

$$K_v = \begin{bmatrix} s_x & 0 & o_x + (v-1) \cdot \frac{D_{min}}{4} \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}, \quad \forall v \in [0, 6],$$

with $s_x = 3740$, $s_y = {}^{H}/_{W} \cdot s_x$, $o_x = {}^{H}/_{2}$ and $o_y = {}^{W}/_{2}$; W and H are the width and the height of the pictures.

BIBLIOGRAPHY

[BMK12] Tali Basha, Yael Moses, and Nahum Kiryati. "Multi-view Scene Flow Estimation: A View Centered Variational Approach." In: *International Journal of Computer Vision* 101.1 (2012), pp. 6–21. DOI: 10.1007/s11263-012-0542-7.

[Bro+04] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. "High Accuracy Optical Flow Estimation Based on a Theory for Warping." In: *Computer Vision - ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part IV*. Ed. by Tomás Pajdla and Jiří Matas. Vol. 3024. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2004, pp. 25–36.

[BW05] Andrés Bruhn and Joachim Weickert. "Towards Ultimate Motion Estimation: Combining Highest Accuracy with Real-Time Performance." In: *Proc. IEEE International Conference on Computer Vision*. Vol. 1. IEEE Computer Society Press, 2005, pp. 749–755. DOI: 10.1109/ICCV.2005.240.

[BWS05] Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. "Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods." In: *International Journal of Computer Vision* 61.3 (2005), pp. 211–231. DOI: 10.1023/B:VISI.0000045324.43199.43.

[FLM92] Olivier D. Faugeras, Quang-Tuan Luong, and Steve J. Maybank. "Camera Self-Calibration: Theory and Experiments." In: *Computer Vision — ECCV'92: Second European Conference on Computer Vision Santa Margherita Ligure, Italy, May 19–22, 1992 Proceedings*. Ed. by Giulio Sandini. Vol. 558. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1992, pp. 321–334.

[FL04] Olivier Faugeras and Quang-Tuan Luong. *The Geometry of Multiple Images: The Laws that Govern the Formation of Multiple Images of a Scene and Some of Their Applications*. In collab. with Theo Papadopoulo. Cambridge, London: The MIT press, 2004.

[Fox50] C. Fox. *An Introduction to the Calculus of Variations*. Dover Books on Mathematics. Dover Publications, 1950.

[Goe+07] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. "Multi-view stereo for community photo collections." In: *Proc. IEEE International Con-*

ference on Computer Vision. IEEE Computer Society Press, 2007, pp. 1–8. DOI: 10.1109/ICCV.2007.4408933.

[Gra+15] Gottfried Graber, Jonathan Balzer, Stefano Soatto, and Thomas Pock. "Efficient Minimal-Surface Regularization of Perspective Depth Maps in Variational Stereo." In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 4. 2015.

[HZ03] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge: Cambridge University Press, 2003.

[Hir05] Heiko Hirschmüller. "Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information." In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. IEEE Computer Society, 2005, pp. 807–814. DOI: 10.1109/CVPR.2005.56.

[HS07] Heiko Hirschmüller and Daniel Scharstein. "Evaluation of Cost Functions for Stereo Matching." In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society Press, 2007, pp. 1–8. DOI: 10.1109/CVPR.2007.383248.

[HSH05] Heiko Hirschmüller, Frank Scholten, and Gerd Hirzinger. "Stereo Vision Based Reconstruction of Huge Urban Areas from an Airborne Pushbroom Camera (HRSC)." In: *Pattern Recognition: 27th DAGM Symposium, Vienna, Austria, August 31 - September 2, 2005. Proceedings*. Ed. by Walter G. Kropatsch, Robert Sablatnig, and Allan Hanbury. Vol. 3663. Lecture Notes in Computer Science. 2005, pp. 58–66.

[HS81] Berthold K. Horn and Brian G. Schunck. "Determining Optical Flow." In: *Artifical Intelligence* 17 (1981), pp. 185–203.

[Hos+12] Ali Hosseininaveh, Margaret Serpico, Stuart Robson, Mona Hess, Jan Boehm, Ivor Pridden, and Giancarlo Amati. "Automatic Image Selection in Photogrammetric Multi-view Stereo Methods." In: *VAST: International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage*. The Eurographics Association, 2012. DOI: 10.2312/VAST/VAST12/009-016.

[Low04] David G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints." In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94.

[Mau14] Daniel Maurer. "Depth-Driven Variational Methods for Stereo Reconstruction." MA thesis. University of Stuttgart, 2014.

[MP98] Etienne Memin and Patrick Perez. "A Multigrid Approach for Hierarchical Motion Estimation." In: *Proc. IEEE International Conference on Computer Vision*. IEEE Computer Society Press, 1998, pp. 933–938.

[Moo20] Eliakim Hastings Moore. "On the Reciprocal of the General Algebraic Matrix." In: *Bulletin of the American Mathematical Society* 26 (1920), pp. 394–395.

[Pap+06] Nils Papenberg, Andrés Bruhn, Thomas Brox, Stephan Didas, and Joachim Weickert. "Highly Accurate Optic Flow Computation with Theoretically Justified Warping." In: *International Journal of Computer Vision* 67.2 (2006), pp. 141–158. DOI: 10.1007/s11263-005-3960-y.

[Pen55] Roger Penrose. "A Generalized Inverse for Matrices." In: *Proc. Cambridge Philosophical Society*. Vol. 51. Cambridge Univ Press, 1955, pp. 406–413.

[Rab+10] Clemens Rabe, Thomas Müller, Andreas Wedel, and Uwe Franke. "Dense, Robust, and Accurate Motion Field Estimation from Stereo Image Sequences in Real-Time." In: *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV*. Ed. by Kostas Daniilidis, Petros Maragos, and Nikos Paragios. Vol. 6314. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010, pp. 582–595.

[RD96] Luc Robert and Rachid Deriche. "Dense Depth Map Reconstruction: A Minimization and Regularization Approach which Preserves Discontinuities." In: *Computer Vision — ECCV '96: 4th European Conference on Computer Vision Cambridge, UK, April 15–18, 1996 Proceedings, Volume I*. Ed. by Bernard Buxton and Roberto Cipolla. Red. by Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen. Vol. 1064. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1996, pp. 439–451.

[ROF92] Leonid I Rudin, Stanley Osher, and Emad Fatemi. "Nonlinear total variation based noise removal algorithms." In: *Physica D: Nonlinear Phenomena* 60.1 (1992), pp. 259–268. DOI: 10.1016/0167-2789(92)90242-F.

[SP07] Daniel Scharstein and Chris Pal. "Learning Conditional Random Fields for Stereo." In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer

Society Press, 2007, pp. 1–8. DOI: 10.1109/CVPR.2007.383191.

[SS02] Daniel Scharstein and Richard Szeliski. "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms." In: *International Journal of Computer Vision* 47.1 (2002), pp. 7–42. DOI: 10.1023/A:1014573219977.

[SHW15] Christopher Schroers, David Hafner, and Joachim Weickert. "Multiview Depth Parameterisation with Second Order Regularisation." In: *Scale Space and Variational Methods in Computer Vision: 5th International Conference, SSVM 2015, Lège-Cap Ferret, France, May 31 - June 4, 2015, Proceedings*. Ed. by Jean-François Aujol, Mila Nikolova, and Nicolas Papadakis. Vol. 9087. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 551–562.

[Sch+12] Christopher Schroers, Henning Zimmer, Levi Valgaerts, Andrés Bruhn, Oliver Demetz, and Joachim Weickert. "Anisotropic Range Image Integration." In: *Pattern Recognition: Joint 34th DAGM and 36th OAGM Symposium, Graz, Austria, August 28-31, 2012. Proceedings*. Ed. by Axel Pinz, Thomas Pock, Horst Bischof, and Franz Leberl. Vol. 7476. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2012, pp. 73–82.

[SBW05] Natalia Slesareva, Andrés Bruhn, and Joachim Weickert. "Optic Flow Goes Stereo: A Variational Method for Estimating Discontinuity-Preserving Dense Disparity Maps." In: *Pattern Recognition: 27th DAGM Symposium, Vienna, Austria, August 31 - September 2, 2005. Proceedings*. Ed. by Walter G. Kropatsch, Robert Sablatnig, and Allan Hanbury. Vol. 3663. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2005, pp. 33–40.

[SGC10] Jan Stühmer, Stefan Gumhold, and Daniel Cremers. "Real-Time Dense Geometry from a Handheld Camera." In: *Pattern Recognition: 32nd DAGM Symposium, Darmstadt, Germany, September 22-24, 2010. Proceedings*. Ed. by Michael Goesele, Stefan Roth, Arjan Kuijper, Bernt Schiele, and Konrad Schindler. Vol. 6376. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010, pp. 11–20.

[Ura+88] Sergio Uras, Frederico Girosi, Alessandro Verri, and Vincent Torre. "A Computational Approach to Motion Perception." In: *Biological Cybernetics* 60.2 (1988), pp. 79–87. DOI: 10.1007/BF00202895.

[Val+10] Levi Valgaerts, Andrés Bruhn, Henning Zimmer, Joachim Weickert, Carsten Stoll, and Christian Theobalt. "Joint Estimation of Motion, Structure and Geometry from Stereo

Sequences." In: *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV*. Ed. by Kostas Daniilidis, Petros Maragos, and Nikos Paragios. Vol. 6314. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010, pp. 568–581.

[You54] David Young. "Iterative Methods for Solving Partial Difference Equations of Elliptic Type." In: *Transactions of the American Mathematical Society* 76.1 (1954), pp. 92–111. DOI: 10.2307/1990745.

[Zac08] Christopher Zach. "Fast and High Quality Fusion of Depth Maps." In: *Proc. International Symposium on 3D Data Processing, Visualization and Transmission*. 2008, pp. 1–8.

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.

Ludwigsburg, April 19, 2016

Oliver Goroll

DEKLARATION

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 übernommenen 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.

Ludwigsburg, 19. April 2016

Oliver Goroll